aws

User Guide

Amazon Q Developer



Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon Q Developer: User Guide

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

Services or capabilities described in AWS documentation might vary by Region. To see the differences applicable to the AWS European Sovereign Cloud Region, see the AWS European Sovereign Cloud User Guide.

Table of Contents

What is Amazon Q Developer? 1
Get started 2
Amazon Q Developer pricing 2
Amazon Q Developer IDE plugins end of support 3
Migrating from Amazon Q Developer to Kiro 3
Getting started 4
Tiers of service 4
Free 4
Pro 5
Get started with a personal account 6
Limitations of Builder IDs 6
Step 1: Sign up 7
Step 2: Install Amazon Q 7
Step 3: (Optional) Upgrade to the Pro tier 7
Get started with IAM Identity Center 7
Step 1: Choose a deployment option 8
Step 2: Subscribe users 13
Pro tier subscriptions 26
Supported Regions 27
Subscription billing 30
Subscription statuses 31
Finding the Start URL 32
Managing the encryption method 33
Q Developer profile 35
Troubleshooting subscriptions 38
Viewing an aggregate list of subscriptions 42
Unsubscribing 44
Upgrading to the Pro tier 47
Upgrade to Kiro 50
Canceling Amazon Q Developer 50
On AWS 52
Authenticating to your Amazon Q Developer Pro subscription 52
Chatting about AWS 53
Using Q artifacts in Amazon Q 54

Add permissions ... 56
Start a conversation ... 56
Manage conversations in the console ... 56
Navigate the Amazon Q chat panel ... 57
Chat settings ... 57
Example prompts ... 58
Chatting about your resources ... 58
Asking Amazon Q to troubleshoot your resources ... 62
Chatting about your costs ... 64
Chatting about your network security ... 67
Chatting about email sending ... 68
Chatting about your telemetry and operations ... 69
Using plugins ... 70
CloudZero ... 71
Datadog ... 78
Wiz ... 85
Console-to-Code ... 92
Console-to-Code ... 92
Where you can use Console-to-Code ... 93
Granting permissions ... 95
Using ... 95
Diagnosing console errors ... 97
Add permissions ... 97
Diagnose common errors in the console ... 98
Chatting with Support ... 98
Prerequisites ... 99
Specify the right service ... 99
Create a support case ... 99
Leave feedback ... 102
In your IDE ... 103
Supported IDEs ... 103
Installing Amazon Q ... 104
Supported IDE versions ... 105
In Eclipse IDEs ... 106
In JetBrains IDEs ... 107
In Visual Studio Code ... 108

In Visual Studio ... 110
IAM principals in your AWS console ... 112
Chatting about code ... 112
Working with Amazon Q in your IDE ... 113
Example tasks ... 115
Example questions ... 116
Reporting issues with responses ... 117
Reviewing code ... 117
Transforming code ... 133
Explaining and updating code ... 185
Chatting inline ... 185
Adding context to the chat ... 188
Chat history compaction ... 197
Managing conversations ... 199
Using shortcut keys in chat ... 201
Selecting models ... 202
Generating inline suggestions ... 203
Pausing suggestions ... 204
Amazon Q code completion in action ... 208
Suggestions in AWS coding environments ... 214
Using shortcut keys ... 222
Using code references ... 228
Code examples ... 238
Supported languages ... 261
Inline suggestions ... 261
Transformations ... 263
Code reviews ... 263
Customizations ... 264
With the Q CLI ... 266
Using MCP ... 267
MCP overview ... 267
In the CLI ... 268
Configuration commands ... 268
Remote MCP servers ... 269
MCP configuration ... 270
Setting up MCP servers with the Q CLI ... 270

Setting up MCP servers with Q in the IDE ... 270
MCP server loading ... 270
Tools and prompts ... 271
Understanding MCP tools ... 271
Discovering available tools ... 271
Using tools ... 272
Working with prompts ... 272
With the IDE ... 272
MCP IDE config files ... 272
Accessing the UI ... 273
Adding an MCP server ... 273
Troubleshooting your MCP configuration ... 275
Enabling an MCP server ... 276
Disabling an MCP server ... 276
Deleting an MCP server that is currently enabled ... 276
Deleting an MCP server that is currently disabled ... 277
Reviewing and adjusting tool permissions ... 277
Key benefits ... 277
MCP architecture ... 277
Core MCP concepts ... 278
Tools ... 278
Prompts ... 279
Resources ... 279
MCP security ... 279
Security model ... 279
Security considerations ... 280
MCP governance ... 280
Disabling MCP for your organization ... 281
Specifying an MCP allow-list for your organization ... 281
Third-party integration ... 295
GitLab Duo with Amazon Q Developer ... 295
Amazon Q Developer for GitHub (Preview) ... 295
Project rules for Amazon Q Developer ... 296
GitLab Duo ... 296
GitLab Duo concepts ... 297
Getting started ... 301

Troubleshooting ... 301
GitHub (Preview) ... 303
Installing Amazon Q Developer app and authorizing access ... 304
Amazon Q Developer agents ... 305
Registering app installation ... 306
Using browser extensions in GitHub ... 306
Using slash commands in GitHub issues and pull requests ... 306
Quickstart ... 307
Developing features and iterating ... 310
Reviewing code ... 313
Increase usage limits and configuration ... 316
Configuring ... 318
Troubleshooting ... 320
Creating project rules ... 322
In chat applications ... 324
Enable Amazon Q chat in your channels ... 324
Ask Amazon Q questions in your channel ... 325
Security ... 326
Data protection ... 327
Data storage ... 328
Data encryption ... 329
Service improvement ... 331
Opt out of data sharing in the IDE and command line ... 332
Cross-region processing ... 340
Identity and access management ... 342
Audience ... 343
Authenticating with identities ... 343
Managing access using policies ... 346
How Amazon Q works with IAM ... 348
Manage access to Amazon Q ... 353
Manage access to Amazon Q Developer for integration ... 394
Amazon Q permissions reference ... 395
AWS managed policies for Amazon Q ... 398
Using service-linked roles ... 404
Compliance validation ... 412
Resilience ... 412

Infrastructure security ... 413
Firewalls, proxies, and data perimeters ... 413
General URLs to allowlist ... 413
Amazon S3 bucket URLs and ARNs to allowlist ... 415
Configuring a corporate proxy in Amazon Q ... 416
VPC endpoints (AWS PrivateLink) ... 423
Considerations for Amazon Q VPC endpoints ... 423
Prerequisites ... 423
Creating an interface VPC endpoint for Amazon Q ... 424
Using an on-premises computer to connect to a Amazon Q endpoint ... 424
Using an in-console coding environment to connect to a Amazon Q endpoint ... 425
Connecting to Amazon Q through AWS PrivateLink from a third-Party IDE on an Amazon EC2 instance ... 425
Monitoring and tracking ... 427
With AWS CloudTrail ... 427
Amazon Q Developer information in CloudTrail ... 428
Understanding Amazon Q Developer log file entries ... 429
With CloudWatch ... 433
Identifying actions by specific users ... 435
Accessing customization-related logs ... 451
Viewing usage metrics (dashboard) ... 452
Dashboard metrics ... 454
Disabling the dashboard ... 458
Troubleshooting the dashboard ... 458
Viewing per-user activity ... 459
User activity report metrics ... 462
Logging users' prompts ... 468
Prompt log examples ... 471
Supported Regions ... 482
Supported Regions (enabled by default) ... 482
Supported opt-in Regions ... 483
Troubleshooting ... 485
Log access and analysis ... 485
Log access overview ... 486
IDE extension logs ... 486
Amazon Q CLI Logs ... 487

Common log patterns and solutions 491
Getting help with log analysis 492
Amazon Q Developer service rename 493
Document history 494

What is Amazon Q Developer?

End of support notice

On April 30, 2027, AWS will discontinue support for Amazon Q Developer IDE plugins. For capabilities similar to Amazon Q Developer IDE plugins, explore Kiro to access the latest models and features, including agentic coding, chat and MCP support. For more information, see Amazon Q Developer IDE plugins end of support.

Note

Powered by Amazon Bedrock: Amazon Q Developer is built on Amazon Bedrock and includes automated abuse detection implemented in Amazon Bedrock to enforce safety, security, and the responsible use of AI.

Amazon Q Developer is a generative artificial intelligence (AI) powered conversational assistant that can help you understand, build, extend, and operate AWS applications. You can ask questions about AWS architecture, your AWS resources, best practices, documentation, support, and more. Amazon Q is constantly updating its capabilities so your questions get the most contextually relevant and actionable answers.

When used in an integrated development environment (IDE), Amazon Q provides software development assistance. Amazon Q can chat about code, provide inline code completions, generate net new code, scan your code for security vulnerabilities, and make code upgrades and improvements, such as language updates, debugging, and optimizations.

Amazon Q is powered by Amazon Bedrock, a fully managed service that makes foundation models (FMs) available through an API. The model that powers Amazon Q has been augmented with high quality AWS content to get you more complete, actionable, and referenced answers to accelerate your building on AWS.

Note

This is the documentation for Amazon Q Developer. If you are looking for documentation for Amazon Q Business, see the Amazon Q Business User Guide.

Get started with Amazon Q Developer

To quickly get started using Amazon Q, you can access it in the following ways:

AWS apps and websites

Add the necessary permissions to your IAM identity, and then choose the Amazon Q icon to start chatting in the AWS Management Console, AWS Documentation website, AWS website, or AWS Console Mobile Application. For more information, see Using Amazon Q Developer on AWS apps and websites.

IDEs

Download the Amazon Q extension and use your AWS Builder ID (no AWS account required) to sign in for free.

 Download Amazon Q in Visual Studio Code

 Download Amazon Q in JetBrains IDEs

 Download Amazon Q in the AWS Toolkit for Visual Studio

- Download Amazon Q in Eclipse IDEs (Preview)

From your IDE, choose the Amazon Q icon to start chatting or initiate a development workflow. For more information, see Installing the Amazon Q Developer extension or plugin in your IDE.

Amazon Q Developer in chat applications

Add the AmazonQDeveloperAccess managed policy to your IAM identity and channel guardrails for Microsoft Teams or Slack applications. For more information, see Chatting with Amazon Q Developer in chat applications.

Amazon Q Developer pricing

Amazon Q Developer is available through a Free tier and the Amazon Q Developer Pro subscription. For more information, see Amazon Q Developer pricing.

Amazon Q Developer IDE plugins end of support

On April 30, 2027, AWS will discontinue support for Amazon Q Developer IDE plugins. For capabilities similar to Amazon Q Developer IDE plugins, explore Kiro to access the latest models and features, including agentic coding, chat and MCP support.

Migrating from Amazon Q Developer to Kiro

If you've been using Amazon Q Developer in your IDE, everything you rely on today (inline suggestions, chat, and code generation) is available in Kiro. For migration steps, see Migrating from Amazon Q Developer to Kiro on the Kiro website.

Getting started with Amazon Q Developer

This section provides comprehensive guidance on getting started with Amazon Q Developer. It covers the different tiers of service (Free and Pro), explains the various sign-in methods available, and walks you through the setup process for personal accounts (Builder IDs) and workforce identities in AWS IAM Identity Center. Whether you're an individual developer or an administrator setting up Amazon Q Developer for your organization, this section will help you choose the right approach for getting started quickly with Amazon Q.

Topics

- Tiers of service for Q Developer – Free and Pro
- Getting started with a personal account (Builder ID)
- Getting started with IAM Identity Center
- Amazon Q Developer Pro subscriptions
- Upgrade to Kiro

Tiers of service for Q Developer – Free and Pro

With Amazon Q Developer, you either use Amazon Q Developer at the Free or Pro tier. Review the following information to understand what's offered at each tier.

Free tier

Amazon Q Developer offers a perpetual Free tier with monthly limits.

For more information about what's available at the Free tier, see the Amazon Q Developer pricing page.

The Free tier is available to users with personal accounts (Builder IDs), users with identities in IAM Identity Center, and users with IAM credentials. Consult the following table to understand the availability of Amazon Q in various interfaces at the Free tier depending on your sign in method.

Interface	Sign in method (Free tier)		
	Personal account (Builder ID)	IAM Identity Center	IAM credentials
AWS Management Console, and AWS apps and websites	⊗ No	⊙ Yes	⊙ Yes
IDE	⊙ Yes	⊗ No	⊗ No
Command line	⊙ Yes	⊗ No	⊗ No

For more information about how content is used to improve the service at the Free tier, see Amazon Q Developer service improvement.

Pro tier

The *Pro tier*, also known as *Amazon Q Developer Pro*, is a paid version of the Amazon Q Developer service that offers higher usage limits than the Free tier. It also gives you access to advanced features.

For information about pricing for the Pro tier, see the Amazon Q Developer pricing page.

The Pro tier is available to users with personal accounts (Builder IDs), and users who have identities in IAM Identity Center. Consult the following table to understand the availability of Amazon Q in various interfaces at the Pro tier depending on your sign in method.

Interface	Sign in method (Pro tier)		
	Personal account (Builder ID)	IAM Identity Center	
AWS Management Console, and AWS apps and websites	⊗ No	⊙ Yes	
IDE	⊙ Yes	⊙ Yes	

	Sign in method (Pro tier)		
Interface	**Personal account (Builder ID)**	**IAM Identity Center**	
Command line	✓ Yes	✓ Yes	

Getting started with a personal account (Builder ID)

If you want to use Amazon Q Developer for personal projects, and you don't need to administer other users, you'll want to get started with a personal account, also known as a Builder ID. A *Builder ID* is a special type of AWS account that gives you the ability to use Amazon Q in your Integrated Development Environment (IDE) and your terminal's command line. Unlike a regular AWS account, a Builder ID is meant to be used by you and you alone, does not allow access to the AWS Management Console, and cannot be assigned IAM roles or permissions.

You can set up a Builder ID for free to start. When you're ready, you can upgrade to the Pro tier by connecting your Builder ID to an AWS account to take advantage of higher usage limits.

For a list of features available at the Pro tier, see the Amazon Q Developer pricing page.

To get started with a personal account (Builder ID) at the Free or Pro tier

- Before you begin: Understand the limitations of personal accounts (Builder IDs)
- Step 1: Sign up
- Step 2: Install Amazon Q
- Step 3: (Optional) Upgrade to the Pro tier

Before you begin: Understand the limitations of personal accounts (Builder IDs)

Before you create a personal account (Builder ID) for use with Amazon Q, understand its limitations.

- With a Builder ID at the Free tier, you will be subject to usage limits. For information about what these limits are, see the pricing page. If you need higher usage limits, subscribe your Builder ID to

the Pro tier using the instructions that follow, or use IAM Identity Center following the guidance in Getting started with IAM Identity Center.

- With a Builder ID at the Pro tier, you'll get *higher usage limits*, but you won't get all Pro tier-only *features*. For a list of Pro tier features that are not available to you, see the footnote in the pricing table of the Amazon Q Developer pricing page. If you need Pro tier features, use IAM Identity Center. For more information, see Getting started with IAM Identity Center.
- With a Builder ID at both the Free and Pro tiers, Amazon Q is only supported in the IDE and at the command line. It is not supported in the AWS Management Console, and on AWS apps and websites. If you need to use Q in the AWS Management Console, and on AWS apps and websites, use IAM Identity Center. For more information, see Getting started with IAM Identity Center.

Step 1: Sign up

Sign up for a free personal account (Builder ID). You can sign up using your email address or an existing Google account. For more information, see Create your AWS Builder ID in the *AWS Sign-In User Guide*.

Step 2: Install Amazon Q

Install Amazon Q in your Integrated Development Environment (IDE) or at the command line, and then authenticate using your personal account (Builder ID). For installation and authentication information, see:

- Installing the Amazon Q Developer extension or plugin in your IDE
- Install the Kiro CLI.

Step 3: (Optional) Upgrade to the Pro tier

Upgrade to the Pro tier to take advantage of increased limits. See Upgrading a personal account (Builder ID).

Getting started with IAM Identity Center

IAM Identity Center is a service that is used by administrators to manage the identities of end users. In the context of Amazon Q Developer, administrators use IAM Identity Center to manage the identities of those whom they plan on subscribing to Amazon Q Developer Pro.

Users who have identities in IAM Identity Center or in a directory or database that is connected to IAM Identity Center are called *IAM Identity Center workforce users* in this guide.

You should get started with IAM Identity Center if:

- **You're an administrator** who wants to set up multiple users with Amazon Q Developer at the Pro tier. By using IAM Identity Center, your users get the full suite of Amazon Q Developer features, plus you get enterprise controls over the Amazon Q Developer subscriptions you administer. For example, you can cancel users' subscriptions, subscribe users in bulk, and track Amazon Q usage on a dashboard.
- **You're an individual user**, and you can't use a personal account (Builder ID) because of its limitations.

Use the following instructions to get started with IAM Identity Center.

To get started with IAM Identity Center

- Step 1: Choose a deployment option
- Step 2: Subscribe workforce users to Amazon Q Developer Pro

Step 1: Choose a deployment option

Before you can subscribe users, you'll need to decide which AWS account or accounts you'll be working in. You'll need to make three key decisions:

- **Decision 1: Where to enable IAM Identity Center** – For more information about IAM Identity Center, see What is IAM Identity Center? in the *AWS IAM Identity Center User Guide*.
- **Decision 2: Where to create the Amazon Q Developer profile** – For more information about the profile, see What is the Amazon Q Developer profile?.
- **Decision 3: Where to subscribe workforce users** – For more information about subscriptions, see Amazon Q Developer Pro subscriptions.

Your specific combination of these three decisions constitutes your *deployment option*.

Deployment options are described in the following table. Pick an option before moving on to Step 2: Subscribe workforce users to Amazon Q Developer Pro.

The table uses the following terms:

- *Standalone account* — An AWS account that is *not* part of an organization managed by AWS Organizations.
- *Management account* — An AWS account that is part of an organization managed by AWS Organizations. It is the ultimate owner of the organization, and is responsible for paying all charges accrued by the accounts in its organization.
- *Member account* — An AWS account, other than the management account, that is part of an organization managed by AWS Organizations.

Deployment option	Description	Advantages	Disadvantages
Deployment option 1 (easiest): Deploy in a standalone account	Use this option if you're an end user and you want to subscribe yourself (and optionally, a small team of users) to quickly evaluate the features of Amazon Q. With this deployment option, in your **standalone** account, you: • enable IAM Identity Center, • create the Amazon Q Developer profile, and • subscribe yourself (and team members). For detailed instructions, see	**Good for demos**. You can try out Pro tier features for yourself without having to do an enterprise-wide implementation. **More features than personal accounts (Builder IDs)**. For more information, see Limitations of Builder IDs.	**Fewer features** Because IAM Identity Center is enabled in a standalone account, it is considered to be an *account instance*, which has fewer features than organization instances[1].

Deployment option	Description	Advantages	Disadvantages
	Subscribe users to Amazon Q Developer Pro in a standalone account.		
Deployment option 2: Deploy in management and member accounts	Use this option if you're an administrator of multiple users. With this deployment option: • In the **management** account, you enable IAM Identity Center. • In a **member** account, you: • create the Amazon Q Developer profile, and • subscribe users. For detailed instructions, see Subscribe users to Amazon Q Developer Pro in a member account.	**More features**. Because IAM Identity Center is installed in a management account, it is considered to be an *organization instance*, which has more features than account instances[2]. **Distributed management**. Subscription management tasks are distributed across member accounts, which is a best practice.	**Complexity**. Requires coordination across accounts by multiple administrators. **Account restrictions**. You can subscribe users in a maximum of 20 accounts per AWS Region, per organization managed by AWS Organizations. If your user base is spread across more than 20 accounts in the same Region under one organization, choose another option.

Deployment option	Description	Advantages	Disadvantages
Deployment option 3: Deploy in a member account only	Use this option if you're an adminstrator of multiple users. With this deployment option, in a **member** account, you: • enable IAM Identity Center, • create the Amazon Q Developer profile, and • subscribe users. For detailed instructions, see Subscribe users to Amazon Q Developer Pro in a member account.	**Quick setup**. Individual member account administrators can deploy without waiting or needing approval for an enterprise-wide implementation. **Flexibility for complex organizations**. Use this option when you don't have a unified identity provider or identity store containing the entire user base that you want to subscribe to the Pro tier.	**Fewer features**. Because IAM Identity Center is enabled in a member account, it is considered to be an *account instance*, which has fewer features than organization instances[1].

Deployment option	Description	Advantages	Disadvantages
Deployment option 4: Deploy in a management account only	Use this option if you're an adminstrator of multiple users. With this deployment option, in the **management** account, you: • enable IAM Identity Center, • create the Amazon Q Developer profile, • subscribe users, and • optionally, share the Amazon Q Developer profile with member accounts. For detailed instructions, see Subscribe users to Amazon Q Developer Pro in a management account.	**More features**. Because IAM Identity Center is installed in a management account, it is considered to be an *organization instance*, which has more features than account instances[2].	**Does not comply with best practices** . Because users are subscribed in the management account, and because of a limitation in Amazon Q Developer where delegated administration is not supported, management account administrators must handle subscription management tasks. You cannot follow the recommended practice of delegating tasks to member accounts.

[1] Account instances support fewer features than organization instances. For example, account instances don't support permission sets, which means that users cannot use their Pro tier subscriptions in the AWS Management Console, and on AWS apps and websites. For a list of the

limitations of account instances, see Account instance considerations in the *AWS IAM Identity Center User Guide*.

[2] Organization instances offer a broader range of features compared to account instances, encompassing all IAM Identity Center capabilities. For a list of features supported by organization instances, see When to use an organization instance in the *AWS IAM Identity Center User Guide*.

Step 2: Subscribe workforce users to Amazon Q Developer Pro

After choosing a deployment option as described in Step 1: Choose a deployment option, you are ready to subscribe workforce users. Subscribing workforce users involves three main steps: Enabling IAM Identity Center, creating the Amazon Q Developer profile, and subscribing users. Instructions on how to complete all steps are included in each of the following sections. You might need to read multiple sections if you're planning on performing steps in multiple accounts.

- Subscribe users to Amazon Q Developer Pro in a standalone account
- Subscribe users to Amazon Q Developer Pro in a management account
- Subscribe users to Amazon Q Developer Pro in a member account

Subscribe users to Amazon Q Developer Pro in a standalone account

A *standalone* account is one that is *not* part of an organization managed by AWS Organizations.

If you are the owner of a standalone AWS account, use the following instructions to subscribe yourself (and a few others) to Amazon Q Developer Pro to evaluate the service's features and functionality.

After completing the steps on this page, read What resources were created? at the end to understand which resources were installed and configured on your behalf when you subscribed. This will help you cleanly remove everything when you're finished testing.

Prerequisites

Before you begin, make sure that:

- You have a **standalone** AWS account.
- You have the minimum permissions required to subscribe users and manage Amazon Q Developer settings. For more information, see Allow administrators to use the Amazon Q console, and Allow administrators to use the Amazon Q Developer console.

- (Optional) You have an account instance of IAM Identity Center set up in your standalone account. This IAM Identity Center contains the identities of the users you want to subscribe to Amazon Q Developer Pro, and must be deployed in a supported AWS Region, as described in IAM Identity Center Regions supported by Amazon Q Developer. If you don't have an IAM Identity Center instance installed, that's ok. One will be installed when you subscribe the first user (yourself). The IAM Identity Center instance will be installed in the AWS Region where you subscribed the first user. For more information about IAM Identity Center, see Organization and account instances of IAM Identity Center in the *AWS IAM Identity Center User Guide*.

> **Note**
>
> The instructions on this page assume you have not already installed an IAM Identity Center instance in your standalone account.

Step 1: Create the Amazon Q Developer Pro profile and subscribe yourself

1. Sign in to the AWS Management Console using your standalone AWS account. Sign in as the root user, or as an IAM user with the permissions described in Prerequisites.
2. Switch to the **Amazon Q Developer** console.
3. Make sure you're in the AWS Region where you want to create the Amazon Q Developer profile and where you want to store user data. For supported Regions, see Supported Regions for the Q Developer console and Q Developer profile.
4. Choose the **Get started** button.

   > **Note**
   >
   > If you see a **Settings** button instead of **Get started** button, it means that you've already run through the 'Get started' workflow and can skip to Step 2: Subscribe team members.

   A **Create your user** dialog box appears.
5. Enter your information. The email address can be the same or different from the one you used to sign up for your AWS account.

   Choose **Continue**.

The **Create Amazon Q Developer profile** dialog box appears.

6. Review the contents of the dialog box and provide a name for your profile in **Profile name**. For help with cross-region inferencing, see Cross-region processing in Amazon Q Developer. For help with disabling dashboard metrics, see Disabling the Amazon Q Developer dashboard.

   Choose **Create application**.

   The Amazon Q Developer profile and managed application are created, and your subscription is created.

7. (Optional) Verify that your subscription was created:

   1. In the Amazon Q Developer console, in the navigation pane, choose **Subscriptions**.
   2. In the main pane, choose the **Users** tab.

   Your subscription should appear in the list in the **Pending** state. If not, refresh your browser tab.

   > **Note**
   >
   > Your subscription will change to the **Active** state after your first use of Amazon Q Developer features.

   Now that you are subscribed, you must activate your subscription. You can do this now, or after you've subscribed team members, as described in the next section. To activate your subscription, check your inbox for emails titled **Invitation to join AWS IAM Identity Center** and **Activate Your Amazon Q Developer Pro Subscription**. Follow the instructions in these emails to activate your Amazon Q Developer Pro subscription and set up Amazon Q Developer Pro in your IDE. You should receive these emails within 24 hours.

Step 2: Subscribe team members

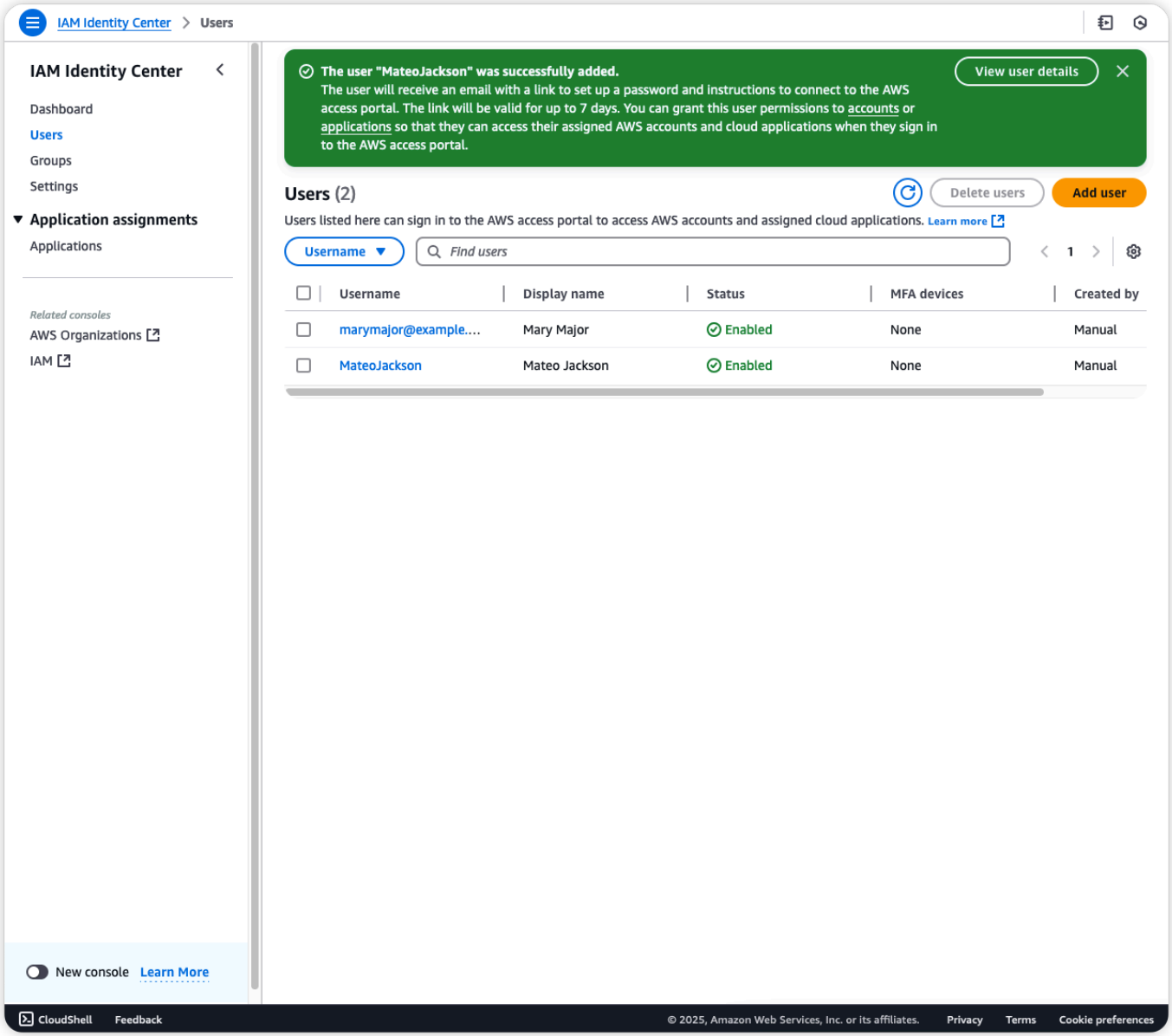
You might want to subscribe other team members so that they can try out Amazon Q Developer Pro with you. To subscribe them, use the following instructions.

To add team members

1. Switch to the IAM Identity Center console (not the IAM console).

   > **Note**
   >
   > IAM Identity Center was set up on your behalf when you subscribed yourself. For more information about the IAM Identity Center that was set up, see What resources were created?.

2. Add users and groups. For instructions, see Add users to your IAM Identity Center directory in the *AWS IAM Identity Center User Guide*.



3. Go to the next procedure to subscribe team members.

To subscribe team members

1. Return to the Amazon Q Developer console.
2. In the navigation pane, choose **Subscriptions**, and then choose **Subscribe**.

   The **Assign users and groups** dialog box appears.
3. Start typing the name of a team member or group that you added. The name should auto-populate.

> **Note**
>
> The dialog box only matches on user names or group names. It does not match on email addresses.

4. Choose **Assign**.
5. Have users check their email. They should receive an email titled **Activate Your Amazon Q Developer Pro Subscription** within 24 hours. In this email, users will find guidance on how to begin using their Amazon Q Developer Pro license in the AWS Management Console and their Integrated Development Environment (IDE). The email includes users' unique Start URL and AWS Region for authentication, and provides quickstart steps for using Amazon Q Developer in their IDE. This email streamlines the onboarding process and saves you valuable time by eliminating the need for you to manually notify each new user.

What resources were created?

When you subscribed yourself (and optionally, team members), Amazon Q created the following AWS resources on your behalf:

- **An account instance of IAM Identity Center**. For more information about account instances of IAM Identity Center, see Account instances of IAM Identity Center in the *AWS IAM Identity Center User Guide*.

> **Note**
>
> Account instances of IAM Identity Center have limitations. For example, account instances don't support console access. (Users can still use Amazon Q in the console, it's just that they'll be subject to the Free tier monthly limits.) If you want to use Amazon Q Developer Pro in the console and other AWS websites, you must be a user in an *organization instance* of IAM Identity Center, in a management account. For more information, see Subscribe users to Amazon Q Developer Pro in a management account.

> **Note**
>
> You can't convert or merge an account instance of IAM Identity Center into an organization instance.

- **The first user**, in IAM Identity Center. You might have manually added team members too.
- **Pro tier subscriptions** for the first user and team members, in Amazon Q Developer.
- **An Amazon Q Developer profile**, in the Amazon Q Developer console, under **Settings**.
- **A managed application** called **QDevProfile-*region***, in the IAM Identity Center that is set up in your standalone account. The application is associated with the Amazon Q Developer profile. Like the Amazon Q Developer profile, the application is created once and shared between all Amazon Q subscribers in your standalone account.

Subscribe users to Amazon Q Developer Pro in a management account

A *management account* is an AWS account that is part of an organization managed by AWS Organizations. It is the ultimate owner of the organization, and is responsible for paying all charges accrued by the accounts in its organization.

If you are the owner of a management account, use the following instructions to subscribe users to Amazon Q Developer Pro in your account.

> **Note**
>
> If possible, subscribe users in member accounts instead of your management account. For more information, see Step 1: Choose a deployment option.

For more information about organizations and management accounts, see Terminology and concepts for AWS Organizations in the *AWS Organizations User Guide*.

Prerequisites

Before you begin, make sure that:

- You have a **management** AWS account.

- You have the minimum permissions required to subscribe users and manage Amazon Q Developer settings. For more information, see Allow administrators to use the Amazon Q console, and Allow administrators to use the Amazon Q Developer console.
- You have an organization instance of IAM Identity Center set up in your management account. This IAM Identity Center contains the identities of the users you want to subscribe to Amazon Q Developer Pro, and must be deployed in a supported AWS Region, as described in IAM Identity Center Regions supported by Amazon Q Developer. For more information about IAM Identity Center, see Organization instances of IAM Identity Center in the *AWS IAM Identity Center User Guide*.

Step 1: Create the Amazon Q Developer profile

1. Sign in to the AWS Management Console using your AWS management account.
2. Switch to the **Amazon Q Developer** console.
3. Make sure you're in the AWS Region where you want to create the Amazon Q Developer profile and where you want to store user data. For supported Regions, see Supported Regions for the Q Developer console and Q Developer profile.
4. Choose **Get started**.

   The **Create Amazon Q Developer profile** dialog box appears.
5. Review the contents of the dialog box and provide a name for your profile in **Profile name**. For help with:

   - Cross-region inferencing, see Cross-region processing in Amazon Q Developer.
   - The **Share Amazon Q Developer settings with member account** check box, see Enabling profile sharing in Amazon Q Developer and Step 1: Choose a deployment option.
   - Disabling dashboard metrics, see Disabling the Amazon Q Developer dashboard.

   Choose **Create application**.

   The Amazon Q Developer profile and managed application are created.

Step 2: Subscribe users

1. In the Amazon Q Developer console, from the navigation pane, choose **Subscriptions**.
2. Choose **Subscribe**.

The **Assign users and groups** dialog box appears.

3. Start typing the group or user you want to subscribe. The group or user will auto-populate with the ones available in the IAM Identity Center set up in your management account.

> **Note**
>
> The dialog box only matches on user names or group names. It does not match on email addresses.

4. Choose **Assign**.
5. Have users check their email. They should receive an email titled **Activate Your Amazon Q Developer Pro Subscription** within 24 hours with instructions on how to begin using their Amazon Q Developer Pro license.

Step 3: Enable identity-enhanced console sessions

If you want to allow users to use their Amazon Q Developer Pro subscription in the AWS Management Console, and on AWS apps and websites, enable identity-enhanced console sessions. For more information, see Enabling identity-enhanced console sessions in the *AWS IAM Identity Center User Guide*.

> **Note**
>
> If you don't enable identity-enhanced console sessions, users can still use Amazon Q in the AWS Management Console, and on AWS apps and websites, but they'll be limited to the Free tier.

What resources were created?

When you created the Amazon Q Developer profile and subscribed users in your management account, Amazon Q created the following resources on your behalf:

- **Pro tier subscriptions** for users, in Amazon Q Developer.
- **An Amazon Q Developer profile**, in the Amazon Q Developer console, under **Settings**.
- **A managed application** called **QDevProfile-*region***, in the IAM Identity Center that is set up in your management account. The application is associated with the Amazon Q Developer profile.

Like the Amazon Q Developer profile, the application is created once and shared between all Amazon Q subscribers in your management account.

> **Note**
>
> Amazon Q can create the **QDevProfile-*region*** managed application in a maximum of 20 AWS accounts per AWS Region within an organization.

Subscribe users to Amazon Q Developer Pro in a member account

A *member account* is an AWS account, other than the management account, that is part of an organization managed by AWS Organizations.

If you are the owner of a member account, use the following instructions to subscribe users to Amazon Q Developer Pro in your account.

Not sure whether to subscribe users in a member or management account? See Step 1: Choose a deployment option for help.

For more information about organizations, member accounts, and management accounts, see Terminology and concepts for AWS Organizations in the *AWS Organizations User Guide*.

Prerequisites

Before you begin, make sure that:

- You have a **member** AWS account.
- You have the minimum permissions required to subscribe users and manage Amazon Q Developer settings. For more information, see Allow administrators to use the Amazon Q console, and Allow administrators to use the Amazon Q Developer console.
- (Optional) You have an organization instance of IAM Identity Center set up in the *management account* or an account instance of IAM Identity Center set up in your *member account*. This IAM Identity Center intance contains the identities of the users you want to subscribe to Amazon Q Developer Pro, and must be deployed in a supported AWS Region, as described in IAM Identity Center Regions supported by Amazon Q Developer. If you don't have an IAM Identity Center instance installed, that's ok. One will be installed in your member account when you subscribe the first user. The IAM Identity Center instance will be installed in the AWS Regionn where you

subscribed the first user. For more information about IAM Identity Center, see Organization and account instances of IAM Identity Center in the *AWS IAM Identity Center User Guide*.

Step 1: Create the Amazon Q Developer Pro profile and subscribe the first user

1. Sign in to the AWS Management Console using your member AWS account.
2. Switch to the **Amazon Q Developer** console.
3. Make sure you're in the AWS Region where you want to create the Amazon Q Developer profile and where you want to store user data. For supported Regions, see Supported Regions for the Q Developer console and Q Developer profile.
4. Choose the **Get started** button.

   > **Note**
   >
   > If you see a **Settings** button instead of **Get started** button, it means that you've already run through the 'Get started' workflow and can skip to Step 2: Subscribe other users.

5. Follow the on-screen prompts to subscribe your first user.

   - If the first user's email address matches one in an existing IAM Identity Center in either your member account or a management account, then Amazon Q connects to that IAM Identity Center.
   - If the first user's email address doesn't match one in an existing IAM Identity Center, then Amazon Q creates an IAM Identity Center account instance in your member account, and adds the first user to it. Note that:
     - Amazon Q only creates an IAM Identity Center account instance if there is no IAM Identity Center already in your member account.
     - If there is an IAM Identity Center account instance in your member account, but the user is not in it, then Amazon Q creates the user in the existing IAM Identity Center.

   The **Create Amazon Q Developer profile** dialog box appears.

6. Review the contents of the dialog box and provide a name for your profile in **Profile name**. For help with cross-region inferencing, see Cross-region processing in Amazon Q Developer. For help with disabling dashboard metrics, see Disabling the Amazon Q Developer dashboard.

Choose **Create application**.

The Amazon Q Developer profile and managed application are created, and the first user is subscribed.

7. (Optional) Verify that the first user's subscription was created:

   1. In the Amazon Q Developer console, in the navigation pane, choose **Subscriptions**.
   2. In the main pane, choose the **Users** tab.

   The subscription of the first user should appear in the list in the **Pending** state. If not, refresh your browser tab.

   > **Note**
   >
   > The subscription will change to the **Active** state after the user's first use of Amazon Q Developer features.

8. Have the first user check their email. They should receive an email titled **Activate Your Amazon Q Developer Pro Subscription** within 24 hours. In this email, users will find guidance on how to begin using their Amazon Q Developer Pro license in the AWS Management Console and their Integrated Development Environment (IDE). The email includes users' unique Start URL and AWS Region for authentication, and provides quickstart steps for using Amazon Q Developer in their IDE. This email streamlines the onboarding process and saves you valuable time by eliminating the need for you to manually notify each new user.

Step 2: Subscribe other users

To subscribe other users, add them to your IAM Identity Center instance if they're not already there, and then subscribe them to Amazon Q Developer Pro by choosing **Subscribe** in the Amazon Q Developer console.

For instructions on adding users to IAM Identity Center, see Add users to your IAM Identity Center directory in the *AWS IAM Identity Center User Guide*.

Step 3: Enable identity-enhanced console sessions

If you want to allow users to use their Amazon Q Developer Pro subscription in the AWS Management Console, and on AWS apps and websites, enable identity-enhanced console sessions.

For more information, see Enabling identity-enhanced console sessions in the *AWS IAM Identity Center User Guide*.

If you don't enable identity-enhanced console sessions, users can still use Amazon Q in the AWS Management Console, and on AWS apps and websites, but they'll be limited to the Free tier.

> **Note**
>
> The ability to enable identity-enhanced console sessions—and therefore the ability to use Amazon Q Developer Pro subscriptions in the AWS Management Console, and on AWS apps and websites—is only supported with organization instances of IAM Identity Center, not account instances.

What resources were created?

When you subscribed users in your member account, Amazon Q created the following AWS resources on your behalf:

- **An account instance of IAM Identity Center**. This instance is only created if the first user you subscribed wasn't found in an existing IAM Identity Center in the member account or management account. For more information about account instances of IAM Identity Center, see Account instances of IAM Identity Center in the *AWS IAM Identity Center User Guide*.

  > **Note**
  >
  > Account instances of IAM Identity Center have limitations. For example, account instances don't support console access. (Users can still use Amazon Q in the console, it's just that they'll be subject to the Free tier monthly limits.) If you want your users to be able to use Amazon Q Developer Pro in the console and other AWS websites, they must exist in an *organization instance* of IAM Identity Center, in a management account. For more information, see Subscribe users to Amazon Q Developer Pro in a management account.

  > **Note**
  >
  > You can't convert or merge an account instance of IAM Identity Center into an organization instance.

- **The first user**, in IAM Identity Center. (You might have added team members too.)
- **Pro tier subscriptions** for the first user and other users, in Amazon Q Developer.
- **An Amazon Q Developer profile**, in the Amazon Q Developer console, under **Settings**.
- **A managed application** called **QDevProfile-*region***, in IAM Identity Center. The application is associated with the Amazon Q Developer profile. Like the Amazon Q Developer profile, the application is created once and shared between all Amazon Q Developer Pro subscribers in your member account.

> **Note**
>
> Amazon Q can create the **QDevProfile-*region*** managed application in a maximum of 20 AWS accounts per AWS Region within an organization.

Amazon Q Developer Pro subscriptions

An *Amazon Q Developer Pro subscription*, also called a *Pro tier subscription*, is a paid version of the Amazon Q Developer service. It offers enhanced AI development capabilities designed for professional developers and teams who require advanced features and higher usage limits beyond the Free tier offering. For more information about pricing tiers and feature availability, see the Amazon Q Developer pricing page.

This chapter covers essential information for managing Amazon Q Developer Pro subscriptions for both personal accounts (Builder IDs) and workforce users (IAM Identity Center). It describes Region availability and support, billing details, and subscription status information. It also provides step-by-step guidance for key tasks such as unsubscribing, and upgrading to the Pro tier.

Topics

- Amazon Q Developer Pro Region support
- Amazon Q Developer Pro subscription billing
- Amazon Q Developer subscription statuses
- Finding the Start URL for use with Amazon Q Developer
- Managing the encryption method in Amazon Q Developer
- What is the Amazon Q Developer profile?
- Troubleshooting Amazon Q Developer Pro subscriptions

- Viewing an aggregated list of Amazon Q Developer subscriptions
- Unsubscribing from Amazon Q Developer Pro
- Upgrading to Amazon Q Developer Pro

Amazon Q Developer Pro Region support

Region information for the Pro tier varies depending on whether you're an end-user with a personal account (Builder ID), or you're an administrator of IAM Identity Center workforce users.

Personal account (Builder ID) users

If you're the owner of a personal account (Builder ID), your Pro tier subscription is supported in the following Region:

- US East (N. Virginia)

IAM Identity Center workforce users

Read this section if you're an administrator of IAM Identity Center workforce users.

Topics

- IAM Identity Center Regions supported by Amazon Q Developer
- Supported Regions for the Q Developer console and Q Developer profile
- Subscribing users to Amazon Q Developer Pro across AWS Regions

IAM Identity Center Regions supported by Amazon Q Developer

The workforce users that you want to subscribe to Amazon Q Developer Pro must have identities in an IAM Identity Center instance (or a connected identity provider) in one of the Regions listed on the Supported Regions page, except for opt-in Regions. If your users have identities in an IAM Identity Center instance in an opt-in Region, they can't be subscribed, which means they will only have access to the Free tier in the AWS Management Console, and on AWS apps and websites, and they won’t have access to Amazon Q in the IDE or at the command line.

Amazon Q stores the subscriptions of IAM Identity Center workforce users in the same Region as your IAM Identity Center instance.

Regardless of the IAM Identity Center Region, data is stored in the Region where you create the Amazon Q Developer profile.

For more information about Amazon Q Developer profiles, see What is the Amazon Q Developer profile?.

For more information about data protection, see Data protection in Amazon Q Developer.

Supported Regions for the Q Developer console and Q Developer profile

The **Amazon Q Developer console** and **Amazon Q Developer profile** are supported in the following Regions:

- US East (N. Virginia)
- Europe (Frankfurt)

For more information about the Amazon Q Developer profile, see What is the Amazon Q Developer profile?.

> **Note**
>
> The following features aren't supported for Q Developer profiles created in the Europe (Frankfurt) Region:
>
> - Chatting with Support
> - Troubleshooting resources with Amazon Q
> - .NET transformations in the IDE
> - Amazon Q in chat applications (for Slack)
> - Amazon Q in the AWS Console Mobile Application
> - GitLab Duo with Amazon Q
> - Amazon Q for GitHub

Subscribing users to Amazon Q Developer Pro across AWS Regions

When you subscribe an IAM Identity Center workforce user to Amazon Q Developer Pro, you might need to work in two different AWS Regions:

- One Region for the IAM Identity Center instance (where user identities are managed, and where subscriptions are stored)
- Another Region for the Amazon Q Developer console (where the Amazon Q Developer profile, customizations, and subscriptions are managed)

The Regions are not always the same because the Amazon Q Developer console is supported in fewer Regions than IAM Identity Center.

In a scenario where your IAM Identity Center is in a different Region from your Amazon Q Developer console, use the guidance in the following example to subscribe users.

Example subscription process in a multi-Region scenario

Let's walk through subscribing a user where:

- The IAM Identity Center instance is in **US West (N. California)**.
- The Amazon Q Developer console is in **US East (N. Virginia)**. This is the closest Region to the IAM Identity Center instance that the Amazon Q Developer console supports.

To subscribe the user

1. Add the user in your IAM Identity Center instance in **US West (N. California)**.
2. Switch to the Amazon Q Developer console in the **US East (N. Virginia)**.
3. Subscribe the user through the Amazon Q Developer console in **US East (N. Virginia)**.

Upon being subscribed:

- The user's subscription is created in **US West (N. California)**.
- The user's subscription is associated with their user entry in **US West (N. California)**.
- The user's subscription is associated with the Amazon Q Developer profile in **US East (N. Virginia)**.

Additionally, any data that Amazon Q Developer needs to store on the user's behalf will be stored in **US East (N. Virginia)**. For more information about data storage and security, see Encryption at rest.

For detailed instructions on subscribing users, see Getting started with IAM Identity Center.

Amazon Q Developer Pro subscription billing

Billing information for the Pro tier varies depending on whether you're an end-user with a personal account (Builder ID), or you're an administrator of IAM Identity Center workforce users.

Personal account (Builder ID) users

If you subscribed to the Pro tier with a personal account (Builder ID), you will be billed monthly. The AWS account that is linked to your Builder ID gets the bill.

The very first month you are subscribed, you are charged a prorated fee. For example, if you subscribe April 15th, you'll be charged half the subscription fee. After that, you'll be charged the full amount.

If you unsubscribe, billing stops at the end of the billing cycle. For more information, see Unsubscribing from Amazon Q Developer Pro.

IAM Identity Center workforce users

If you're an administrator who has subscribed a set of IAM Identity Center workforce users to the Pro tier, you will be billed monthly for each user that you subscribe. For more information, see Amazon Q Developer pricing.

If your business has AWS Organizations set up, billing for Amazon Q Developer Pro usage is per AWS organization. The management account gets the bill. If the same user is subscribed to Amazon Q Developer in multiple accounts within the same organization, you will not be double-billed.

If you don't have AWS Organizations set up, the AWS account under which your users are subscribed gets the bill.

You can view your bill in the Billing and Cost Management console. The Amazon Q expenses are listed on the **Charges by service** tab, under **Q**. For more information about the Billing and Cost Management console, see What is AWS Billing and Cost Management? in the *AWS Billing User Guide*.

You can identify the cost of Amazon Q subscriptions for specific users with resource IDs through AWS Billing and Cost Management. To do so, in the Billing and Cost Management console under Data Exports, create either a standard data export or a legacy CUR export with the **Include resource IDs** option selected. To learn more, refer to Creating data exports in the *AWS Data Exports User Guide*.

If you unsubscribe users, billing stops at the end of the billing cycle. For more information, see Unsubscribing from Amazon Q Developer Pro.

Amazon Q Developer subscription statuses

Subscription status information varies depending on whether you're an end-user with a personal account (Builder ID), or you're an administrator of IAM Identity Center workforce users.

Personal account (Builder ID) users

If you are a user with a personal account (Builder ID), you can view the status of your Amazon Q Developer Pro subscription on the **Subscriptions** page of the Amazon Q Developer console.

The possible statuses of your subscription are:

- **Active** – You have activated your subscription by using Amazon Q Developer features. You are being charged for your subscription.
- **Canceled** – You canceled your subscription by unsubscribing from the Pro tier. You can no longer access Amazon Q Developer features and limits. For more information, see Unsubscribing from Amazon Q Developer Pro.

IAM Identity Center workforce users

If you're an administrator who has subscribed a set of IAM Identity Center workforce users to the Pro tier, you can view the status of your users' subscriptions on the **Subscriptions** page of the Amazon Q Developer console.

The statuses will be slightly different depending on whether you're looking at the **Groups** tab or the **Users** tab.

The statuses on the **Groups** tab are:

- **Subscribed** – The group is subscribed to Amazon Q Developer Pro. You will be charged for active user subscriptions in the group.
- **Canceled** – The group was canceled (unsubscribed) by an administrator. Users in the group can no longer access Amazon Q Developer Pro features. For more information, see Unsubscribing from Amazon Q Developer Pro.

The statuses on the **Users** tab are:

- **Active** – The user has activated their subscription by using Amazon Q Developer features. You are being charged for this subscription.
- **Pending** – The user is subscribed but has not activated their subscription. You are not being charged for this subscription.
- **Canceled** – The user's subscription was canceled (unsubscribed) by an administrator, and the user can no longer access Amazon Q Developer features. For more information, see Unsubscribing from Amazon Q Developer Pro.

> **Note**
>
> The **Users** tab of the Amazon Q Developer console does *not* show users who are subscribed as part of a group. To see these users, navigate to the Amazon Q console's (*not* the Amazon Q Developer console's) **Subscriptions** page. On this page, group-subscribed users will appear with a status of **Unavailable**. To see their actual status, choose a user from the table, and look for their status under **User associations**.

Finding the Start URL for use with Amazon Q Developer

> **Note**
>
> This section does not apply to personal accounts (Builder IDs).

If you're an administrator who has subscribed a set of IAM Identity Center workforce users to the Pro tier, those users will need to sign in to Amazon Q in their IDE or at the command line using your IAM Identity Center's Start URL and Region. If you need to provide this URL to your users, you can find it in the Amazon Q Developer console, on the **Settings** page. The start URL is specific to your organization.

To find the Start URL

1. Sign in to the AWS Management Console.
2. Switch to the Amazon Q Developer console.

   To use the Amazon Q Developer console, you must have the permissions defined in Allow administrators to use the Amazon Q Developer console .

3. Choose **Settings**.

   The start URL is shown in **Start URL** near the top of the page. The start URL is specific to your organization.

Managing the encryption method in Amazon Q Developer

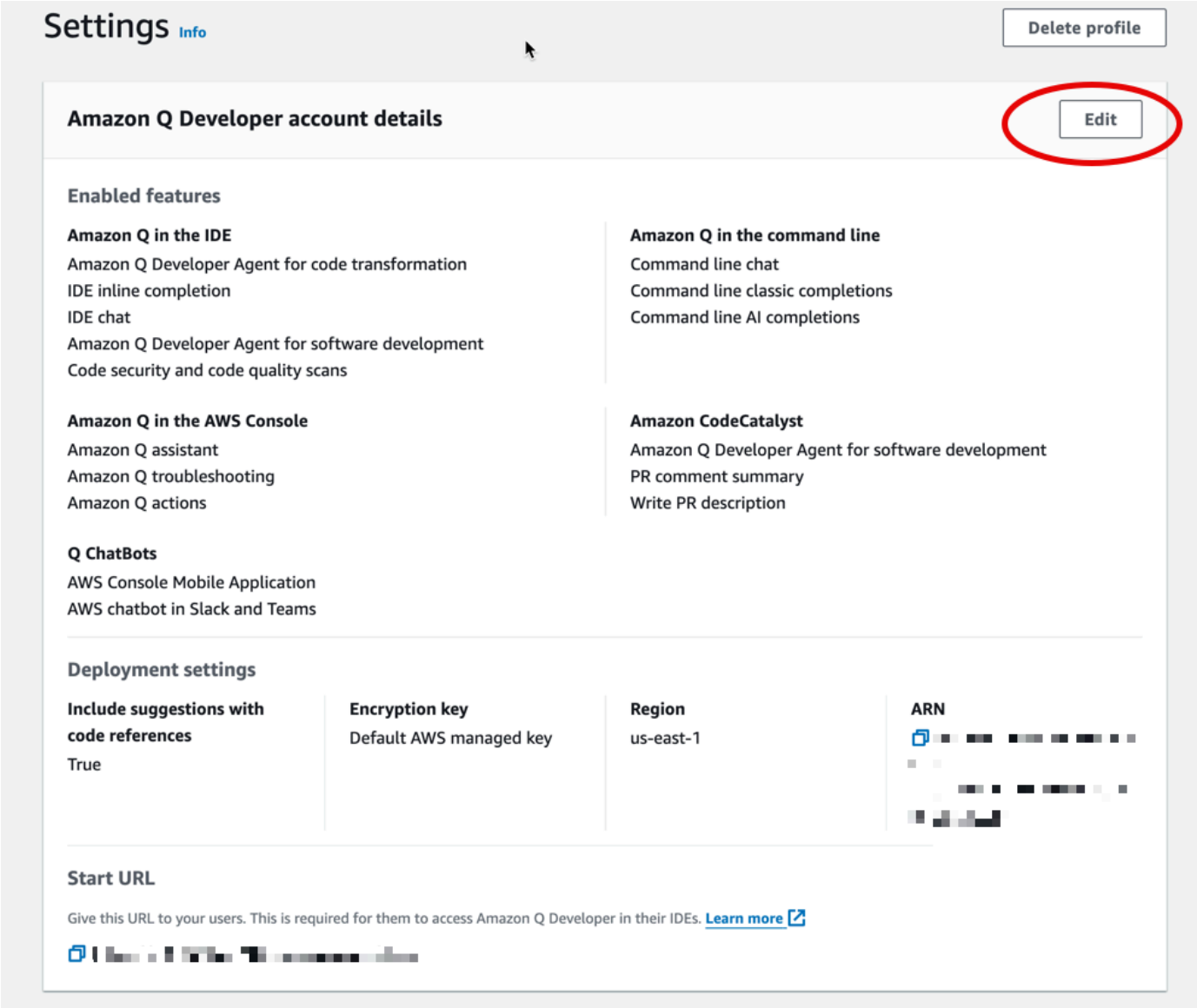
> **Note**
>
> This section does not apply to personal accounts (Builder IDs).

By default, Amazon Q Developer uses an AWS managed key for encryption. For some features, you can set up a customer managed key to encrypt data. For a list of features that support encryption with customer managed keys, see Data encryption.

To set the key used for encryption, complete the following procedure.

1. Sign in to the AWS Management Console.
2. Switch to the Amazon Q Developer console.

   To use the Amazon Q Developer console, you must have the permissions defined in Allow administrators to use the Amazon Q Developer console .
3. Choose **Settings**.
4. Choose **Edit** in the Amazon Q Developer account details panel.



5. On the **Edit details** page, expand the **Encryption key - optional** section.
6. To use a customer managed key for encryption, select **Customize encryption settings (advanced)**.
7. In the search bar that appears, search for the name of the key you want to use for encryption or enter the key ARN.

   If you haven't created a key yet, choose **Create an AWS KMS key**, and then return to this page to add your key.
8. To disable encryption with your customer managed key and revert to an AWS managed key for encryption, deselect **Customize encryption settings (advanced)**.

What is the Amazon Q Developer profile?

> **Note**
>
> This section does not apply to personal accounts (Builder IDs).

An *Amazon Q Developer profile*, also known as a *settings profile*, a collection of Amazon Q Developer settings associated with a set of IAM Identity Center workforce user Pro tier subscriptions. The profile is also associated with an Amazon Q Developer managed application which ties the user's identity in IAM Identity Center to their subscription in Amazon Q Developer.

The very first time you subscribe users, you will be prompted to create this profile. Creating the profile causes several pages to appear in the side navigation of the Amazon Q Developer console where you can configure Amazon Q Developer Pro features. All subscriptions that you add to your account (during the initial subscription process, and later) will be associated with this profile.

Other characteristics of the Amazon Q Developer profile are:

- The profile is mandatory for use with IAM Identity Center workforce users. You cannot subscribe workforce users without it. It must be created in the AWS account where you want to subscribe users to Amazon Q Developer Pro.
- The profile can be created once per supported AWS Region, per AWS account. For a list of AWS Regions supported by the Amazon Q Developer profile, see Supported Regions for the Q Developer console and Q Developer profile.

Creating the Amazon Q Developer profile

> **Note**
>
> This section does not apply to personal accounts (Builder IDs).

Creating the Amazon Q Developer profile unlocks settings in the Amazon Q Developer console and is a prerequisite for subscribing users to Amazon Q Developer Pro. For more information about the profile, see What is the Amazon Q Developer profile?.

To create the Amazon Q Developer profile

- Subscribe users according to the instructions in Getting started with IAM Identity Center. During the subscription process, you will be asked to create the profile.

Deleting the Amazon Q Developer profile

> **Note**
>
> This section does not apply to personal accounts (Builder IDs).

You might want to delete the Amazon Q Developer profile to quickly cancel all subscriptions and remove all Q Developer configurations from your AWS account. When you delete the Amazon Q Developer profile, the following occurs:

- All subscriptions associated with the profile marked as **Canceled**, and users will no longer be able to access Amazon Q Developer features. The final monthly subscription fee is charged at the end of the current billing cycle for all users who had active subscriptions. You'll be charged for the full month; the fee won't be prorated. Subscriptions will continue to be visible in the Amazon Q Developer console until the end of the month, at which time they will be removed from view.
- All settings and options in the Amazon Q Developer console that became available as a result of creating the profile will no longer be visible or take effect. For example, the Q Developer dashboard will no longer be visible, customizations will no longer be configurable or applied, and user activity reports will no longer be configurable or generated.
- The managed application (called **QDevProfile-*region***) will be removed from the IAM Identity Center instance that is connected to Amazon Q Developer. (This IAM Identity Center instance might be a different account from the one where the profile is being deleted, depending on your deployment option.)

> **Note**
>
> If you accidentally delete the profile:
>
> - You'll have to recreate it and then resubscribe users. You'll also have to reset any settings you configured previously through the Amazon Q Developer console. Your IAM Identity Center instance will remain, so there is no need to recreate user identities.

- You won't be able to see historical data in the Amazon Q Developer dashboard after recreating the profile; you can only see data starting from the date of the new profile creation onward.

Use the following instructions to delete the Amazon Q Developer profile.

Before you begin

- Remove any customizations you might have created to allow for the successful deletion of the profile.

To delete the Amazon Q Developer profile

1. Sign in to the AWS Management Console.
2. Switch to the Amazon Q Developer console.

   To use the Amazon Q Developer console, you must have the permissions defined in Allow administrators to use the Amazon Q Developer console.
3. Choose **Settings**.
4. Choose **Delete profile**.

Enabling profile sharing in Amazon Q Developer

Note

This section does not apply to personal accounts (Builder IDs).

If you are a management account administrator within an organization managed by AWS Organizations, you can enable the *profile sharing* feature. When profile sharing is enabled, the Amazon Q Developer profile that has been created in a management account will be shared with member accounts. Sharing the profile has one benefit: it allows IAM Identity Center workforce users who are subscribed to Amazon Q Developer Pro in a *management* account to use their Amazon Q Developer Pro subscription in the AWS Management Console, and on AWS apps and websites while signed in to a *member* account. When profile sharing is disabled, these users can still

use Amazon Q in the AWS Management Console, and on AWS apps and websites, while signed in to a member account, but they'll be subject to Free tier limits and features.

Enabling profile sharing has no effect on users' ability to use Amazon Q in the integrated development environment (IDE) or on the command line.

Use the following instructions to enable profile sharing.

Prerequisites

Before you begin, make sure that:

- You are an administrator of an AWS *management* account.
- You have an IAM Identity Center instance set up in your management account and connected to Amazon Q. To check, sign in to your management account, go to the Amazon Q Developer console, choose **Settings**, and make sure that a **Start URL** appears.
- You have subscribed users to Amazon Q Developer Pro in your management account.
- You have the minimum permissions required to access the Amazon Q Developer console. For more information, see Allow administrators to use the Amazon Q Developer console.

To enable profile sharing

1. Sign in to the AWS Management Console using your AWS management account.
2. Switch to the Amazon Q Developer console.
3. Choose **Settings**.
4. Scroll to the **Member account settings** section and choose **Edit**.
5. Enable **Q Developer managed application and settings profile**.
6. Choose **Save**.

   Users who are subscribed to Amazon Q Developer Pro in your management account will now be able to use their Amazon Q Developer Pro subscription in the AWS Management Console, and on AWS apps and websites while signed in to a member account.

Troubleshooting Amazon Q Developer Pro subscriptions

If you are having trouble with Amazon Q Developer Pro subscriptions, review the following issues to understand how to resolve them.

Topics

- Unable to subscribe users
- Users not receiving activation emails
- Users unable to use their subscription on AWS websites
- Users unable to use their subscription in the IDE
- Can't see subscribed users

Unable to subscribe users

Problem: Unable to subscribe users to Amazon Q Developer Pro

Solutions:

- Verify that you have the minimum permissions required to subscribe users. For more information, see Allow administrators to use the Amazon Q Developer console. After you obtain the necessary permissions, reload the console page to access Amazon Q.
- Check that you're in the Amazon Q Developer console (not the Amazon Q console).
- Check that you're in a supported AWS Region for Amazon Q Developer. For more information, see Supported Regions for the Q Developer console and Q Developer profile.
- Ensure you're following the correct workflow for the type of account you have. For more information, see Getting started with a personal account (Builder ID), Subscribe users to Amazon Q Developer Pro in a standalone account, Subscribe users to Amazon Q Developer Pro in a management account, or Subscribe users to Amazon Q Developer Pro in a member account.

Users not receiving activation emails

Problem: Users are not receiving activation emails

Solutions:

- Verify the email address is correct in AWS IAM Identity Center.
- Have users check their spam or junk folders for emails titled **Activate Your Amazon Q Developer Pro Subscription**.
- Allow up to 24 hours for activation emails to be delivered.
- Verify the user was properly added to IAM Identity Center. For more information, see Add users to your IAM Identity Center directory in the *AWS IAM Identity Center User Guide*.

Users unable to use their subscription on AWS websites

Problem: Users are unable to use their subscriptions on AWS websites

When attempting to use Amazon Q in the AWS Management Console, and on AWS apps and websites, users see the following message in their browser:

```
Your account has not been configured to use an Amazon Q subscription.
You currently have access to the Free tier of Amazon Q. Contact your AWS
administrator to configure your subscription.
```

Solutions:

- Verify that identity-enhanced console sessions are enabled (only available with organization instances of IAM Identity Center). For information about how to enable identity-enhanced console sessions, see Enabling identity-enhanced console sessions.
- Check that the user has an active Amazon Q Developer Pro subscription. For more information, see Amazon Q Developer subscription statuses.
- Verify that you're not using an account instance of IAM Identity Center. Account instances don't support console access. For more information, see Account instance considerations in the *AWS IAM Identity Center User Guide*.

  Users with identities in an account instance of IAM Identity Center can still use Amazon Q in the console but will be limited to Free tier.
- If the user recently switched from the Free tier to the Pro tier, have them sign out of the AWS Management Console or another AWS website and sign back in again.
- If you subscribed the user as part of a group, allow up to 24 hours for their subscription to be activated. There might be a delay between the time the user is added to the group and the time their subscription becomes active.
- Verify that the user's access to the Amazon Q Developer Pro managed application was not revoked or that the managed application was not deleted. Restore access to the managed application if needed.
- If users don't have an active subscription, try getting them to refresh their page so that they can use the Free tier.

Users unable to use their subscription in the IDE

Problem: IAM Identity Center workforce users are unable to use their Pro subscriptions in the IDE

Solutions:

- Check that the user has an active Amazon Q Developer Pro subscription. For more information, see Amazon Q Developer subscription statuses.
- If the user recently switched from the Free tier to the Pro tier, have them sign out of Amazon Q in the IDE and sign back in again.
- If you subscribed the user as part of a group, allow up to 24 hours for their subscription to be activated. There might be a delay between the time the user is added to the group and the time their subscription becomes active.
- Verify that the user's access to the Amazon Q Developer Pro managed application was not revoked or that the managed application was not deleted. Restore access to the managed application if needed.
- Have users sign in with a Builder ID to use the Free tier while they wait for their subscription to become active. For more information, see Installing the Amazon Q Developer extension or plugin in your IDE.

Can't see subscribed users

Problem: Subscribed users are not appearing in the Amazon Q Developer console.

You have subscribed one or more users to the Pro tier, but when you navigate to the Amazon Q Developer console's **Subscriptions** page, you can't see them.

Solutions:

- Make sure you're signed in to the correct AWS account and AWS Region.
- Try switching to the **Amazon Q** console. The Amazon Q console is able to display users who were subscribed as part of a group, and is also able to display subscriptions across multiple accounts in an organization managed by AWS Organizations.
- If you switched to the **Amazon Q** console, and still can't see users, do the following:

- Make sure you're in the correct AWS Region. You will need to be in the Region where your IAM Identity Center instance is deployed. This might be a different Region from your Amazon Q Developer console and profile.
- If you're using AWS Organizations, try enabling trusted access so that you can see subscriptions in both management and member accounts. For more information, see Viewing an aggregated list of Amazon Q Developer subscriptions.

Viewing an aggregated list of Amazon Q Developer subscriptions

> **Note**
>
> This section does not apply to personal accounts (Builder IDs).

If you are a management account administrator within an organization managed by AWS Organizations, you can configure Amazon Q to display Amazon Q Developer Pro subscriptions from both *management* and *member* accounts in a single, unified list on the **Subscriptions** page of the Amazon Q console (*not* the Amazon Q Developer console) while signed in to your management account. This organization-wide visibility eliminates the need to sign in to multiple accounts to track your subscriptions.

> **Note**
>
> Unified subscription information also appears on the **Dashboard** page of the Amazon Q Developer console when you enable organization-wide visibility.

If you are a member account administrator, you will only ever be able to view the subscriptions within the member accounts that you administer. This is true regardless of whether organization-wide visibility was enabled in the management account.

To enable organization-wide visibility of Amazon Q Developer subscriptions, you must enable trusted access to Amazon Q in your organization. *Trusted access* is an AWS Organizations feature that lets you designate Amazon Q as a *trusted service* that is allowed to query your organization's structure. This querying is required in order to show the status of subscriptions.

To learn more about trusted access, see Enabling trusted access for AWS Account Management in the *AWS Organizations User Guide*.

To learn more about member and management accounts, see Terminology and concepts for AWS Organizations in the *AWS Organizations User Guide* for explanations.

Use the following instructions to enable trusted access to Amazon Q in your organization.

Prerequisites

Before you begin, make sure that:

- You are an administrator of an AWS *management* account.
- You have an *organization instance* of IAM Identity Center set up in your management account and connected to Amazon Q Developer.
- Your organization instance of IAM Identity Center contains users who are subscribed to Amazon Q Developer Pro in member accounts.
- You have the minimum permissions required to perform actions in the Amazon Q or Amazon Q Developer console (you can use either console to enable trusted access). For more information, see Allow administrators to use the Amazon Q console and Allow administrators to use the Amazon Q Developer console.

To enable trusted access (enable organization-wide visibility of subscriptions)

1. Sign in to the AWS Management Console using your AWS management account.
2. Do one of the following depending on the console you want to use:

   - Switch to the Amazon Q console.

     Choose **Subscriptions**.

     In the **Subscription view settings** section, choose **Edit**.

     Choose **On**.
   - Switch to the Amazon Q Developer console.

     Choose **Settings**.

     In the **Subscription view settings** section, choose **Edit**.

     Enable the toggle.
3. Choose **Save**.

Trusted access to Amazon Q is now enabled. Users and groups who are subscribed in member accounts now appear in the Amazon Q console (not the Amazon Q Developer console) when you're signed in as a management account administrator.

Unsubscribing from Amazon Q Developer Pro

How you unsubscribe from the Pro tier depends on whether you're an end-user with a personal account (Builder ID) or an administrator of IAM Identity Center workforce users.

Unsubscribing a personal account (Builder ID)

Read this section if you want to unsubscribe your personal account (Builder ID) from the Pro tier.

When you unsubscribe your Builder ID, your subscription is marked as **Canceled**, and you can no longer access Amazon Q Developer features. (You can still use the Free tier though, provided you have not exceeded your Free tier limits.) The final monthly subscription fee is charged at the end of the current billing cycle. You'll be charged for the full month; the fee won't be prorated.

> **Warning**
>
> Deleting your Builder ID does not unsubscribe you. To stop being billed, you must actively unsubscribe yourself using the instructions in this section. Similarly, deleting your Amazon Q Developer Profile or Kiro Profile does not cancel your Builder ID subscription.

You can unsubscribe starting from one of the following interfaces:

- IDE
- command line
- AWS Management Console

IDE

To unsubscribe your Builder ID starting from your IDE

1. Authenticate to Amazon Q in your IDE using your personal account (Builder ID). For more information, see Installing the Amazon Q Developer extension or plugin in your IDE.

2. From the Amazon Q menu, choose **Manage Q Developer Pro Subscription**.

   A browser window opens.

3. If prompted, sign in to the AWS Management Console using the AWS account that's linked to your personal account (Builder ID). You specified this AWS account when you upgraded your Builder ID to the Pro tier. For more information, see Upgrading a personal account (Builder ID).

   The **Subscriptions** page in the Amazon Q Developer console appears.

4. In the **Builder ID users** section, choose **Unsubscribe**.

Command line

To unsubscribe your Builder ID starting from the command line

1. On a computer where Amazon Q for the command line is installed, open a terminal.
2. Make sure you're signed in with your personal account (Builder ID) by typing `q whoami` at the command line.

   You should see a `Logged in with Builder ID` message.

3. Start a chat session by typing `q chat` at your terminal's prompt.

   An interactive chat sessions opens.

4. Type **`/subscribe --manage`**.

   The AWS Management Console launches in a browser window.

   > **Note**
   >
   > If the AWS Management Console doesn't automatically launch, copy and paste the URL from the terminal into a browser window.

5. If prompted, sign in to the AWS Management Console using the AWS account that's linked to your personal account (Builder ID). You specified this AWS account when you upgraded your Builder ID to the Pro tier. For more information, see Upgrading a personal account (Builder ID).

   The **Subscriptions** page in the Amazon Q Developer console appears.

6. In the **Builder ID users** section, choose **Unsubscribe**.

AWS Management Console

To unsubscribe your Builder ID starting from the AWS Management Console

1. Sign in to the AWS Management Console using the AWS account that's linked to your personal account (Builder ID). You specified this AWS account when you upgraded your Builder ID to the Pro tier. For more information, see Upgrading a personal account (Builder ID).
2. If you don't have a Kiro Profile, create one in the US East (N. Virginia) Region (IAD) by following the Kiro onboarding quickstart. If you have a Kiro Profile but not a Amazon Q Developer Profile, choose **Settings**, then under **Other applications**, choose **Enable**.
3. Switch to the Amazon Q Developer console in the US East (N. Virginia) Region (IAD).
4. Choose **Subscriptions**.
5. In the **Builder ID users** section, select the subscription and choose **Unsubscribe**.

Unsubscribing IAM Identity Center workforce users

If you are the administrator of an AWS standalone account, an AWS management account, or an AWS member account, use the following procedure to unsubscribe IAM Identity Center workforce users from your account.

For more information about AWS management and member accounts, see Managing AWS accounts in an organization with AWS Organizations.

Notes about unsubscribing users:

- If you are an administrator of either a management or member account within an organization managed by AWS Organizations, you can only unsubscribe users if your created their subscriptions.
- If a user has been subscribed in both member and management accounts, both account administrators must unsubscribe the user from their respective accounts for the user to be fully unsubscribed.
- If you are a management account administrator, you can view other accounts the user is subscribed in by choosing **View subscriptions from member accounts** on the **Settings** page of the Amazon Q Developer console. This allows you to coordinate with member account

administrators for unsubscription. Alternatively, if you have the appropriate permissions, you can sign in as a member account administrator and unsubscribe the user directly. For more information about viewing member account subscriptions as a management account administrator, see Viewing an aggregated list of Amazon Q Developer subscriptions.

- After unsubscribing users or groups, their subscriptions are marked as **Canceled**, and they can no longer access Amazon Q Developer features. (They can still use the Free tier though, provided they have not exceeded their Free tier limits.) The final monthly subscription fee is charged at the end of the current billing cycle for all users who had active subscriptions. You'll be charged for the full month; the fee won't be prorated.

To unsubscribe a user or group you manage

1. Sign in to the AWS Management Console using your AWS standalone, management, or member account.
2. Switch to the Amazon Q Developer console.
3. In the **Identity provider users and groups** section, choose the **Users** or **Groups** tab.
4. Choose the user or group you want to unsubscribe.
5. Choose **More actions**.
6. Choose **Unsubscribe**.

Upgrading to Amazon Q Developer Pro

How you upgrade to the Pro tier from the Free tier depends on whether you're an end-user with a personal account (Builder ID) or whether you're an administrator of IAM Identity Center workforce users.

Upgrading a personal account (Builder ID)

Read this section if you want to upgrade your personal account (Builder ID) from the Free tier to a Pro tier monthly subscription. For reasons why you might consider upgrading, see Tiers of service, and Limitations of Builder IDs.

You can upgrade starting from one of the following interfaces:

- IDE
- command line

Before you begin

- Create an AWS account if you don't have one yet. This account will be linked to your Builder ID, and will be charged the monthly subscription fee. For more information about creating an account, see Create an AWS account in the *AWS Account Management Reference Guide*.

> **Note**
>
> You cannot link multiple Builder IDs to a single AWS account. If you have multiple Builder IDs that you want to upgrade, create separate AWS accounts for each.

IDE

To upgrade starting from the IDE

1. Authenticate to Amazon Q in your IDE using your personal account (Builder ID). For more information, see Installing the Amazon Q Developer extension or plugin in your IDE.
2. In your IDE, do one of the following:

   - If you see a `Monthly request limit reached` message in the Amazon Q chat window, choose **Subscribe to Q Developer Pro**.

     Or

   - From the Amazon Q menu, choose **Subscribe to Q Developer Pro**.

     A browser window opens.
3. If prompted, sign in using your AWS account. You should have created this account previously, as described in **Before you begin**.

   A **Review upgrade details for Q Developer Pro** page appears. The page shows your Builder ID, AWS account number, and subscription fee, among other information. For more information about the subscription fee, see Subscription billing. The **Activation token** field shows a one-time token that links your AWS account with your Builder ID. This token is never used again.
4. Choose **Confirm upgrade**.

The **Subscriptions** page of the Amazon Q Developer console appears with an **Upgrade to Q Developer Pro successful** message. You should see your user name listed in the **Builder ID users** section.

Your personal account (Builder ID) is now subscribed to the Pro tier.

5. (Optional) Return to the Amazon Q chat window in your IDE.

   You should see a **Successfully subscribed to Amazon Q Developer Pro** message.

Command line

To upgrade starting from the command line

1. On a computer where Amazon Q for the command line is installed, open a terminal.
2. Make sure you're signed in with your personal account (Builder ID) by typing `q whoami` at the command line.

   You should see a `Logged in with Builder ID` message.
3. Start a chat session by typing `q chat` at your terminal's prompt.

   An interactive chat sessions opens.
4. Type **`/subscribe`**.

   The AWS Management Console launches in a browser window.

   > **Note**
   >
   > If the AWS Management Console doesn't automatically launch, copy and paste the URL from the terminal into a browser window.

5. If prompted, sign in using your AWS account. You should have created this account previously, as described in **Before you begin**.

   A **Review upgrade details for Q Developer Pro** page appears. The page shows your Builder ID, AWS account number, and subscription fee, among other information. For more information about the subscription fee, see Subscription billing. The **Activation token** field shows a one-time token that links your AWS account with your Builder ID. This token is never used again.

6. Choose **Confirm upgrade**.

   The **Subscriptions** page of the Amazon Q Developer console appears with a **Upgrade to Q Developer Pro successful** message. You should see your user name listed in the **Builder ID users** section.

   Your personal account (Builder ID) is now subscribed to the Pro tier.
7. (Optional) Verify that you're subscribed by typing **`/subscribe`** at the Q CLI prompt. Amazon Q should indicate that you're already subscribed.

Upgrading IAM Identity Center workforce users to the Pro tier

If you are the administrator of IAM Identity Center workforce users, you can upgrade these users to the Pro tier following the instructions in Getting started with IAM Identity Center.

Upgrade to Kiro

The Amazon Q Developer CLI has been rebranded to Kiro.

What does this rebrand mean for you?

- **If you're a user with a personal account**: Optionally upgrade the Q CLI and Amazon Q extension in your IDE to Kiro to take advantage of all the updates that will only be available in Kiro going forward. On upgrade, your extension and Q CLI installation will be rebranded to Kiro, and you'll also get additional features.
- **If you're an administrator of Amazon Q Developer Pro subscriptions**: Start using the **Kiro** console to manage your users' subscriptions. The Kiro console is a rebrand of the Amazon Q Developer console, with all the same functionality. Have your users upgrade their Amazon Q IDE extensions and CLI installations to Kiro to access the rebranded interface and receive additional features. After upgrading, all Amazon Q components will automatically update to their Kiro equivalents.

Canceling Amazon Q Developer

After you upgrade to Kiro, the Amazon Q Developer console is no longer visible in the left navigation pane by default. To cancel your Amazon Q Developer subscription, you must first enable the Amazon Q Developer console, and then follow the unsubscribe instructions.

> **Note**
>
> The **Users & Groups** page under **Kiro** in the left navigation pane is only for managing Kiro subscriptions. To manage Amazon Q Developer subscriptions, you must enable the Amazon Q Developer console as described in the following procedure.

To cancel Amazon Q Developer

1. Open the Kiro console.
2. In the left navigation pane, choose **Settings**.
3. In the **Other applications** section, choose **Enable** to make Amazon Q Developer visible in the left navigation pane. An **Amazon Q Developer** section appears in the left navigation pane with links to the Dashboard, Subscriptions, Customizations, and Plugins pages.
4. In the left navigation pane, under **Amazon Q Developer**, choose **Subscriptions**, and then follow the instructions in Unsubscribing from Amazon Q Developer Pro to cancel your subscription.

To learn about Kiro in the IDE and at the command line, and how to administer Kiro subscriptions in your enterprise, see the Kiro Docs.

Using Amazon Q Developer on AWS apps and websites

Use Amazon Q Developer in the AWS Management Console, AWS Console Mobile Application, AWS Marketing website, AWS Documentation website, and supported chat applications to ask questions about AWS. You can ask Amazon Q about AWS architecture, best practices, support, and documentation. Amazon Q can also help with code that you're writing with the AWS SDKs and AWS Command Line Interface (AWS CLI).

In the AWS Management Console, you can ask Amazon Q about your AWS resources and costs, contact Support directly, and diagnose common console errors.

To quickly provide access to features of Amazon Q Developer on AWS, attach the `AmazonQDeveloperAccess` AWS managed policy to the IAM identity using Amazon Q. For permissions needed for specific features, see the topic for the feature you want to use.

Topics

- Authenticating to your Amazon Q Developer Pro subscription
- Chatting with Amazon Q Developer about AWS
- Using Amazon Q Developer plugins
- Automating AWS services with Amazon Q Developer Console-to-Code
- Diagnosing common errors in the console with Amazon Q Developer
- Using Amazon Q Developer to chat with Support

Authenticating to your Amazon Q Developer Pro subscription

To access Amazon Q at the Free tier, sign in to the AWS Management Console. Any Free tier features are available as long as you have the required permissions.

To access Amazon Q at the Pro tier, sign to the console with IAM Identity Center. When you sign in with IAM Identity Center, including authenticating through an external identity provider that is connected to IAM Identity Center, you will automatically have access to the Pro tier if your IAM Identity Center identity is subscribed to Amazon Q Developer Pro.

For more information on the Amazon Q Developer Pro tier, see Tiers of service for Q Developer – Free and Pro.

> **Note**
>
> If you see an error message that starts with, `Your account has not been configured to use an Amazon Q subscription`, see Troubleshooting Amazon Q Developer Pro subscriptions for troubleshooting tips.

If you sign in to the AWS console with IAM or federation with IAM, then you will be prompted to authenticate with IAM Identity Center when you reach a Free tier limit or attempt to use a feature only available at the Pro tier.

Chatting with Amazon Q Developer about AWS

> **Introducing generative AI-based Q artifacts**
>
> Amazon Q can now provide answers to questions with rich, interactive visualizations including tables, charts, dashboards, time-series graphs, key value pairs etc.

Chat with Amazon Q in the AWS Management Console, AWS Console Mobile Application, AWS website, AWS Documentation website, and chat applications to learn about AWS services.

You can ask Amazon Q about best practices, recommendations, step-by-step instructions for AWS tasks, and architecting your AWS resources and workflows. You can also ask about your AWS resources and account costs. Amazon Q additionally generates short scripts or code snippets to help you get started using the AWS SDKs and AWS CLI.

The following topics describe how to use Amazon Q chat and topics you can chat about.

Topics

- Using Q artifacts in Amazon Q
- Add permissions
- Start a conversation
- Manage conversations in the console
- Navigate the Amazon Q chat panel
- Chat settings

- Example prompts
- Chatting about your resources with Amazon Q Developer
- Asking Amazon Q to troubleshoot your resources
- Chatting about your costs
- Chatting about your network security
- Chatting about email sending
- Chatting about your telemetry and operations

Using Q artifacts in Amazon Q

Amazon Q artifacts enable Amazon Q to deliver responses enriched with visualizations such as tables, charts, and dashboards. When you ask natural language questions about your resources, Amazon Q may display an artifact that helps you quickly understand your resources at a glance. You can paginate, filter, and sort results directly within the response.

The Q experience is now more usable and useful. Access Q easily from the navigation bar next to search. The Q chat panel opens and can expand to full screen. A new prompt library helps you discover useful example prompts.

To get started, ensure you have the required permissions, and then review the example prompts to get the most out of Amazon Q artifacts. For more information, see Prerequisites and Example prompts.

Prerequisites

To view visualizations with Amazon Q see Allow users to chat with Amazon Q and Chatting about your costs.

How it works

> **Note**
>
> All data associated with Amazon Q visualizations is saved in us-east-1.

To view Q artifacts in Amazon Q in the AWS management Console:

1. Sign in to the AWS Management Console.
2. Access Amazon Q by choosing the Q icon in the Unified Navigation bar.
3. Describe your task to Amazon Q using natural language. For example:
   a. "List my running EC2 instances"
   b. "Create a chart of my costs by region last month"
4. If Amazon Q determines a visual interface would be helpful, it automatically displays an artifact in a new panel next to Q chat with either a table or chart visualization.
   a. If you are asking about resources, the panel will include:
      i. A table with the resources you asked about, categorized based on any properties you specify.
      ii. Deep links to the resources that redirect you to the resource page in the service console.
   b. If you are asking about cost and billing information with chart visualization, the panel will include a chart widget.

Example prompts

The following categories and associated prompts are examples of the types of tasks you can complete with Amazon Q artifacts.

- **View resource information** – Visualize resource information in table or chart format.
- **Get billing recommendations and forecasts** – Show me a line chart of my forecasted costs for the next 6 months, Graph RDS costs by instance type by month for the last 6 months.
- **Security and compliance** – Check traffic and internet accessibility to EC2 resources, verify internet connectivity for EC2 instances across regions.
- **Resource detail** – Show me the configuration details for my EC2 instance i-xxxx.
- **Time-series charts** – Graph network utilization for my EC2 instances over 12 hours across all regions.
- **Permissions/Security** – Which instances across all regions are open to the world?
- **Troubleshooting diagnostics** – Why is my ECS task failing in cluster my-ecs-cluster?

For a list of suggested use cases, choose the Amazon Q prompt library icon in the Q chat panel and filter by table or visualization response type.

Add permissions

For an IAM policy that grants permissions needed for chatting with Amazon Q, see Allow users to chat with Amazon Q.

Start a conversation

To open up the Amazon Q chat panel in the AWS Management Console, choose the Amazon Q icon in the navigation bar. To open up the panel on the AWS website or any AWS service's documentation page, choose the Amazon Q icon.

To ask Amazon Q a question, enter your question into the text bar in the Amazon Q panel. Amazon Q generates a response to your question with a sources section that links to its references.

After you receive a response, you can optionally leave feedback by using the thumbs-up and thumbs-down icons. You can also copy the response to your clipboard by choosing the copy icon.

To start a new conversation in the console:

1. You can start a new conversation by choosing the plus icon (+) in the chat panel header.
2. To name or rename a conversation, choose the conversation name field in the chat panel and enter your conversation name.

Manage conversations in the console

You can view, switch to, and delete your past conversations in Amazon Q.

Amazon Q maintains the history of previously asked questions and responses within a given conversation to use as context to inform responses. You can save up to 1,000 separate conversations with Amazon Q chat in the AWS console.

When you start a conversation, it's automatically saved as a new conversation. You can title the conversation, or Amazon Q will generate a title based on the example prompt you select or the first few questions in the conversation.

You can switch between conversations to continue chatting with Amazon Q about previous topics. Inactive conversations, in which you don't ask a new question, will be deleted after 90 days of inactivity. Messages older than 90 days will be deleted, even if a conversation is still active.

To switch conversations:

1. Choose the clock icon in the chat panel header. The **Conversations** pop-up opens.
2. Choose the name of the conversation you want to resume. All previous messages from that conversation appear in the chat panel where you can continue chatting with Amazon Q.

To delete conversations:

1. Choose the clock icon in the chat panel header. The **Conversations** pop-up opens.
2. Choose the delete icon next to the name of the conversation you want to delete.

If you're using Amazon Q in the console, your current conversation and associated context are maintained when you navigate to another place in the console or to another browser or tab. If you're using Amazon Q on the AWS website, Documentation website, or Console Mobile Application, a new conversation starts without any context when you navigate to a new page, browser, or tab.

Navigate the Amazon Q chat panel

Note: You can switch between the Amazon Q chat panel and service consoles at any time:

1. To expand the Q chat panel in full-screen mode, choose the maximize icon in the chat panel header. To toggle full-screen mode, choose the resize icon.
2. To close the Q chat panel, choose < in the chat panel header. To close the panel with visualizations, choose X in the chat panel header.
3. To adjust the chat panel size, use the divider.
4. To reopen the chat panel, choose the Q icon in the Unified Navigation bar.
5. Your work is automatically saved when switching between views.

Chat settings

To view your chat settings in Amazon Q, choose the gear icon in the chat panel header.

- **Region** — Amazon Q defaults to the AWS Region set in the AWS Management Console when you open the chat panel. To update the Region used by Amazon Q, change your console Region.

Example prompts

You can ask Amazon Q questions about AWS and AWS services, such as finding the right service, understanding best practices or reviewing the state of your resources. If Amazon Q determines a visual interface would be helpful, it generates a rich visualization such as a table, chart, health dashboard, time-series graph, or diagnostic view.

You can also ask about software development with the AWS SDKs and AWS CLI. Amazon Q in the console can generate short scripts or code snippets to help you get started using the AWS SDKs and AWS CLI.

The following are example questions that demonstrate how Amazon Q can help you build on AWS:

- List RDS databases without CloudWatch alarms
- What's the maximum runtime for a Lambda function?
- List S3 buckets with tag value *<tag value>*
- Create a chart showing my cost per GB for different S3 storage classes
- Graph EC2 cost per vCPU hour over the last 3 weeks
- What's the best container service to use to run my workload if I need to keep my costs low?
- Show me a bar chart of potential savings by optimization recommendation
- Graph network utilization for my EC2 instances over 12 hours
- Which instances across all regions are open to the world?

To help you get started, Q recommends prompts when you start a new conversation. You can also view the list of supported prompts in the prompt library. To view prompts in the prompt library, choose the book icon in the chat panel header.

Chatting about your resources with Amazon Q Developer

Amazon Q Developer answers questions about your AWS account resources to help you understand your AWS infrastructure through natural language prompts. Using advanced reasoning capabilities, Amazon Q analyzes and provides insights about your resources so you can quickly get the information you need without relying on multiple service consoles, APIs, or complicated scripts.

The type of resource analysis Amazon Q can perform includes:

- **Resource listing and details** – Ask for lists or specific details about resources in your account.

- **Filtered queries** – Request resource information based on criteria such as region or configuration state.
- **Cross-service analysis** – Ask complex questions about your infrastructure, configurations, and dependencies across multiple AWS resources and services.
- **Troubleshooting assistance** – Get help identifying and resolving issues with your resources. For more information, see Asking Amazon Q to troubleshoot your resources.

For examples of questions you can ask, see Ask Amazon Q for resource information.

Topics

- How it works
- Prerequisites
- Ask Amazon Q for resource information
- Count resources with AWS Resource Explorer

How it works

To respond to questions about resources, Amazon Q uses service APIs and AWS Cloud Control API to retrieve the requested information. To allow Amazon Q to call the APIs required to retrieve requested resource information, your IAM identity must have permissions to use those APIs. For more information, see Prerequisites.

Amazon Q can perform get, list, and describe actions to retrieve information about multiple AWS resources at a time. When asked complex resource questions, Amazon Q creates dynamic, multi-step plans that explain the reasoning behind the actions it's taking to further your understanding of your AWS environment. If the initial plan fails, Amazon Q attempts alternative methods or prompts you for any additional information required to continue.

Amazon Q can provide answers to questions enriched with read-only Q artifacts. For example, when you ask a question about your resources or cost and billing, Amazon Q generates visualizations like tables and charts to help you quickly understand the state of your account resources.

Amazon Q can't answer questions about the data stored in your resources, such as listing objects in an Amazon S3 bucket, or questions related to your account security, identity, credentials, or cryptography.

Prerequisites

You can chat about your account resources with Amazon Q in the AWS Management Console, AWS Console Mobile Application, and in configured chat applications.

To chat about your resources, your IAM identity must have the following permissions:

- Permissions to chat with Amazon Q, to use Cloud Control API, and to allow Amazon Q to access your resources. For an IAM policy that grants the required permissions, see Allow users to chat about resources with Amazon Q.
- Permissions to access the resources you ask about. For example, if you ask Amazon Q to list your Amazon S3 buckets, you must have the `s3:ListAllMyBuckets` permission.

  Amazon Q will never access resources that your IAM identity doesn't have access to.

> **Important**
>
> Normal fees apply when you ask Amazon Q to perform read, list, or describe actions. For more information, see the pricing page for the AWS service you are asking Amazon Q about.

Ask Amazon Q for resource information

When you ask Amazon Q about your resources, you can specify the AWS Region that Amazon Q calls to locate your resources. If no Region is specified in a given query, Amazon Q will use a Region previously specified in your conversation if applicable, and otherwise uses your current console Region (or the most recent console Region if you are using a global console Region).

Amazon Q might need additional information to answer to your resource questions. When Amazon Q asks a follow up, reply with the requested details.

Following are example questions you can ask Amazon Q about your resources:

- Describe the encryption settings for S3 bucket *<name>*
- What SQS queues invoke my Lambda functions?
- Do I have any MySQL RDS clusters that need updates?
- List my EC2 instances in *<region>*

- Get the configuration for my lambda function *<name>*
- What alarms are configured for instance *<instance ID>*?
- List RDS databases without CloudWatch alarms
- List S3 buckets with tag value *<tag value>*
- Show me chart of my costs by service last week
- Show me a bar chart of my top 10 most expensive resources
- Create a chart showing budget vs forecasted spend

Count resources with AWS Resource Explorer

When you ask a question that requires resource counting, such as 'How many EC2 resources are running in my account?', Amazon Q uses Cloud Control API by default to return a count of the requested resources. You also have the option to enable and configure Resource Explorer for faster resource counting with Amazon Q.

If Resource Explorer is enabled, Amazon Q will attempt to use it when generating a response that requires counting your resources. Amazon Q can use Resource Explorer to count a single type of resource across all AWS Regions. Using Resource Explorer enables Amazon Q to count resources faster by returning the count from the Resource Explorer index, as opposed to calling service APIs to list resources and count the results.

If you choose to enable Resource Explorer for resource counting, note that resource information can be out of date. Resource Explorer indexes resources in your account by taking a periodic inventory, and if resources have been created or deleted after the last inventory, the resource count will be incorrect. Resource Explorer also doesn't support resource filtering. If you ask to count resources matching a specific criteria, Amazon Q will fall back to Cloud Control API.

If you don't have Resource Explorer enabled and configured for use, or if Amazon Q can't use Resource Explorer to answer your question, Amazon Q uses Cloud Control API to count resources. Using Cloud Control API ensures an accurate resource count and supports resource filtering, however this can also lead to increased latency compared to counting with Resource Explorer. If you are counting a large number of resources, Cloud Control API can also time out.

To use Resource Explorer for resource counting, the following configuration is required:

- The user interacting with Amazon Q must be in account where an Resource Explorer default view is configured and an aggregator index has been created in the same Region as the default

view. For more information, see Setting up Resource Explorer using Advanced setup in the *AWS Resource Explorer User Guide*.

- The user's IAM identity must have read permissions for the default view. For more information, see Granting access to Resource Explorer views for search in the *AWS Resource Explorer User Guide*.

Asking Amazon Q to troubleshoot your resources

In the AWS Management Console, you can ask Amazon Q to troubleshoot issues you're having with your AWS resources. When you encounter a problem, open the chat panel and describe the situation to Amazon Q. For instance, you might enter, "I can't add an object to my S3 bucket" or "My load balancer is returning a 503 error". Amazon Q analyzes the information you provided to identify potential root causes. It then offers tailored solutions, step-by-step instructions, or best practices to resolve your issue efficiently.

The following examples show the types of resource issues Amazon Q can help you troubleshoot, organized by AWS service.

Amazon S3: Permissions issues

- Why can’t I put objects into my S3 bucket? The bucket ID is amzn-s3-demo-bucket.
- Why can’t I delete the object s3://amzn-s3-demo-bucket-locked/Q-Stream2.jpg?
- Why can't I delete an object in S3?

AWS Glue: Job failures

- My Glue job with the job name 'Run111B11B11-<...>' and the job run id 'bb_b1b111<...>' in the 'us-west-2' region failed.
- Why did my Glue job called GlueRun00AA00A00A-<...> fail?

Amazon Athena: Query issues

- My Athena query didn't return any results. query ID: 222c22cc-2c022-<...> region id: us-east-2
- I ran an Athena query with an execution ID of 333d33dd-3d33-<...> and a region of us-east-1, and it didn't return any results.

Amazon ECS: Task stoppage issues; Fargate health check issues; disconnected agent issues

- My ECS task is stopped and I don't know why. The details of the task are: Cluster: my-ecs-cluster, Service: my-ecs-service, Task Definition: my-task-definition, Task ARN: arn:aws:ecs:us-west-2:444444444444:task/my-ecs-cluster/4ee4ee4ee4444<...>
- I'm having a problem with my ECS task. The task health check always fails for the task in the 'my-ecs-cluster' cluster and service.
- The Amazon ECS agent on one of my container instances appears to be disconnected. The agent is not responding or updating its status, which is causing tasks to be stuck in a pending state.

Amazon EC2 Elastic Load Balancing: Health check issues; 504, 503, 502, and 500 errors

- Why are the health checks for the target group called 'my-target-group' failing?
- Why am I receiving 503 errors from my load balancer 'my-elb'?

Amazon EKS: Application Load Balancer (ALB) ingress controller issues; managed add-on issues

- I have an ALB ingress controller in my EKS cluster, and am seeing a failure with the error message 'WebIdentityErr:failed to retrieve credentials'. The AWS region is us-west-2.
- There seems to be an issue with the add-ons in my EKS cluster called my-eks-cluster, in the us-west-2 region.

Amazon ECR: Secondary account access issues

- I'm having difficulty granting access to an Amazon ECR image repository from a different AWS account. Specifically, I need to allow account 222222222222 to push and pull images from the repository named "my-ecr-repo" in my account (111111111111) in the region (us-west-2).

Amazon CloudWatch: Metric and performance monitoring

- Show me the CPU utilization for my EC2 instance i-<...> over the last 24 hours.

For Amazon Q to troubleshoot your resources, you'll need the same permissions as those outlined in Chatting about your resources with Amazon Q Developer.

Chatting about your costs

Amazon Q Developer is a generative artificial intelligence (AI) powered conversational assistant that can help you understand, build, extend, and operate AWS applications. Amazon Q Developer provides powerful capabilities to help you manage your AWS costs through natural conversation. You can analyze your historical and forecasted costs from Cost Explorer, discover cost-saving recommendations from Cost Optimization Hub and AWS Compute Optimizer, understand Savings Plans and reservation opportunities, and get instant answers about AWS product attributes or service pricing. Amazon Q Developer can both answer specific questions (e.g., "What were net unblended costs for EC2 instances last month?") or perform complex or open-ended analysis (e.g., "What were the biggest drivers of last week's cost decrease?"). Amazon Q Developer transforms how you interact with AWS cost data by letting you ask questions in your own words instead of learning query syntax or navigating multiple console pages, while providing precise answers backed by real data from your AWS account and showing exactly which APIs were called and where to find the information in the console.

For more information about the cost management capabilities in Amazon Q Developer, see Managing your costs using generative AI with Amazon Q Developer in the *AWS Cost Management User Guide*.

What you can do

With Amazon Q Developer, you can:

- **Analyze your costs** – Ask questions about your historical spending patterns, cost trends, and forecasted costs. For example, "What were my EC2 costs last month?" or "Why did my costs increase last week?"
- **Find optimization opportunities** – Discover ways to reduce your AWS spending by asking about recommendations from Cost Optimization Hub, AWS Compute Optimizer, and Savings Plans. For example, "What are my top cost optimization opportunities?" or "Which EC2 instances are over-provisioned?"
- **Understand pricing** – Get instant answers about AWS service pricing. For example, "How much does a c8g.2xlarge instance cost in us-east-1?" or "What would it cost to store 1 PB in S3 in Dublin?"
- **Check payment status** – List recent invoices and check payment balance. For example, “List my invoices for the last 6 months” and “Do I have an outstanding payment balance?”

- **Visualize your costs** – Generate custom charts and graphs of historical costs and usage, service pricing, budgets, and more. For example, “Show me a graph of how much we’re spending in each region” or “Create a chart breaking down EC2-Other costs last month”.

Amazon Q Developer adapts to however you phrase your questions. You can ask specific questions when you know exactly what you want, or ask open-ended exploratory questions and let Q investigate on your behalf. Q maintains context throughout your conversation, so you can ask follow-up questions to dive deeper or guide the analysis in a specific direction.

How it works

When you ask Amazon Q Developer about your costs, Q retrieves data from AWS Cost Explorer, Cost Optimization Hub, AWS Compute Optimizer, and other AWS services. Q performs calculations, analyzes patterns, and provides insights based on your actual usage and spending data. With each response, Q provides transparency into how it arrived at its answer by showing you the API calls it made, the parameters used, and links to matching views in the AWS Management Console where available. This helps you verify the data and explore further.

Getting started

To chat about your AWS costs, you need:

- **Appropriate IAM permissions** – Your IAM identity must have permissions to chat with Amazon Q and access your billing data. For an IAM policy that grants the required permissions, see Allow Amazon Q to access cost data and provide cost optimization recommendations.
- **Cost Explorer opt-in** – You must enable AWS Cost Explorer in your AWS account. To enable Cost Explorer, open the Cost Explorer console. For more information, see Enabling Cost Explorer in the *AWS Cost Management User Guide*.

To take advantage of the full range of Amazon Q Developer's cost management capabilities, you can also enable additional services such as AWS Cost Optimization Hub or AWS Budgets. To learn more, see Overview of cost management capabilities in Amazon Q Developer in the *AWS Cost Management User Guide*.

To get started:

1. Sign in to the AWS Management Console at https://console.aws.amazon.com.
2. Choose the Amazon Q icon in the console navigation bar.

3. Ask a question about your costs, such as:
   - "What were my costs last month?"
   - "What are my top cost optimization opportunities?"
   - "How much does a c8g.2xlarge instance running Linux cost in us-east-1?"
   - “Show me a pie chart of my costs by region last week”

You can also configure Amazon Q Developer in chat applications such as Slack and Microsoft Teams. For more information about using Amazon Q Developer in chat applications, see Chatting with Amazon Q Developer in chat applications.

Example questions

Following are example questions about costs that you can ask Amazon Q Developer:

Cost analysis

- "What were my costs last month?"
- "Show me my EC2 spending trends for the past six months."
- "What are the top contributing services to my AWS bill in the eu-central-1 region?"
- "Why did my costs increase last week?"
- "Analyze my spending data for the last month and give me the most important insights."

Cost optimization

- "What are my top cost optimization opportunities?"
- "Which EC2 instances are over-provisioned?"
- "Do I have any idle resources?"
- "Which Savings Plans should I purchase?"

Budget and anomaly monitoring

- "Have any teams exceeded their budgets?"
- "Do I have any cost anomalies?"

Pricing estimation

- "How much does a c8g.2xlarge instance cost in us-east-1?"
- "What would it cost to store 1 PB in S3 in Dublin?"
- "What's the monthly cost of a t4g.xlarge RDS instance with Multi-AZ and 300 GB gp2 storage?"
- "What would be the price to build a basic three tier web app, with a small EC2 instance, API gateway, a ~5GB SQL database, and a basic JS front-end hosted in CloudFront?"

Cost visualization

- “Graph my support charges by month for the last 12 months”
- “Show me an area chart of EC2 costs by instance type by day this month”
- “Create a chart of S3 storage pricing by tier in us-east-1”"
- “Line chart of Savings Plans coverage and utilization % over the last 3 months”
- “Graph EC2 cost per vCPU hour over the last 3 weeks”

Chatting about your network security

> Chatting about network security is in preview, and is subject to change.

Amazon Q can help you analyze your network security configurations, identify missing or misconfigured AWS network security services, and provide recommendations for a stronger network security posture. This helps you understand network security findings, implement remediation steps, and follow security best practices without interrupting your workflow.

When you ask Amazon Q about your network security, its responses include specific information about your resources, related security findings, and detailed remediation instructions as well as links to learn more in the AWS Management Console.

Prerequisites

You can chat about your AWS network security in the AWS Management Console and in configured chat applications.

For Amazon Q to answer questions about your network security, the following prerequisites must be met.

Add permissions

To chat about your network security, your IAM identity must have permissions to chat with Amazon Q. For an IAM policy that grants the required permissions, see Allow users to chat with Amazon Q.

Example questions

Following are example questions about network security that you can ask Amazon Q:

- Identify my top network security findings
- Summarize the network security of my environment
- Are my systems at risk of DDoS attacks?
- How can I improve my network security?
- Do I have any resources without WAF protection?
- Which resources are not protected from common web vulnerabilities?
- What are the common network security issues on my EC2 instances?
- Do I have any WAF WebACLs that aren't protecting anything?

Chatting about email sending

Amazon Q can help you set up email sending in Amazon Simple Email Service (Amazon SES), helping you to optimize your sending delivery and engagement rates, and troubleshoot sending problems. When you ask Amazon Q about Amazon SES, it's responses include information about the sending identities, configuration sets, and other Amazon SES resources in your account. It is also able to answer questions about your email sending patterns, as well as patterns of responses from mailbox providers such as Gmail and Yahoo.

Prerequisites

You can chat about your Amazon SES in the AWS Management Console and in configured chat applications.

For Amazon Q to answer questions about your email sending, the following prerequisites must be met.

Add permissions

To chat about your email sending, your IAM identity must have permissions to chat with Amazon Q. For an IAM policy that grants the required permissions, see Allow users to chat with Amazon Q. You must also have permissions to access the Amazon SES resources you ask about.

Example questions

Following are example questions about email sending that you can ask Amazon Q:

- Do I need to do anything to finish setting up SES for email sending?
- Tell me which sending identities have the best deliverability performance.
- How is my deliverability for emails sent to Yahoo?
- Do you have any recommendations to improve my sending?
- Tell me if there have been any recent events where my deliverability performance suddenly improved or worsened.

Chatting about your telemetry and operations

Amazon Q analyzes your CloudWatch telemetry and operational data to help manage your AWS environment. It retrieves resource health information, monitors alarms, and provides troubleshooting guidance. When you ask questions, Amazon Q may prompt you for specific details like resource names and time ranges to ensure accurate assistance.

AWS service health check: Evaluate the health of resources of specified AWS services, assisting customers in troubleshooting and resolving issues or errors they encounter with these resources.

- Is my Lambda function X healthy?
- Is anything wrong with my Amazon ECS clusters?
- Help me troubleshoot my DynamoDB tables between time X and Y.
- Investigate anomalies related to Amazon S3 between time X and Y.
- Graph network utilization for my Amazon EC2 instances over 12 hours across all regions.
- Show me CPU utilization trends for my Lambda functions this week.

Alarm troubleshooting: Identifies alarms in Alarm state and the underlying telemetry that triggered the alarm, helping customers diagnose the reasons behind the alarm/alert/pages.

- Why is my alarm with name X firing?
- Show me the metrics that triggered my alarm X over the last 6 hours.

Application Signals specific troubleshooting: Analyzes CloudWatch Application Signals service-level objectives and indicators to determine the overall health of a service, enabling you to assess and maintain application performance.

- Is my Service X in environment Y healthy?
- Create a latency chart for my API Gateway endpoints over the last 6 hours.

For more information about how Amazon Q analyzes your CloudWatch telemetry and operational data, see *CloudWatch investigations* in the Amazon CloudWatch User Guide.

Using Amazon Q Developer plugins

Amazon Q Developer integrates with third party monitoring tools and security platforms so you can access your AWS application insights without leaving the AWS builder environment. In the AWS Management Console, you can chat about metrics provided by these tools to understand and address application performance, errors, or vulnerabilities.

After you configure a plugin, add the plugin alias to the beginning of your question when you chat with Amazon Q in the AWS console. Amazon Q calls the third party provider APIs to retrieve resources and generates a response with deep links to the external resources.

When Amazon Q calls a third party API, the API will not appear in AWS CloudTrail logs. The CloudTrail log will only show when an AWS Secrets Manager secret is accessed by Amazon Q to retrieve credentials to connect to the third party provider.

Amazon Q doesn't share any information with third party providers when you configure or use plugins. For more information on how Amazon Q uses your data, see Data protection.

> **Note**
>
> Member accounts within an AWS organization don't have access to plugins that are configured in the organization's management account profile. Each member account must create their own Q Developer profile before they can configure and use plugins in their account.

Warning

Third party provider user permissions are not detected by Amazon Q Developer plugins. When an administrator configures a plugin in an AWS account, users with plugin permissions in that account have access to any resources in the third party provider account retrievable by the plugin.
You can configure IAM policies to restrict which plugins users have access to. For more information, see Allow users to chat with plugins from one provider.

To get started, see the topic for the plugin you want to use with Amazon Q Developer.

Topics

- Configuring the Amazon Q Developer CloudZero plugin
- Configuring the Amazon Q Developer Datadog plugin
- Configuring the Amazon Q Developer Wiz plugin

Configuring the Amazon Q Developer CloudZero plugin

CloudZero is a cloud cost optimization platform that evaluates costs to improve cloud efficiency. If you use CloudZero to monitor your AWS costs, you can use the CloudZero plugin in Amazon Q Developer chat to access cost insights without leaving the AWS Management Console.

You can use the CloudZero plugin to understand your AWS costs, get cost optimization insights, and track billing. After you receive a response, you can ask follow up questions, such as the status or cost impact of CloudZero insights.

To configure the plugin, you provide authentication credentials from your CloudZero account to enable a connection between Amazon Q and CloudZero. After you configure the plugin, you can access CloudZero data by adding **`@cloudzero`** to the beginning of your question in Amazon Q chat.

Warning

CloudZero user permissions are not detected by the CloudZero plugin in Amazon Q. When an administrator configures the CloudZero plugin in an AWS account, users with plugin permissions in that account have access to any resources in the CloudZero account retrievable by the plugin.

You can configure IAM policies to restrict which plugins users have access to. For more information, see Configure user permissions.

Prerequisites

Add permissions

To configure plugins, the following administrator level permissions are required:

- Permissions to access the Amazon Q Developer console. For an example IAM policy that grants needed permissions, see Allow administrators to use the Amazon Q Developer console.
- Permissions to configure plugins. For an example IAM policy that grants the needed permissions, see Allow administrators to configure plugins.

Acquire credentials

Before you begin, note the following information from your CloudZero account. These authentication credentials will be stored in an AWS Secrets Manager secret when you configure the plugin.

- **API key** – An access key that allows Amazon Q to call the CloudZero API to access your organization's cost insights and billing information. You can find the API key in your CloudZero account settings. For more information, see the Authorization in the CloudZero documentation.

For more information on acquiring credentials from your CloudZero account, see the CloudZero documentation.

Secrets and service roles

AWS Secrets Manager secret

When you configure the plugin, Amazon Q creates a new AWS Secrets Manager secret for you to store CloudZero authentication credentials. Alternatively, you can use an existing secret that you create yourself.

If you create a secret yourself, enter the API key as plaintext:

```
your-api-key
```

For more information about creating secrets, see Create a secret in the *AWS Secrets Manager User Guide*.

Service roles

To configure the CloudZero plugin in Amazon Q Developer, you need to create a service role that gives Amazon Q permission to access your Secrets Manager secret. Amazon Q assumes this role to access the secret where your CloudZero credentials are stored.

When you configure the plugin in the AWS console, you have the option to create a new secret or use an existing one. If you create a new secret, the associated service role is created for you. If you use an existing secret and an existing service role, make sure your service role contains the following permissions, and has the following trust policy attached. The service role required depends on your secret encryption method.

If your secret is encrypted with an AWS managed KMS key, the following IAM service role is required:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [
                "arn:aws-eusc:secretsmanager:eusc-de-
east-1:111122223333:secret:secret-id"
            ]
        }
    ]
}
```

If your secret is encrypted with a customer managed AWS KMS key, the following IAM service role is required:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource": "arn:aws-eusc:secretsmanager:eusc-de-
east-1:111122223333:secret:secret-id"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "arn:aws-eusc:kms:eusc-de-east-1:111122223333:key/key-
id",
            "Condition": {
                "StringEquals": {
                "kms:ViaService": "secretsmanager.eusc-de-east-1.amazonaws.com"
                }
            }
        }
    ]
}
```

To allow Amazon Q to assume the service role, the service role needs the following trust policy:

Note

The `codewhisperer` prefix is a legacy name from a service that merged with Amazon Q Developer. For more information, see Amazon Q Developer rename - Summary of changes.

JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "q.amazonaws.com"
      },
      "Action": ["sts:AssumeRole", "sts:SetContext"],
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "111122223333",
          "aws:SourceArn": "arn:aws-eusc:codewhisperer:eusc-de-
east-1:111122223333:profile/profile-id"
        }
      }
    }
  ]
}
```

For more information about service roles, see Create a role to delegate permissions to an AWS service in the *AWS Identity and Access Management User Guide*.

Configure the CloudZero plugin

You configure plugins in the Amazon Q Developer console. Amazon Q uses credentials stored in AWS Secrets Manager to enable interactions with CloudZero.

To configure the CloudZero plugin, complete the following procedure:

1. Open the Amazon Q Developer console at https://console.aws.amazon.com/amazonq/developer/home
2. On the Amazon Q Developer console home page, choose **Settings**.
3. In the navigation bar, choose **Plugins**.
4. On the plugins page, choose the plus sign on the **CloudZero** panel. The plugin configuration page opens.
5. For **Configure AWS Secrets Manager**, choose either **Create a new secret** or **Use an existing secret**. The Secrets Manager secret is where your CloudZero authentication credentials will be stored.

   If you create a new secret, enter the following information:

a. For **CloudZero API key**, enter the API key for your CloudZero organization.

b. A service role will be created that Amazon Q will use to access the secret where your CloudZero credentials are stored. Do not edit the service role that is created for you.

If you use an existing secret, choose a secret from the **AWS Secrets Manager secret** dropdown menu. The secret should include the CloudZero authentication credentials specified in the previous step.

For more information about the required credentials, see Acquire credentials.

6. For **Configure AWS IAM service role**, choose either **Create new service role** or **Use existing service role**.

> **Note**
>
> If you chose **Create a new secret** for step 6, you can't use an existing service role. A new role will be created for you.

If you create a new service role, a service role will be created that Amazon Q will use to access the secret where your CloudZero credentials are stored. Do not edit the service role that is created for you.

If you use an existing service role, choose a role from the dropdown menu that appears. Make sure your service role has the permissions and trust policy defined in Service roles.

7. Choose **Save configuration**.

8. After the CloudZero plugin panel appears in the **Configured plugins** section on the Plugins page, users will have access to the plugin.

If you want to update the credentials for a plugin, you must delete your current plugin and configure a new one. Deleting a plugin removes all previous specifications. Any time you configure a new plugin, a new plugin ARN is generated.

Configure user permissions

To use plugins, the following permissions are required:

- Permissions to chat with Amazon Q in the console. For an example IAM policy that grants permissions needed to chat, see Allow users to chat with Amazon Q.
- The `q:UsePlugin` permission.

When you grant an IAM identity access to a configured CloudZero plugin, the identity gains access to any resources in the CloudZero account retrievable by the plugin. CloudZero user permissions are not detected by the plugin. If you want to control access to a plugin, you can do so by specifying the plugin ARN in an IAM policy.

Each time you create or delete and re-configure a plugin, it is assigned a new ARN. If you use a plugin ARN in a policy, it will need to be updated if you want to grant access to the newly configured plugin.

To locate the CloudZero plugin ARN, go to the **Plugins** page in the Amazon Q Developer console and choose the configured CloudZero plugin. On the plugin details page, copy the plugin ARN. You can add this ARN to a policy to allow or deny access to the CloudZero plugin.

If you create a policy to control access to CloudZero plugins, specify `CloudZero` for the plugin provider in the policy.

For examples of IAM policies that control plugin access, see Allow users to chat with plugins from one provider.

Chat with the CloudZero plugin

To use the CloudZero plugin, enter **`@cloudzero`** at the beginning of a question about CloudZero or your AWS application monitors and cases. Follow up questions or responses to questions from Amazon Q must also include **`@cloudzero`**.

Following are some example use cases and associated questions you can ask to get the most of out of the Amazon Q CloudZero plugin:

- **Learn about using CloudZero with AWS** – Ask about how CloudZero features work. Amazon Q might ask you for more information about what you're trying to do to provide the best answer.
  - **`@cloudzero how do I use CloudZero?`**
  - **`@cloudzero how do I get started with CloudZero?`**
- **List cost insights** – Get a list of cost insights or find out more about a specific insight.
  - **`@cloudzero list my top cost insights`**

- **`@cloudzero tell me more about insight <insight ID>`**

- **Get billing information** – Ask the Amazon Q CloudZero plugin about your AWS billing information.
  - **`@cloudzero what were my AWS costs for December 2024?`**

Configuring the Amazon Q Developer Datadog plugin

Datadog is a monitoring and security platform that provides infrastructure, application, and network monitoring and analytics. If you use Datadog to monitor your AWS applications, you can use the Datadog plugin in Amazon Q Developer chat to access monitoring information without leaving the AWS Management Console.

You can use the Datadog plugin to learn about Datadog, understand how it works with AWS services, and ask about your Datadog cases and monitors. After you receive a response, you can ask follow up questions, including how to address an issue or for details about Datadog resources.

To configure the plugin, you provide authentication credentials from your Datadog account to enable a connection between Amazon Q and Datadog. After you configure the plugin, you can access Datadog metrics by adding **`@datadog`** to the beginning of your question in Amazon Q chat.

> **Warning**
>
> Datadog user permissions are not detected by the Datadog plugin in Amazon Q. When an administrator configures the Datadog plugin in an AWS account, users with plugin permissions in that account have access to any resources in the Datadog account retrievable by the plugin.
> You can configure IAM policies to restrict which plugins users have access to. For more information, see Configure user permissions.

Prerequisites

Add permissions

To configure plugins, the following administrator level permissions are required:

- Permissions to access the Amazon Q Developer console. For an example IAM policy that grants needed permissions, see Allow administrators to use the Amazon Q Developer console.

- Permissions to configure plugins. For an example IAM policy that grants the needed permissions, see Allow administrators to configure plugins.

Acquire credentials

Before you begin, note the following information from your Datadog account. These authentication credentials will be stored in an AWS Secrets Manager secret when you configure the plugin.

- **Site parameter** – The Datadog site parameter you use. For example, `us3.datadoghq.com`. For more information, see Getting Started with Datadog Sites in the Datadog documentation.
- **API key and application key** – Access keys that allow Amazon Q to call the Datadog API to access events and metrics. You can find these under **Organization Settings** in your Datadog account. For more information, see API and Application Keys in the Datadog documentation.

Secrets and service roles

AWS Secrets Manager secret

When you configure the plugin, Amazon Q creates a new AWS Secrets Manager secret for you to store Datadog authentication credentials. Alternatively, you can use an existing secret that you create yourself.

If you create a secret yourself, make sure it includes the following credentials and uses the following JSON format:

```
{
   "ApiKey": "<your-api-key>",
   "AppKey": "<your-applicaiton-key>"
}
```

For more information about creating secrets, see Create a secret in the *AWS Secrets Manager User Guide*.

Service roles

To configure the Datadog plugin in Amazon Q Developer, you need to create a service role that gives Amazon Q permission to access your Secrets Manager secret. Amazon Q assumes this role to access the secret where your Datadog credentials are stored.

When you configure the plugin in the AWS console, you have the option to create a new secret or use an existing one. If you create a new secret, the associated service role is created for you. If you use an existing secret and an existing service role, make sure your service role contains the following permissions, and has the following trust policy attached. The service role required depends on your secret encryption method.

If your secret is encrypted with an AWS managed KMS key, the following IAM service role is required:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [
                "arn:aws-eusc:secretsmanager:eusc-de-
east-1:111122223333:secret:secret-id"
            ]
        }
    ]
}
```

If your secret is encrypted with a customer managed AWS KMS key, the following IAM service role is required:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
```

```
            "Resource": "arn:aws-eusc:secretsmanager:eusc-de-
east-1:111122223333:secret:secret-id"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "arn:aws-eusc:kms:eusc-de-east-1:111122223333:key/key-
id",
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": "secretsmanager.eusc-de-
east-1.amazonaws.com"
                }
            }
        }
    ]
}
```

To allow Amazon Q to assume the service role, the service role needs the following trust policy:

> **Note**
>
> The `codewhisperer` prefix is a legacy name from a service that merged with Amazon Q Developer. For more information, see Amazon Q Developer rename - Summary of changes.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "q.amazonaws.com"
      },
      "Action": ["sts:AssumeRole", "sts:SetContext"],
      "Condition": {
        "StringEquals": {
```

```
          "aws:SourceAccount": "111122223333",
          "aws:SourceArn": "arn:aws-eusc:codewhisperer:eusc-de-
east-1:111122223333:profile/profile-id"
        }
      }
    }
  ]
}
```

For more information about service roles, see Create a role to delegate permissions to an AWS service in the *AWS Identity and Access Management User Guide*.

Configure the Datadog plugin

You configure plugins in the Amazon Q Developer console. Amazon Q uses credentials stored in AWS Secrets Manager to enable interactions with Datadog.

To configure the Datadog plugin, complete the following procedure:

1. Open the Amazon Q Developer console at https://console.aws.amazon.com/amazonq/developer/home
2. On the Amazon Q Developer console home page, choose **Settings**.
3. In the navigation bar, choose **Plugins**.
4. On the plugins page, choose the plus sign on the **Datadog** panel. The plugin configuration page opens.
5. For **Site URL**, enter the URL of the Datadog site you use.
6. For **Configure AWS Secrets Manager**, choose either **Create a new secret** or **Use an existing secret**. The Secrets Manager secret is where your Datadog authentication credentials will be stored.

   If you create a new secret, enter the following information:

   a. For **Datadog API key**, enter the API key for your Datadog organization.

   b. For **Datadog application key**, enter the application key for your Datadog account.

   c. A service role will be created that Amazon Q will use to access the secret where your Datadog credentials are stored. Do not edit the service role that is created for you.

If you use an existing secret, choose a secret from the **AWS Secrets Manager secret** dropdown menu. The secret should include the Datadog authentication credentials specified in the previous step.

For more information about the required credentials, see Acquire credentials.

7. For **Configure AWS IAM service role**, choose either **Create new service role** or **Use existing service role**.

> **Note**
>
> If you chose **Create a new secret** for step 6, you can't use an existing service role. A new role will be created for you.

If you create a new service role, a service role will be created that Amazon Q will use to access the secret where your Datadog credentials are stored. Do not edit the service role that is created for you.

If you use an existing service role, choose a role from the dropdown menu that appears. Make sure your service role has the permissions and trust policy defined in Service roles.

8. Choose **Save configuration**.
9. After the Datadog plugin panel appears in the **Configured plugins** section on the Plugins page, users will have access to the plugin.

If you want to update the credentials for a plugin, you must delete your current plugin and configure a new one. Deleting a plugin removes all previous specifications. Any time you configure a new plugin, a new plugin ARN is generated.

Configure user permissions

To use plugins, the following permissions are required:

- Permissions to chat with Amazon Q in the console. For an example IAM policy that grants permissions needed to chat, see Allow users to chat with Amazon Q.
- The `q:UsePlugin` permission.

When you grant an IAM identity access to a configured Datadog plugin, the identity gains access to any resources in the Datadog account retrievable by the plugin. Datadog user permissions are not detected by the plugin. If you want to control access to a plugin, you can do so by specifying the plugin ARN in an IAM policy.

Each time you create or delete and re-configure a plugin, it is assigned a new ARN. If you use a plugin ARN in a policy, it will need to be updated if you want to grant access to the newly configured plugin.

To locate the Datadog plugin ARN, go to the **Plugins** page in the Amazon Q Developer console and choose the configured Datadog plugin. On the plugin details page, copy the plugin ARN. You can add this ARN to a policy to allow or deny access to the Datadog plugin.

If you create a policy to control access to Datadog plugins, specify `Datadog` for the plugin provider in the policy.

For examples of IAM policies that control plugin access, see Allow users to chat with plugins from one provider.

Chat with the Datadog plugin

To use the Datadog plugin, enter **`@datadog`** at the beginning of a question about Datadog or your AWS application monitors and cases. Follow up questions or responses to questions from Amazon Q must also include **`@datadog`**.

Following are some example use cases and associated questions you can ask to get the most of out of the Amazon Q Datadog plugin:

- **Learn about using Datadog features in your AWS workload** – Ask about how Datadog features work with certain AWS services. Amazon Q might ask you for more information about what you're trying to do to provide the best answer.
  - **`@datadog how do I use APM on EC2?`**
- **Retrieve and summarize cases and monitors** – Ask about a specific case or monitor, or specify properties to get information about monitors and cases like create date, status, or author. For more information about properties, see Properties in the Datadog documentation.
  - **`@datadog summarize the global outage case`**
  - **`@datadog summarize my top cases`**

- **Check monitors that are in an alarm state** – Ask the Amazon Q Datadog plugin to find your AWS application monitors that are in alarm. You can follow up with questions about the monitors it lists.
  - `@datadog what monitors are in alarm?`
  - `@datadog what is the status for monitor <monitor ID>?`

Configuring the Amazon Q Developer Wiz plugin

Wiz is a cloud security platform that provides security posture management, risk assessment and prioritization, and vulnerability management. If you use Wiz to evaluate and monitor your AWS applications, you can use the plugin in Amazon Q chat to access insights from Wiz without leaving the AWS Management Console.

You can use the plugin to identify and retrieve Wiz issues, assess your riskiest assets, and understand vulnerabilities or exposures. After you receive a response, you can ask follow up questions, including how to remediate an issue.

To configure the plugin, you provide authentication credentials from your Wiz account to enable a connection between Amazon Q and Wiz. After you configure the plugin, you can access Wiz metrics by adding `@wiz` to the beginning of your question in Amazon Q chat.

> **Warning**
>
> Wiz user permissions are not detected by the Wiz plugin in Amazon Q. When an administrator configures the Wiz plugin in an AWS account, users with plugin permissions in that account have access to any resources in the Wiz account retrievable by the plugin. You can configure IAM policies to restrict which plugins users have access to. For more information, see Configure user permissions.

Prerequisites

Add permissions

To configure plugins, the following administrator level permissions are required:

- Permissions to access the Amazon Q Developer console. For an example IAM policy that grants needed permissions, see Allow administrators to use the Amazon Q Developer console.

- Permissions to configure plugins. For an example IAM policy that grants the needed permissions, see Allow administrators to configure plugins.

Acquire credentials

Before you begin, note the following information from your Wiz account. These authentication credentials will be stored in an AWS Secrets Manager secret when you configure the plugin.

- **API endpoint URL** – The URL where you access Wiz. For example, `https://api.us1.app.Wiz.io/graphql`. For more information, see API endpoint URL in the Wiz documentation.
- **Client ID and Client secret** – Credentials that allow Amazon Q to call Wiz APIs to access your application. For more information, see Client ID and Client secret in the Wiz documentation.

Secrets and service roles

AWS Secrets Manager secret

When you configure the plugin, Amazon Q creates a new AWS Secrets Manager secret for you to store Wiz authentication credentials. Alternatively, you can use an existing secret that you create yourself.

If you create a secret yourself, make sure it includes the following credentials and uses the following JSON format:

```
{
   "ClientId": "<your-client-id>",
   "ClientSecret": "<your-client-secret>"
}
```

For more information about creating secrets, see Create a secret in the *AWS Secrets Manager User Guide*.

Service roles

To configure the Wiz plugin in Amazon Q Developer, you need to create a service role that gives Amazon Q permission to access your Secrets Manager secret. Amazon Q assumes this role to access the secret where your Wiz credentials are stored.

When you configure the plugin in the AWS console, you have the option to create a new secret or use an existing one. If you create a new secret, the associated service role is created for you. If you use an existing secret and an existing service role, make sure your service role contains these permissions, and has the following trust policy attached. The service role required depends on your secret encryption method.

If your secret is encrypted with an AWS managed KMS key, the following IAM service role is required:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [
                "arn:aws-eusc:secretsmanager:eusc-de-
east-1:111122223333:secret:secret-id"
            ]
        }
    ]
}
```

If your secret is encrypted with a customer managed AWS KMS key, the following IAM service role is required:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
```

```
            "Resource": "arn:aws-eusc:secretsmanager:eusc-de-
east-1:111122223333:secret:secret-id"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "arn:aws-eusc:kms:eusc-de-east-1:111122223333:key/key-
id",
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": "secretsmanager.eusc-de-
east-1.amazonaws.com"
                }
            }
        }
    ]
}
```

To allow Amazon Q to assume the service role, the service role needs the following trust policy:

> **Note**
>
> The `codewhisperer` prefix is a legacy name from a service that merged with Amazon Q Developer. For more information, see Amazon Q Developer rename - Summary of changes.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "q.amazonaws.com"
      },
      "Action": ["sts:AssumeRole", "sts:SetContext"],
      "Condition": {
        "StringEquals": {
```

```
          "aws:SourceAccount": "111122223333",
          "aws:SourceArn": "arn:aws-eusc:codewhisperer:eusc-de-
east-1:111122223333:profile/profile-id"
        }
      }
    }
  ]
}
```

For more information about service roles, see Create a role to delegate permissions to an AWS service in the *AWS Identity and Access Management User Guide*.

Configure the Wiz plugin

You configure plugins in the Amazon Q Developer console. Amazon Q uses credentials stored in AWS Secrets Manager to enable interactions with Wiz.

To configure the Wiz plugin, complete the following procedure:

1. Open the Amazon Q Developer console at https://console.aws.amazon.com/amazonq/developer/home
2. On the Amazon Q Developer console home page, choose **Settings**.
3. In the navigation bar, choose **Plugins**.
4. On the plugins page, choose the plus sign on the **Wiz** panel. The plugin configuration page opens.
5. For **API endpoint URL**, enter the URL of API endpoint where you access Wiz.
6. For **Configure AWS Secrets Manager**, choose either **Create a new secret** or **Use an existing secret**. The Secrets Manager secret is where your Wiz authentication credentials will be stored.

   If you create a new secret, enter the following information:

   a. For **Client ID**, enter the Client ID for your Wiz account.

   b. For **Client Secret**, enter the Client Secret for your Wiz account.

   c. A service role will be created that Amazon Q will use to access the secret where your Wiz credentials are stored. Do not edit the service role that is created for you.

If you use an existing secret, choose a secret from the **AWS Secrets Manager secret** dropdown menu. The secret should include the Wiz authentication credentials specified in the previous step.

For more information about the required credentials, see Acquire credentials.

7. For **Configure AWS IAM service role**, choose either **Create new service role** or **Use existing service role**.

> **Note**
>
> If you chose **Create a new secret** for step 6, you can't use an existing service role. A new role will be created for you.

If you create a new service role, a service role will be created that Amazon Q will use to access the secret where your Wiz credentials are stored. Do not edit the service role that is created for you.

If you use an existing service role, choose a role from the dropdown menu that appears. Make sure your service role has the permissions and trust policy defined in Service roles.

8. Choose **Save configuration**.
9. After the Wiz plugin panel appears in the **Configured plugins** section on the Plugins page, users will have access to the plugin.

If you want to update the credentials for a plugin, you must delete your current plugin and configure a new one. Deleting a plugin removes all previous specifications. Any time you configure a new plugin, a new plugin ARN is generated.

Configure user permissions

To use plugins, the following permissions are required:

- Permissions to chat with Amazon Q in the console. For an example IAM policy that grants permissions needed to chat, see Allow users to chat with Amazon Q.
- The `q:UsePlugin` permission.

When you grant an IAM identity access to a configured Wiz plugin, the identity gains access to any resources in the Wiz account retrievable by the plugin. Wiz user permissions are not detected by the plugin. If you want to control access to a plugin, you can do so by specifying the plugin ARN in an IAM policy.

Each time you create or delete and re-configure a plugin, it is assigned a new ARN. If you use a plugin ARN in a policy, it will need to be updated if you want to grant access to the newly configured plugin.

To locate the Wiz plugin ARN, go to the **Plugins** page in the Amazon Q Developer console and choose the configured Wiz plugin. On the plugin details page, copy the plugin ARN. You can add this ARN to a policy to allow or deny access to the Wiz plugin.

If you create a policy to control access to Wiz plugins, specify `Wiz` for the plugin provider in the policy.

For examples of IAM policies that control plugin access, see Allow users to chat with plugins from one provider.

Chat with the Wiz plugin

To use the Amazon Q Wiz plugin, enter **`@Wiz`** at the beginning of a question about your Wiz issues. Follow up questions or responses to questions from Amazon Q must also include **`@Wiz`**.

Following are some example use cases and associated questions you can ask to get the most of out of the Amazon Q Wiz plugin:

- **View issues with critical severity** – Ask the Amazon Q Wiz plugin to list your issues with critical or high severity. The plugin can return up to 10 issues. You can also ask to list up to the top 10 most severe issues.
  - **`@wiz what are my critical severity issues?`**
  - **`@wiz can you specify the top 5?`**
- **List issues based on date or status** – Ask to list issues based on create date, due date, or resolved date. You can also specify issues based on properties like status, severity, and type.
  - **`@wiz which issues are due before <date>?`**
  - **`@wiz what are my issues that have been resolved since <date>?`**
- **Assess issues with security vulnerabilities** – Ask about the vulnerabilities or exposures that are posing security threats in your issues.

- **`@wiz which issues are associated with vulnerabilities or external exposures?`**

Automating AWS services with Amazon Q Developer Console-to-Code

What is Console-to-Code?

Console-to-Code is a feature of Amazon Q Developer that can help you write code to automate your use of other AWS services. Console-to-Code records your console actions, then uses generative AI to suggest the equivalent AWS CLI commands and code in your preferred language and format.

Tiers of service

Since Console-to-Code is a part of Amazon Q Developer, your use of it is subject to Amazon Q Developer's tiers of service.

- At the Free tier, there is no fixed monthly limit to the number of times you can record your console actions and generate CLI commands based on those actions. However, there is a limit to how many times per month you can generate code to use with the AWS CDK or AWS CloudFormation based on your recorded actions.

  To access the Free tier, sign into the AWS Management Console. After you reach the monthly code generations limit, you must authenticate to the Pro tier in order to generate more code.
- At the Pro tier, there is no fixed monthly limit to the number of times you can generate code for the AWS CDK or CloudFormation.

  To access the Pro tier, you must be a user registered with IAM Identity Center, and your IAM Identity Center identity must be subscribed to Amazon Q Developer Pro. For more information, see Authenticating to your Amazon Q Developer Pro subscription or contact your AWS administrator.

For more information on pricing tiers, visit the Amazon Q Developer pricing page.

> **Note**
>
> When you record an action, you will still be charged for the action itself, if applicable. For example, if you record yourself provisioning an Amazon EC2 instance, then you will still be charged for the instance. There is no additional cost for recording the action.

Supported code formats

Console-to-Code can currently generate infrastructure-as-code (IaC) in the following languages and formats:

- CDK Java
- CDK Python
- CDK TypeScript
- CloudFormation JSON
- CloudFormation YAML

Where can you use Console-to-Code?

Using Console-to-Code across multiple services

Console-to-Code works across multiple services, saving its own state for as long as your browser tab is open.

For example, you may record your actions during a complete setup of a web server:

- In the Amazon VPC console, you provision two subnets (one public and one private), security groups, NACLs, a custom routing table, and an internet gateway.
- In the Amazon EC2 console, you provision an Amazon EC2 instance and place it in the public subnet.
- In the Amazon RDS console, you provision an Amazon RDS DB instance and place it in the private subnet.

Even if you perform your actions in different parts of the console and they use different AWS services, Console-to-Code can include them in a single recording.

AWS services that support Console-to-Code

Currently, Console-to-Code is available to record your actions when using the AWS management console with the following services:

- Amazon EC2
- Amazon VPC
- Amazon DynamoDB
- Amazon Cognito
- AWS IoT
- Amazon RDS
- Amazon CloudWatch
- AWS Batch
- Amazon Simple Queue Service (Amazon SQS)
- AWS CloudFormation
- EC2 Image Builder
- AWS Systems Manager
- Amazon Athena
- Amazon EventBridge
- Amazon Elastic Container Registry (Amazon ECR)
- Amazon Simple Notification Service (Amazon SNS)
- AWS Secrets Manager
- Amazon ElastiCache
- AWS Step Functions
- Amazon Elastic Kubernetes Service (Amazon EKS)
- AWS Lambda
- Amazon Elastic Container Service (Amazon ECS)
- Amazon API Gateway
- Amazon S3
- Amazon CloudFront

- AWS WAF
- AWS Shield
- AWS Firewall Manager
- Amazon Bedrock
- AWS CodeCommit
- AWS CodeDeploy
- AWS CodeBuild

Granting permissions to use Console-to-Code

To use Console-to-Code, the following permissions are required:

- `q:GenerateCodeFromCommands` to use Console-to-Code. For an example IAM policy that grants the needed permission, see Allow users to generate code from CLI commands with Amazon Q.
- Permissions to take the actions that you're going to record.

Using Console-to-Code

Using Console-to-Code consists of three steps.

Step 1: Start recording

To start recording with Console-to-Code, use the following procedure.

1. Go to the console of one of the integrated services (Amazon VPC, Amazon RDS, or Amazon EC2).
2. Choose the Console-to-Code icon: 
3. In the Console-to-Code side panel, choose **Start recording**.

Step 2: Take actions

In the consoles of any of the integrated services, proceed to take any actions that you want to record.

The Console-to-Code side panel retains its own state. You can move between the consoles of the integrated services, creating one recording that involves actions for multiple services.

The Console-to-Code side panel will retain your actions until your Console-to-Code session ends. The session will end when you close the browser tab, or when your AWS Management Console session ends, whichever comes first.

When you have finished taking actions that you want to convert to code, choose **Stop** from the top of the Console-to-Code panel.

Step 3: Gather CLI commands and generating code

You can follow either Step 3a or Step 3b.

Step 3a: Gather CLI commands

To use Console-to-Code to generate CLI commands based on your actions, use the following procedure.

1. In the Console-to-Code panel, review your recorded actions.

   You can filter the recorded actions using the dropdown, search box, or filter widget in the Console-to-Code panel.
2. Do one of the following:

   - To copy an individual CLI command, choose the copy button to the left of the command.
   - To run an individual CLI command in AWS CloudShell, choose the CloudShell icon  to the left of the command. This opens CloudShell and populates it with the CLI command ready for you to execute.
   - To view or run a set of CLI commands, select the commands and choose either **Copy CLI** to copy all selected commands, or **Run CLI** to open CloudShell and populate it with all commands.

To learn more about the AWS CLI, see What is the AWS Command Line Interface? in the *AWS Command Line Interface User Guide.*

Step 3b: Generate code

1. In the Console-to-Code panel, review your recorded actions. You can filter the recorded actions using the dropdown, search box, or filter widget in the Console-to-Code panel.

2. Select the actions that you want to convert into code. Only the actions with checked boxes will be used in the following steps.
3. Indicate the type of code that you want to generate. From the language menu in the Console-to-Code panel, select the language and (if applicable) format of the code to be generated.
4. Choose **Generate chosen language**.

   The generated code will appear, along with the equivalent CLI commands.

Diagnosing common errors in the console with Amazon Q Developer

In the AWS Management Console, Amazon Q Developer diagnoses common errors you encounter while working with AWS services, such as insufficient permissions, incorrect configuration, and exceeding service limits. Amazon Q troubleshoots errors you receive while using the following services in the AWS console:

- Amazon Elastic Compute Cloud (Amazon EC2)
- Amazon Elastic Container Service (Amazon ECS)
- Amazon Simple Storage Service (Amazon S3)
- AWS Lambda
- AWS Step Functions

In addition, Amazon Q troubleshoots IAM permission errors across all AWS console pages and a limited number of service-specific errors for some AWS services. Amazon Q doesn't maintain a history of previous error diagnosing sessions.

If you're unable to diagnose your error with Amazon Q, you can use Amazon Q to create a support case with Support. For more information, see Using Amazon Q Developer to chat with Support. If you have an issue specific to the Amazon Q error diagnosing feature, you can use the thumbs-down icon to report an issue.

Add permissions

For an IAM policy that grants permissions needed for diagnosing console errors, see Allow users to diagnose console errors with Amazon Q.

Diagnose common errors in the console

To use Amazon Q to diagnose an error in the AWS Management Console, use the following procedure.

1. If you receive an error that Amazon Q can help you with, a **Diagnose with Amazon Q** button appears in the error message. If you want to use Amazon Q to diagnose the error, choose **Diagnose with Amazon Q** to proceed.
2. A window appears where Amazon Q first provides information about the error. It then provides a series of steps you can take to resolve the error. It can take several seconds for Amazon Q to generate instructions.
3. To provide feedback, you can use the thumbs-up and thumbs-down icons. To provide detailed feedback, choose the **Tell me more** button that appears after you select an icon.

Using Amazon Q Developer to chat with Support

You can use Amazon Q Developer to create a support case and contact Support from anywhere in the AWS Management Console, including the AWS Support Center Console. Amazon Q uses the context of your conversation to draft a support case on your behalf automatically. It also adds your recent conversation to the support case description. After creating the case, Amazon Q can transfer you to a support agent in the method of your choice, including live chat in the same interface.

When you create a support case in Amazon Q, the case is also updated in the Support Center Console. To track updates on cases created with Amazon Q, use the Support Center Console.

The type of Support available to you depends on the support plan for your AWS account. All AWS users have access to account and billing support as part of the Basic Support plan. For technical support questions, only users with support plans other than the Basic Support plan can contact Support with Amazon Q. For more information about AWS Support, see Getting started with AWS Support in the *AWS Support User Guide*.

> **Tip**
>
> Before you create a support ticket, try asking Amazon Q to resolve the issue. For more information, see Asking Amazon Q to troubleshoot your resources. You can also try the **Diagnose with Amazon Q** button, if it's available. For more information, see Diagnosing console errors.

Prerequisites

To create cases in Amazon Q, you must meet the following requirements:

- You have a support plan higher than the Basic Support plan. Only users with support plans other than the Basic Support plan can contact Support with Amazon Q.
- You have permissions to chat with Amazon Q. For more information, see Allow users to chat with Amazon Q.
- You have permissions to create Support cases. For more information, see Manage access to Support Center.

Specify the right service

When you create a support case with Amazon Q, it populates the service field based on your question. If Amazon Q chooses the wrong service, update the case with the correct service. If your question has to do with multiple services, specify the service that's most applicable.

To contact Support about an Amazon Q feature that is part of another AWS service, create a support case for the other AWS service, not for Amazon Q. For example, if you're using Amazon Q network troubleshooting in Amazon VPC Reachability Analyzer, choose Amazon VPC for the service in the support case.

To contact Support about features in either Amazon Q Developer or Amazon Q Business, create a support case for Amazon Q.

Create a support case

To create an Support case with Amazon Q, use the following steps.

1. You can create an Support case through Amazon Q in one of two ways:

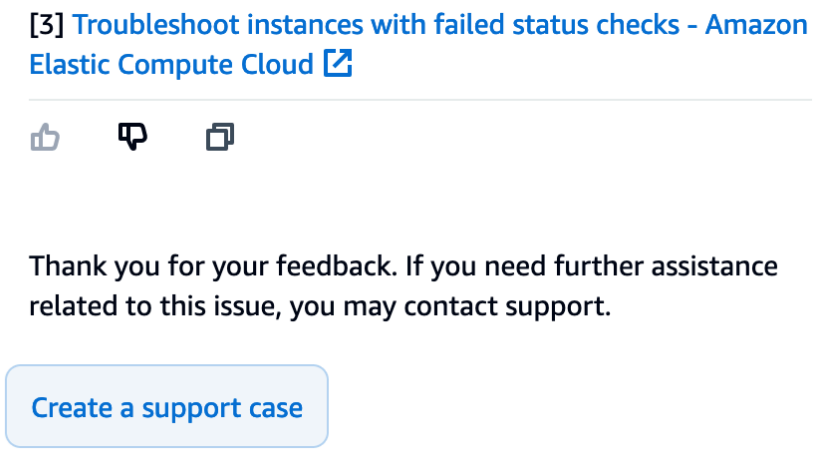
    a. Ask for help directly by entering a question such as “I want to speak to someone” or “Get support”.

    To provide more context for Amazon Q to create the support case, you can add more information when requesting support directly. Following is an example of providing more information in a request:

"I am unable to connect to my bastion instance. I have tried restarting it and generating new key pairs but still nothing works. This started this morning after a planned deployment. I can confirm that no other network related changes were made. Can I talk to someone?"

b. If an Amazon Q response didn't help you, choose the thumbs-down icon on the response and then choose a reason that you're providing the feedback. To contact Support, choose **Create a support case**.

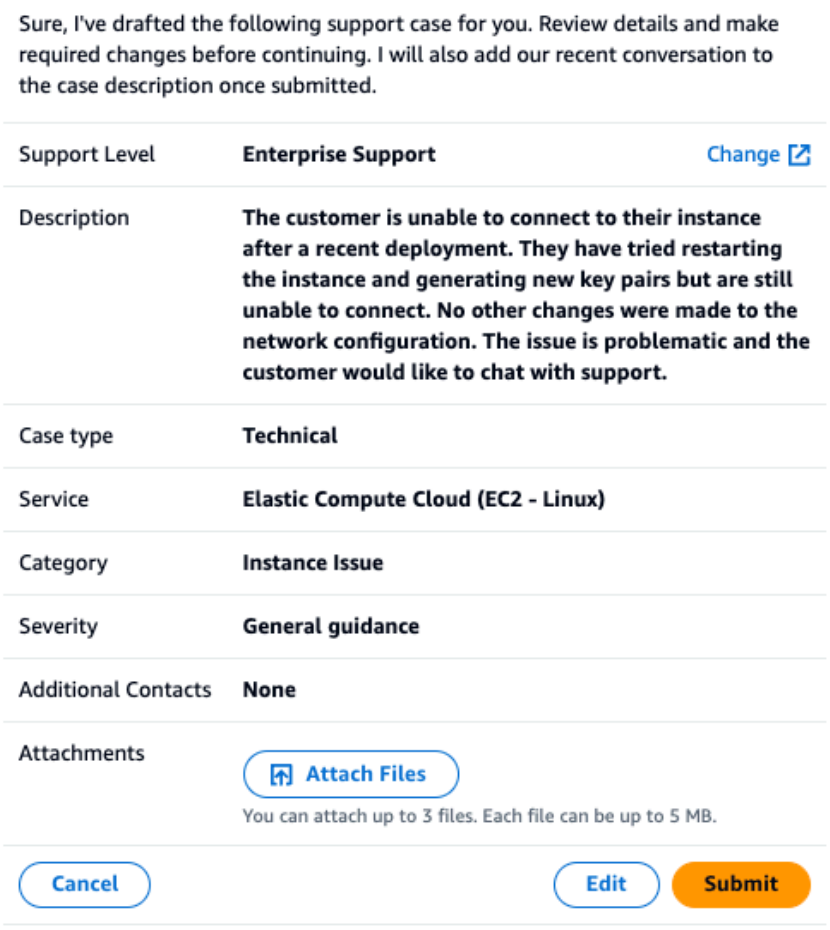
The following image shows the **Create a support case** button in the Amazon Q chat panel that appears after you leave feedback.



2. A support case appears in the chat panel. If you had a conversation with Amazon Q before requesting support, it will use the context of your conversation to autopopulate the fields in the case. To update any field in the support case, choose **Edit**. You can also attach files that help explain your issue.

   If you didn't chat with Amazon Q before requesting support or Amazon Q otherwise can't complete the fields in the support case, you can input your support case information into the case manually.

   The following image is an example of a filled-out support case in the Amazon Q chat panel.



3. After confirming that the support case describes your needs, choose **Submit** to create the support case. If you no longer want to create the case, choose **Cancel**.
4. To contact Support, choose the method that you want to use. Depending on your case details, you can chat, email, or request a phone call from a live support agent:

   a. **Chat** – If you choose to chat with an agent, a live support agent will enter the conversation. To end the chat with the support agent, choose **End this chat** at any time during the chat.

      If you refresh your page, navigate to a different console, or get signed out of the console because of session expiration, the conversation will end.

      If you minimize the chat panel or leave the page, you might miss notifications and be disconnected because of inactivity. We recommend that you keep the chat panel open throughout the duration of your support chat.

   b. **Email** – If you choose to send an email message to an agent, a support agent will contact you at the email address that's associated with your AWS account.

c. **Call** – If you choose to call an agent, enter your phone number when prompted, and choose **Submit**. You will be added to the call queue.

5. You can leave feedback or choose **Skip** to return to the Amazon Q chat panel.

Leave feedback

After the support chat has ended, you can optionally leave feedback.

Rate your experience, enter any additional feedback, and then choose **Submit feedback**.

Using Amazon Q Developer in the IDE

End of support notice

On April 30, 2027, AWS will discontinue support for Amazon Q Developer IDE plugins. For capabilities similar to Amazon Q Developer IDE plugins, explore Kiro to access the latest models and features, including agentic coding, chat and MCP support. For more information, see Amazon Q Developer IDE plugins end of support.

Use Amazon Q Developer in integrated development environments (IDEs) to learn about AWS and get assistance with your software development needs. In IDEs, Amazon Q includes capabilities to provide guidance and support across various aspects of software development, such as answering questions about building on AWS, generating and updating code, security scanning, and optimizing and refactoring code.

To install Amazon Q in your IDE, see Installing the Amazon Q Developer extension or plugin in your IDE .

Topics

- Supported IDEs and available features
- Installing the Amazon Q Developer extension or plugin in your IDE
- Chatting with Amazon Q Developer about code
- Generating inline suggestions with Amazon Q Developer
- Supported languages for Amazon Q Developer in the IDE

Supported IDEs and available features

The features you have access to depend on the IDE where you use Amazon Q. The following table describes the IDEs supported by Amazon Q and the availability and limitations of features in each IDE.

If no language support is specified, the IDE supports languages listed in the Supported languages topic.

Feature	VSCode	JetBrains	Eclipse	Visual Studio
Chat	Yes	Yes	Yes	Yes
Agentic coding	Yes	Yes	Yes	Yes
MCP servers	Yes	Yes	Yes	Yes
Context in chat	Yes	Yes	Yes	Yes
Inline chat	Yes	Yes	Yes	No
Workspace context in chat	Yes	Yes	Yes	Yes
Inline suggestions	Yes	Yes	Yes	Yes
Transformations	Yes	Yes	No	Yes
	Yes	Yes	Yes	No

You can also generate inline suggestions in AWS coding environments. For more information, see Generating inline suggestions in AWS coding environments .

Installing the Amazon Q Developer extension or plugin in your IDE

End of support notice

On April 30, 2027, AWS will discontinue support for Amazon Q Developer IDE plugins. For capabilities similar to Amazon Q Developer IDE plugins, explore Kiro to access the latest models and features, including agentic coding, chat and MCP support. For more information, see Amazon Q Developer IDE plugins end of support.

To set up Amazon Q Developer in your integrated development environment (IDE), complete the following steps. After installing the Amazon Q extension or plugin, authenticate through IAM

Identity Center or AWS Builder ID. You can use Amazon Q for free, without an AWS account, by authenticating with Builder ID.

To get started, download the Amazon Q extension or plugin for your IDE:

- Download Amazon Q for Eclipse
- Download Amazon Q for Visual Studio Code
- Download Amazon Q for JetBrains IDEs
- Download Amazon Q in the AWS Toolkit for Visual Studio

> **Note**
>
> In general, the default duration for a session that is authenticated through IAM Identity Center is 8 hours. However, in the case of Amazon Q, the default session lasts 90 days (if you set up IAM Identity Center on April 18, 2024 or later). For more information refer to How to extend the session duration for Amazon Q in the IDE in the *AWS IAM Identity Center User Guide*

To sign in and authenticate, complete the steps in this section.

Steps

- Prerequisite: Supported IDE versions
- Authenticating in Eclipse IDEs
- Authenticating in JetBrains IDEs
- Authenticating in Visual Studio Code
- Authenticating in Visual Studio
- Using an IAM principal in your AWS console

Prerequisite: Supported IDE versions

- The minimum version of Eclipse supported by Amazon Q is 2024-06 (4.32).
- The minimum version of JetBrains IDEs (including IntelliJ and PyCharm) supported by Amazon Q is 2024.3.

- The minimum version of Visual Studio Code supported by Amazon Q is 1.85.0.
- Only Visual Studio for Windows is supported by Amazon Q. The minimum version of Visual Studio supported is Visual Studio 2022 version 17.7. All Visual Studio 2022 editions are supported.

Authenticating in Eclipse IDEs

You can authenticate for free with AWS Builder ID or with IAM Identity Center with a Amazon Q Developer Pro subscription. Choose your authentication method to see steps to start using Amazon Q in Eclipse.

Builder ID

This procedure does not require you to have Builder ID. If you have not yet signed up for Builder ID, you will have the opportunity to do so during the sign-in process.

1. Install the Amazon Q plugin in Eclipse.
2. Choose the Amazon Q icon in the IDE toolbar.
3. An Amazon Q tab opens in the IDE. Under **Choose a sign-in option**, choose **Personal account**, and then choose **Continue**. You are redirected to your browser.
4. Follow the instructions in your browser to authenticate with Builder ID. When you've completed authentication, return to the Eclipse IDE.
5. To begin using Amazon Q, choose the Amazon Q icon to open the chat Amazon Q panel.

IAM Identity Center

Before you begin this procedure, your administrator should have:

- Created an identity for you in IAM Identity Center
- Subscribed that identity to Amazon Q Developer Pro

After your identity has been subscribed to Amazon Q Developer Pro, complete the following steps to authenticate:

1. Install the Amazon Q plugin in Eclipse.
2. Choose the Amazon Q icon in the IDE toolbar.

3. An Amazon Q tab opens in the IDE. Under **Choose a sign-in option**, choose **Company account**, and then choose **Continue**.
4. Enter the **Start URL** that your administrator got from the Amazon Q subscription console.
5. Choose the AWS Region in which your administrator set up your IAM Identity Center instance.
6. Choose **Continue**. You are redirected to your browser.
7. Follow the instructions in your browser to authenticate with IAM Identity Center. When you've completed authentication, return to the Eclipse IDE.
8. To begin using Amazon Q, choose the Amazon Q icon to open the chat Amazon Q panel.

Authenticating in JetBrains IDEs

You can authenticate for free with AWS Builder ID or with IAM Identity Center with a Amazon Q Developer Pro subscription. Choose your authentication method to see steps to start using Amazon Q in your JetBrains IDE.

Builder ID

This procedure does not require you to have Builder ID. If you have not yet signed up for Builder ID, you will have the opportunity to do so during the sign-in process.

1. Install the Amazon Q plugin in your JetBrains IDE.
2. Choose the Amazon Q icon in your IDE.

   The icon will be on the side of the interface by default.
3. Follow the instructions in your browser to authenticate with Builder ID.
4. To begin using Amazon Q, choose the Amazon Q icon to chat with Amazon Q, or choose **Amazon Q** from the IDE navigation bar.

IAM Identity Center

Before you begin this procedure, your administrator should have:

- Created an identity for you in IAM Identity Center
- Subscribed that identity to Amazon Q Developer Pro

After your identity has been subscribed to Amazon Q Developer Pro, complete the following steps to authenticate:

1. Install the Amazon Q plugin in your JetBrains IDE.
2. Choose the Amazon Q icon in your IDE.

   The icon will be on the side of the interface by default.
3. Choose **Company account**.
4. Provide the **Start URL** that your administrator got from the Amazon Q subscription console.
5. Provide the AWS Region in which your administrator set up your IAM Identity Center instance.
6. Choose **Continue**. The focus will switch to your web browser.
7. Follow the instructions in your browser to authenticate with IAM Identity Center, and then return to the IDE.
8. If your administrator has configured more than one Amazon Q Developer profile, you will see the profiles you have access to. Choose the profile that meets your current working needs, or that your administrator has instructed you to use. For more information about profiles, see What is the Amazon Q Developer profile?.

   If there is only one profile available, that profile will automatically be chosen and you can begin using Amazon Q.

   To change your Amazon Q Developer profile, choose **Amazon Q** from the IDE navigation bar, and then choose **Change profile**. From the window that appears, choose the profile you'd like to use.
9. To begin using Amazon Q, choose the Amazon Q icon to chat with Amazon Q, or choose **Amazon Q** from the IDE navigation bar.

Authenticating in Visual Studio Code

You can authenticate for free with AWS Builder ID or with IAM Identity Center with a Amazon Q Developer Pro subscription. Choose your authentication method to see steps to start using Amazon Q in VS Code.

Builder ID

This procedure does not require you to have Builder ID. If you have not yet signed up for Builder ID, you will have the opportunity to do so during the sign-in process.

1. Install the Amazon Q extension in VS Code.
2. Choose the Amazon Q icon in your IDE.

   The icon will be on the side of the interface by default.
3. Follow the instructions in your browser to authenticate with Builder ID.
4. To begin using Amazon Q, choose the Amazon Q icon to chat with Amazon Q, or choose **Amazon Q** from the IDE navigation bar.

IAM Identity Center

Before you begin this procedure, your administrator should have:

- Created an identity for you in IAM Identity Center
- Subscribed that identity to Amazon Q Developer Pro

After your identity has been subscribed to Amazon Q Developer Pro, complete the following steps to authenticate:

1. Install the Amazon Q extension in VS Code.
2. Choose the Amazon Q icon in your IDE.

   The icon will be on the side of the interface by default.
3. Choose **Company account**.
4. Provide the **Start URL** that your administrator got from the Amazon Q subscription console.
5. Provide the AWS Region in which your administrator set up your IAM Identity Center instance.
6. Choose **Continue**. The focus will switch to your web browser.
7. Follow the instructions in your browser to authenticate with IAM Identity Center, and then return to the IDE.

8. If your administrator has configured more than one Amazon Q Developer profile, you will see the profiles you have access to. Choose the profile that meets your current working needs, or that your administrator has instructed you to use. For more information about profiles, see What is the Amazon Q Developer profile?.

   If there is only one profile available, that profile will automatically be chosen and you can begin using Amazon Q.

   To change your Amazon Q Developer profile, choose **Amazon Q** from the IDE navigation bar, and then choose **Change profile**. From the command palette, choose the profile you'd like to use.
9. To begin using Amazon Q, choose the Amazon Q icon to chat with Amazon Q, or choose **Amazon Q** from the IDE navigation bar.

Authenticating in Visual Studio

To connect to your AWS accounts from the Toolkit for Visual Studio, open the **Getting Started with the AWS Toolkit** User Interface (connection UI) by completing the following procedure.

1. From the Visual Studio main menu, expand **Extensions** then expand the **AWS Toolkit**.
2. From the **AWS Toolkit** menu options choose **Getting Started**.
3. The **Getting Started with the AWS Toolkit** connection UI opens in Visual Studio.

You can authenticate for free with AWS Builder ID or with IAM Identity Center with a Amazon Q Developer Pro subscription. Choose your authentication method to see steps to start using Amazon Q in Visual Studio.

Builder ID

1. From Visual Studio, expand **Extensions** from the main menu and then expand the **AWS Toolkit** sub-menu.
2. Choose **Getting started**. The **Getting Started** tab opens in the Visual Studio editor window.
3. In the **Amazon Q** section, choose **Enable**.
4. From the **Free Tier** section, choose the **Sign up or Sign in** button.
5. Confirm that you want to open the AWS Authorize request portal in your default web browser.

6. Follow the prompts in your default web browser. You're notified when the authorization process is complete, and it's safe to close your browser and return to Visual Studio.

IAM Identity Center

1. From Visual Studio, expand **Extensions** from the main menu and then expand the **AWS Toolkit** sub-menu.
2. Choose **Getting started**. The **Getting Started** tab opens in the Visual Studio editor window.
3. In the **Amazon Q** section, choose **Enable**. You will fill out the Professional Tier section to authenticate.
4. The credentials profile is made up of the Profile Name, Start URL, Profile Region, or SSO Region provided by an administrator at your company or organization. For detailed information about IAM Identity Center credentials, see What is IAM Identity Center? in the *IAM Identity Center User Guide*.

   If you have an existing credentials profile, choose it from the dropdown menu in the Professional tier panel, and then choose **Connect**.

   To create a new credentials profile, fill out the following fields from the Professional tier section:

   a. In the **Profile Name** text field, enter the name of the IAM Identity Center profile you want to authenticate with.

   b. In the **Start URL** text field, enter the start URL that's attached to your IAM Identity Center credentials.

   c. From the **Profile Region (defaults to us-east-1)** drop-down menu, choose the AWS Region that's defined by the IAM Identity Center user profile you're authenticating with.

   d. From the **SSO Region (defaults to us-east-1)** drop-down menu, choose the **SSO Region** that's defined by your IAM Identity Center credentials, then choose the **Connect** button to open the **Log in with AWS IAM Identity Center** dialog.
5. Confirm that you want to open the AWS Authorize request portal in your default web browser.
6. Follow the prompts in your default web browser. You're notified when the authorization process is complete, and it's safe to close your browser and return to Visual Studio.

7. A **Sign into Amazon Q** window appears. In the credentials profile dropdown, choose the profile you used to authenticate in the previous steps.
8. If your administrator has configured more than one Amazon Q Developer profile, you are then prompted to choose a Q Developer profile from the dropdown menu. Choose the profile that meets your current working needs, or that your administrator has instructed you to use. For more information about profiles, see What is the Amazon Q Developer profile?.

   If there is only one profile available, that profile will automatically be chosen and you can begin using Amazon Q.

   To change your Amazon Q Developer profile, choose **Amazon Q** from the IDE navigation bar, and then choose **Change Q Developer Profile**. From the window that appears, choose the profile you'd like to use.

   You can also change your profile by choosing the overflow menu in the chat panel header, and then choosing **Change Q Developer Profile**.

For more information about authenticating in the Toolkit for Visual Studio, see Getting Started in the *AWS Toolkit for Visual Studio User Guide*.

Using an IAM principal in your AWS console

Depending on how you use AWS, you may be accustomed to using your IAM credentials to sign in to the console for all AWS services. However, you cannot use Amazon Q Developer in the IDE as an IAM principal, or with an IAM role. You must authenticate with credentials from either IAM Identity Center or Builder ID.

Chatting with Amazon Q Developer about code

End of support notice

On April 30, 2027, AWS will discontinue support for Amazon Q Developer IDE plugins. For capabilities similar to Amazon Q Developer IDE plugins, explore Kiro to access the latest models and features, including agentic coding, chat and MCP support. For more information, see Amazon Q Developer IDE plugins end of support.

Chat with Amazon Q Developer in your integrated development environment (IDE) to ask questions about building at AWS and for assistance with software development. Amazon Q can explain coding concepts and code snippets, generate code and unit tests, and improve code, including debugging or refactoring.

Topics

- Working with Amazon Q in your IDE
- Example tasks
- Example questions
- Reporting issues with responses from Amazon Q
- Reviewing code with Amazon Q Developer
- Transforming code in the IDE with Amazon Q Developer
- Explaining and updating code with Amazon Q Developer
- Chatting inline with Amazon Q Developer
- Adding context to Amazon Q Developer chat in the IDE
- Chat history compaction in Amazon Q Developer
- Viewing, deleting, and exporting the Amazon Q Developer conversation history
- Using shortcut keys in chat with Amazon Q Developer
- Selecting a model for Amazon Q chat in IDEs

Working with Amazon Q in your IDE

Using chat

To start chatting with Amazon Q, choose the Amazon Q icon from the navigation bar in your IDE and enter your question in the text bar. To start chatting with Amazon Q in Visual Studio, choose **View** from the main menu and then choose **Amazon Q chat**.

When you ask Amazon Q a question, it uses the current file that is open in your IDE as context, including the programming language and the file path. You can add more context in your prompt or specify files, folders, or your whole workspace as context throughout a chat session. For more information, see Adding context to the chat.

If Amazon Q includes code in its response, you can copy the code or insert it directly into your file by choosing **Insert at cursor**. Amazon Q might include inline references to its sources in its response.

Amazon Q maintains the context of your conversation within a given session inform future responses. You can ask follow up questions or refer to previous questions and responses throughout the duration of your session. To start a new conversation with Amazon Q, open a new tab in the panel. You can open up to 10 tabs at a time. Amazon Q doesn't retain context across different conversations.

Chat commands

The following commands help you manage your chats with Amazon Q.

- **/clear** - Use this command to clear a current conversation. This removes all previous conversation from the chat panel and clears the context that Amazon Q has about your previous conversation.
- **/compact** - Use this command to compact your chat history when the context window approaches its capacity limit. This creates a concise summary of your conversation while preserving essential information.
- **/help** - Use this command to see an overview of what Amazon Q can and can't do, example questions, and available features.

Agentic coding

With agentic coding, Amazon Q acts as your coding partner, chatting with you as you develop. Agentic coding is on by default in the IDE. You can toggle agentic coding on or off with the </> icon in the chat panel.

When you ask Amazon Q to improve your code, it updates your files directly. You can view the changes in a diff and have the option to undo them.

While Amazon Q is thinking or working on a task, you can continue to add instructions in the chat panel, and it will incorporate them into its work.

As you discuss your project with Amazon Q, it will offer suggestions for shell commands. Sometimes, when it deems those commands to be low-risk, it will run them on its own.

Chatting in natural languages

Amazon Q Developer provides multi-language support when you chat in the IDE. Supported natural languages include Mandarin, French, German, Italian, Japanese, Spanish, Korean, Hindi, and Portuguese, with more languages available. To utilize this functionality, you can start a conversation with Amazon Q in the IDE using your preferred natural language. Amazon Q automatically detects the language and provides responses in the appropriate language.

Example tasks

Developing code features

> **Note**
>
> This capability used to be referred to as /dev in this documentation and in the IDE.

Amazon Q can help you develop code features, make code changes to projects, and answer questions about software development tasks in your integrated development environment (IDE). You explain the task you want to accomplish, and Amazon Q uses the context of your current project or workspace to generate code that you can apply to your codebase. Amazon Q can help you build AWS projects or your own applications.

Unit test generation

> **Note**
>
> This capability used to be referred to as /test in this documentation and in the IDE.

Amazon Q can generate unit tests so you can automate testing throughout the software development lifecycle. This feature helps developers focus on accelerating feature development while ensuring code quality.

Documentation generation

> **Note**
>
> This capability used to be referred to as /doc in this documentation and in the IDE.

Amazon Q helps you understand your code and keep documentation up to date by generating READMEs and other documentation for your code. It can produce new documentation and update existing documentation in your codebase.

Code reviews

> **Note**
>
> This capability used to be referred to as /review in this documentation and in the IDE.

Amazon Q can review your codebase for security vulnerabilities and code quality issues to improve the posture of your applications throughout the development cycle. For more information on how to use this feature, see Reviewing code with Amazon Q Developer.

Transforming code

Amazon Q can transform your code in integrated development environments (IDEs) by performing automated language and operating system (OS)-level upgrades and conversions. You provide the code to be transformed, and Amazon Q generates changes that you can review and apply to your files. For more information, see Transforming code.

Example questions

Within IDEs, Amazon Q can answer questions about AWS services and software development, in addition to generating code. Amazon Q is particularly useful for answering questions related to the following subject areas.

- Building on AWS, including AWS service selection, limits, and best practices
- General software development concepts, including programming language syntax and application development
- Writing code, including explaining code, debugging code, and writing unit tests

Following are some example questions that you can ask to get the most out of Amazon Q in your IDE:

- How do I debug issues with my Lambda functions locally before deploying to AWS?

- How do I choose between AWS Lambda and Amazon EC2 for a scalable web application backend?
- What is the syntax of declaring a variable in TypeScript?
- How do I write an app in React?
- Provide me a description of what this [*selected code or application*] does and how it works.
- Generate test cases for [*selected code or function*].

Reporting issues with responses from Amazon Q

You can optionally leave feedback for every response Amazon Q generates by using the thumbs-up and thumbs-down icons. To report an issue with a response, choose the thumbs-down icon, and enter information in the feedback window that appears.

Reviewing code with Amazon Q Developer

End of support notice

On April 30, 2027, AWS will discontinue support for Amazon Q Developer IDE plugins. For capabilities similar to Amazon Q Developer IDE plugins, explore Kiro to access the latest models and features, including agentic coding, chat and MCP support. For more information, see Amazon Q Developer IDE plugins end of support.

Amazon Q Developer can review your codebase for security vulnerabilities and code quality issues to improve the posture of your applications throughout the development cycle. You can review an entire codebase, analyzing all files in your local project or workspace, or review a single file. You can also enable auto reviews that assess your code as you write it.

Reviews are powered by both generative AI and rule-based automatic reasoning. Amazon Q detectors, informed by years of AWS and Amazon.com security best practices, power the rule-based security and quality reviews. As security policies are updated and detectors are added, reviews automatically incorporate new detectors to ensure your code is compliant with the most up-to-date policies.

For information on supported IDEs for this feature, see Supported IDEs. For information on supported languages, see Language support for code reviews.

Topics

- How it works
- Types of code issues
- Quotas
- Starting a code review with Amazon Q Developer
- Addressing code issues with Amazon Q Developer
- Filtering code issues
- Code issue severity in Amazon Q Developer code reviews

How it works

During a code review, Amazon Q assesses both your custom code and third-party libraries in your code. Before starting a code review, Amazon Q applies filtering to ensure that only relevant code is reviewed. As part of the filtering process, Amazon Q excludes unsupported languages, test code, and open source code.

Amazon Q can either review your recent code changes, or an entire file or project. To initiate a review, you open your code folder in your IDE, and then ask Amazon Q to review your code from the chat panel.

By default, if you simply ask Amazon Q to review your code, it will review only the code changes in the active file in your IDE. Code changes are determined by the output of the `git diff` command on your file. If there is no diff file is present, Amazon Q will review the entire code file. If no file is open, it will search for any code changes in the project to review.

Similarly, if you ask Amazon Q to review your entire project or workspace, it will first attempt to review your code changes. If there is no diff file present, it will review your entire codebase.

Types of code issues

Amazon Q reviews your code for the following types of code issues:

- **SAST scanning — Detect security vulnerabilities in your source code.** Amazon Q identifies various security issues, such as resource leaks, SQL injection, and cross-site scripting.
- **Secrets detection — Prevent the exposure of sensitive or confidential information in your code.** Amazon Q reviews your code and text files for secrets such as hardcoded passwords,

database connection strings, and usernames. Secrets findings include information about the unprotected secret and how to protect it.

- **IaC issues — Evaluate the security posture of your infrastructure files.** Amazon Q can review your infrastructure as code (IaC) code files to detect misconfiguration, compliance, and security issues.
- **Code quality issues — Ensure your code is meeting quality, maintainability, and efficiency standards.** Amazon Q generates code issues related to various quality issues, including but not limited to performance, machine learning rules, and AWS best practices.
- **Code deployment risks — Assess risks related to deploying code.** Amazon Q determines if there are any risks to deploying or releasing your code, including application performance and disruption to operations.
- **Software composition analysis (SCA) — Evaluate third-party code.** Amazon Q examines third-party components, libraries, frameworks, and dependencies integrated into your code, ensuring third-party code is secure and up to date.

For a complete list of the detectors Amazon Q uses to review your code, see the Amazon Q Detector Library.

Quotas

Amazon Q security scans maintain the following quotas:

- **Input artifact size** – The maximum size of all the files within an IDE project workspace, including third-party libraries, build JAR files, and temporary files.
- **Source code size** – The maximum size of the source code that Amazon Q scans after filtering all third-party libraries and unsupported files.

The following table describes the quotas maintained for auto scans and full project scans.

Resource	Auto reviews	File or project reviews
Input artifact size	200 KB	500 MB
Source code size	200 KB	50 MB

Starting a code review with Amazon Q Developer

Amazon Q can review your entire file or codebase, or auto-review your code as you write it.

Before you get started, make sure you've installed Amazon Q in an IDE that supports code reviews. For more information, see Installing the Amazon Q Developer extension or plugin in your IDE.

Topics

- Review a file, project, or workspace
- Example tasks and prompts
- Review as you code

Review a file, project, or workspace

You can initiate a review from the chat panel to have Amazon Q review a particular file or project. File and project reviews include both rule-based and generative AI-powered reviews.

After Amazon Q completes a review, you can investigate the issue and get a code fix to remediate the issue. For more information, see Addressing code issues.

To start a file or project review, complete the following steps for your IDE:

JetBrains

1. Open a file or project you want to review in your IDE.
2. Choose the Amazon Q icon to open the chat panel.
3. Using natural language, describe the type of code review you want to run. You can review only your recent code changes, or an entire file. Code changes are determined based on the output of the git diff command on your file. If applicable, Amazon Q will only review your code changes by default unless otherwise specified.
4. With your code project or file open in the IDE, you can enter things like:

    a. **`Review my code changes`** – Amazon Q will review any code changes in your codebase

    b. **`Run a code review on this entire file`** – Amazon Q will review all code in your file, not only changes

    c. **`Review this repository`** – Amazon Q will review your entire codebase, not only changes

For more detailed code review scenarios and associated prompts, see Example prompts.

5. Amazon Q will begin reviewing your file or project. Once complete, it will summarize the highest priority issues and observations.
6. If any issues were detected, the **Code Issues** tab opens with a list of the issues Amazon Q found.
7. To learn more about a code issue, navigate to the **Code Issues** panel. From there, you can do the following:

    a. Select an issue to be redirected to the specific area of the file where the vulnerable or low-quality code was detected.

    b. To get an explanation of the code issue, choose the magnifying glass icon next to the name of the code issue. Amazon Q will provide details about the issue and suggest a remediation that you can insert into your code.

    c. To fix the code issue, choose the wrench icon next to the name of the code issue. Amazon Q will provide a brief explanation of the fix and then make an in-place fix in your code file. You will see the code change in your file, and have the option to undo the change from the chat panel.

    d. You can also use natural language to ask more about an issue, get an explanation of proposed fixes, or ask for alternative solutions.
8. For more information about addressing code issues, see Addressing code issues with Amazon Q Developer.

Visual Studio Code

1. Open a file or project you want to review in your IDE.
2. Choose the Amazon Q icon to open the chat panel.
3. Using natural language, describe the type of code review you want to run. You can review only your recent code changes, or an entire file. Code changes are determined based on the output of the git diff command on your file. If applicable, Amazon Q will only review your code changes by default unless otherwise specified.
4. With your code project or file open in the IDE, you can enter things like:

    a. **`Review my code changes`** – Amazon Q will review any code changes in your codebase

b. **`Run a code review on this entire file`** – Amazon Q will review all code in your file, not only changes

c. **`Review this repository`** – Amazon Q will review your entire codebase, not only changes

For more detailed code review scenarios and associated prompts, see Example prompts.

5. Amazon Q will begin reviewing your file or project. Once complete, it will summarize the highest priority issues and observations.
6. If any issues were detected, the **Code Issues** tab opens with a list of the issues Amazon Q found.
7. To learn more about a code issue, navigate to the **Code Issues** panel. From there, you can do the following:

   a. Select an issue to be redirected to the specific area of the file where the vulnerable or low-quality code was detected.

   b. To get an explanation of the code issue, choose the magnifying glass icon next to the name of the code issue. Amazon Q will provide details about the issue and suggest a remediation that you can insert into your code.

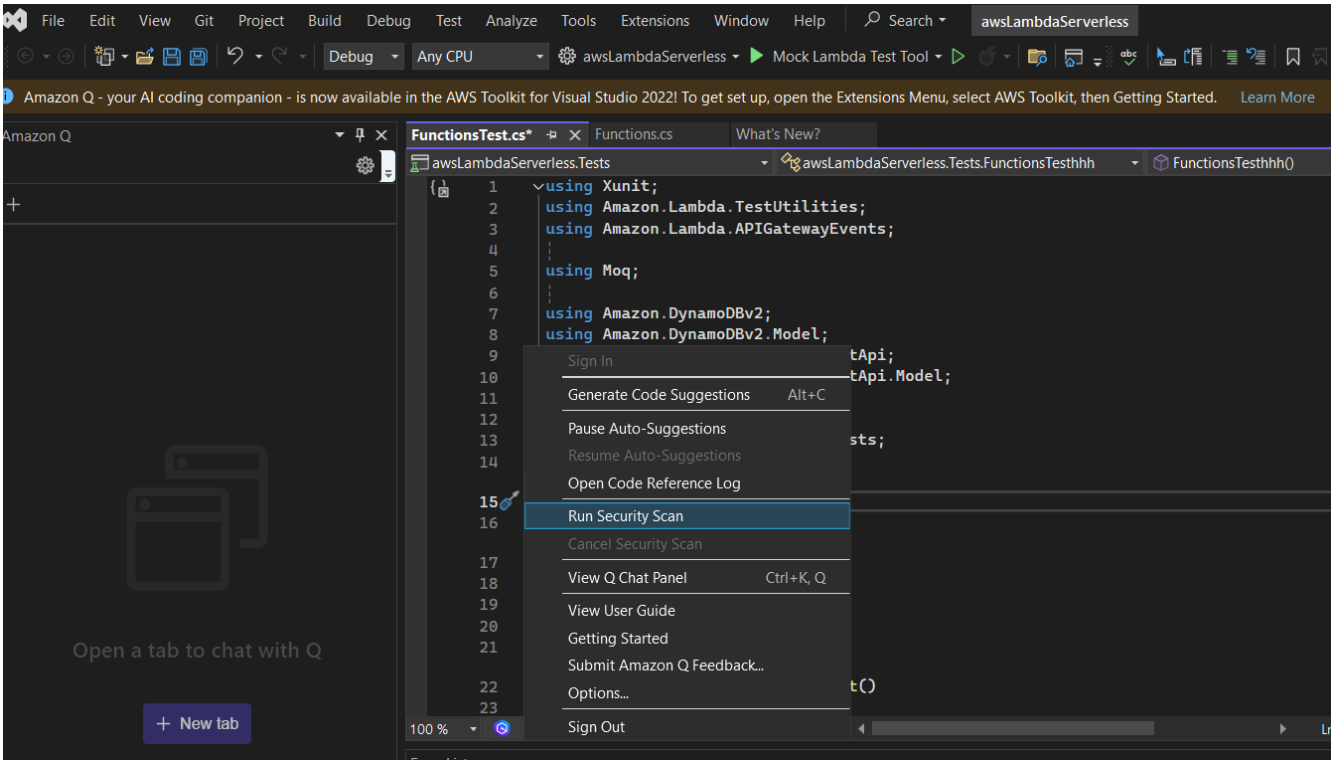
   c. To fix the code issue, choose the wrench icon next to the name of the code issue. Amazon Q will provide a brief explanation of the fix and then make an in-place fix in your code file. You will see the code change in your file, and have the option to undo the change from the chat panel.

   d. You can also use natural language to ask more about an issue, get an explanation of proposed fixes, or ask for alternative solutions.
8. For more information about addressing code issues, see Addressing code issues with Amazon Q Developer.

Visual Studio

1. Open up a file from the project you want to scan in Visual Studio.
2. Choose the Amazon Q icon in the status bar to open the Amazon Q task bar.
3. From the task bar, choose **Run Security Scan**. Amazon Q begins scanning your project.

   In the following image, in Visual Studio, the user chooses the **Amazon Q** icon, prompting a task bar from which the user may choose **Run Security Scan**.



4. The status of your scan is updated in the Visual Studio output pane. You're notified when the scan is complete.

   For information about viewing and addressing findings, see [Addressing code issues with Amazon Q Developer].

Example tasks and prompts

There are several scenarios that you might be in when initiating a code review. Following is an overview of some of the ways to initiate a code review and how to prompt Amazon Q to run the review you want.

- To review just the code changes for a single file:
  - Open the file in your IDE and enter **`Review my code`**
  - Enter **`Review the code in <filename>`**
- To review an entire code file:
  - Open a file without changes and enter **`Review my code`**
  - Open a file with changes and enter **`Review my entire code file`**
  - Enter **`Review all the code in <filename>`**

- To review all code changes in your repository:
  - Open the repository in your IDE and enter **`Review my code`**
- To review your entire repository, not just the changes:
  - Open the repository in your IDE and enter **`Review my repository`**

Review as you code

> **Note**
>
> Amazon Q auto-reviews are only available with a Amazon Q Developer Pro subscription.

Auto-reviews are rule-based reviews powered by Amazon Q detectors . Amazon Q automatically reviews the file you are actively coding in, generating code issues as soon as they are detected in your code. When Amazon Q performs auto reviews, it doesn't generate in-place code fixes.

Auto-reviews are enabled by default when you use Amazon Q. Use the following procedure to pause or resume auto-reviews.

Pause and resume auto-reviews

To pause auto-reviews, complete the following steps.

1. Choose **Amazon Q** in the status bar of the IDE window.

   The Amazon Q task bar opens.
2. Choose **Pause Auto-Reviews**. To resume auto-reviews, choose **Resume Auto-Reviews**.

Addressing code issues with Amazon Q Developer

The topics in this section explain how to address and resolve code issues, and, where applicable, how to ignore issues.

Topics

- Address code issues in JetBrains and Visual Studio Code
- Address code issues in Visual Studio

Address code issues in JetBrains and Visual Studio Code

To address a code issue in JetBrains and Visual Studio Code, you will either have the option to generate an in-place fix or generate an explanation that you can use to manually update your code.

You can take the following actions:

- Generate an in-place code fix
- Explain the issue and get new code
- Ignore the issue, or ignore all similar issues

Generate in place fixes for your file

Amazon Q can update your files in-place to automatically remediate a code issue it detected.

To automatically fix a code issue in your file:

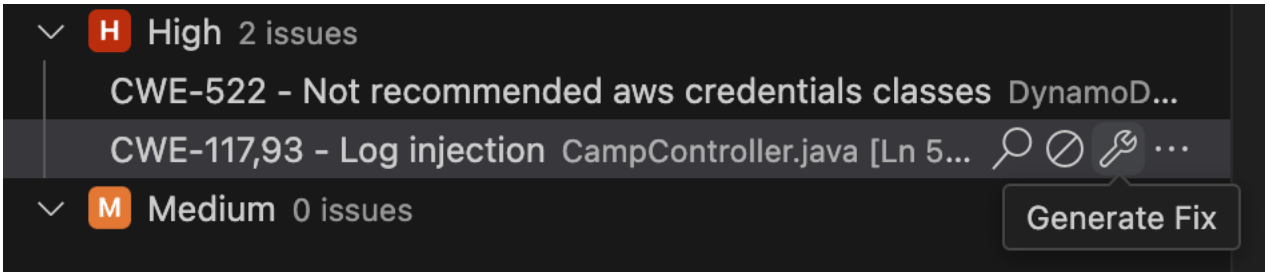
JetBrains

1. In the **Problems** tool window, in the **Amazon Q Code Issues** tab, choose the code issue you want to address.
2. A panel opens with more information about the code issue. If applicable, you'll see details about the Amazon Q detector that was used to identify the code issue.
3. Along the bottom of the panel, choose **Fix**.
4. In the chat panel, Amazon Q provides a brief explanation of the fix and then applies an in-place fix in your code file.
5. You will see the code change in your file, and have the option to undo the change from the chat panel.

Visual Studio Code

1. In the **Code Issues** tab, choose the code issue you want to address.
2. Choose the wrench icon.

   The following image shows the wrench icon for a code issue in Visual Studio Code.



3. In the chat panel, Amazon Q provides a brief explanation of the fix and then applies an in-place fix in your code file.
4. You will see the code change in your file, and have the option to undo the change from the chat panel.

Explain the code issue and get new code

Amazon Q can provide an in-depth explanation of a code issue and provide remediation options with accompanying code for you to add to your files.

To get an explanation of a code issue:

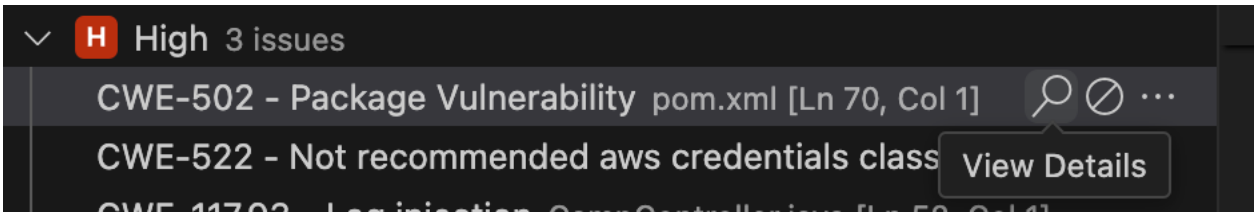
JetBrains IDEs

1. In the **Problems** tool window, in the **Amazon Q Code Issues** tab, choose the code issue you want to address.
2. A panel opens with more information about the code issue. If applicable, you'll see details about the Amazon Q detector that was used to identify the code issue.
3. Along the bottom of the panel, choose **Explain**.
4. In the chat panel, Amazon Q provides details about the issue and suggests how to fix it, with code that you can insert into your file.
5. To update your file, follow Amazon Q's instructions for where to add or replace code, and copy the provided code in the correct location in your file. Make sure to remove the vulnerable code when adding the updated code.

Visual Studio Code

1. In the **Code Issues** tab, choose the code issue you want to address.
2. Choose the magnifying glass icon.

The following image shows the magnifying glass icon for a code issue in Visual Studio Code.



3. In the chat panel, Amazon Q provides details about the issue and suggests how to fix it, with code that you can insert into your file.
4. To update your file, follow Amazon Q's instructions for where to add or replace code, and copy the provided code in the correct location in your file. Make sure to remove the vulnerable code when adding the updated code.

Ignore a code issue

If a detected code issue isn't applicable, you can choose to ignore it, or ignore it and all similar issues (issues with the same CWE). The issues will be removed from the Code Issues tab.

To ignore a code issue:

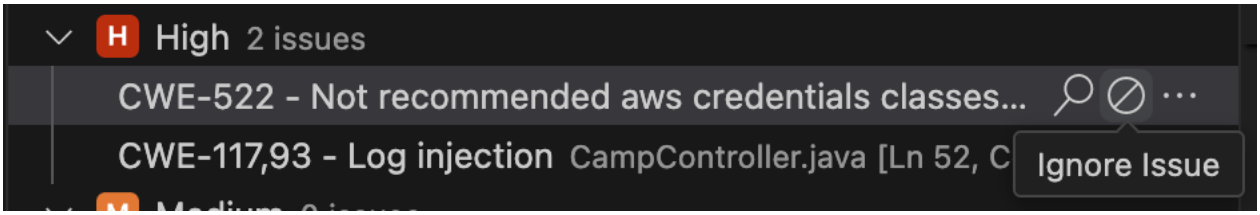
JetBrains

1. In the **Problems** tool window, in the **Amazon Q Code Issues** tab, choose the code issue you want to ignore.
2. A panel opens with more information about the code issue. Along the bottom of the panel, choose **Ignore**. The code issue is removed from the Code Issue panel.
3. You can also choose **Ignore All** to ignore this and other code issues with the same CWE.

Visual Studio Code

1. In the **Code Issues** tab, choose the code issue you want to ignore.
2. Choose the ignore icon.

   The following image shows the ignore icon for a code issue in Visual Studio Code.



3. The code issue is removed from the Code Issue panel.
4. To ignore similar issues, choose the elipses icon, and then the choose **Ignore Similar Issues** button that appears.

Address code issues in Visual Studio

To view code issues detected by Amazon Q in Visual Studio, open the Visual Studio **Error List** by expanding the **View** heading in the Visual Studio main menu and choosing **Error List**.

You can use the information in the code issue to update your code. After updating your code, review your code again to see if the issues were addressed.

By default, the Visual Studio **Error List** displays all of the warnings and errors for your code base. To filter your Amazon Q code issues from the Visual Studio **Error List**, create a filter by completing the following procedure.

> **Note**
>
> Code issues are only visible after you've run a code review in which Amazon Q detected issues.
> Code issues appear as warnings in Visual Studio. In order to view issues detected by Amazon Q in the **Error List**, the **Warnings** option in the **Error List** heading must be selected.

Filter code issues in the Error List

1. From the Visual Studio main menu, choose view and then **Error List** to open the **Error List** pane.
2. From the **Error List** pane, right-click the header row to open the context menu.
3. From the context menu, expand **Show Columns**, and then select **Tool** in the expanded menu.
4. The **Tool** column is added to your **Error List**.

5. From the **Tool** column header, select the **Filter** icon and choose **Amazon Q** to filter for Amazon Q code issues.

Filtering code issues

> **Note**
>
> You can only filter code issues in JetBrains IDEs and Visual Studio Code.

When you filter code issues, only issues that meet the selected criteria are shown in the Code Issues panel. You can filter the issues based on their severity so you only see issues with the selected severity in the panel.

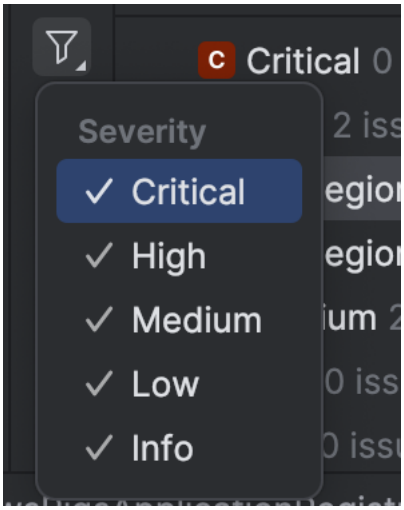
You can also control how code issues are organized in the Code Issues panel. You can group issues based on severity or based on their file location.

To filter code issues:

JetBrains IDEs

1. From the **Amazon Q Code Issues** tab, choose the filter icon.
2. A pop-up menu with Severity levels opens.

   The following image shows the Severity menu in the Code Issues tab in IntelliJ IDEA.



3. Select or deselect the the severity levels you want to filter for, and then choose **OK**. Only the issues with the severity you selected will appear in the **Amazon Q Code Issues** panel.

Visual Studio Code

1. From the **Code Issues** panel, choose the filter icon.

   The following image shows the filter icon in the Code Issues tab in Visual Studio Code.



2. The **Filter Issues** menu opens.

   Select or deselect the boxes next to the severity levels you want to filter for, and then choose **OK**. Only the issues with the severity you selected will appear in the **Code Issues** panel.

To group code issues:

JetBrains IDEs

1. From the **Amazon Q Code Issues** panel, choose the grouping icon.
2. A **Group By** pop-up menu opens.
3. Select **Severity** to group issues in the Code Issues panel based on their severity. Select **Location** to group issues based on which code file they are located in.

Visual Studio Code

1. From the **Code Issues** panel, choose the grouping icon.

   The following image shows the grouping icon in the Code Issues tab in Visual Studio Code.



2. The **Group Issues** menu opens.
3. Select **Severity** to group issues in the Code Issues panel based on their severity. Select **Location** to group issues based on which code file they are located in.

Code issue severity in Amazon Q Developer code reviews

Amazon Q defines the severity of the code issues detected in your code so you can prioritize what issues to address and track the security posture of your application. The following sections explain what methods are used to determine the severity of code issues and what each level of severity means.

How severity is calculated

The severity of a code issue is determined by the detector that generated the issue. Detectors in the Amazon Q Detector Library are each assigned a severity using the Common Vulnerability Scoring System (CVSS). The CVSS considers how the finding can be exploited in its context (for example, can it be done over internet, or is physical access required) and what level of access can be obtained.

The following table outlines how severity is determined based on the level of access and level of effort required for a bad actor to successfully attack a system.

Severity determination matrix

Level of access	Level of effort	Severity
Full control of system or its output	Requires access to system	High
Full control of system or its output	Internet with high level of effort	Critical
Full control of system or its output	Over internet	Critical
Access to sensitive informati on	Requires access to system	Medium
Access to sensitive informati on	Internet with high level of effort	High

Level of access	Level of effort	Severity
Access to sensitive informati on	Over internet	High
Can crash or slow down the system	Requires access to system	Low
Can crash or slow down the system	Internet with high level of effort	Medium
Can crash or slow down the system	Over internet	Medium
Provides additional security	Not exploitable	Info
Provides additional security	Requires access to system	Info
Provides additional security	Internet with high level of effort	Low
Provides additional security	Over internet	Low
Best practice	Not exploitable	Info

Severity definitions

The severity levels are defined as follows.

Critical – The code issue should be addressed immediately to avoid it escalating.

Critical code issues suggest that an attacker can gain control of the system or modify its behavior with moderate effort. It is recommended that you treat critical findings with the utmost urgency. You also should consider the criticality of the resource.

High – The code issue must be addressed as a near-term priority.

High severity code issues suggest that an attacker can gain control of the system or modify its behavior with high effort. It is recommended that you treat a high severity finding as a near-term priority and that you take immediate remediation steps. You also should consider the criticality of the resource.

Medium – The code issue should be addressed as a midterm priority.

Medium severity findings can lead to crash, unresponsiveness, or unavailability of the system. It is recommended that you investigate the implicated code at your earliest convenience. You also should consider the criticality of the resource.

Low – The code issue does not require action on its own.

Low severity findings suggest programming errors or anti-patterns. You do not need to take immediate action on low severity findings, but they can provide context when you correlate them with other issues.

Informational – No recommended action.

Informational findings include suggestions for quality or readability improvements, or alternative API operations. No immediate action is necessary.

Transforming code in the IDE with Amazon Q Developer

End of support notice

On April 30, 2027, AWS will discontinue support for Amazon Q Developer IDE plugins. For capabilities similar to Amazon Q Developer IDE plugins, explore Kiro to access the latest models and features, including agentic coding, chat and MCP support. For more information, see Amazon Q Developer IDE plugins end of support.

Amazon Q Developer can transform your code in integrated development environments (IDEs) by performing automated language and operating system (OS)-level upgrades and conversions. You provide the code to be transformed, and Amazon Q generates changes that you can review and apply to your files.

To get started, install Amazon Q in an IDE that supports transformations. Then, see the topic for the type of transformation you'd like to perform with Amazon Q.

For more information on IDEs that support transformation and how to install Amazon Q, see Using Amazon Q Developer in the IDE.

Topics

- Transforming Java applications with Amazon Q Developer
- Transforming .NET applications with Amazon Q Developer

Transforming Java applications with Amazon Q Developer

> **Note**
>
> AWS Transform custom now available for Java upgrades. Agentic AI that handles version upgrades, SDK migration, and more, and improves with every execution. Get started

Amazon Q supports the following types of transformations for Java applications:

- Java language and dependency version upgrades
- Embedded SQL conversion for Oracle to PostgreSQL database migration

To get started, see the topic for the type of transformation you'd like to perform.

Topics

- Quotas
- Upgrading Java versions with Amazon Q Developer
- Converting embedded SQL in Java applications with Amazon Q Developer
- Transforming code on the command line with Amazon Q Developer
- Viewing transformation job history
- Troubleshooting issues with Java transformations

Quotas

Java application transformations with Amazon Q in the IDE and command line maintain the following quotas:

- **Lines of code per job** – The maximum number of code lines that Amazon Q can transform in a given transformation job.
- **Lines of code per month** – The maximum number of code lines that Amazon Q can transform in a month.

- **Concurrent jobs** – The maximum number of transformation jobs you can run at the same time. This quota applies to all transformations in the IDE, including .NET transformations in Visual Studio.
- **Jobs per month** – The maximum number of transformation jobs you can run in one month.

Resource	Quotas
Lines of code per job	Free tier: 1000 lines of code
Lines of code per month	Free tier: 2000 lines of code
Concurrent jobs	1 job per user 25 jobs per AWS account
Jobs per month	Pro tier: 1000 jobs Free tier: 100 jobs

Upgrading Java versions with Amazon Q Developer

Amazon Q Developer can upgrade your Java applications to newer language versions in the integrated development environment (IDE). Changes Amazon Q can make to upgrade your code include updating deprecated code components and APIs as well as upgrading libraries, frameworks, and other dependencies in your code.

To transform your code, Amazon Q first builds your code in the source language version and verifies that it has the information necessary to perform the transformation. After Amazon Q successfully transforms your code, you verify and accept the changes in your IDE. Since Amazon Q Developer makes the minimal changes necessary to make your upgraded code compatible with the target JDK, an additional transformation is required to upgrade your project's libraries and dependencies. For more information about how Amazon Q transforms your code, see How Amazon Q Developer transforms code for Java language upgrades.

Topics

- Supported Java upgrades and IDEs
- Step 1: Prerequisites

- Step 2: Configure your project
- Step 3: Create a dependency upgrade file (optional)
- Step 4: Transform your code
- How Amazon Q Developer transforms code for Java language upgrades

Supported Java upgrades and IDEs

Amazon Q currently supports the following Java source code versions and target versions for transformations. Transforming code to the same Java version includes upgrading libraries and other dependencies in the source code version.

Supported Java upgrades

Source code version	Supported target versions
Java 8	Java 17 and Java 21
Java 11	Java 17 and Java 21
Java 17	Java 17 and Java 21
Java 21	Java 21

Amazon Q supports Java upgrades in the following IDEs:

- Modules in JetBrains IDEs
- Projects and workspaces in Visual Studio Code

Step 1: Prerequisites

Before you continue, make sure you've completed the steps in Set up Amazon Q in your IDE.

Make sure that the following prerequisites are met before you begin a Code Transformation job:

- Your project is written in a supported Java version and is built on Maven.
- Your project successfully builds with Maven in your IDE. Maven 3.8 or later is currently supported.
- Your project source JDK is available locally and is the version of your source code. For example, if you are transforming Java 8 code, your local JDK installation should be JDK 8.

- Your project builds in 55 minutes or less.
- Your project is configured correctly, and the correct JDK version is specified. For more information, see Step 2: Configure your project.
- Your project doesn't require access to resources on your private network, including a virtual private cloud (VPC) or on-premise network. For example, if your project contains unit tests that connect to a database in your network, the transformation will fail.
- Your project doesn't use plugins that package languages other than Java in your Java project. For example, if your project uses the frontend-maven-plugin for executing front-end JavaScript code in addition to your Java source code, the transformation will fail.
- Your local network allows uploads to Amazon S3 buckets that Amazon Q uses to transform your code. For more information, see Allow access to Amazon S3 buckets in data perimeters.
- Your application only uses UTF-8 characters. If your application uses non-UTF-8 characters, Amazon Q will still attempt to transform your code.

Step 2: Configure your project

To configure your project, use the following information for the IDE you're using.

Configure a project in JetBrains

To configure your project in JetBrains, you might need to specify the following project and module settings.

If your modules use the same JDK and language level as your project, you don't need to update module settings.

- Project SDK – The JDK used to compile your project.
- Project language level – The Java version used in your project.
- Module SDK – The JDK used to compile your module.
- Module language level – The Java version used in your module.
- Maven Runner JRE – The JDK you build your module with.

Update project and module settings

To update your SDK and language level settings for your project or module, complete the following steps:

1. From your JetBrains IDE, choose **File** and then **Project Structure**.
2. The Project Structure window opens. Under **Project Settings**, choose **Project**.
   a. To update your project JDK, choose from the dropdown list next to **SDK**.
   b. To update your project language, choose from the dropdown next to **Language level**.
3. Under **Project Settings**, choose **Modules**.
   a. To update your module JDK, choose from the dropdown list next to **SDK**.
   b. To update your module language, choose from the dropdown next to **Language level**.

For more information, see Project structure settings and Module structure settings in the JetBrains documentation.

Update Maven settings

To update your Maven Runner JRE, complete the following steps:

1. From your JetBrains IDE, choose the gear icon, and then choose **Settings** in the menu that appears.
2. In the **Settings** window, choose **Build, Execution, Deployment**, then **Build Tools**, then **Maven**, and then **Runner**.
3. In the JRE field, choose the JDK used to build the module you're transforming.

Configure a project in VS Code

To configure your project in VS Code, your project must contain the following:

- A `pom.xml` file in the project root folder
- A `.java` file in the project directory

If your project contains a Maven wrapper executable (`mvnw` for macOS or `mvnw.cmd` for Windows), make sure it's at the root of your project. Amazon Q will use the wrapper, and no other Maven configuration is necessary.

If you aren't using a Maven wrapper, install Maven. For more information, see Installing Apache Maven in the Apache Maven documentation.

After installing Maven, add it to your PATH variable. For more information, see How do I add Maven to my PATH? Your Java `runtime` variable should also be pointing to a JDK and not to a JRE. To confirm your configuration is correct, run `mvn -v`. The output should show your Maven version and the `runtime` variable pointing to the path to your JDK.

Step 3: Create a dependency upgrade file (optional)

You can provide Amazon Q with a *dependency upgrade file*, a YAML file that lists your project's dependencies and which versions to upgrade to during a transformation. By providing a dependency upgrade file, you can specify third and first party dependencies that Amazon Q might not otherwise know to upgrade.

First party dependencies refer to the libraries, plugins, and frameworks that your organization maintains and are only available locally or on your organization's private network. Amazon Q is able to access your first party dependencies when it performs builds in your local environment. For more information, see Building code in your local environment. Third party dependencies are publicly available or open source dependencies that aren't unique to your organization.

You can specify first party dependencies you want to upgrade in a YAML file, and Amazon Q upgrades them during the JDK upgrade (for example, Java 8 to 17). You can initiate a separate transformation (17 to 17 or 21 to 21) after the initial JDK upgrade to upgrade third-party dependencies.

Once Amazon Q performs a minimum JDK upgrade, you can initiate a separate transformation to upgrade all third party dependencies. Alternatively, you can specify third party dependencies and their versions in a YAML file to only upgrade those dependencies during the library upgrade transformation.

Amazon Q will prompt you to provide a dependency upgrade file during the transformation. If you want to provide one, first make sure you've configured the file properly. The following fields are required in the YAML file:

- name - The name of the dependency upgrade file.
- description (optional) - A description of the dependency upgrade file, and for which transformation.
- dependencyManagement - Contains the list of dependencies and plugins to upgrade.
- dependencies - Contains the name and version of the libraries to upgrade.
- plugins - Contains the names and versions of the plugins to upgrade.

- identifier - The name of the library, plugin, or other dependency.
- targetVersion - The version of the dependency to upgrade to.
- versionProperty (optional) - The version of the dependency you're defining, as set with the `properties` tag in your application's `pom.xml` file.
- originType - Whether the dependency is first or third party, specified by either FIRST_PARTY or THIRD_PARTY.

Following is an example of a dependency upgrade YAML file, and the required configuration for Amazon Q to parse:

```
name: dependency-upgrade

description: "Custom dependency version management for Java migration from JDK 8/11/17
 to JDK 17/21"


dependencyManagement:

  dependencies:

    - identifier: "com.example:library1"

      targetVersion: "2.1.0"

      versionProperty: "library1.version"  # Optional

      originType: "FIRST_PARTY"

    - identifier: "com.example:library2"

      targetVersion: "3.0.0"

      originType: "THIRD_PARTY"

  plugins:

    - identifier: "com.example.plugin"

      targetVersion: "1.2.0"

      versionProperty: "plugin.version"  # Optional
```

```
    originType: "THIRD_PARTY"
```

Step 4: Transform your code

To test your IDE setup, download and unzip the sample project, and complete the following steps for your IDE. If you are able to view the proposed changes and transformation summary, you are ready to transform your own code project. If the transformation fails, your IDE is not configured correctly. To address configuration issues, review Step 2: Configure your project and Troubleshooting.

> **Note**
>
> Do not turn off, close, or put your local machine to sleep during the code transformation. The initial and validation builds use the client-side environment, which requires a stable network connection.

To upgrade the language version of your code project or module, complete the following steps for your IDE.

JetBrains

1. Open the module that you want to upgrade in JetBrains. Make sure you've successfully built your project in the IDE.
2. Choose the Amazon Q logo, and ask Amazon Q to transform your application in the chat panel that opens.
3. A **Transform your application** pop-up appears. Choose the project that you want to upgrade from the dropdown list, and then choose **Transform**.
4. Amazon Q prompts you to provide an upgrade dependency file. If you have configured a YAML with the dependencies and version to upgrade to, add your file. Amazon Q will validate the file to ensure it's configured correctly. If you get an error, review the format and required fields described in Step 3: Create a dependency upgrade file (optional).
5. Amazon Q begins the transformation. You can view progress on the **Transformation details** tab.
6. After the transformation is complete, you can verify the upgraded code before updating your project. To view the new code, go to the **Transformation details** tab and then choose

View diff. In the **Apply patch** window that appears, choose a file to open a diff view with your source code and upgraded code.

7. To accept the changes that Amazon Q made, choose **View diff** to open the **Apply patch** window. Select all the updated files, and choose **OK** to update your project in place.
8. To get details about how your code was upgraded and suggested next steps, on the **Transformation details** tab, choose **View transformation summary**.

Visual Studio Code

1. Open the project or workspace that you want to upgrade in VS Code. Make sure that you've successfully built your project in the IDE.
2. Choose the Amazon Q logo, and ask Amazon Q to transform your application in the chat panel that opens.
3. Choose the project that you want to upgrade from the search bar at the top of the IDE.
4. If Amazon Q can't find the version of your source code, it prompts you to choose your code version. Choose the version that your source code is written in, and then choose **Transform** in the pop-up to proceed.
5. If prompted, enter the `JAVA_HOME` path to your JDK. For more information, see Configure your VS Code project.
6. Amazon Q prompts you to provide an upgrade dependency file. If you have configured a YAML with the dependencies and version to upgrade to, add your file. Amazon Q will validate the file to ensure it's configured correctly. If you get an error, review the format and required fields described in Step 3: Create a dependency upgrade file (optional).
7. Amazon Q begins the transformation. You can view progress on the **Transformation Hub** tab.
8. After the transformation is complete, the **Proposed Changes** tab opens. To verify the upgraded code before updating your project, choose **Download proposed changes**. Choose a file to open a diff view with your source code and upgraded code.
9. To accept the changes Amazon Q made, go to the **Proposed Changes** tab and choose **Accept**.
10. To get details about how your code was upgraded and suggested next steps, on the **Transformation Hub**, choose the **Views and More Actions** ellipsis button, and then choose **Show Transformation Summary**.

How Amazon Q Developer transforms code for Java language upgrades

To transform your code, Amazon Q Developer generates a transformation plan that it uses to upgrade the code language version of your project. After transforming your code, it provides a transformation summary and a file diff for you to review changes before accepting them. Since Amazon Q Developer makes the minimal changes necessary to make your upgraded code compatible with the target JDK, an additional transformation is required to upgrade your project's libraries and dependencies. The following sections provide more details on how Amazon Q performs the transformation.

Building your code and creating a transformation plan

To begin transforming your code, Amazon Q builds your project locally and generates a build artifact that contains your source code, project dependencies, and build logs.

After generating the build artifact, Amazon Q builds your code in a secure build environment and creates a transformation plan, which is customized to the project or module you're upgrading. The transformation plan outlines the specific changes Amazon Q will attempt to make, including new dependency versions, major code changes, and suggested replacements for deprecated code. These changes are based on the preliminary build of your code, and might change during the transformation.

Transforming your code

To transform your code, Amazon Q attempts to upgrade your code to the target Java version based on the proposed changes in the transformation plan. As it makes changes, it re-builds and runs existing unit tests in your source code to iteratively fix any encountered errors. The JDK upgrade can be made from the following source code version to the target version:

- Java 8 to 17
- Java 8 to 21
- Java 11 to 17
- Java 11 to 21
- Java 17 to 21

Amazon Q makes the minimal changes necessary to make your code compatible with the target Java version. Once Amazon Q performs a minimum JDK upgrade, you can initiate a separate transformation to upgrade all third party dependencies. Alternatively, you can specify third party

dependencies and their versions in a YAML file to only upgrade those dependencies during the library upgrade transformation.

Amazon Q attempts to make the following changes when upgrading your code:

- Update deprecated code components according to the target Java version recommendations
- Upgrade popular libraries and frameworks to a version compatible with the target Java version. This includes updating the following libraries and frameworks to their latest available major versions:
  - Apache Commons IO
  - Apache HttpClient
  - bc-fips
  - Cucumber-JVM
  - Hibernate
  - jackson-annotations
  - JakartaEE
  - Javax
  - javax.servlet
  - jaxb-api
  - jaxb-impl
  - jaxen
  - jcl-over-slf4j
  - json-simple
  - jsr305
  - junit
  - junit-jupiter-api
  - Log4j
  - Micronaut
  - Mockito
  - mockito-core
  - Okio
  - PowerMockito

- Quarkus
- slf4j
- slf4j-api
- Spring Boot
- Spring Framework
- Spring Security
- Swagger
- testng

> **Note**
>
> Do not turn off or close your local machine during the code transformation because client-side build requires a stable network connection.

Building code in your local environment

During a transformation, Amazon Q performs verification builds in your local environment. Amazon Q transforms your code on the server side in multiple steps. After each step, Amazon Q sends the code to your local environment to build and test the changes it made. The code is then sent back to the server side to continue the transformation.

The build in your local environment helps verify the transformed code by allowing Amazon Q to run tests that require access to private resources. To minimize security risks associated with building AI-generated code in your local environment, Amazon Q reviews and updates the code it generates to address security concerns.

Reviewing the transformation summary and accepting changes

After the transformation is complete, Amazon Q provides a transformation summary with details about the changes it made, including the status of the final build which indicates whether your entire project was upgraded. You can also view a build log summary to understand any issues that prevented Amazon Q from building your code in the upgraded version.

The transformation summary additionally includes the differences between the changes proposed in the transformation plan and the changes Amazon Q ultimately made to upgrade your code, and any additional changes that weren't in the original plan.

After you review the transformation summary, you can view the changes Amazon Q is proposing in a file diff view. Any code changes Amazon Q suggests will not affect your current project files until you accept the changes. The transformed code is available up to 30 days after the transformation completes.

Completing partially successful transformations

Depending on the complexity and specifics of your codebase, there might be instances where the transformation is partially successful. This means that Amazon Q was able to transform only certain files or areas of code in your project. In this case, you have to manually update the remaining code for your project to be buildable in the updated language version.

To help transform the rest of your code, you can use Amazon Q chat in the IDE. You can ask Amazon Q to review the partially updated files and provide new code to address issues, such as compilation errors. You can also use features like Feature development and Workspace context to include more of your project as context and get suggestions for multiple files at a time.

Converting embedded SQL in Java applications with Amazon Q Developer

The Amazon Q Developer agent for code transformation in the IDE can help you convert embedded SQL to complete Oracle to PostgreSQL database migration with AWS Database Migration Service (AWS DMS).

AWS DMS is a cloud service that makes it possible to migrate relational databases, data warehouses, NoSQL databases, and other types of data stores. DMS Schema Conversion in AWS DMS helps you convert database schemas and code objects that you can apply to your target database. For more information, see What is AWS Database Migration Service? in the *AWS Database Migration Service User Guide*.

When you use AWS DMS and DMS Schema Conversion to migrate a database, you might need to convert the embedded SQL in your application to be compatible with your target database. Rather than converting it manually, you can use Amazon Q in the IDE to automate the conversion. Amazon Q uses metadata from a DMS Schema Conversion to convert embedded SQL in your application to a version that is compatible with your target database.

Currently, Amazon Q can convert SQL in Java applications for Oracle databases migrating to PostgreSQL. You will only see the option to transform SQL code in the IDE if your application contains Oracle SQL statements. See the prerequisites for more information.

Step 1: Prerequisites

Before you continue, make sure you've completed the steps in Set up Amazon Q in your IDE.

Before you begin a code transformation job for SQL conversion, make sure the following prerequisites are met:

- You are migrating a Java application with embedded SQL from an Oracle database to a PostgreSQL database. Your application must contain Oracle SQL statements for it to be eligible for transformation.
- You have completed the process for converting your database schema using AWS DMS Schema Conversion. For more information, see Migrating Oracle databases to Amazon RDS for PostgreSQL with DMS Schema Conversion in the *Database Migration Guide*.
- After schema conversion is complete, you have downloaded the migration project file from the AWS DMS console.

Step 2: Configure your application

To convert your embedded SQL code, your Java project must contain at least one `.java` file.

If you are using a JetBrains IDE, you must set the SDK field in Project Structure settings to the applicable JDK. For information on configuring Project Structure settings, see Project structure settings in the JetBrains documentation.

Step 3: Convert embedded SQL

To convert the embedded SQL code in your Java application to a format that is compatible with your PostgreSQL target database, complete the following steps:

1. In your IDE where Amazon Q is installed, open the Java codebase that contains the embedded SQL you need to convert.
2. Choose the Amazon Q icon to open the chat panel.
3. Ask Amazon Q to transform your application in the chat panel.
4. If your Java application is eligible for SQL conversion, Amazon Q will prompt you to choose which type of transformation you'd like to perform. Enter **`SQL conversion`**.
5. Amazon Q prompts you to upload the schema metadata file you retrieved from Amazon S3. In the chat, Amazon Q provides instructions for retrieving the file.

6. Amazon Q prompts you to provide the project that contains the embedded SQL as well as the database schema file. Choose the appropriate files from the dropdown menus in the chat panel.
7. Confirm the details Amazon Q retrieved from the database schema are accurate.
8. Amazon Q begins converting your SQL code. This might take a few minutes.
9. After Amazon Q converts the SQL code, it provides a diff with any updates it has made to your files. Review the changes in the diffs, and then accept the changes to update your code.

   Amazon Q also provides a transformation summary with details about the changes it made.
10. After updating your code, return to the AWS DMS console to verify the new SQL is compatible with the migrated database.

Transforming code on the command line with Amazon Q Developer

You can transform your applications from the command line with the Amazon Q Developer command line transformation tool. To transform your code, you provide the path to your source code and any necessary configuration files, and Amazon Q generates new code in a series of steps. Throughout the transformation, Amazon Q builds code on your local environment to verify changes. For more information, see Building code in your local environment. Amazon Q creates a new branch in your repository where it commits the code changes. When the transformation is complete, you can merge the branch into your original branch to incorporate the changes into your codebase.

To get started, install the command line tool and authenticate, and then see the commands to configure and start a transformation.

Topics

- Building code in your local environment
- Commands
- Running a transformation on the command line with Amazon Q Developer
- Troubleshooting transformations on the command line
- Amazon Q Developer command line transformation tool version history

Building code in your local environment

During a transformation, Amazon Q performs verification builds in your local environment. Amazon Q transforms your code on the server side in multiple steps. After each step, Amazon Q sends the code to your local environment to build and test the changes it made. The code is then sent back to the server side to continue the transformation.

The build in your local environment helps verify the transformed code by allowing Amazon Q to run tests that require access to private resources. To minimize security risks associated with building AI-generated code in your local environment, Amazon Q reviews and updates the code it generates to address security concerns.

> **Note**
>
> Amazon Q performs transformations based on your project's requests, descriptions, and content. To maintain security, avoid including external, unvetted artifacts in your project repository and always validate transformed code for both functionality and security.

Commands

For step-by-step instructions for running these commands, see Running a transformation on the command line with Amazon Q Developer.

To configure a transformation and authenticate to Amazon Q Developer Pro, run:

```
qct configure
```

To start a transformation for a Java upgrade, run the following command. For *<your-source-java-version>*, you can enter `JAVA_1.8`, `JAVA_8`, `JAVA_11`, `JAVA_17`, or `JAVA_21`. For *<your-target-java-version>*, you can enter either `JAVA_17` or `JAVA_21`. Both `--source_version` and `--target_version` are optional. The `--trust` flag enables a transformation to run while vetting code to maintain security.

```
qct transform --source_folder <path-to-folder>
    --source_version <your-source-java-version>
    --target_version <your-target-java-version>
    --trust
```

To start a transformation for a SQL conversion, run:

```
qct transform --source_folder <path-to-folder>
    --sql_conversion_config_file <path-to-sql-config-file>
```

To see what version of the command line tool for transformation you are using, run:

```
qct -v
```

To get help with transformations, run:

```
qct -h
```

To view your transformation job history, run:

```
qct history
```

For more information about viewing and managing your transformation job history, see Viewing job history on the command line.

Running a transformation on the command line with Amazon Q Developer

Complete these steps to transform your code on the command line with the Amazon Q Developer command line tool.

Prerequisites

Before you begin a transformation on the command line, the following prerequisites must be met:

- If you're upgrading your Java code version, your project meets the prerequisites for upgrading Java versions with Amazon Q.
- If you're converting embedded SQL in a Java application, your application meets the prerequisites for converting embedded SQL with Amazon Q.
- You have Python installed on your command line environment. This is how you will install the command line tool. The minimum supported Python version is 3.12.
- You are running the transformation on macOS or Linux.
- The size of your application is 2 GB or smaller.
- If you have specific dependencies you want Amazon Q to upgrade, you have configured a dependency upgrade file.

Step 1: Choose authentication method and add permissions

You can authenticate IAM Identity Center to run transformations on the command line. Ensure you have the proper permissions.

> **Note**
>
> Customer managed keys aren't supported for transformations performed on the command line.

Add permissions

The IAM identity associated with the Amazon Q Developer subscription you are using to authenticate must have permissions to perform transformations on the command line. Before you proceed, ensure your IAM identity has the permissions defined in Allow users to run transformations on the command line.

Authenticate with IAM Identity Center through a Amazon Q Developer subscription

To authenticate with IAM Identity Center, you must be subscribed to Amazon Q Developer Pro as a workforce user by your administrator, and you must provide the Start URL to authenticate through your subscription. You or your administrator can find the Start URL in the Amazon Q Developer console. For more information see, Finding the Start URL for use with Amazon Q Developer.

To add required permissions, see Add permissions.

You provide the Start URL in Step 4: Configure and authenticate.

Step 2: Install the tool

1. Download the Amazon Q command line tool for transformations and unzip it.

   To download a previous version of the command line tool, see Version history.
2. We recommend that you set up a virtual environment in Python to install the tool. To create a virtual environment, open a terminal window in the directory where you want to install the tool and run:

   ```
   python -m venv qct-cli
   ```

3. To activate the virtual environment, run:

```
source qct-cli/bin/activate
```

4. To install the tool on your command line, run the following command with the path to where you unzipped the tool, based on your machine architecture:

   Linux_aarch64

   ```
   pip install <path/to/unzipped-tool>/Linux_aarch64/amzn_qct_cli-1.2.2-py3-none-any.whl
   ```

   Linux_x86_64

   ```
   pip install <path/to/unzipped-tool>/Linux_x86_64/amzn_qct_cli-1.2.2-py3-none-any.whl
   ```

   Note

   If you are using an older version of the command line tool for transformations, replace `1.2.2` with the version you downloaded.

5. To verify that the tool was installed, run:

```
which qct
```

Step 3: Create a dependency upgrade file (optional)

You can provide Amazon Q with a *dependency upgrade file*, a YAML file that lists your project's dependencies and which versions to upgrade to during a transformation. By providing a dependency upgrade file, you can specify third and first party dependencies that Amazon Q might not otherwise know to upgrade.

First party dependencies refer to the libraries, plugins, and frameworks that your organization maintains and are only available locally or on your organization's private network. Amazon Q is able to access your first party dependencies when it performs builds in your local environment. For more information, see Building code in your local environment. Third party dependencies are publicly available or open source dependencies that aren't unique to your organization.

You can specify first party dependencies you want to upgrade in a YAML file, and Amazon Q upgrades them during the JDK upgrade (for example, Java 8 to 17). You can initiate a separate transformation (17 to 17 or 21 to 21) after the initial JDK upgrade to upgrade third-party dependencies.

Once Amazon Q performs a minimum JDK upgrade, you can initiate a separate transformation to upgrade all third party dependencies. Alternatively, you can specify third party dependencies and their versions in a YAML file to only upgrade those dependencies during the library upgrade transformation.

Amazon Q will prompt you to provide a dependency upgrade file during the transformation. If you want to provide one, first make sure you've configured the file properly. The following fields are required in the YAML file:

- name - The name of the dependency upgrade file.
- description (optional) - A description of the dependency upgrade file, and for which transformation.
- dependencyManagement - Contains the list of dependencies and plugins to upgrade.
- dependencies - Contains the name and version of the libraries to upgrade.
- plugins - Contains the names and versions of the plugins to upgrade.
- identifier - The name of the library, plugin, or other dependency.
- targetVersion - The version of the dependency to upgrade to.
- versionProperty (optional) - The version of the dependency you're defining, as set with the `properties` tag in your application's `pom.xml` file.
- originType - Whether the dependency is first or third party, specified by either FIRST_PARTY or THIRD_PARTY.

Following is an example of a dependency upgrade YAML file, and the required configuration for Amazon Q to parse:

```
name: dependency-upgrade

description: "Custom dependency version management for Java migration from JDK 8/11/17
 to JDK 17/21"
```

```
dependencyManagement:

  dependencies:

    - identifier: "com.example:library1"

      targetVersion: "2.1.0"

      versionProperty: "library1.version"  # Optional

      originType: "FIRST_PARTY"

    - identifier: "com.example:library2"

      targetVersion: "3.0.0"

      originType: "THIRD_PARTY"

  plugins:

    - identifier: "com.example.plugin"

      targetVersion: "1.2.0"

      versionProperty: "plugin.version"  # Optional

      originType: "THIRD_PARTY"
```

Step 4: Configure and authenticate

Before you can begin a transformation, you must authenticate with IAM Identity Center and provide configuration details for your transformation.

1. To start the transformation configuration process, run the following command:

   ```
   qct configure
   ```

2. You are prompted to enter a JDK path for each supported Java version. You only need to specify the path to the JDK of the source version of your Java application, not the target version.
3. Next, to authenticate with IAM Identity Center, you are prompted to enter the start URL for your Amazon Q Developer Pro subscription profile.

Then, enter the AWS Region where you were subscribed in the following format: `us-east-1`. For a list of supported Regions, see Supported Regions. For a list of Region codes, see Regional endpoints in the *AWS General Reference guide*.

4. Your configuration preferences are saved to a configuration.ini file.

Step 5: Run a transformation

Choose the type of transformation you're performing to see the required configuration and commands.

> **Note**
>
> Do not turn off or close your local machine during the code transformation because client-side build requires a stable network connection.

Java upgrade

Modifying the transformation plan

During Java version upgrades, Amazon Q generates a transformation plan that you can review before the transformation begins. You have the option to request the following changes to the plan:

- Which libraries Amazon Q upgrades, from the list included in the plan
  - Example prompts:
    - Only upgrade <dependency1>, <dependency2>, and <dependency5>
    - Don't upgrade <dependency1> or <dependency2>
- The target version to upgrade a library to
  - Example prompts:
    - Upgrade <dependency> to this version instead <version>
- Which steps Amazon Q should perform
  - Example prompts:
    - Only complete steps 1-7
    - Don't run steps 5-9

- Add additional dependencies to upgrade (only an option when upgrading to a newer JDK version)
  - Example prompts:
    - Also upgrade <dependency1> to <version2>

Upgrade Java code

1. Run the following command to start a transformation for a Java upgrade. Replace `<path-to-folder>` with the path to the folder with the code you're transforming and `<your-target-java-version>` with either `JAVA_17` or `JAVA_21`.

   ```
   qct transform --source_folder <path-to-folder>
       --target_version <your-target-java-version>
   ```

   Additional command options:

   - If you are specifying dependencies to upgrade, add the `--dependency_upgrade_file` option with the path to your dependency upgrade file.
   - If you don't want to review or update the transformation plan, add the `--no-interactive` flag to your command. Amazon Q won't ask you for feedback on the plan, and you won't have the opportunity to request changes.
2. Your Maven version is verified before the transformation begins. If you have at least the minimum supported version, you will see the following output:

   ```
   Running command: mvn --version at: path/to/current/directory
   Your Maven version is supported for transformations.
   ```

   If you don't have a supported version of Maven, you must update it to continue. For more information, see the Prerequisites.
3. If you didn't add the `--no-interactive` flag, Amazon Q will prompt you to provide feedback on the transformation plan. You can explain the changes you want to make in English natural language, and Amazon Q will update the plan if it can support the changes you request.
4. Amazon Q begins the transformation. It will output status updates throughout the transformation. When it's complete, Amazon Q provides the path where the transformation results, logs, and configuration files are outputted.

Your upgraded code will be committed to the new branch Amazon Q created. Amazon Q will commit the code in one or multiple commits, depending on the selection you made when you ran `qct configure`.

5. If you're running another transformation after upgrading your Java version, start the second transformation in the same branch where you committed the changes from the first transformation.

SQL conversion

Before you begin, make you sure you've read Converting embedded SQL in Java applications with Amazon Q Developer to understand the prerequisites for this type of transformation.

1. To convert embedded SQL, you must first create a YAML file that contains the path to the schema metadata file from your AWS DMS Schema Conversion.

   Following is the required format of the file:

   ```
   schema_conv_metadata_path: <path-to-metadata-zip-file>
   ```

2. Run the following command to start a transformation for a SQL conversion. Replace `<path-to-folder>` with the path to the folder with the code you're transforming and `<path-to-sql-config-file>` with the path to the YAML file you created in step 1.

   ```
   qct transform --source_folder <path-to-folder>
       --sql_conversion_config_file <path-to-sql-config-file>
   ```

3. If Amazon Q finds multiple schemas in your schema metadata file, it will stop the transformation and provide a list of the detected schemas. Choose which schema to use for the SQL conversion, and then add a new field `schema: <schema-name>` to the YAML file.

4. Amazon Q begins the transformation. It will output status updates throughout the transformation. When it's complete, Amazon Q provides the path where the transformation results, logs, and configuration files are outputted.

   Your upgraded code will be committed to the new branch Amazon Q created.

Pause or cancel a transformation

You can choose to pause or cancel your current transformation job. You can pause a transformation job for up to 12 hours before you can resume again.

To pause or cancel a code transformation job

1. In your CLI terminal, press **Ctrl+C** on your keyboard.
2. Select whether you want to pause or cancel your tranformation.

   - Enter 1 if you want to puase the code transformation job. You can resume the job within 12 hours to continue the code transformation using the following QCT command: `` `qct transform --source_folder=≤/Path/Given/Originally/To/QCT>` ``.
   - Enter 2 if you want to cancel the code tranformation job.

Troubleshooting transformations on the command line

The following information can help you troubleshoot common issues when transforming applications on the command line with Amazon Q Developer.

Why is my bearer token not refreshing?

If you see the following error, it means you need to refresh the bearer token used for authentication.

```
Refreshing bearer token
('Error refreshing bearer token due to: ', InvalidGrantException('An error occurred
 (InvalidGrantException) when calling the CreateToken operation: '))
('Error getting bearer token due to: ', RuntimeError(('Error refreshing bearer token
 due to: ', InvalidGrantException('An error occurred (InvalidGrantException) when
 calling the CreateToken operation: '))))
```

To address this error, run the following command:

```
rm ~/.aws/qcodetransform/credentials.json
```

Once you remove the outdated credentials file, run `qct transform` again to restart the transformation.

Why isn't the most recent version of the command line tool being used?

When you download a new version of the command line tool for transformations, sometimes a previous version of the tool still gets used.

To make you're using the most recent version of the tool, download the most recent version. Then run the following command with the path to where you unzipped the tool, based on your machine architecture:

Linux_aarch64

```
pip install <path/to/unzipped-tool>/Linux_aarch64/amzn_qct_cli-1.2.2-py3-none-
any.whl --force-reinstall
```

Linux_x86_64

```
pip install <path/to/unzipped-tool>/Linux_x86_64/amzn_qct_cli-1.2.2-py3-none-any.whl
 --force-reinstall
```

Note

If you're using an older version of the command line tool for transformations, replace `1.2.2` with the version you downloaded.

Amazon Q Developer command line transformation tool version history

Review the following information for details about current and past releases of the Amazon Q Developer command line transformation tool. The table includes the download link, release date, and release notes for each version.

Version	Release date	Release notes
1.2.2 (latest)	February 26, 2026	Added promotional banner for AWS Transform custom to QCT CLI. Banner display on transform command execution and help text. New

Version	Release date	Release notes
		--skip-banner flag to suppress banner output.
1.2.1	September 9, 2025	Updated Maven extension to include first-party parent POMs during initial project upload
1.2.0	August 7, 2025	Added support for viewing job history and for module structure visualization for Maven Java projects.
1.1.0	July 21, 2025	Includes support for collectin g telemetry about transform ations.
1.0.0	June 27, 2025	The command line transform ation tool is generally available and supports authentication through AWS IAM Identity Center with a Amazon Q Developer Pro subscription only. Added support for subscriptions in the Europe (Frankfurt) Region.
0.6.0	June 6, 2025	Includes support for providing a dependency upgrade file and iterating on the transformation plan.

Version	Release date	Release notes
0.5.2	April 16, 2025	Bug fixes to resolve issues when resuming jobs and failures for applications with first-party dependencies.
0.5.1	March 13, 2025	When you authenticate with IAM, you no longer need to provide an AWS Region. Also includes bug fix to include job status in output logs.
0.5.0	February 28, 2025	Includes support for authentic ating with IAM through the AWS CLI.
0.4.1	February 17, 2025	Bug fix to include support for entering the AWS Region where your Amazon Q Developer subscription is configured.
0.4.0	February 14, 2025	Includes support for upgrading Java applications to Java 21.
0.3.0	February 12, 2025	Includes support for convertin g embedded SQL in Java applications.
0.2.0	February 3, 2025	Includes support for receiving upgraded Java code in multiple commits.

Version	Release date	Release notes
0.1.0	November 27, 2024	Initial release. Includes support for upgrading Java code versions from the command line.

Viewing transformation job history

Amazon Q provides a comprehensive overview of your Java transformation job history, allowing you to track and review your transformation jobs in both IDEs and the command line.

Transformation job history includes the following information about a job:

- **Date** – When the transformation job was executed
- **Project name** – The name of the project that was transformed
- **Status** – The current status of the transformation job
- **Duration** – How long the transformation took to complete
- **Job ID** – A unique identifier for the transformation job
- **Diff patch** – A link or path to the final diff patch file showing all code changes
- **Summary** – A link or path to the transformation summary file with details about the changes made

> **Note**
>
> Only transformations run since this feature was released will be available in the job history. For the feature release date, see Document history for Amazon Q Developer User Guide.

Viewing job history in IDEs

> **Note**
>
> This feature is currently available in Visual Studio Code only.

The Transformation Hub in Visual Studio Code provides a comprehensive view of your Java transformation job history.

A table in the Transformation Hub lists your 10 most recent transformation jobs from the last 30 days. From the table, you can access transformation artifacts and refresh jobs to track progress and get missing artifacts.

Retrieve transformation artifacts

You can access transformation artifacts, such as the diff patches and summary files, from the job history table. Choose the appropriate links to open the diff or summary in your IDE.

Artifacts are stored locally in the `.aws/transform` directory, so you can also access previously downloaded transformation artifacts from past jobs.

Refresh job status

You can refresh the job status from the job history table. Refresh a failed job to get an updated status from the server side that may not have reached your server yet, such as when Amazon Q is able to resume a failed job. You can also refresh completed jobs to download artifacts that may not have appeared yet.

How job history is stored for jobs run in the IDE

For Visual Studio Code, all transformation job information and artifacts are stored locally in the `.aws/transform` directory. The storage is organized as follows:

```
.aws/transform/
### [project-name-1]/
#   ### [job-id-1]/
#   #   ### diff.patch
#   #   ### [summary-1]/
#   #   #   ### summary.md
#   #   #   ### buildCommandOutput.log
#   ### [job-id-2]/
#       ### diff.patch
#       ### [summary-2]/
#       #   ### summary.md
#       #   ### buildCommandOutput.log
### [project-name-2]/
    ### [job-id-3]/
```

```
        ### diff.patch
        ### [summary-3]/
        #   ### summary.md
        #   ### buildCommandOutput.log
```

Viewing job history on the command line

For transformations on the command line, the **qct history** command provides access to your transformation job history with customization options.

For the CLI, transformation job history information is stored locally in the `.aws/qcodetransform/history/` directory.

Using the qct history command

The basic command to view your transformation job history is:

```
qct history
```

By default, this command displays the 10 most recent transformation jobs, in addition to any paused or in-progress jobs.

You can also specify how many job history entries to display with the **--limit** flag. For example, to show 20 jobs, run:

```
qct history --limit 20
```

Troubleshooting issues with Java transformations

The following information can help you troubleshoot common issues when transforming Java applications with Amazon Q Developer.

Topics

- Why can't Amazon Q upload my project?
- Why are my Maven commands failing?
- How do I add Maven to my PATH?
- Why can't Amazon Q build my code?
- Why did my transformation fail after 55 minutes?

- Why can't I download my transformed code?
- How do I access code transformation logs?
- How do I find my transformation job ID?

Why can't Amazon Q upload my project?

If your project fails to upload, it's likely due to one of the following issues. See the topic that corresponds to the error you see from Amazon Q.

Topics

- Reduce project size
- Configure proxy settings in your IDE
- Allow access to Amazon S3

Reduce project size

To transform your code, Amazon Q generates a project artifact, which includes your source code, project dependencies, and build logs. The maximum project artifact size for a transformation job is 2 GB. If you get an error related to project artifact size, you must decrease the size of your project or try transforming a smaller project. You can view the size of your project artifact file in the code transformation logs. For more information, see How do I access code transformation logs?

Configure proxy settings in your IDE

To transform your code, Amazon Q uploads your project artifact to a service-owned Amazon S3 bucket. Part of the upload process involves using SSL or TLS certificates to establish communication between Amazon S3 and your IDE. If you are using a proxy server, the SSL or TLS certificates used by your proxy server must be trusted, otherwise Amazon Q is not able to upload your project.

If you receive an error related to your proxy or certificates, you likely need to configure your IDE or operating system to trust your certificates or update other proxy settings.

> **Note**
>
> You might also encounter issues unrelated to certificates if you are behind your organization's proxy server or firewall. If you complete the following procedures to configure your certificates and still have issues, contact your network administrator

to ensure you are allowed to communicate with Amazon S3 from your IDE. For more information, see Allow access to Amazon S3.

Configure certificates in JetBrains

To configure your JetBrains IDE Java Runtime Environment (JRE) to trust the SSL or TLS certificates used by your proxy server, you must import the SSL or TLS certificates to the `cacerts` file in the JRE. The `cacerts` file is a file that contains trusted root certificates for secure connections such as HTTPS and SSL, and it's part of the JRE's security settings. To import a certificate, complete the following procedure.

Note

We recommend making a backup of the `cacerts` file before modifying it, as any mistakes can cause issues with secure connections.

1. Determine the path to the `cacerts` file in your JRE. The path of the `cacerts` file in the internal JRE shipped with your JetBrains IDE depends on the operating system and the version of the JetBrains IDE you're using.

   Following are examples of paths to the `cacerts` file in common operating systems. Choose your operating system to see examples.

   Note

   `<JetBrains Installation Folder>` refers to the directory where JetBrains products are installed. This directory is typically chosen during the installation process. The `jbr` folder represents the JRE bundled with JetBrains IDEs, which is a specific version of the JRE tailored for use with JetBrains IDEs.

   Windows

   The `cacerts` file path for a JetBrains IDE installed on Windows is:

   ```
   <JetBrains Installation Folder>\jbr\bin\cacerts
   ```

For example, if you installed a JetBrains IDE on Windows in the default location, the path might be:

```
C:\Program Files\JetBrains\jbr\bin\cacerts
```

macOS

The `cacerts` file path for a JetBrains IDE installed on macOS is:

```
/Applications/JetBrains Toolbox/<version>/JetBrains Toolbox.app/Contents/jbr/
Contents/Home/lib/security/cacerts
```

For example, if you installed a JetBrains IDE on macOS in the default location, the path might be:

```
/Applications/JetBrains Toolbox/2022.3.4/JetBrains Toolbox.app/Contents/jbr/
Contents/Home/lib/security/cacerts
```

Linux

The `cacerts` file path for a JetBrains IDE installed on Linux is:

```
/opt/jetbrains/jbr/lib/security/cacerts
```

2. Determine the certificate you need to import to the `cacerts` file. The certificate file typically has a `.cer`, `.crt`, or `.der` file extension. If you aren't sure which certificates you need to add, contact your network administrator.
3. Import the certificate to the `cacerts` keystore. You can do this with the Java `keytool` command.

   a. Open a command prompt and enter the following command:

   ```
   keytool -import -alias <alias> -file <certificate_file> -keystore
    <path_to_cacerts>
   ```

   b. For `<alias>`, you can add a name for the certificate you are importing to refer to it later. This option is optional.

c. For <certificate_file>, specify the path to the certificate you are importing. This should be a path to the .cer, .crt, or .der file containing the certificate.

d. For <path_to_cacerts>, specify the path to the cacerts keystore file you saved in step 1. This is the file where you are importing the certificate.

For example, if you want to import a certificate named my_certificate.cer into the cacerts keystore of the bundled JRE in IntelliJ IDEA on Windows, and you want to give the alias myalias to the certificate, the command might be:

```
keytool -import -alias myalias -file my_certificate.cer -keystore "C:\Program Files
\JetBrains\IntelliJ IDEA 2022.3.2\jbr\bin\cacerts"
```

4. During the import process, you'll be prompted to enter the keystore password. The default password for the cacerts keystore is changeit.
5. After running the command, you'll be asked to trust the certificate. To confirm the certificate is trusted and complete the import, enter yes.
6. You might also need to add the certificates to the IDE itself, in addition to the JRE. For more information, see Server Certificates in the JetBrains documentation.

Configure certificates in Visual Studio Code

To configure Visual Studio Code to trust the SSL or TLS certificates used by your proxy server, make sure you have configured the following proxy settings for your operating system.

Configure certificates in Visual Studio Code on macOS

Configure the following proxy settings for Visual Studio Code on macOS.

Add certificates to your macOS keychain

If you haven't already, you must add the certificates used by your proxy server to your macOS keychain. For information on adding certificates to your keychain, see Add certificates to a keychain using Keychain Access on Mac in the Keychain Access User Guide.

Install the Mac CA VSCode extension

The Mac CA VSCode extension allows Amazon Q to access the certificates you added to Keychain Access on your Mac.

To install the extension:

1. Search for `mac-ca-vscode` in the VS Code extensions pane, and choose **Install**.
2. Restart VS Code.

Update proxy settings in VS Code on macOS

Update the following settings to make sure VS Code is configured properly for your proxy.

1. Open settings in VS Code.
2. Enter `proxy` in the search bar.
3. In the **Http: Proxy** field, add your proxy URL.
4. Deselect **Http: Proxy Strict SSL**.
5. In the **Http: Proxy Support** dropdown list, choose **on**.
6. In the settings search bar, enter `http.experimental.systemCertificatesV2`. Select **Http › Experimental: System Certificates V2**.

Configure certificates in Visual Studio Code on Windows

Configure the following proxy settings for Visual Studio Code on Windows.

Add certificate as a trusted root certificate on Windows

If you haven't already, you must add the certificates used by your proxy server to your Trusted Root Certification Authorities store on Windows. To add a certificate, complete the following procedure:

1. Open the search tool or a Run command window.
2. Enter the following to open the Certificate Manager tool:

   ```
   certmgr.msc
   ```

3. Choose the **Trusted Root Certification Authorities** store.
4. Right-click **Certificates**, choose **All Tasks**, and then choose **Import...**.
5. Follow the instructions given to import your proxy certificate.
6. After you've imported your certificate, confirm the certificate was added.

In the **Trusted Root Certification Authorities** store, double click **Certificates**. Right-click the certificate you added and choose **Properties**. Under **Certificate purposes**, the option **Enable all purposes for this certificate** should be selected.

Install the Win-CA VSCode extension

The Win-CA VSCode extension allows Amazon Q to access the certificates you added to Trusted Root Certificates in Windows.

To install the extension:

1. Search for `win-ca` in the VS Code settings pane.
2. In the **Inject** dropdown list, choose **append**.

Update proxy settings in VS Code on Windows

Update the following settings to make sure VS Code is configured properly for your proxy.

1. Open settings in VS Code.
2. Enter `proxy` in the search bar.
3. In the **Http: Proxy** field, add your proxy URL.
4. Deselect **Http: Proxy Strict SSL**.
5. In the **Http: Proxy Support** dropdown list, choose **on**.
6. In the settings search bar, enter `http.experimental.systemCertificatesV2`. Select **Http › Experimental: System Certificates V2**.
7. Restart VS Code.

Allow access to Amazon S3

During a transformation, Amazon Q uploads your code to a service-owned Amazon S3 bucket. If your network or organization hasn't configured access to Amazon S3, Amazon Q isn't able to upload your project.

To ensure Amazon Q can upload your project, make sure your proxy configuration and other network components, such as Data Lost Prevention (DLP) policies, are configured to allow access to

Amazon S3. You might also need to allowlist the Amazon S3 bucket where Amazon Q uploads your project. For more information, see Amazon S3 bucket URLs and ARNs to allowlist.

If you transform a large project, DLP policies or other network components might cause delays and prevent a successful upload if they aren't configured to allowlist the Amazon S3 bucket. If you choose not to allowlist the bucket, you might need to transform a smaller project so that Amazon Q can upload it.

Why are my Maven commands failing?

Following are Maven configuration issues that you might see in the JetBrains and Visual Studio Code IDEs. If you address the issues and still see Maven errors, there might be an issue with your project. Use the information in the error logs to address any issues with your project, and then try transforming your project again.

Update Maven configuration in JetBrains

If a transformation fails in JetBrains due to Maven command issues, the error logs appear on the **Run** tab. Use the information in the logs to address the issue. Following are some issues that you might need to address:

- Make sure that your Maven home path is set to **Bundled**. Go to **Settings**, and then expand the **Build, Execution, Deployment** section. Expand the **Build Tools** section and then expand **Maven**. In the **Maven home path** dropdown list, choose **Bundled**.
- Make sure that the Java runtime environment (JRE) is using your project JDK. Go to **Settings**, and then expand the **Build, Execution, Deployment** section. Expand **Maven** and choose **Runner**. In the **JRE** dropdown list, choose **Use Project JDK**.
- Make sure that Maven is enabled. Go to **Settings** and choose **Plugins**. Search for Maven and choose the Maven plugin. If you see an **Enable** button, choose it to enable Maven.

Update Maven configuration in Visual Studio Code

If a transformation fails in VS Code because of Maven command issues, a text file that contains the error logs opens in a new tab. Use the information in the logs to address the issue.

Make sure that you have configured either one of the following options:

- Your project contains a Maven wrapper in the project root folder
- A version of Maven supported by Amazon Q is available on your PATH

For more information, see How do I add Maven to my PATH?

How do I add Maven to my PATH?

To transform your code in VS Code without using a Maven wrapper, you must install Maven and add it to your PATH variable.

To check if you have Maven installed correctly already, run `mvn -v` in a new OS terminal outside of Visual Studio Code. You should see an output with your Maven version.

If you get an output in your Visual Studio Code terminal but not in your OS terminal, or if the command isn't found, you need to add Maven to your PATH.

To add Maven to your PATH, follow the instructions for your machine.

macOS

To add Maven to your macOS PATH, complete the following steps.

1. Locate your Maven installation directory, or the folder where you installed Maven, and save the path to that folder.
2. Open the configuration file for your shell in an editor of your choice. For recent macOS versions, the default shell is `zsh` and the default configuration file is located at `~/.zshrc`.

   Add the following lines to the bottom of the configuration file. Set the value of `M2_HOME` to the path you saved in step 1:

   ```
   export M2_HOME="your Maven installation directory"
   export PATH="${M2_HOME}/bin:${PATH}"
   ```

   These commands make the `mvn` command available in all terminals.
3. Close all OS terminal windows and quit all Visual Studio Code instances.
4. To verify that Maven was added to your PATH, open a new OS terminal and run the following command:

   ```
   mvn -v
   ```

   You should see an output with your Maven version.
5. After seeing your Maven output, restart Visual Studio Code. You might also need to restart your machine. Open a new Visual Studio Code terminal and run the following command:

```
mvn -v
```

The output should be identical to the output in step 4. If the Visual Studio Code output is different, try the following to make sure your setup is correct:

- Check your PATH variable in Visual Studio Code. An IDE extension might be altering the PATH such that it differs from your local PATH variable. Uninstall the extension to remove it from your PATH.
- Check your default shell in Visual Studio Code. If it's set to something other than `zsh`, repeat these steps for your shell.

Windows

To add Maven to your Windows PATH, complete the following steps:

1. Locate your Maven installation directory, or the folder where you installed Maven, and save the path to that folder.
2. Open the Environment Variables window:
   a. Choose the Windows button to open the search bar.
   b. Enter `Edit environment variables for your account` and choose it.
3. In the **Environment Variables** window, look for the Path variable. If you have a Path variable already, choose **Edit...** to update it. If you don't see a Path variable, choose **New...** to add one.
4. In the **Edit environment variable** window that appears, double click the existing path to edit it, or choose **New** to add a new path entry.

   Replace the existing Maven path entry with the path you saved in step 1, or add the path as a new entry. At the end of the path, add `\bin` as a suffix, as in the following example:

   ```
   C:\Users\yourusername\Downloads\apache-maven-3.9.6-bin\apache-maven-3.9.6\bin
   ```
5. Choose **OK** to save the path entry, and then choose **OK** again in the **Environment Variables** window.
6. Open a new Command Prompt and run the following command:

```
mvn -v
```

You should see an output with your Maven version.

Why can't Amazon Q build my code?

If the transformation fails when Amazon Q is building your code, your project may not be configured properly for the environment where Amazon Q builds your code. You might need to update your build configuration or code implementation.

Review the build log output Amazon Q provides to determine if there are changes you can make to your project. Following are some common issues that might prevent Amazon Q from building your code.

Remove absolute paths in pom.xml

If you have an absolute path in your pom.xml file, Amazon Q won't be able to find the relevant files, and as a result might not be able to build your code.

Following is an example of an absolute path that you could have in your `pom.xml` file:

```
<toolspath>
    <path>/Library/Java/JavaVirtualMachines/jdk-11.0.11.jdk/Contents/Home/lib/
tools.jar</path>
</toolspath>
```

Instead of using an absolute path, you can create a relative path using a pointer. Following is an example of how you can replace the previous absolute path with a relative path:

```
<toolspath>
    <path>${java.home}/../lib/tools.jar</path>
</toolspath>
```

Remove local or external databases in unit tests

Amazon Q runs any unit tests in your project when it builds your code. If a unit test calls a local or external database, Amazon Q won't have access to the database, causing the build to fail. To prevent the build from failing, you must either remove the database call from the unit test or remove the unit test before submitting the transformation.

Why did my transformation fail after 55 minutes?

If your code transformation job fails after 55 minutes, your code build time likely exceeds the build time limit. There is currently a time limit of 55 minutes for building your code.

If your local build time takes 55 minutes or longer, reduce your project's build time to transform your code. If your local build is faster than the build with Code Transformation, check your project for tasks that might be failing or take a longer time in a different environment. Consider disabling long-running test cases. Also consider using timeouts for attempts to access resources that might not be available from the secure IDE environment or the internet.

Why can't I download my transformed code?

If you aren't able to download your code after your transformation is complete, it's likely due to one of the following issues. See the topic that corresponds to the error you see from Amazon Q.

Topics

- Reduce project size
- Download code diff within 30 days
- Configure proxy settings in your IDE
- Remove wildcard characters in JetBrains proxy settings

Reduce project size

After the transformation is complete, Amazon Q generates an output artifact that contains a diff with your upgraded code and a transformation summary with information about the changes it made. The output artifact must be 1 GB or less in order for the IDE to download it.

If the output artifact exceeds the limit, you will not be able to download your upgraded code or transformation summary. Try transforming a smaller project to prevent a large output artifact. If the issue persists, contact Support. For information about contacting Support with Amazon Q, see Using Amazon Q Developer to chat with Support.

Download code diff within 30 days

The code diff file with your upgraded code is only available for 30 days after the transformation is complete. If it's been over 30 days since the transformation completed, restart the transformation to download the diff file.

Configure proxy settings in your IDE

Amazon Q downloads your upgraded code from a service-owned Amazon S3 bucket. Part of the download process involves using SSL or TLS certificates to establish communication between Amazon S3 and your IDE. If you are using a proxy server, the SSL or TLS certificates used by your proxy server must be trusted, otherwise Amazon Q is not able to upload your project.

To download your code, you might need to configure your IDE to trust certificates or update other proxy settings. For more information on updating your proxy settings, see Configure proxy settings in your IDE.

Remove wildcard characters in JetBrains proxy settings

If you have configured proxy settings in your JetBrains IDE, you might see the following error when downloading your upgraded code:

```
software.amazon.awssdk.core.exception.SdkClientException:
Unable to execute HTTP request: Dangling meta character '*' near index 0
```

This is likely caused by the presence of a wildcard character (*) in the **No proxy for** field of your IDE's proxy settings. The Java SDK used by Amazon Q doesn't support wildcard entries in this field.

To download your code, remove any wildcards from the **No proxy for** field, and then restart your IDE. If you need to specify hosts that should bypass the proxy, use a regular expression instead of a wildcard. To update proxy settings in your JetBrains IDE, see HTTP Proxy in the JetBrains documentation.

How do I access code transformation logs?

Access logs in JetBrains

For information about how to access JetBrains log files, see Locating IDE log files in the JetBrains documentation.

To find logs emitted by Amazon Q in JetBrains, search the IDE logs for the following string:

```
software.aws.toolkits.jetbrains.services.codemodernizer
```

Code transformation logs start with the preceding string. Logs generated by Maven are displayed on the **Run** tab and have the preceding string before and after the log entry.

Access logs in Visual Studio Code

To find logs emitted by Amazon Q in VS Code, complete the following steps:

1. In the IDE menu bar, choose **View**, and then choose **Command Palette**.
2. Search `Amazon Q: View Logs` in the command palette that appears.
3. The logs open in the IDE. To search the log files for `CodeTransformation`, use `CMD + F` or `Control + F`.

Code transformation logs in VS Code are prefixed with `CodeTransformation:`. Following is an example of a log generated in VS Code for a Maven copy dependencies error:

```
2024-02-12 11:29:16 [ERROR]: CodeTransformation: Error in running Maven copy-
dependencies command mvn = /bin/sh: mvn: command not found
```

How do I find my transformation job ID?

Find your job ID in JetBrains

To find a transformation job ID in JetBrains, go to the **Transformation details** tab in the **Transformation Hub** and choose the **Show Job Status** (clock) icon.

Find your job ID in Visual Studio Code

To find a transformation job ID in VS Code, go to the **Transformation Hub** and choose the **Show Job Status** (clock) icon.

Transforming .NET applications with Amazon Q Developer

Amazon Q Developer can port your Windows-based .NET applications to Linux-compatible cross-platform .NET applications through a generative AI-powered refactoring workflow. Amazon Q also helps you upgrade outdated versions of cross-platform .NET applications to newer versions.

To transform a .NET solution or project, Amazon Q analyzes your codebase, determines the necessary updates to port your application, and generates a transformation plan before the transformation begins. During this analysis, Amazon Q divides your .NET solution or project into code groups that you can view in the transformation plan. A *code group* is a project and all its dependencies that together generate a buildable unit of code such as a dynamic link library (DLL) or an executable.

During the transformation, Amazon Q provides step-by-step updates in a Transformation Hub where you can monitor progress. After transforming your application, Amazon Q generates a summary with the proposed changes in a diff view for you to optionally verify the changes before you accept them. When you accept the changes, Amazon Q makes in-place updates to your .NET solution or project.

Amazon Q performs four keys tasks to port .NET applications to Linux:

- **Upgrades language version** – Replaces outdated C# versions of code with Linux-compatible C# versions.
- **Migrates from .NET Framework to cross-platform .NET** – Migrates projects and packages from Windows dependent .NET Framework to cross-platform .NET compatible with Linux.
- **Rewrites code for Linux compatibility** – Refactors and rewrites deprecated and inefficient code components.
- **Generates a Linux compatibility readiness report** – For open-ended tasks where user intervention is needed to make the code build and run on Linux, Amazon Q provides a detailed report of actions needed to configure your application after transformation.

For more information about how Amazon Q performs .NET transformations, see How it works.

Topics

- Quotas
- Porting a .NET application with Amazon Q Developer in Visual Studio
- How Amazon Q Developer transforms .NET applications
- Troubleshooting issues with .NET transformations in the IDE

Quotas

.NET transformations with Amazon Q in the IDE maintain the following quotas:

- **Lines of code per job** – The maximum number of code lines that Amazon Q can transform in a given transformation job. This is also the monthly total limit for .NET transformations.
- **Concurrent jobs** – The maximum number of transformation jobs you can run at the same time. This quota applies to all transformations in the IDE, including Java transformations.

Resource	Quotas
Lines of code per job	100,000 lines of code
Concurrent jobs	1 job per user 2 jobs per AWS account

Porting a .NET application with Amazon Q Developer in Visual Studio

Complete these steps to port a Windows-based .NET application to a Linux-compatible cross-platform .NET application with Amazon Q Developer in Visual Studio.

Step 1: Prerequisites

Before you continue, make sure you've completed the steps in Set up Amazon Q in your IDE.

Make sure that the following prerequisites for your application are met before you begin a .NET transformation job:

- Your application contains only .NET projects written in C#.
- Your application only has Microsoft-authored NuGet package dependencies
- Your application only uses UTF-8 characters. If your application uses non-UTF-8 characters, Amazon Q will still attempt to transform your code.
- If your application is dependent on Internet Information Services (IIS), only default IIS configurations are used
- Amazon Q will evaluate the type of the project you selected and its dependencies to create a code group. Your code group can only have the following project types:
  - Console application
  - Class library
  - Web API
  - WCF Service
  - Business logic layers of Model View Controller (MVC) and Single Page Application (SPA)
  - Test projects

> **Note**
>
> Amazon Q doesn't support transforming UI layer components such as Razor views or WebForms ASPX files. If Amazon Q detects UI layer components in your solution or project, it will perform a partial transformation by excluding UI layer components, and you might need to refactor further to make your code buildable on the target .NET version.

Step 2: Transform your application

To transform your .NET solution or project, complete the following procedure:

1. Open any C# based solution or project in Visual Studio that you want to transform.
2. Open any C# code file in the editor.
3. Choose **Solution Explorer**.
4. From the Solution Explorer, right click a solution or project you want to transform, and then choose **Port with Amazon Q Developer**.
5. The **Port with Amazon Q Developer** window appears.

   The solution or project you selected will be chosen in the **Choose a solution or project to transform** dropdown menu. You can expand the menu to choose a different solution or project to transform.

   In the **Choose a .NET target** dropdown menu, choose the .NET version you want to upgrade to.
6. Choose **Confirm** to begin the transformation.
7. Amazon Q begins transforming your code. You can view the transformation plan it generates for details about how it will transform your application.

   A **Transformation Hub** opens where you can monitor progress for the duration of the transformation. After Amazon Q has completed the **Awaiting job transformation startup** step, you can navigate away from the project or solution for the duration of the transformation.
8. After the transformation is complete, navigate to the **Transformation Hub** and choose **View diffs** to review the proposed changes from Amazon Q in a diff view.
9. Choose **View code transformation summary** for details about the changes Amazon Q made. You can also download the transformation summary by choosing **Download summary as .md**.

If any of the items in the **Code groups** table require input under the Linux porting status, you must manually update some files to run your application on Linux.

a. From the **Actions** dropdown menu, choose **Download Linux readiness report**.

b. A .csv file opens with any changes to your project or solution that you must complete before your application is Linux compatible. It includes the project and file that need to be updated, a description of the item to be updated, and an explanation of the issue. Use the **Recommendation** column for ideas on how to address a Linux readiness issue.

10. To update your files in place, choose **Accept changes** from the **Actions** dropdown menu.

How Amazon Q Developer transforms .NET applications

Review the following sections for details about how .NET transformation with Amazon Q Developer works.

Analyzing your application and generating a transformation plan

Before a transformation begins, Amazon Q builds your code locally to ensure it's buildable and configured correctly for transformation. Amazon Q then uploads your code to a secure and encrypted build environment on AWS, analyzes your codebase, and determines the necessary updates to port your application.

During this analysis, Amazon Q divides your .NET solution or project into code groups. A code group is a project and all its dependencies that together generate a buildable unit of code such as a dynamic link library (DLL) or an executable. Even if you didn't select all project dependencies to be transformed, Amazon Q determines the dependencies needed to build your selected projects and transforms them too, so that your transformed application will be buildable and ready for use.

After analyzing your code, Amazon Q generates a transformation plan that outlines the proposed changes that it will make, including a list of code groups and their dependencies that will be transformed.

Transforming your application

To start the transformation, Amazon Q builds your code again in the secure build environment to ensure it's buildable remotely. Amazon Q then begins porting your application. It works from the bottom up, starting with the lowest level dependency. If Amazon Q runs into an issue with porting a dependency, it stops the transformation and provides information about what caused the error.

The transformation includes the following updates to your application:

- Replacing outdated C# versions of code with Linux-compatible C# versions
- Upgrading .NET Framework to cross-platform .NET, including:
  - Identifying and iteratively replacing packages, libraries, and APIs
  - Upgrading and replacing NuGet packages and APIs
  - Transitioning to cross-platform runtime
  - Setting up middleware and updating runtime configurations
  - Replacing private or third-party packages
  - Handling IIS and WCF components
  - Debugging build errors
- Rewriting code for Linux compatibility, including refactoring and rewriting deprecated and inefficient code to port existing code

Reviewing transformation summary and accepting changes

After the transformation is complete, Amazon Q provides a transformation summary with information about the proposed updates it made to your application, including the number of files changed, packages updated, and APIs changed. It flags any unsuccessful transformations, including affected files or portions of files and the errors encountered during an attempted build. You can also view a build summary with build logs to learn more about what changes were made.

The transformation summary also provides a Linux porting status, which indicates whether or not additional user input is needed to make the application Linux compatible. If any of the items in a code group require input from you, you download a Linux readiness report that contains Windows-specific considerations that Amazon Q could not address at build time. If input is needed for any code groups or files, review the report for details about what type of change still needs to be made and, if applicable, for recommendations for how to update your code. These changes must be made manually before your application can be run on Linux.

You can review the proposed changes Amazon Q made in a diff view before accepting them as in-place updates to your files. After updating your files and addressing any items in the Linux readiness report, your application is ready to run on cross-platform .NET.

Troubleshooting issues with .NET transformations in the IDE

Use the following sections to troubleshoot common issues with .NET transformations in the IDE with Amazon Q Developer.

How do I know if a job is progressing?

If Amazon Q appears to be spending a long time on a step in the Transformation Hub, you can check whether the job is still active in the output logs. If diagnostic messages are being generated, the job is still active.

To check the outputs, choose the **Output** tab in Visual Studio. In the **Show output from:** menu, choose **Amazon Q Language Client**.

The following screenshot shows an example of the outputs Amazon Q generates during a transformation.



Why are some projects not selected for transformation?

Amazon Q can only transform supported project types in the C# language. Currently, Amazon Q does not support porting UI layer components or projects written in the VB.NET or F# languages. For a list of supported project types and other prerequisites for transforming your .NET projects, see Step 1: Prerequisites.

How can I get support if my project or solution isn't transforming?

If you aren't able to troubleshoot issues on your own, you can reach out to Support or your AWS account team to submit a support case.

To get support, provide the transformation job ID so AWS can investigate a failed job. To find a transformation job ID, choose the **Output** tab in Visual Studio. In the **Show output from:** menu, choose **Amazon Q Language Client**.

How can I prevent my firewall from interfering with transformation jobs?

If your organization uses a firewall, it might interfere with transformations in Visual Studio. You can temporarily disable security checks in Node.js to troubleshoot or test what is preventing the transformation from running.

The environment variable `NODE_TLS_REJECT_UNAUTHORIZED` controls important security checks. Setting `NODE_TLS_REJECT_UNAUTHORIZED` to "0" disables Node.js's rejection of unauthorized TLS/SSL certificates. This means:

- Self-signed certificates will be accepted
- Expired certificates will be allowed
- Certificates with mismatched hostnames will be permitted
- Any other certificate validation errors will be ignored

If your proxy uses a self-certificate, you can set the following environment variables instead of disabling `NODE_TLS_REJECT_UNAUTHORIZED`:

```
NODE_OPTIONS = –use-openssl-ca
NODE_EXTRA_CA_CERTS = Path/To/Corporate/Certs
```

Otherwise, you must specify the CA certs used by the proxy to disable `NODE_TLS_REJECT_UNAUTHORIZED`.

To disable NODE_TLS_REJECT_UNAUTHORIZED on Windows:

1. Open the Start menu and search for **Environment Variables**.
2. Choose **Edit the system environment variables**.
3. In the **System Properties** window, choose **Environment Variables**.
4. Under **System variables**, choose **New**.
5. Set **Variable name** to NODE_TLS_REJECT_UNAUTHORIZED and **Variable value** to 0.
6. Choose **OK** to save the changes.
7. Restart Visual Studio.

Explaining and updating code with Amazon Q Developer

Amazon Q Developer can explain and update specific lines of code in your integrated development environment (IDE). To update your code, ask Amazon Q to make changes to a given line or block of code, and it will generate new code that reflects the changes that you asked it to make. Then, you can insert the updated code directly into the file where the code originated.

You can choose from the following options:

- **Explain** – Get your code explained in natural language.
- **Refactor** – Improve code readability or efficiency, among other improvements.
- **Fix** – Debug code.
- **Generate tests** – Create unit tests for the current file or selected code.
- **Optimize** – Enhance code performance.
- **Send to prompt** – Send the highlighted code to the Amazon Q chat panel, and ask questions that you have about the code.

Send code to Amazon Q

To get your code explained or updated by Amazon Q, complete the following steps.

1. Highlight a section of a code file in your IDE.
2. Right-click your highlighted code to open a context window. Choose **Amazon Q**, and then choose **Explain**, **Refactor**, **Fix**, **Generate tests**, **Optimize**, or **Send to prompt**.

   If you choose **Send to prompt**, Amazon Q copies the highlighted code to the chat panel, where you can enter questions that you have about the code.
3. To replace the highlighted code with the newly generated code, you can copy the code or insert it directly into your file by choosing **Insert code**. Amazon Q replaces the original code with the updated code.

Chatting inline with Amazon Q Developer

The *inline chat* feature lets you chat with Amazon Q from your IDE's main coding window. To use the inline chat feature, you highlight code that you want suggestions for, and provide instructions

in the small input screen. Amazon Q proceeds to generate code for you, which it presents in a diff within the main coding window. You can then choose to accept or reject the changes.

The advantage of inline chat is that it eliminates the context switching that occurs when moving between a chat window and the main coding window.

You would typically use the inline chat feature when you're reviewing code, writing unit tests, or performing other tasks that require code-based answers. For situations where you want text-based answers (for example, an answer to "Explain this code") then using the chat window is a better option.

Amazon Q considers the code in the current file when generating a code recommendation through the inline chat. It won't look at code in other files or projects.

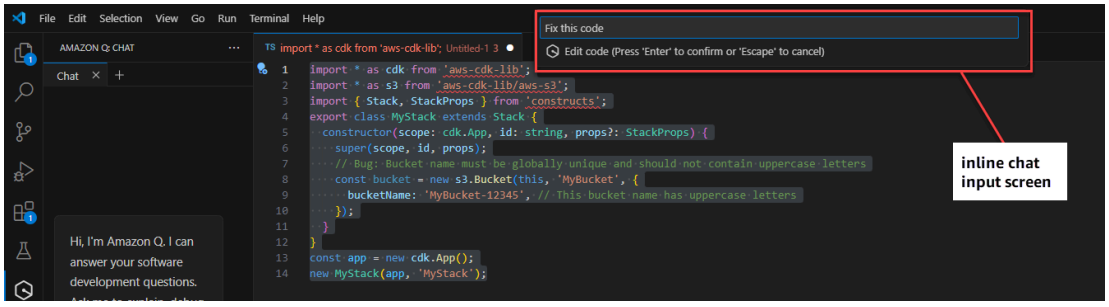
Amazon Q inline chat in action

An inline chat session unfolds as follows.

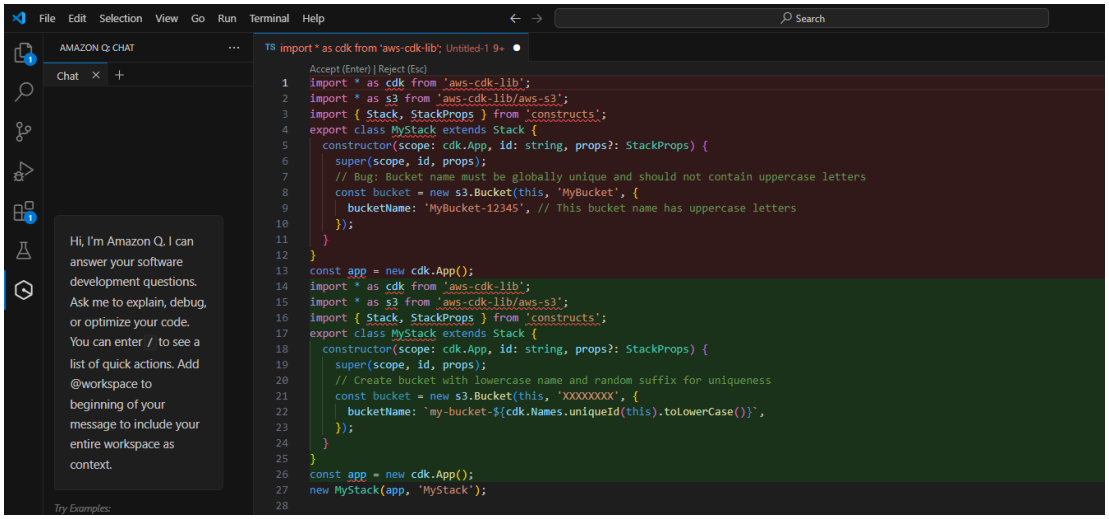
1. You highlight the code that you want suggestions for, and then choose from the following options based on your IDE:

   - In Visual Studio Code and JetBrains, press `#+I` (Mac) or `Ctrl+I` (Windows)
   - In Eclipse, press `#+Shift+I` (Mac) or `Ctrl+Shift+I` (Windows)
   - Alternatively, you can right-click the selection and choose **Amazon Q** and then **Inline chat**

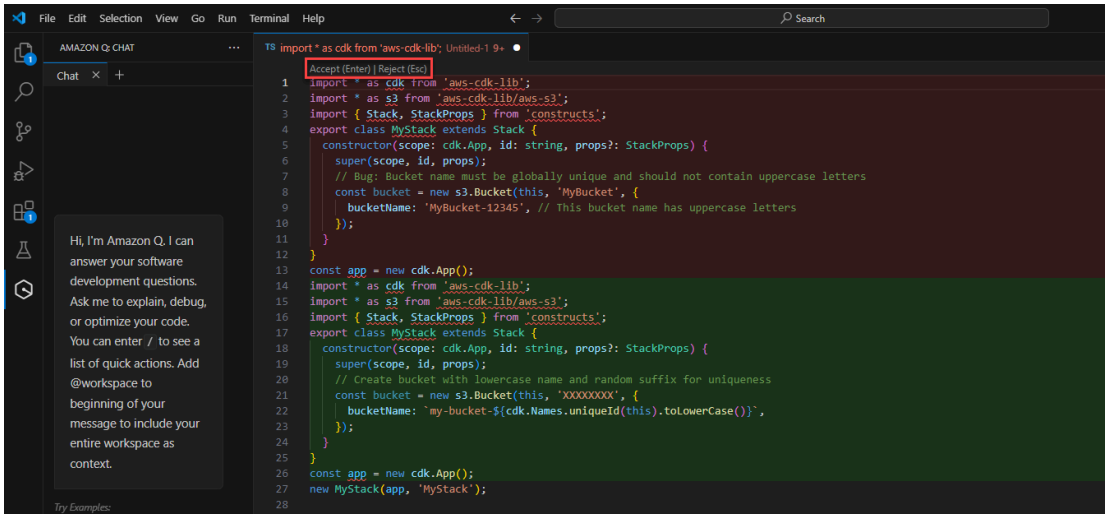
   This launches an input field in the editor where you can enter a prompt, such as **`Fix this code`**.



2. Amazon Q generates code and presents it in a diff.



3. You accept or reject the change by choosing **Accept** or **Reject**, or by pressing the keyboard equivalents (`Enter` or `Esc`).



Example topics and questions

The inline chat always returns code as the answer, so you can enter prompts like:

- Document this code
- Refactor this code
- Write unit tests for this function

Diff format

The inline chat displays the diff in multiple blocks, with the existing code on the top, and the suggested code on the bottom. A side-by-side diff is not supported.

Adding context to Amazon Q Developer chat in the IDE

When you chat with Amazon Q in the integrated development environment (IDE), you can provide Amazon Q with additional *context*, such as files and folders, that Amazon Q can use to tailor and improve its answers.

There are two ways to provide context to Amazon Q:

- **Explicitly** – To provide context explicitly, you enter **@** in the chat window. The **@** launches a context picker pop-up from which you select items to include as context. Alternatively, you can type **@** and begin typing the name of the file, folder, or other context type to have it auto-complete. For more information, see Explicit context types.
- **Automatically** – To provide context automatically, you set up the context separately, outside of the chat. Amazon Q automatically references the context whenever any developer working on the project types a question into the chat window. For more information, see Automatic context types.

After Amazon Q generates an answer, it shows you the files it used as context in the **Context** drop-down list, which appears immediately above the start of the answer.

Explicit context types

When you type **@** in the chat, you can select from the following context types:

- **@workspace** – Amazon Q uses your project's workspace as context for its answers. The **@workspace** option requires configuration. For more information, see Adding workspace context to Amazon Q Developer chat in the IDE.
- **Folders** – Amazon Q shows you a list of folders in the current project, and uses the folder you select as context for its answers.
- **Files** – Amazon Q shows you a list of files in the current project, and uses the file you select as context for its answers.
- **Code** – Amazon Q shows you a list of classes, functions, global variables in the current project, and uses your selection as context for its answers.

- **Images** – Amazon Q allows you to add images as context for your prompts, useful for scenarios like generating code from UI mockups or sequence diagrams. Images must be in JPEG, PNG, GIF, or WebP format, with a maximum size of 3.75 MB and dimensions not exceeding 8,000 x 8,000 pixels. You can include up to 20 images in a single message, including any images pinned to context.
- **Prompts** – Amazon Q shows you a list of prompts that you have saved, and uses the prompt you select as context for its answers. The **Prompts** option requires some configuration. For more information, see Saving prompts to a library for use with Amazon Q Developer chat.

Automatic context types

The following types of contexts will be used automatically by Amazon Q, if you've set them up:

- **Project rules** – Amazon Q will automatically use a set of project rules that you define as context. For more information, see Creating project rules for use with Amazon Q Developer chat.
- **Customizations** – Amazon Q will automatically use a repository of source code as context.

Adding workspace context to Amazon Q Developer chat in the IDE

> **Important**
>
> The **`@workspace`** feature is deprecated. You can still add context by using **`@file`**, **`@folder`**, or **`@code`**.

When you chat with Amazon Q in the integrated development environment (IDE), you can add **`@workspace`** to your question to automatically include the most relevant chunks of your workspace code as context. Amazon Q Developer determines relevance based on an index that is updated periodically.

With workspace context, Amazon Q has enhanced capabilities, including locating files, understanding how code is used across files, and generating code that leverages multiple files, including files that aren't opened.

Topics

- Setup
- Ask questions with workspace context

Setup

Before you continue, make sure you have the latest version of your IDE installed. You can then complete the following setup steps.

Enable indexing

To use your workspace as context, Amazon Q creates a local index of your workspace repository, including code files, configuration files, and project structure. During indexing, non-essential files like binaries or those specified in `.gitignore` files are filtered out.

It can take 5 to 20 minutes to index a new workspace. During this time, you can expect elevated CPU usage in your IDE. After initial indexing, the index is incrementally updated when you make changes to your workspace.

The first time you add workspace context, you must enable indexing in your IDE. Complete the following steps to enable indexing:

1. Add **`@workspace`** to your question in the Amazon Q chat panel.
2. Amazon Q prompts you to enable indexing. Choose **Settings** to be redirected to Amazon Q settings in your IDE.

   If you aren't prompted, you can go to settings by choosing **Amazon Q** in the IDE status bar. Then, choose **Open Settings** from the Amazon Q task bar that opens.
3. Select the box next to **Workspace Index**.

Configure indexing (optional)

No configuration is necessary for the indexing process, however you can choose to specify the number of threads dedicated to indexing. If you increase the number of threads used, indexing will complete faster, and it will use more of your CPU. To update the indexing configuration, specify the number of threads for the **Workspace Index Worker Threads** setting. You can also set the maximum size of the files that can be indexed for workspace context, and enable the use of your graphics processing unit (GPU) for indexing.

Ask questions with workspace context

To add your workspace as context to your conversation with Amazon Q, open the workspace you want to ask questions about, and then add **`@workspace`** to your question in the chat panel. You must add @workspace to any question that you want to add workspace context to.

If you want to start chatting about a different workspace, open the workspace, and then open a new chat tab. Include **`@workspace`** in your question to add the new workspace as context.

You can ask Amazon Q about any file in your workspace, including unopened files. Amazon Q can explain files, locate code, and generate code across files, in addition to existing conversational coding capabilities.

Following are example questions you can ask Amazon Q that leverage workspace context in the chat:

- @workspace where is the code that handles authorization?
- @workspace what are the key classes with application logic in this project?
- @workspace explain main.py
- @workspace add auth to this project
- @workspace what third-party libraries or packages are used in this project, and for what purpose?
- @workspace add unit tests for function *<function name>*

Saving prompts to a library for use with Amazon Q Developer chat

You can build a library of common prompts that you can use when chatting with Amazon Q in the IDE. By storing these prompts in your library, you can easily insert them into the chat without having to retype the prompt each time. You can use saved prompts across multiple conversations and projects.

Prompts are saved in the `~/.aws/amazonq/prompts` folder.

To save a prompt to a prompt library

1. In your IDE, open an Amazon Q chat window.
2. Type **@**, and select **Prompts**.
3. Choose **Create a new prompt**. (You might have to scroll down to find it.)
4. In **Prompt name**, enter a prompt name such as **`Create_sequence_diagram`** and press Enter. Note that prompt names cannot include spaces.

   Amazon Q creates a prompt file called `Create_sequence_diagram.md` in the `~/.aws/amazonq/prompts` folder, and opens the file in your IDE.

5. In the prompt file, add a detailed prompt. For example:

   ```
   Create a sequence diagram using Mermaid that shows the sequence of
   calls between resources. Ignore supporting resources like IAM policies
   and security group rules.
   ```

6. Save the prompt file.

To use a saved prompt

1. In your IDE, open an Amazon Q chat window.
2. Type **@**, and select **Prompts**.
3. Choose your saved prompt, for example, **Create_sequence_diagram**.
4. (Optional) In the chat input window, add details, as required. You can type more text and add more context types. An example prompt might look like this...

   `@Create_sequence_diagram` `using the files in the` **`@lib`** `folder`

5. Submit the prompt and wait for Amazon Q to generate an answer.

Pinning context items

> **Note**
>
> Context pinning is currently only available in the VS Code IDE.

Context pinning lets you specify context items that get added to all messages within your chat session. When you pin a context item, it's automatically included in every message within your current conversation, eliminating the need to repeatedly type commands like **`@workspace`**, **`@file`**, or **`@folder`**.

Pinned items can come from two sources: you can manually pin items you frequently reference, or Amazon Q may automatically add context, (such as your current active file,) to improve response quality. Pinned context items appear in the pinned context area of your chat panel, and you have control to remove any context you don't want included.

To help maintain clear context boundaries, pinned items only apply to your current chat tab. When you open a new tab, you'll start fresh with only the default pinned context, such as the active file.

Using pinned context

To add pinned context items

1. In your IDE, open the Amazon Q chat panel.
2. After using a context command like **`@workspace`**, **`@file`**, **`@folder`**, or **`@prompt`** in a chat, click the desired context to pin it.

   Alternatively, you can click the "@ Pin Context" button to view the available options and select a context to pin.
3. The pinned context will appear in the pinned context area in your chat panel.

Methods to pin context items

There are three ways to pin context items:

1. Using the @Pin Context menu:
   - Click the "@Pin Context" button in your chat panel.
   - Select the desired context item from the available options.
2. Using the context menu and keyboard shortcut:
   - Type "@" in the chat input to bring up the context menu.
   - Navigate to the desired item.
   - Press Option/Alt + Enter to pin the selected item.
3. Pinning from the input prompt:
   - If you've already typed a context command (like **`@workspace`**, **`@file`**, **`@folder`**, or **`@prompt`**) in your input, hover over the context item in your input.
   - Click on the item to pin it.

After pinning, the context item will appear in the pinned context area.

To remove pinned context items

- To remove a pinned context item, choose the remove (X) icon on the context pill. This works for both user-pinned and system-added context items.

Creating project rules for use with Amazon Q Developer chat

You can build a library of project rules that you can use when chatting with Amazon Q in the IDE. These rules describe coding standards and best practices across your team. For example, you could have a rule that states that all Python code must use type hints, or that all Java code must use Javadoc comments. By storing these rules in your project, you can ensure consistency across developers, regardless of their experience level.

Project rules are defined in Markdown files in the project's *project-root*`/.amazonq/rules` folder.

Once you've created your project rules, Amazon Q will automatically use them as context whenever a developer chats with Amazon Q within your project, and will make sure to adhere to them when generating answers. For more information about adding context to the chat, see Adding context to Amazon Q Developer chat in the IDE.

You can create project rules either directly in the file system or through the Amazon Q chat interface.

To create a project rule using the Amazon Q chat interface

1. In your IDE, open the Amazon Q chat panel.
2. In the chat input box, click the **Rules** button.
3. Select **Create new rule**.
4. In the dialog that appears, enter a name for your rule.

   This will create a Markdown file with that name in your project's *project-root*`/.amazonq/rules` folder.
5. Add your rule content in the editor.
6. Save the file.

To create a project rule using the file system

1. In your IDE, open the project's root folder.
2. In the project root folder, create the following folder:

   project-root`/.amazonq/rules`

   This folder holds all your project rules.

3. In *project-root*/.amazonq/rules, create a project rule file. It must be a Markdown file. For example:

   cdk-rules.md

4. Open your project rule Markdown file.
5. Add a detailed prompt to the file. For example:

```
All Amazon S3 buckets must have encryption enabled, enforce SSL, and block public
 access.
All Amazon DynamoDB Streams tables must have encryption enabled.
All Amazon SNS topics must have encryption enabled and enforce SSL.
All Amazon SNS queues must enforce SSL.
```

6. Save the file.
7. (Optional) Add more project rule Markdown files.

   You have now created one or more project rules. Amazon Q will use these rules as context automatically whenever a developer chats with Amazon Q within your project.

To manage rules in the Amazon Q chat interface

1. In your IDE, open the Amazon Q chat panel.
2. In the chat input box, click the **Rules** button to see all available rules.
3. Click on a rule to toggle it on or off for the current chat session:

   - Rules with a check mark are active and will be applied to your conversation.
   - Rules without a check mark are inactive for the current session.

Generating a memory bank for Amazon Q chat

Amazon Q can automatically generate memory bank files that provide a quick index of your project's structure, technology stack, and product information. This feature analyzes key files in your project to create summary files that help Amazon Q understand your codebase without having to analyze the entire project each time you ask a question.

When you generate memory bank files, Amazon Q creates a memory-bank subfolder under .amazonq/rules that contains the following auto-generated files:

- `product.md` – Overview of your project and its capabilities.
- `structure.md` – Your project's architecture, folder organization, and key components.
- `tech.md` – Your technology stack, frameworks, dependencies, and coding standards.
- `guidelines.md` – Development standards and patterns for your project.

These files are automatically used as context when you chat with Amazon Q, providing it with background information about your project.

Generate a memory bank for your project

To generate a memory bank, complete the following procedure.

1. In your IDE, open the Amazon Q chat panel.
2. In the chat input box, choose the **Rules** button.
3. Select **Generate Memory Bank**.
4. A new chat tab opens where Amazon Q begins analyzing your project to create the memory bank files.
5. Once complete, you can view the files by choosing the **Rules** button.

   You can select and deselect individual files to be used as context when you ask a question.
6. If your project changes, you can have Amazon Q generate new memory bank files to update its context. To do so, choose the **Rules** button and then select **Regenerate Memory Bank**.

Customize memory bank generation

You can customize how memory bank files are generated by creating custom project rules. For example, you can create a rule that specifies the language or format for the generated files:

```
When generating the memory bank files like product.md, structure.md, and tech.md,
always generate content in Spanish and include detailed code examples.
```

Save your memory bank rules in a file in your project's `project-root/.amazonq/rules` folder.

For more information about creating custom project rules, see Creating project rules for use with Amazon Q Developer chat.

Chat history compaction in Amazon Q Developer

As you interact with Amazon Q Developer in your IDE, your conversation accumulates in the chat history. This history provides important context that helps Amazon Q understand your project and deliver more relevant responses. However, there are limits to how much conversation history can be included in each request to the underlying model.

Understanding context window limits

The context window represents the maximum amount of information that can be processed in a single interaction with Amazon Q. This includes:

- Your current question or request
- Previous messages in your conversation
- Code snippets and files you've shared
- System information about your project

When this context window approaches its capacity limit, Amazon Q's ability to reference earlier parts of your conversation may be affected.

How chat history compaction works

Chat history compaction allows you to preserve the essential information from your conversation while reducing the amount of context used. When compaction occurs:

1. Amazon Q analyzes your conversation history
2. It creates a concise summary of key points, questions, and decisions
3. This summary replaces the detailed conversation history in the context window
4. Your complete conversation remains visible in the chat interface

Compaction helps you continue your conversation without losing important context, while avoiding the need to start a completely new chat when you reach context window limits.

Using chat history compaction

You can use compaction in two ways:

Manual compaction

To manually compact your chat history:

1. Enter **`/compact`** in the chat input field
2. Amazon Q will process your request and display a confirmation message with a summary of the compacted conversation

Use manual compaction when you want to continue your current conversation but notice slower response times or less relevant answers.

Automatic compaction nudge

When your context window reaches approximately 80% of its capacity, Amazon Q will display a notification suggesting compaction. This notification includes:

- An explanation of why compaction is recommended
- A button to trigger compaction immediately

After compaction

After compaction occurs:

- Your complete conversation history remains visible in the chat interface until the end of the current session
- Amazon Q uses the compacted summary (not the full history) for generating responses
- The compacted summary is included in the context window instead of the detailed history
- The detailed chat history will reset when you restart your IDE

Related commands

Clearing chat history

As an alternative to compaction, you can completely clear your chat history using the **`/clear`** command:

1. Enter **`/clear`** in the chat input field

2. Amazon Q will remove all previous conversation history from both the display and the context window

When to choose compaction vs. clearing history

Choose compaction when:

- You want to continue your current conversation topic
- Previous context is still relevant to your current task
- You want to preserve the general direction and knowledge from your conversation

Choose to clear history when:

- You're starting a completely new task or topic
- Previous conversation is no longer relevant
- You want to ensure no previous context influences new responses
- You want to remove potentially sensitive information from the conversation

Viewing, deleting, and exporting the Amazon Q Developer conversation history

When you chat with Amazon Q in the integrated development environment (IDE), Amazon Q saves each of your chat tabs as a separate conversation. You can view, search for, and delete these conversations. You can also export them to Markdown or HTML format files.

Amazon Q stores your conversations on your local computer, in your home directory.

Amazon Q saves conversations for each workspace separately, so if you don't see your conversation history, it might be because you're in the wrong workspace. Amazon Q only displays the conversation history for the current workspace.

Use the following instructions to view, search for, delete, and export your conversations.

To view and search for past conversations

1. In your IDE, sign in to Amazon Q.
2. Open an Amazon Q chat tab.

3. Open the chat history by doing one of the following:

   - In the chat panel header, choose the **View chat history** button.
   - Press `ctrl+F` (Windows and Linux) or `# F` (Mac).

4. Do one of the following:

   - Choose the conversation you want to view. Conversations are organized by date.
   - Use the search bar near the top of the chat history to find a conversation. Amazon Q finds conversations that match exactly the text you enter.

To delete a single conversation

1. In your IDE, sign in to Amazon Q.
2. Do one of the following:

   - In the chat tab of an open chat session, enter **`/clear`** to delete the contents of the chat tab.
   - Open an Amazon Q chat tab, and then open the chat history by doing one of the following:
     - In the chat panel header, choose the **View chat history** button.
     - Press `ctrl+F` (Windows and Linux) or `# F` (Mac).

     In the conversation that you want to delete, choose the vertical ellipsis (⋮) and choose **Delete**.

To export a conversation to Markdown or HTML

1. In your IDE, sign in to Amazon Q.
2. Do one of the following:

   - With a chat session already started, in the chat panel header, choose the **Export** button to export the conversation displayed in the tab.
   - Open an Amazon Q chat tab, and then open the chat history by doing one of the following:
     - In the chat panel header, choose the **View chat history** button.
     - Press `ctrl+F` (Windows and Linux) or `# F` (Mac).

     In the conversation that you want to export, choose the vertical ellipsis (⋮) and choose **Export** to export the conversation to a Markdown or HTML format file.

By default, Amazon Q names the file `q-dev-chat-yyyy-mm-dd.md|html` and saves it in the root of your project.

Using shortcut keys in chat with Amazon Q Developer

Amazon Q provides keyboard shortcuts to help you interact with the agentic chat interface efficiently. Using keyboard shortcuts can help you maintain your workflow without switching between keyboard and mouse.

> **Note**
>
> Keyboard shortcuts are only currently available in the Visual Studio Code IDE.

Keyboard shortcuts

Keyboard shortcuts for Amazon Q chat

Action	Visual Studio Code Shortcut
Run command	Shift + Cmd + Enter (Mac) Shift + Ctrl + Enter (Windows) Shift + Meta + Enter (Linux)
Reject command	Shift + Cmd + R (Mac) Shift + Ctrl + R (Windows) Shift + Meta + R (Linux)
Stop generation	Shift + Cmd + Backspace (Mac) Shift + Ctrl + Backspace (Windows) Shift + Meta + Backspace (Linux)

Customizing keyboard shortcuts

Shortcuts can be modified through the standard keyboard shortcut customization interface in your IDE. Amazon Q uses each IDE's native shortcut system, so any changes you make will be reflected in the Amazon Q interface. The shortcuts shown in tooltips and UI elements will automatically update to match your customized key mappings.

Shortcut discoverability

Amazon Q provides several ways to discover keyboard shortcuts:

- **Tooltips**: Hover over the Run command button, Reject command button, or Stop button to see the current keyboard shortcut
- **Settings**: View all shortcuts in your IDE's keyboard shortcut customization interface

Shortcut behavior

- Keyboard shortcuts for agentic chat are only active when your focus is in the Amazon Q chat panel
- This prevents conflicts with existing IDE keybindings
- If you modify a shortcut, you may need to refresh your IDE for the changes to be reflected in tooltips

Selecting a model for Amazon Q chat in IDEs

You can select the model you want Amazon Q to use while chatting in the IDE. The model you select persists across all future chat sessions.

The following table describes the models that are available for Amazon Q chat in the IDE and their context windows.

Model	Context window
Claude Sonnet 3.7	200k
Claude Sonnet 4 (default)	200k

Select the model used for chatting in the IDE

To select the model Amazon Q uses when you chat in your IDE:

1. Open the Amazon Q chat panel in your IDE.
2. In the chat input area, choose the model menu dropdown. Select the model you want to use from the available options.

   The model you select persists until you change it.

Context windows

The context window is the amount of context, including your conversation history and any explicit or automatic context, that Amazon Q can use to process and respond to your requests. Context is measured in tokens, which includes text and code.

For more information about context, see Adding context to the chat.

Generating inline suggestions with Amazon Q Developer

End of support notice

On April 30, 2027, AWS will discontinue support for Amazon Q Developer IDE plugins. For capabilities similar to Amazon Q Developer IDE plugins, explore Kiro to access the latest models and features, including agentic coding, chat and MCP support. For more information, see Amazon Q Developer IDE plugins end of support.

Amazon Q can provide you with code recommendations in real time. As you write code, Amazon Q automatically generates suggestions based on your existing code and comments. Your personalized recommendations can vary in size and scope, ranging from a single line comment to fully formed functions.

When you start typing out single lines of code or comments, Amazon Q makes suggestions based on your current and previous inputs. Filenames are also taken into consideration.

Inline suggestions are automatically enabled when you download the Amazon Q extension. To get started, start writing code, and Amazon Q will begin generating code suggestions.

You can also customize the suggestions Amazon Q generates to your software development team's internal libraries, proprietary algorithmic techniques, and enterprise code style.

Topics

- Pausing suggestions with Amazon Q
- Amazon Q code completion in action
- Generating inline suggestions in AWS coding environments
- Using shortcut keys
- Using code references
- Code examples

Pausing suggestions with Amazon Q

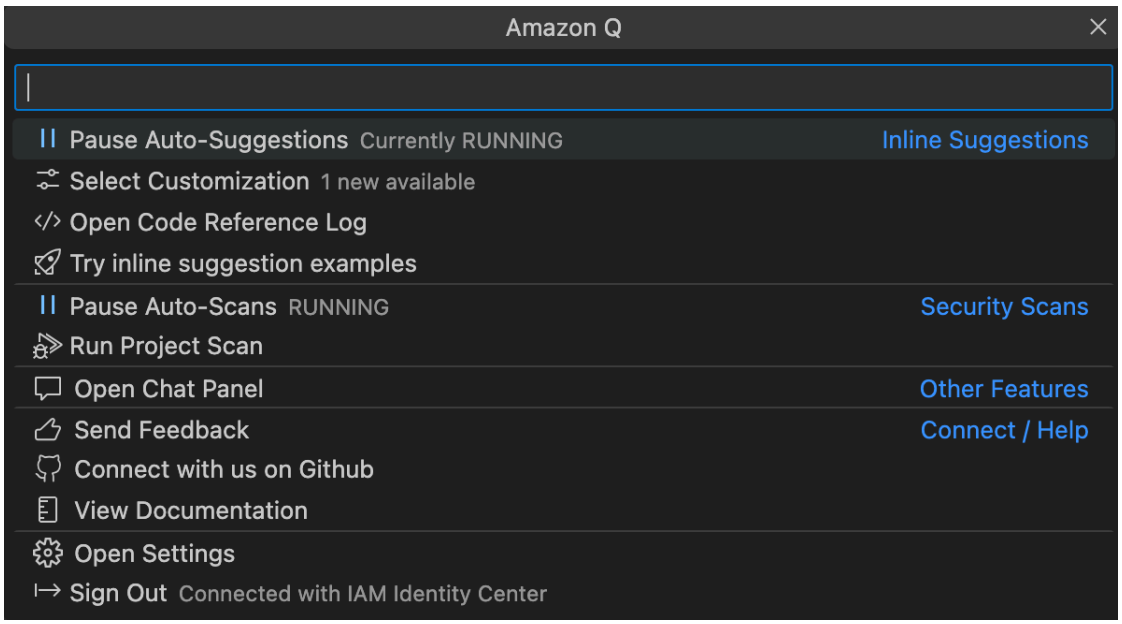
Choose your IDE to see steps for pausing and resuming inline code suggestions in Amazon Q.

Visual Studio Code

1. In VS Code, choose **Amazon Q** from the IDE component tray.

   The Amazon Q task bar opens.
2. Choose **Pause Auto-Suggestions** or **Resume Auto-Suggestions**.

The following image shows the Amazon Q task bar in VS Code.



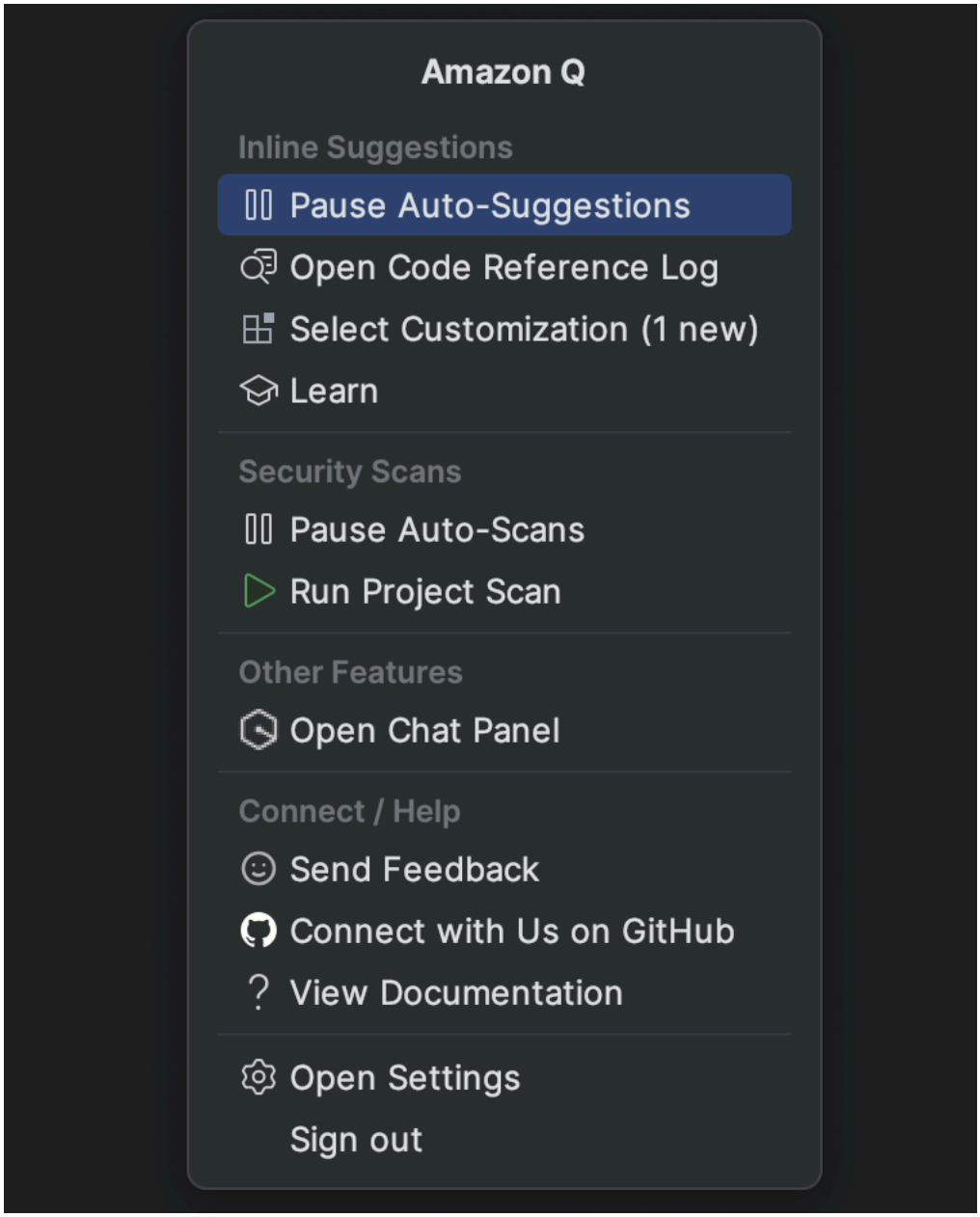
JetBrains

1. In your JetBrains IDE, choose **Amazon Q** from the IDE status bar.

   The Amazon Q task bar opens.

2. Choose **Pause Auto-Suggestions** or **Resume Auto-Suggestions**.

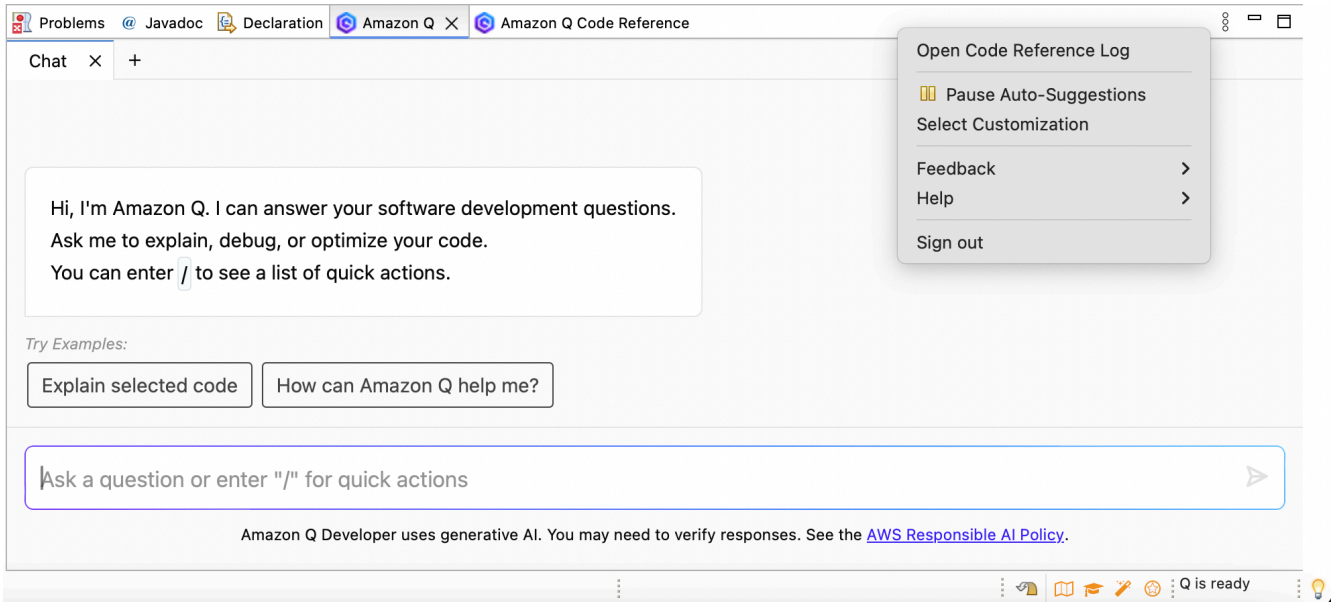
The following image shows the Amazon Q task bar in a JetBrains IDE.



Eclipse

1. In your Eclipse IDE, choose the **Amazon Q** icon in the IDE toolbar.
2. With the Amazon Q chat tab open, choose the ellipsis icon in the chat panel header. The Amazon Q task bar opens.

   The following image shows the Amazon Q task bar in an Eclipse IDE.

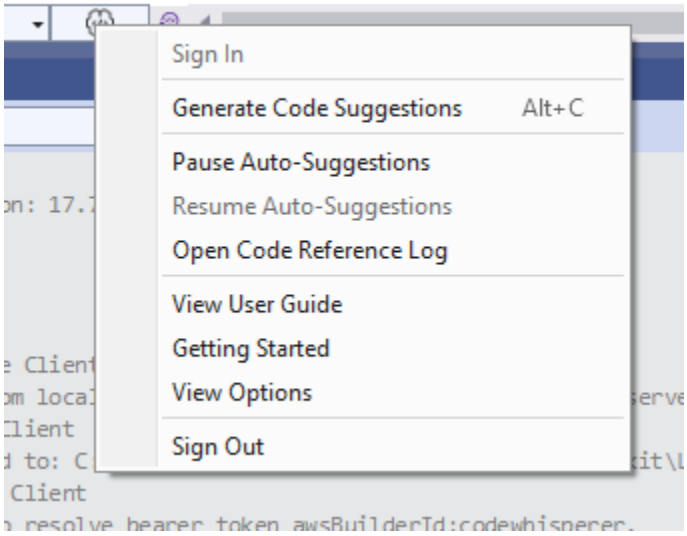


3. Choose **Pause Auto-Suggestions** or **Resume Auto-Suggestions**.

Visual Studio

1. From the edge of the window, choose the Amazon Q icon.
2. Select **Pause Auto-Suggesions** or **Resume Auto-Suggestions**

The following image shows the Amazon Q task bar in a Visual Studio.



AWS Cloud9

Amazon Q does not support toggling suggestions on and off in AWS Cloud9.

To stop receiving Amazon Q suggestions in AWS Cloud9, remove the IAM policy that gives Amazon Q access to AWS Cloud9 from the role or user that you are using to access AWS Cloud9.

AWS Lambda

To deactivate or re-activate Amazon Q code suggestions in Lambda:

1. In the Lambda console, open the screen for a particular Lambda function.
2. In the **Code source** section, from the toolbar, choose **Tools**.
3. From the dropdown menu, choose **Amazon Q Code Suggestions.**

Amazon SageMaker AI Studio

1. In the SageMaker AI Studio console, choose Amazon Q from the IDE status bar.

   The Amazon Q panel will open.
2. Choose **Pause Auto-Suggestions** or **Resume Auto-Suggestions**.

JupyterLab

1. In the JupyterLab console, choose Amazon Q from the IDE status bar.

The Amazon Q panel will open.

2. Choose **Pause Auto-Suggestions** or **Resume Auto-Suggestions**.

AWS Glue Studio Notebook

1. In the AWS Glue Studio Notebook console, choose Amazon Q from the IDE status bar.

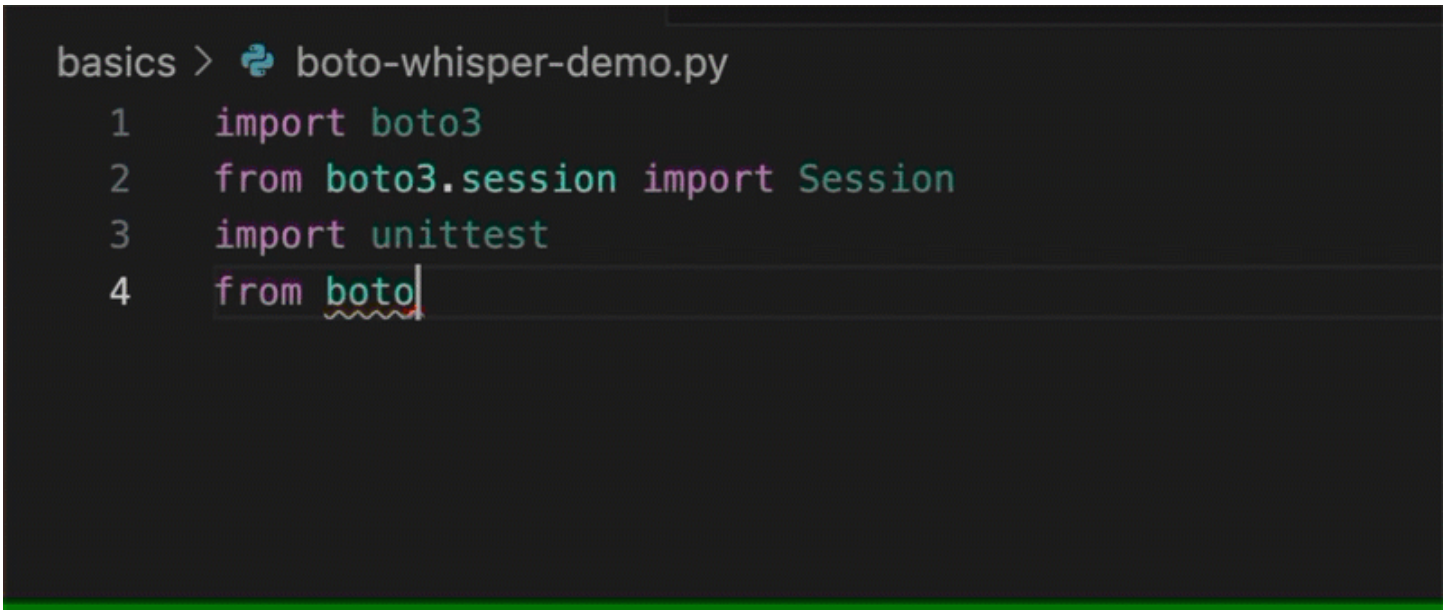
   The Amazon Q panel will open.

2. Choose **Pause Auto-Suggestions** or **Resume Auto-Suggestions**.

Amazon Q code completion in action

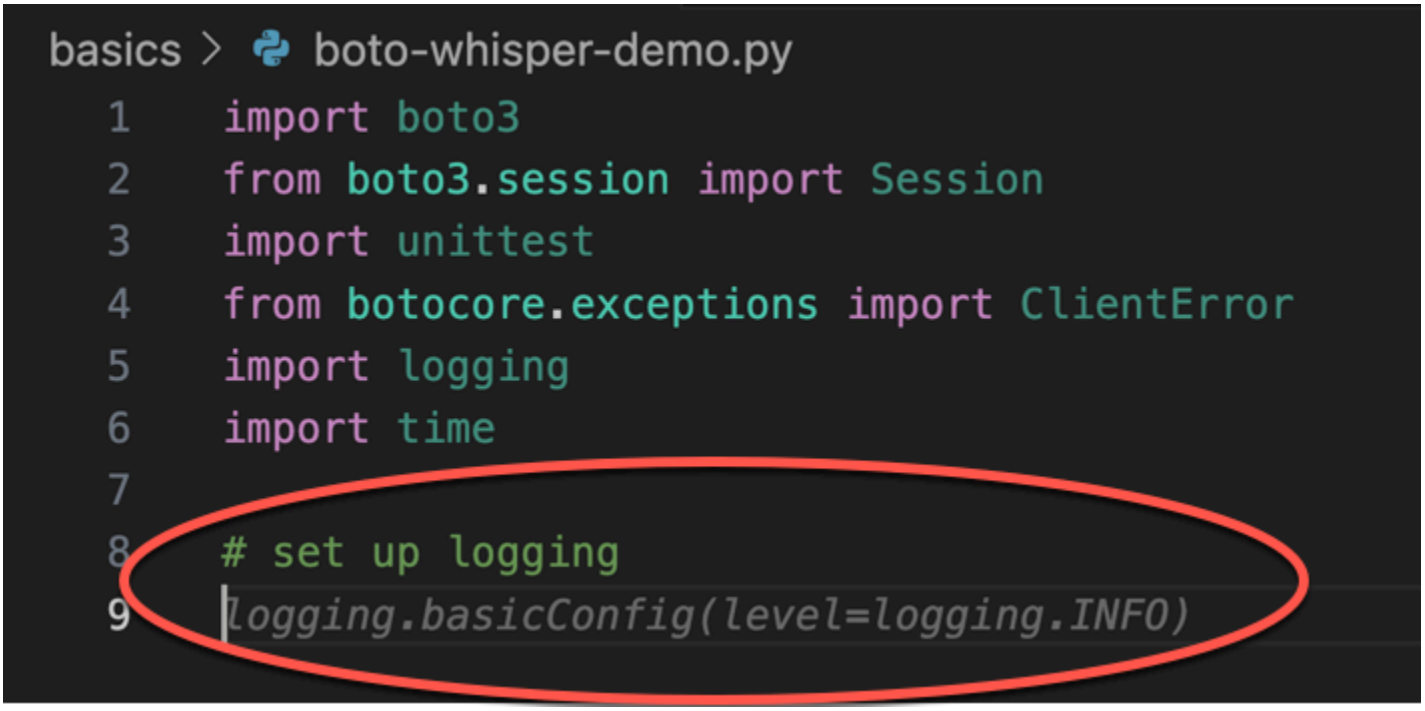
This section demonstrates how Amazon Q can help you write a complete application. This application creates an Amazon S3 bucket and a Amazon DynamoDB table, plus a unit test that validates both tasks.

Here, Amazon Q helps the developer choose which libraries to import. Using the arrow keys, the developer toggles through multiple suggestions.



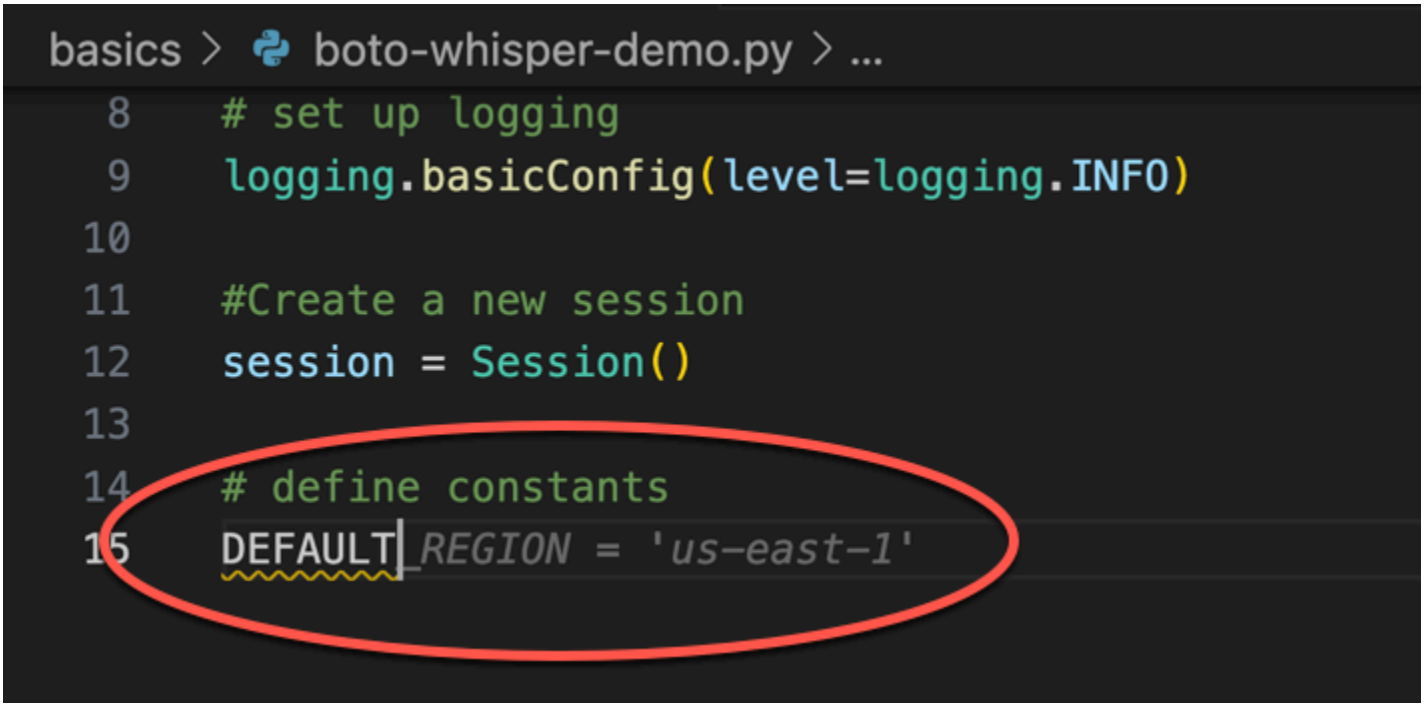
Here, the developer enters a comment, describing the code they intend to write on the next line.

Amazon Q correctly anticipates the method to be called. The developer can accept the suggestion with the tab key.



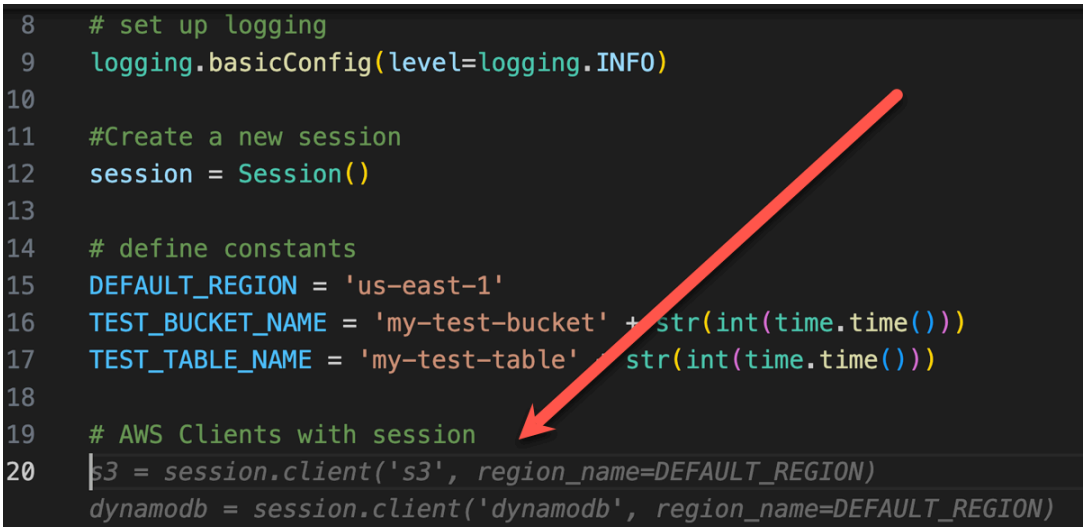
Here, the developer prepares to define constants.

Amazon Q correctly anticipates that the first constant will be `REGION` and that its value will be `us-east-1`, which is the default.



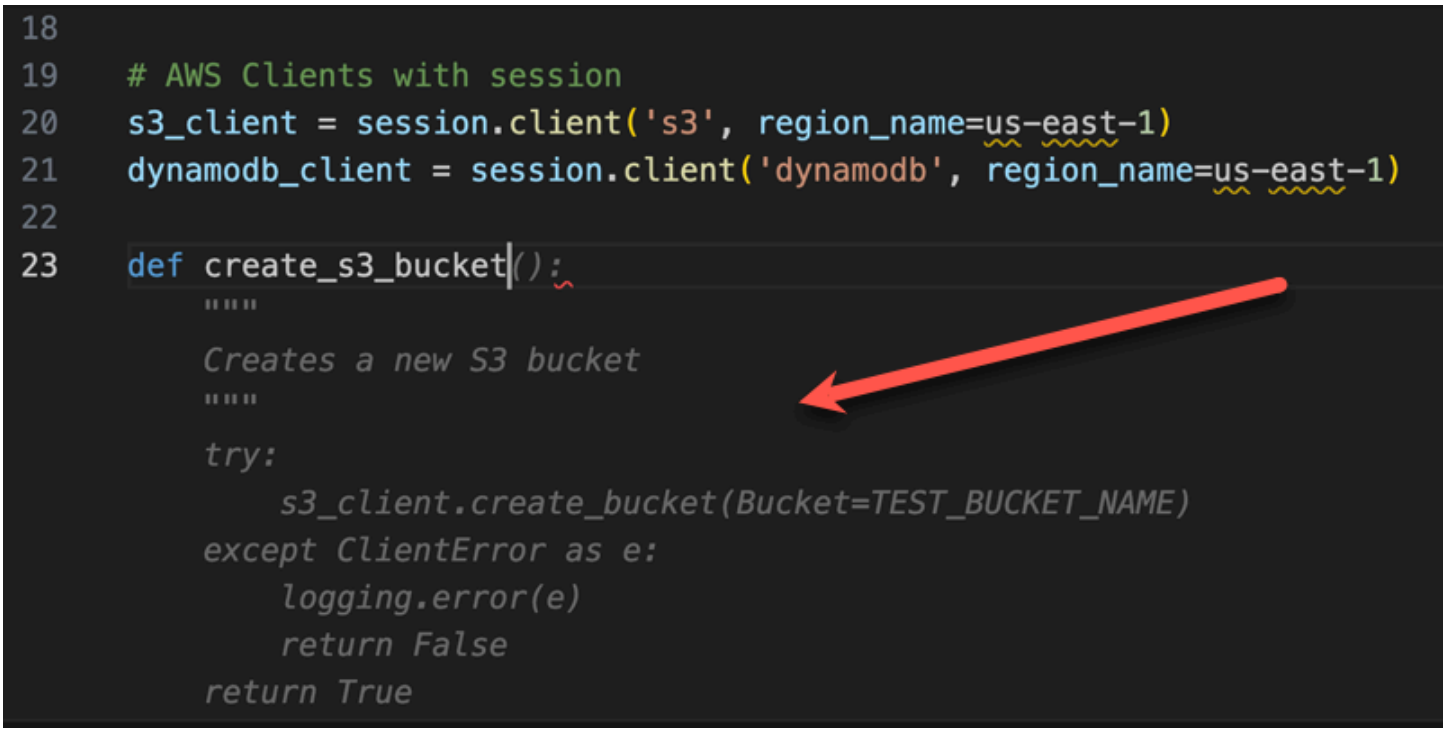
Here, the developer prepares to write code that will open sessions between the user and both Amazon S3 and DynamoDB.

Amazon Q, familiar with AWS APIs and SDKs, suggests the correct format.



The developer has merely written the name of the function that will create the bucket. But based on that (and the context), Amazon Q offers a full function, complete with try/except clauses.

Notice the use of `TEST_BUCKET_NAME`, which is a constant declared earlier in the same file.



The developer has only just begun to type in the name of the function that will create a DynamoDB table. But Amazon Q can tell where this is going.

Notice that the suggestion accounts for the DynamoDB session created earlier, and even mentions it in a comment.



The developer has done little more than write the name of the unit test class, when Amazon Q offers to complete it.

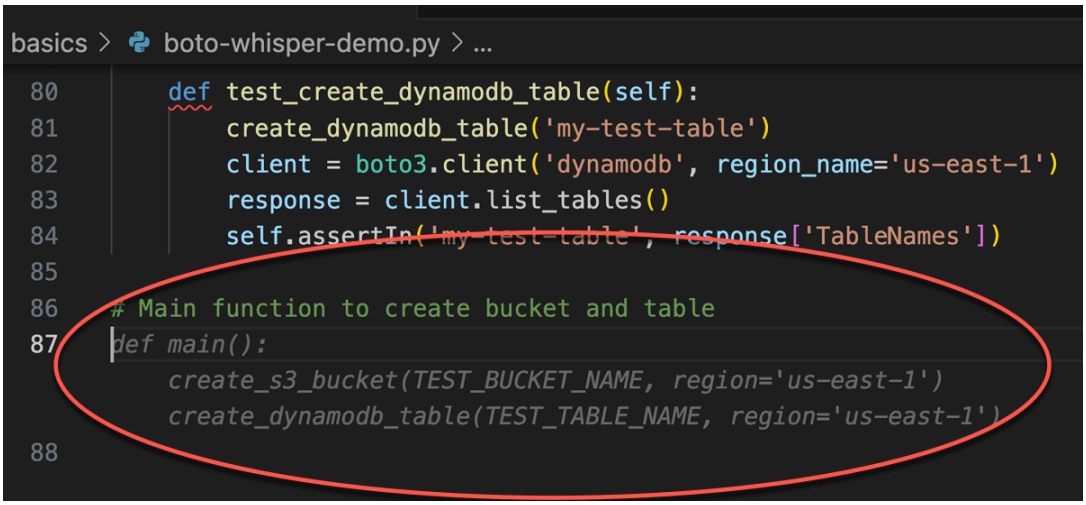
Notice the built-in references to the two functions created earlier in the same file.

The developer has only just begun to type in the name of the function that will create a DynamoDB table. But Amazon Q can tell where this is going.

Notice that the suggestion accounts for the DynamoDB session created earlier, and even mentions it in a comment.

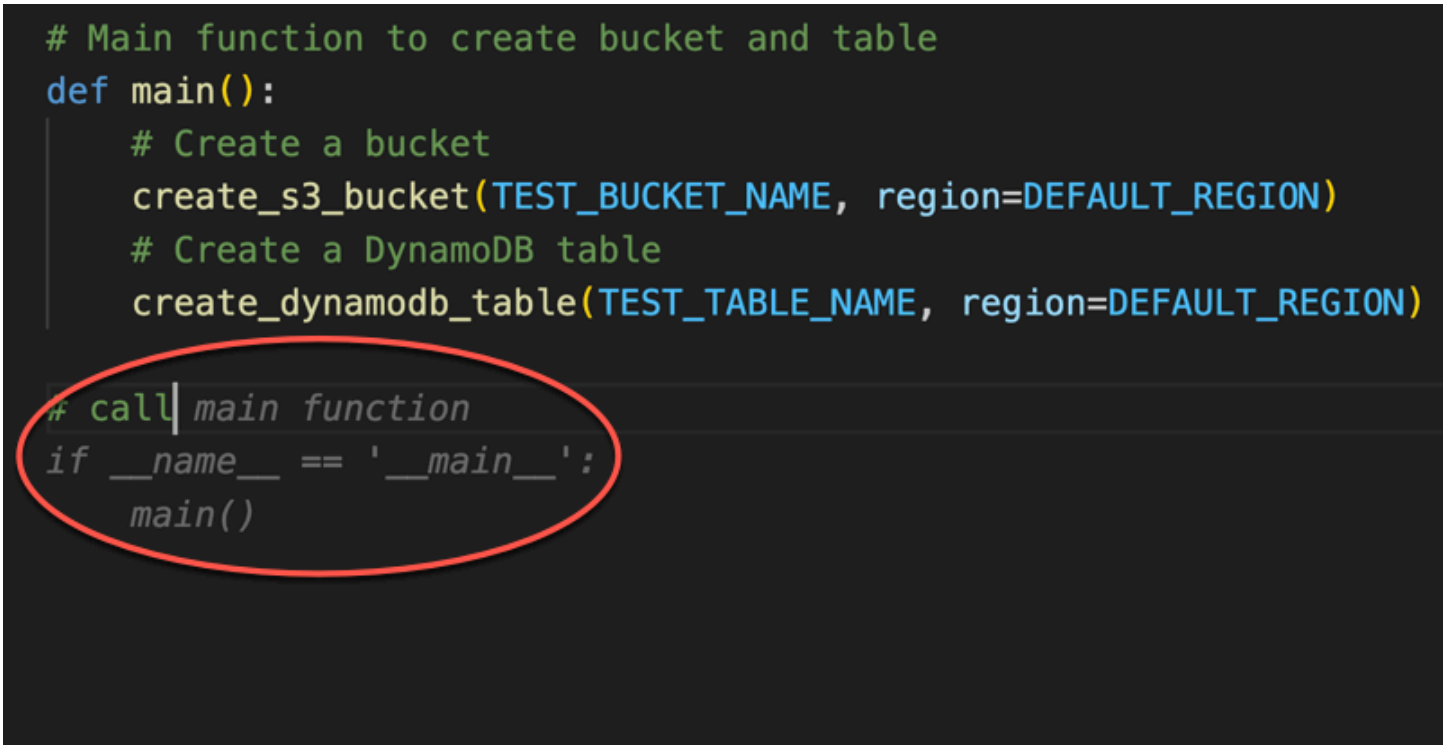


Based only on a comment and the context, Amazon Q supplies the entire main function.

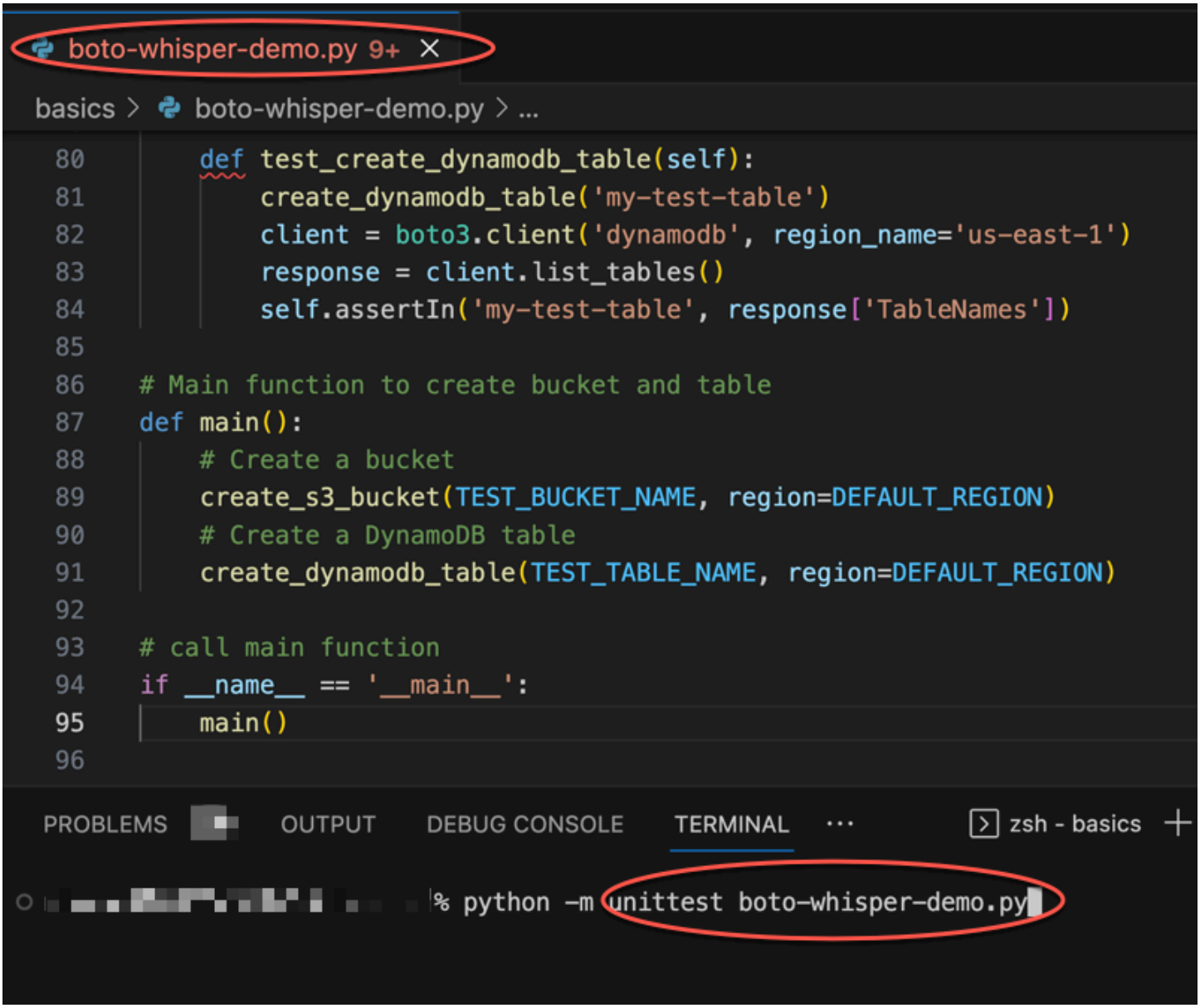


All that's left is the main guard, and Amazon Q knows it.

Based only on a comment and the context, Amazon Q supplies the entire main function.



Finally, the developer runs the unit test from the terminal of the same IDE where the coding took place.



Generating inline suggestions in AWS coding environments

In addition to third-party IDEs, Amazon Q Developer can generate inline suggestions within AWS services that provide their own coding environments.

The following sections describe how to set up Amazon Q inline code suggestions within integrated AWS services.

> **Note**
>
> If you are using Amazon Q as part of an enterprise, then you are using Amazon Q Developer Pro. In that case, administrators at your organization must complete additional steps before you can start coding. For more information, see Getting started with Amazon Q Developer.

Topics

- Using Amazon Q Developer with Amazon SageMaker AI Studio
- Using Amazon Q Developer with JupyterLab
- Using Amazon Q Developer with Amazon EMR Studio
- Using Amazon Q Developer with AWS Glue Studio
- Using Amazon Q Developer with AWS Lambda
- Using Amazon Q Developer with other services

Using Amazon Q Developer with Amazon SageMaker AI Studio

You can chat with Amazon Q inside Amazon SageMaker AI Studio. You can also make code recommendations automatically as you write your code.

To use Amazon Q Developer with Amazon SageMaker AI Studio, you must add Amazon Q permissions to your SageMaker AI execution role. The way you configure permissions depends on whether you are using the Amazon Q Developer Free tier or the Pro tier.

To set up and activate Amazon Q for Amazon SageMaker AI Studio, see Set up Amazon Q Developer for your users in the *Amazon SageMaker AI User Guide*.

Using Amazon Q Developer with JupyterLab

This page describes how to set up and activate Amazon Q Developer for JupyterLab. Once activated, Amazon Q can make code recommendations automatically as you write your code.

> **Note**
>
> Python is the only programming language that Amazon Q supports in JupyterLab.

Installing JupyterLab

Install JupyterLab on your computer or if you already have JupyterLab installed, check its version by running the following command.

```
pip show jupyterlab
```

Note the version in the response, and follow the corresponding directions in one of the following sections.

Installation using pip for Jupyter Lab version >= 4.0

You can install and enable the Amazon Q extension for JupyterLab 4 with the following commands.

```
# JupyterLab 4
pip install amazon-q-developer-jupyterlab-ext
```

Installation using pip for Jupyter Lab version >= 3.6 and < 4.0

You can install and enable the Amazon Q extension for JupyterLab 3 with the following commands.

```
# JupyterLab 3
pip install amazon-q-developer-jupyterlab-ext~=3.0
jupyter server extension enable amazon-q-developer-jupyterlab-ext
```

Authenticating with AWS Builder ID

In the following procedure, you will set up Builder ID, which you will use to authenticate when you enable Amazon Q.

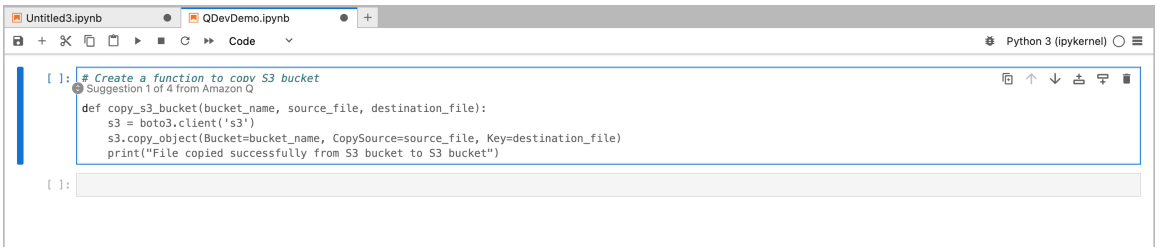
1. Refresh the browser tab on which you are using JupyterLab.
2. From the Amazon Q panel, choose **Get Started**.
3. From the pop-up window, choose **Copy Code and Proceed**.
4. On the **Get started** page, sign in or sign up for a Builder ID using your email address or Google account. For more information, see Getting started with a personal account (Builder ID).

   If you already have a Builder ID, skip to the step about the **Authorize request** page.

5. After you receive your email verification code, enter it in the blank field and choose **Verify**.
6. On the next screen, choose and confirm a password, then choose **Create AWS Builder ID**
7. On the next page choose **Allow** to allow Amazon Q to access your data.

Now you should be logged into Amazon Q in JupyterLab with Builder ID.

To begin coding, see Using shortcut keys.



Using Amazon Q Developer with Amazon EMR Studio

This page describes how to set up and activate Amazon Q Developer for Amazon EMR Studio. Once activated, Amazon Q can make code recommendations automatically as you write your ETL code.

> **Note**
>
> Amazon Q supports Python, which can be used to code ETL scripts for Spark jobs in Amazon EMR Studio.

Use the following procedure to set up Amazon EMR Studio to work with Amazon Q.

1. Set up Amazon EMR Studio Notebook.
2. Attach the following policy to the IAM user role for Amazon EMR Studio Notebook.

> **Note**
>
> The `codewhisperer` prefix is a legacy name from a service that merged with Amazon Q Developer. For more information, see Amazon Q Developer rename - Summary of changes.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AmazonQDeveloperPermissions",
            "Effect": "Allow",
            "Action": [
                "codewhisperer:GenerateRecommendations"
            ],
            "Resource": "*"
        }
    ]
}
```

3. Open the Amazon EMR console.
4. Under Amazon EMR Studio, choose **Workspaces (Notebooks).**
5. Select your desired Workspace and choose **Quick launch**.

Using Amazon Q Developer with AWS Glue Studio

This page describes how to set up and activate Amazon Q Developer for AWS Glue Studio Notebook. Once activated, Amazon Q can make code recommendations automatically as you write your ETL code.

> **Note**
>
> Amazon Q supports both Python and Scala, the two languages used for coding ETL scripts for Spark jobs in AWS Glue Studio.

In the following procedure, you will set up AWS Glue to work with Amazon Q.

1. Set up AWS Glue Studio Notebook.
2. Attach the following policy to your IAM role for Glue Studio notebook

> **Note**
>
> The `codewhisperer` prefix is a legacy name from a service that merged with Amazon Q Developer. For more information, see Amazon Q Developer rename - Summary of changes.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AmazonQDeveloperPermissions",
            "Effect": "Allow",
            "Action": [
                "codewhisperer:GenerateRecommendations"
            ],
            "Resource": "*"
        }
    ]
}
```

3. Open the Glue console
4. Under **ETL jobs**, choose **Notebooks**.
5. Verify that **Jupyter Notebook** is selected. Choose **Create**.
6. Enter a **Job name**.
7. For IAM role, select the role that you configured to interact with Amazon Q
8. Choose **Start notebook**.

Using Amazon Q Developer with AWS Lambda

This document describes how to set up and activate Amazon Q Developer for the Lambda console. Once activated, Amazon Q can make code recommendations on demand in the Lambda code editor as you develop your function.

> **Note**
>
> In the Lambda console, Amazon Q only supports functions using the Python and Node.js runtimes.

AWS Identity and Access Management permissions for Lambda

For Amazon Q to provide recommendations in the Lambda console, you must enable the correct IAM permissions for either your IAM user or role. You must add the `codewhisperer:GenerateRecommendations` permission, as outlined in the sample IAM policy below:

> **Note**
>
> The `codewhisperer` prefix is a legacy name from a service that merged with Amazon Q Developer. For more information, see Amazon Q Developer rename - Summary of changes.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AmazonQDeveloperPermissions",
      "Effect": "Allow",
      "Action": ["codewhisperer:GenerateRecommendations"],
      "Resource": "*"
    }
  ]
}
```

It is best practice to use IAM policies to grant restrictive permissions to IAM principals. For details about working with IAM for AWS Lambda, see Identity and access management in AWS Lambda in the *AWS Lambda Developer Guide*.

Activating Amazon Q Developer with Lambda

To activate Amazon Q in the Lambda console code editor, complete these steps.

1. Open the Functions page of the Lambda console, and choose the function that you want to edit.
2. As you type in the code editor, automatic code suggestions from Amazon Q are enabled by default. To pause suggestions, choose **Amazon Q** in the status bar of the **Code source** panel. The command palette opens. From there, choose **Pause auto-suggestions**.

For shortcut keys, see Using shortcut keys.

Using Amazon Q Developer with other services

AWS Identity and Access Management permissions for other services

For Amazon Q to provide recommendations in the context of another service, you must enable the correct IAM permissions for either your IAM user or role. You must add the `codewhisperer:GenerateRecommendations` permission, as outlined in the sample IAM policy below:

> **Note**
>
> The `codewhisperer` prefix is a legacy name from a service that merged with Amazon Q Developer. For more information, see Amazon Q Developer rename - Summary of changes.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AmazonQDeveloperPermissions",
      "Effect": "Allow",
      "Action": ["codewhisperer:GenerateRecommendations"],
      "Resource": "*"
    }
  ]
```

```
}
```

It is best practice to use IAM policies to grant restrictive permissions to IAM principals. For details about working with IAM, see Security best practices in the *IAM user guide*.

Using shortcut keys

While getting inline suggestions from Amazon Q, you can use keyboard shortcuts for common actions you take, such as initiating Amazon Q or accepting a recommendation.

Choose the integrated development environment (IDE) where you are developing code to see keyboard shortcuts for your IDE.

Visual Studio Code

Action	Keyboard shortcut
Manually initiate Amazon Q	MacOS: Option + C Windows: Alt + C
Accept a recommendation	Tab
Next recommendation	Right arrow
Previous recommendation	Left arrow
Reject a recommendation	ESC, backspace, or keep typing and the recommendation will disappear as soon as there is a character mismatch.
Accept next word	Cmd + right arrow

To change keybindings in VS Code, see Key Bindings for Visual Studio Code on the VS Code website.

> **Note**
>
> The inline suggestions toolbar in VS Code is disabled by default. For more information, see Redesigned inline suggestions toolbar on the VS Code website.

JetBrains

Action	Keyboard shortcut
Manually initiate Amazon Q	MacOS: Option + C Windows: Alt + C
Accept a recommendation	Tab
Next recommendation	Right arrow
Previous recommendation	Left arrow
Reject a recommendation	ESC, backspace, or keep typing and the recommendation will disappear as soon as there is a character mismatch.

To change keybindings in IntelliJ, see IntelliJ IDEA keyboard shortcuts on the JetBrains website.

Eclipse

Action	Keyboard shortcut
Manually initiate Amazon Q	MacOS: Option + C Windows: Alt + C
Accept a recommendation	Tab
Next recommendation	MacOS: Option +] Windows: Alt +]

Action	Keyboard shortcut
Previous recommendation	MacOS: Option + [Windows: Alt + [
Reject a recommendation	ESC, backspace, or keep typing and the recommendation will disappear as soon as there is a character mismatch.

To change keybindings in Eclipse, see Changing the key bindings in the Eclipse documentation.

Toolkit for Visual Studio

Action	Keyboard shortcut
Manually initiate Amazon Q `AWSToolkit.CodeWhisperer.GetSuggestion` in the keybindings	Windows: Alt + C
Accept a recommendation	Tab
Next recommendation `Edit.NextSuggestion` in the keybindings	Windows: Alt + .
Previous recommendation `Edit.PreviousSuggestion` in the keybindings	Windows: Alt + ,
Reject a recommendation	ESC, backspace, or keep typing and the recommendation will disappear as soon as there is a character mismatch.

See also Microsoft's Visual Studio default keyboard shortcuts.

To change keybindings in Visual Studio, use Tools -> Options -> Keyboard.

Amazon SageMaker AI

Action	Keyboard shortcut
Manually initiate Amazon Q	MacOS: Option + C Windows: Alt + C
Accept a recommendation	Tab
Next recommendation	Down arrow
Previous recommendation	Up arrow
Reject a recommendation	ESC

JupyterLab

Action	Keyboard shortcut
Manually initiate Amazon Q	MacOS: Option + C Windows: Alt + C
Accept a recommendation	Tab
Next recommendation	Down arrow
Previous recommendation	Up arrow
Reject a recommendation	ESC

AWS Glue Studio Notebook

Action	Keyboard shortcut
Manually initiate Amazon Q	MacOS: Option + C

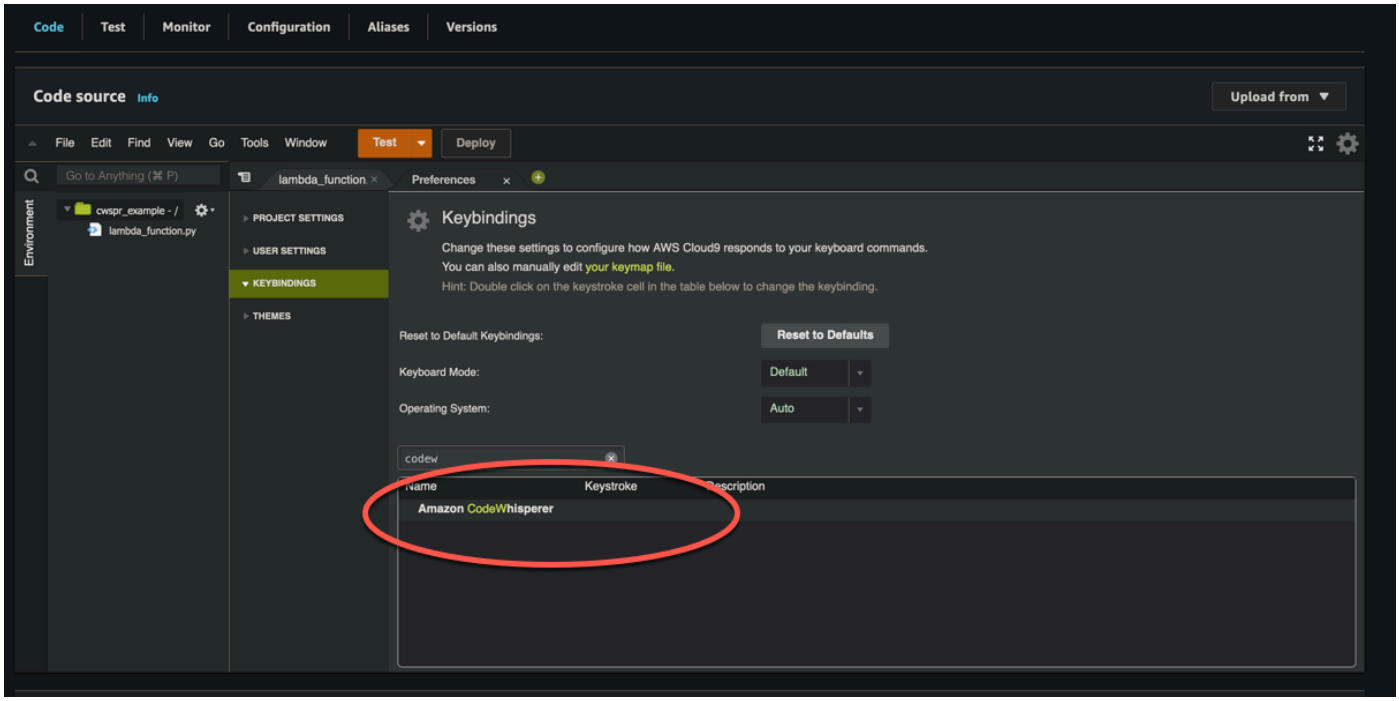
Action	Keyboard shortcut
	Windows: Alt + C
Accept a recommendation	Tab
Next recommendation	Down arrow
Previous recommendation	Up arrow
Reject a recommendation	ESC

AWS Lambda

Action	Keyboard shortcut
Manually fetch a code suggestion	MacOS: Option + C Windows: Alt + C
Accept a suggestion	Tab
Reject a suggestion	ESC, Backspace, scroll in any direction, or keep typing and the recommendation automatically disappears.

To change the key bindings, use the following procedure.

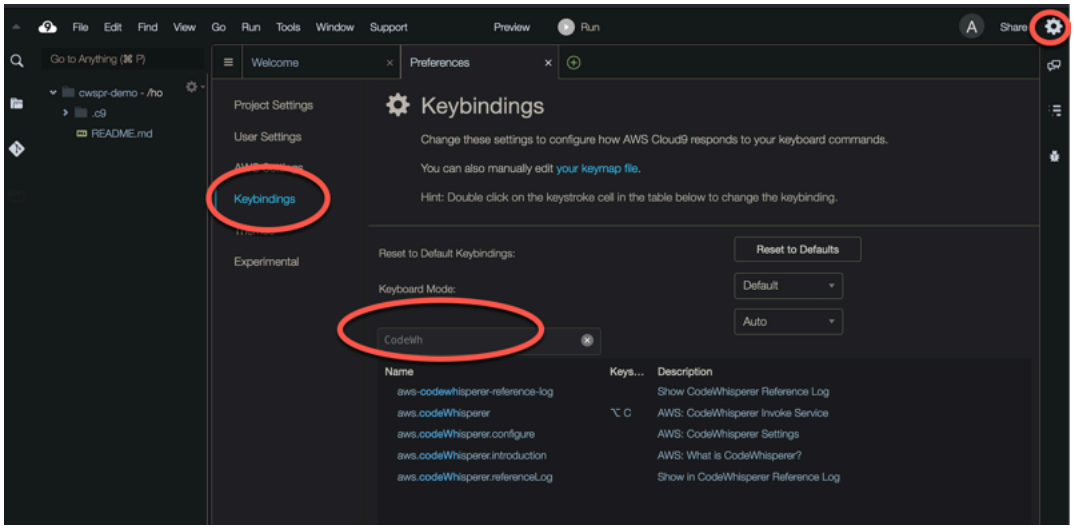
1. While viewing a particular function, choose the gear icon to open the **Preferences** tab.
2. On the **Preferences** tab, select **Keybindings**.
3. In the keybindings search box, enter Amazon Q.



AWS Cloud9

Action	Keyboard shortcut
Manually fetch a code suggestion	MacOS: Option + C Windows: Alt + C
Accept a suggestion	Tab
Reject a suggestion	ESC, Backspace, scroll in any direction, or keep typing and the recommendation automatically disappears.

1. While viewing a particular environment, choose the gear icon to open the **Preferences** tab.
2. On the **Preferences** tab, select **Keybindings**.
3. In the keybindings search box, enter Amazon Q.
4. In the Keystroke column, double-click the space corresponding to the function you're interested in.
5. Enter the keys that you want to bind the function to.



Using code references

Amazon Q learns, in part, from open-source projects. Sometimes, a suggestion it's giving you may be similar to publicly available code. Code references include information about the source Amazon Q used to generate a recommendation.

Topics

- View and update code references
- Turn code references off and on
- Opt out of code with references

View and update code references

With the reference log, you can view references to code recommendations that are similar to publicly available code. You can also update and edit code recommendations suggested by Amazon Q.

Choose your IDE to see steps for how to view and update code references.

Visual Studio Code

To display the Amazon Q reference log in VS Code, use the following procedure.

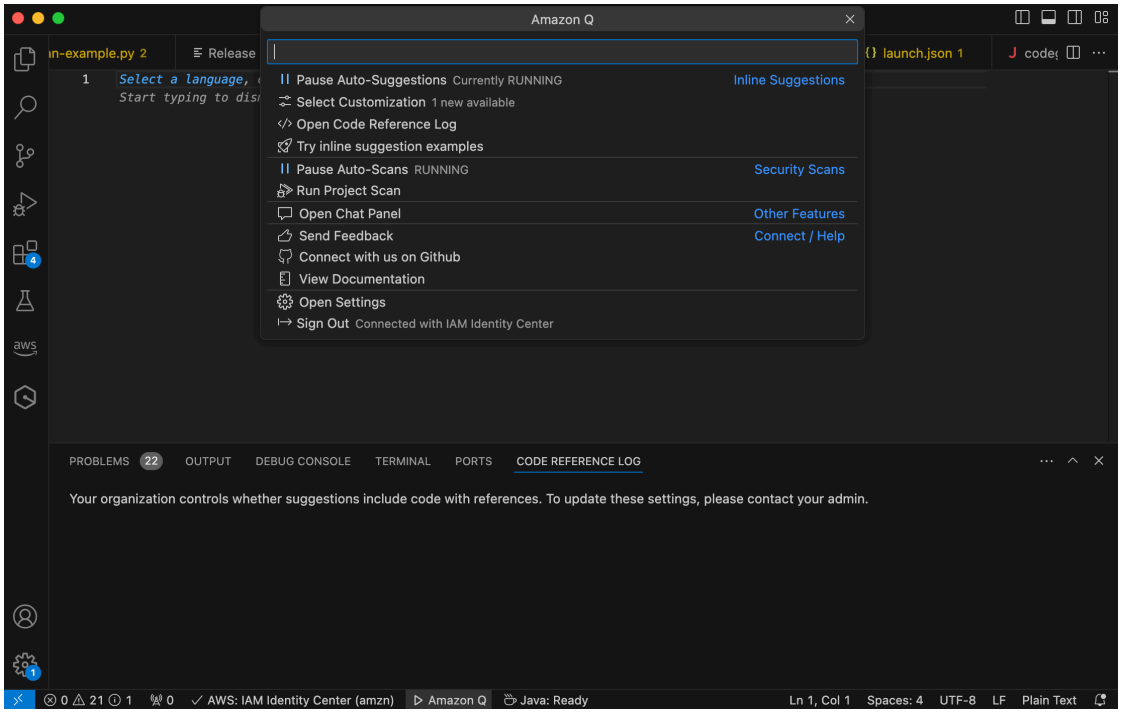
1. Make sure you are using the latest version of both VS Code and the Amazon Q extension.
2. In VS Code, choose **Amazon Q** from the IDE component tray.

The Amazon Q task bar opens.

3. Choose **Open Code Reference Log**.

   The code reference log tab opens. Any references to code recommendations are listed.

The following image shows the open Amazon Q task bar and code reference log tab.



JetBrains

To display the Amazon Q reference log in JetBrains IDEs, use the following procedure.

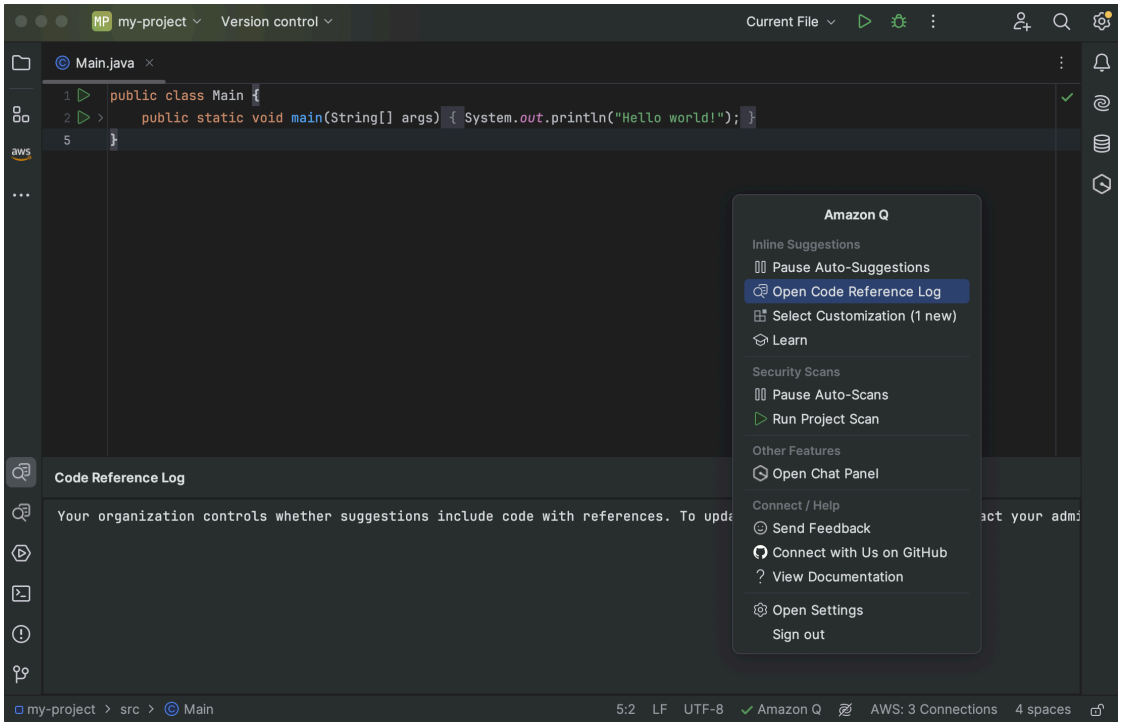
1. Make sure you are using the latest version of both your JetBrains IDE and the Amazon Q plugin.
2. In JetBrains, choose **Amazon Q** from the IDE status bar.

   The Amazon Q task bar opens.

3. Choose **Open Code Reference Log**.

   The code reference log tab opens. Any references to code recommendations are listed.

The following image shows the open Amazon Q task bar and code reference log tab.

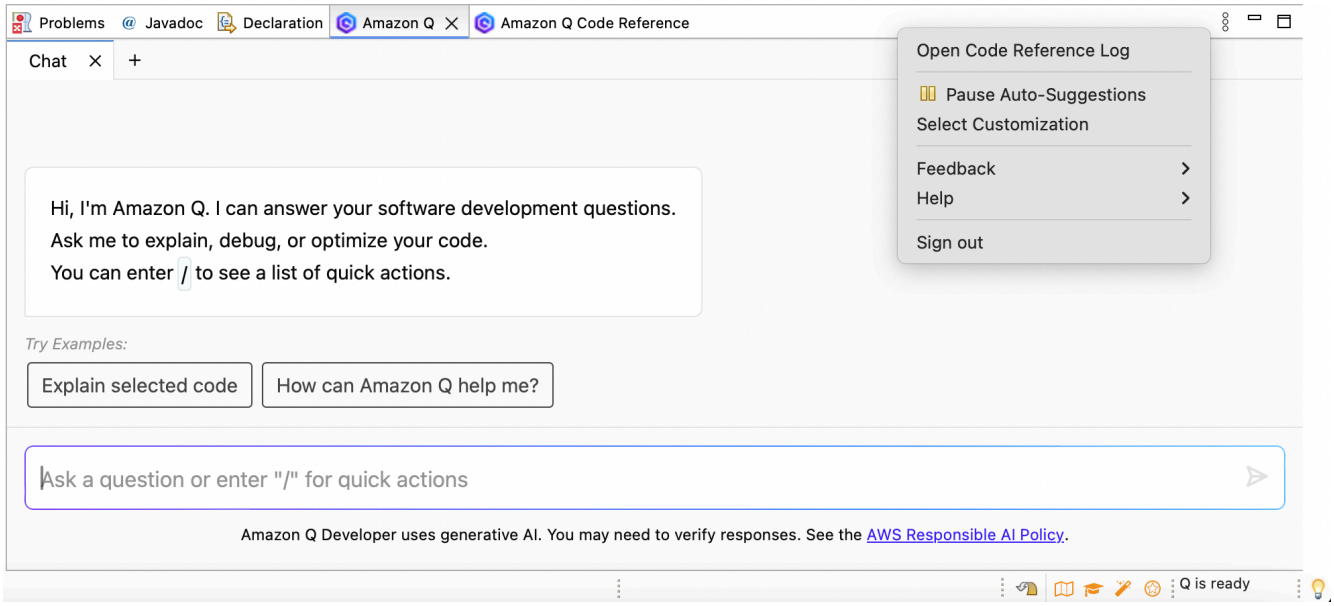


Eclipse

To display the Amazon Q reference log in Eclipse IDEs, use the following procedure.

1. Make sure you are using the latest version of both the Eclipse IDE and the Amazon Q plugin.
2. In your Eclipse IDE, choose the **Amazon Q** icon in the IDE toolbar.
3. With the Amazon Q chat tab open, choose the ellipsis icon in the chat panel header. The Amazon Q task bar opens.

   The following image shows the Amazon Q task bar in an Eclipse IDE.

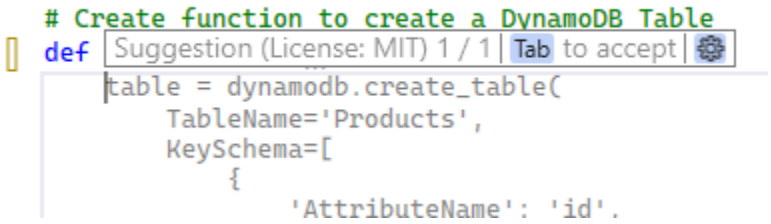


4. Choose **Open Code Reference Log**.

   The code reference log tab opens. Any references to code recommendations are listed.

Toolkit for Visual Studio

When Amazon Q suggests code that contains a reference in the Toolkit for Visual Studio, the reference type appears in the suggestion description.



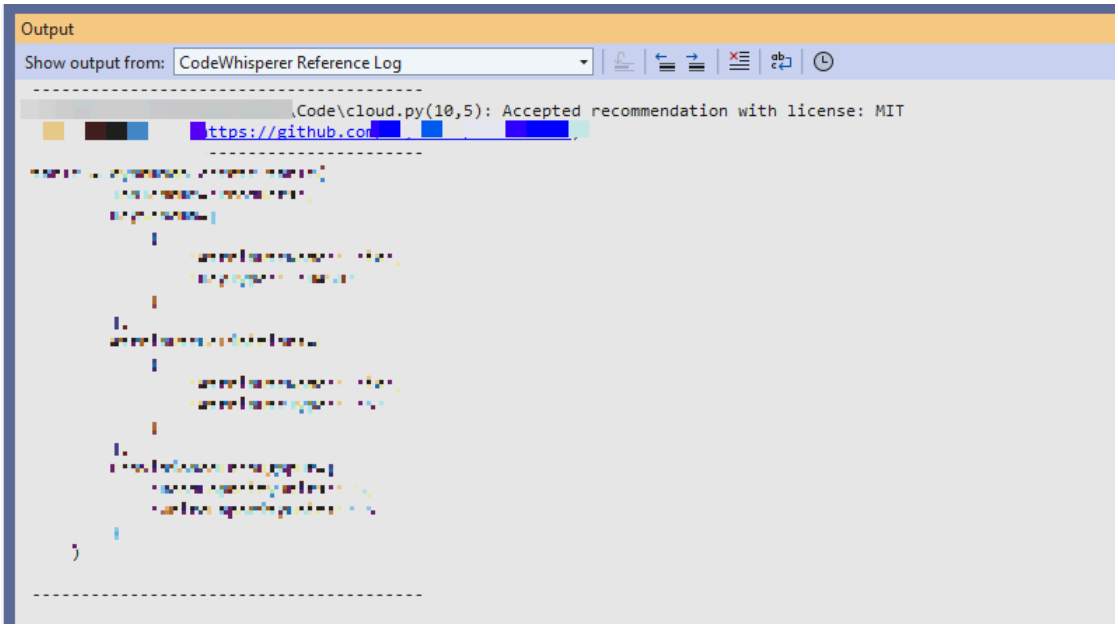
All accepted suggestions that contain references are captured in the reference log.

To access the reference log, choose the AWS icon, then select **Open Code Reference Log**.

A list of accepted suggestions that contain references will appear. This list includes:

- The location where the suggestion was accepted. Double clicking on this will take you to that location in your code.
- The associated license
- The referenced source code

- The fragment of code attributed to the reference



AWS Cloud 9

When you use Amazon Q with AWS Cloud 9, code references are on by default.

To turn them off, or to turn them back on later, use the following procedure.

1. On the AWS Cloud 9 console, choose the AWS Cloud 9 logo.
2. From the dropdown menu, choose **Preferences**.

   The **Preferences** tab opens.
3. On the **Preferences** tab, under **Project Settings**, under **Extensions**, select **AWS Toolkit**.
4. Select or deselect **Amazon Q: Include Suggestions With Code References**.

Lambda

Amazon Q in Lambda does not support code references. When you use Amazon Q with Lambda, any code suggestions with references are omitted.

SageMaker AI Studio

To display the Amazon Q reference log in SageMaker AI Studio, use the following procedure.

1. In the SageMaker AI Studio window, open the Amazon Q panel.

2. Choose **Open Code Reference Log**.

JupyterLab

To display the Amazon Q reference log in JupyterLab, use the following procedure.

1. In the JupyterLab window, open the Amazon Q panel.
2. Choose **Open Code Reference Log**.

AWS Glue Studio Notebook

To display the Amazon Q reference log in AWS Glue Studio Notebook, use the following procedure.

1. In the AWS Glue Studio Notebook window, open the Amazon Q panel.
2. Choose **Open Code Reference Log**.

Turn code references off and on

In most IDEs, code references are on by default. Choose your IDE to see steps for how to turn code references off or on.

Visual Studio Code

When you use Amazon Q with VS Code, code references are on by default.

To turn them off, or to turn them back on later, use the following procedure.

1. Make sure you are using the latest version of both VS Code and the Amazon Q extension.
2. In VS Code, choose **Amazon Q** from the IDE component tray.

   The Amazon Q task bar opens.
3. Choose **Open Settings**. The settings tab opens with the options related to Amazon Q displayed.
4. Select or deselect the box next to **Show Code With References**.

JetBrains

When you use Amazon Q with your JetBrains IDE, code references are on by default.

To turn them off, or to turn them back on later, use the following procedure.

1. Make sure you are using the latest version of both your JetBrains IDE and the Amazon Q plugin.
2. In JetBrains, choose **Amazon Q** from the IDE status bar.

   The Amazon Q task bar opens.
3. Choose **Open Settings**. The settings window opens with the options related to Amazon Q displayed.
4. Select or deselect the box next to **Show Code With References**.

Eclipse

When you use Amazon Q with Eclipse, code references are on by default.

To turn them off, or to turn them back on later, use the following procedure.

1. Make sure you are using the latest version of both the Eclipse IDE and the Amazon Q plugin.
2. Open **Settings** in your Eclipse IDE.
3. Choose **Amazon Q** from the left navigation bar.
4. Select or deselect the box next to **Show Code With References**.
5. Choose **Apply** to save your changes.

Toolkit for Visual Studio

When you use Amazon Q in the Toolkit for Visual Studio, code references are on by default.

To turn them off, or to turn them back on later, use the following procedure.

1. Make sure you are using the latest version of the Toolkit for Visual Studio.
2. Open **Options** in Visual Studio.
3. Choose **AWS Toolkit** from the left navigation bar, and then choose **Amazon Q**.
4. From the dropdown next to **Include Suggestions With References**, select True or False.

5. Choose **OK** to save your changes.

AWS Cloud 9

When you use Amazon Q with AWS Cloud 9, code references are on by default.

To turn them off, or to turn them back on later, use the following procedure.

1. On the AWS Cloud 9 console, choose the AWS Cloud 9 logo.
2. From the dropdown menu, choose **Preferences**.

   The **Preferences** tab opens.
3. On the **Preferences** tab, under **Project Settings**, under **Extensions**, select **AWS Toolkit**.
4. Select or deselect **Amazon Q: Include Suggestions With Code References**.

Lambda

Amazon Q in Lambda does not support code references. When you use Amazon Q with Lambda, any code suggestions with references are omitted.

SageMaker AI Studio

When you use Amazon Q with SageMaker AI Studio, code references are on by default.

To turn them off, or to turn them back on later, use the following procedure.

1. From the top of the SageMaker AI Studio window choose **Settings**.
2. From the **Settings** dropdown, choose **Advanced Settings Editor**.
3. In the Amazon Q dropdown, select or deselect the box next to **Enable suggestions with code references**.

JupyterLab

When you use Amazon Q with JupyterLab, code references are on by default.

To turn them off, or to turn them back on later, use the following procedure.

1. From the top of the JupyterLab window choose **Settings**.
2. From the **Settings** dropdown, choose **Advanced Settings Editor**.

3. In the Amazon Q dropdown, select or deselect the box next to **Enable suggestions with code references**.

AWS Glue Studio Notebook

1. In the AWS Glue Studio Notebook window, choose **Amazon Q**.
2. From the pop-up menu, toggle the switch next to **Code with references**.

> **Note**
>
> Pausing code references will be valid only for the duration of the current AWS Glue Studio Notebook.

Opt out of code with references

In some IDEs, you can opt out of receiving suggestions with references at the administrator level.

Choose your IDE to see steps for opting out as an administrator.

Visual Studio Code

If you are an enterprise administrator, you can opt out of suggestions with code references for your entire organization. If you do this, individual developers in your organization will not be able to opt back in through the IDE. Those developers will be able to select and deselect the box discussed in the previous section, but it will have no effect if you have opted out at the enterprise level.

To opt out of suggestions with references at the enterprise level, use the following procedure.

1. In the Amazon Q Developer console, choose **Settings**.
2. In the **Amazon Q Developer account details** pane, choose **Edit**.
3. On the Edit details page, in the **Advanced settings** pane, deselect **Include suggestions with code references**.
4. Choose **Save changes**.

JetBrains

If you are an enterprise administrator, you can opt out of suggestions with code references for your entire organization. If you do this, individual developers in your organization will not be able to opt back in through the IDE. Those developers will be able to select and deselect the box discussed in the previous section, but it will have no effect if you have opted out at the enterprise level.

To opt out of suggestions with references at the enterprise level, use the following procedure.

1. In the Amazon Q Developer console, choose **Settings**.
2. In the **Amazon Q Developer account details** pane, choose **Edit**.
3. On the Edit details page, in the **Advanced settings** pane, deselect **Include suggestions with code references**.
4. Choose **Save changes**.

Eclipse

If you are an enterprise administrator, you can opt out of suggestions with code references for your entire organization. If you do this, individual developers in your organization will not be able to opt back in through the IDE. Those developers will be able to select and deselect the box discussed in the previous section, but it will have no effect if you have opted out at the enterprise level.

To opt out of suggestions with references at the enterprise level, use the following procedure.

1. In the Amazon Q Developer console, choose **Settings**.
2. In the **Amazon Q Developer account details** pane, choose **Edit**.
3. On the Edit details page, in the **Advanced settings** pane, deselect **Include suggestions with code references**.
4. Choose **Save changes**.

Toolkit for Visual Studio

To opt out of suggestions with references at the enterprise level, use the following procedure.

1. You can get to the code references setting in one of two ways:

a. Choose the Amazon Q icon at the edge of the window, and then choose **Options...**

b. Go to **Tools** -> **AWS Toolkit** -> **Amazon Q**

2. Change the toggle to **True** or **False**, depending on whether you want to include suggestions with references.

AWS Cloud 9

Amazon Q in AWS Cloud 9 does not support opting out of code suggestions with references at the enterprise level.

To opt out at the individual developer level, see Toggling code references.

Lambda

Amazon Q in Lambda does not support code references. When you use Amazon Q with Lambda, any code suggestions with references are omitted.

SageMaker AI Studio

Amazon Q does not support opting out of code suggestions with references at the enterprise level in SageMaker AI Studio.

JupyterLab

Amazon Q does not support opting out of code suggestions with references at the enterprise level in JupyterLab.

AWS Glue Studio Notebook

Amazon Q does not support opting out of code suggestions with references in AWS Glue Studio Notebook.

Code examples

Amazon Q can suggest code in different scenarios. To understand how it can help you as you write code in your programming language of choice, view the following code examples.

Topics

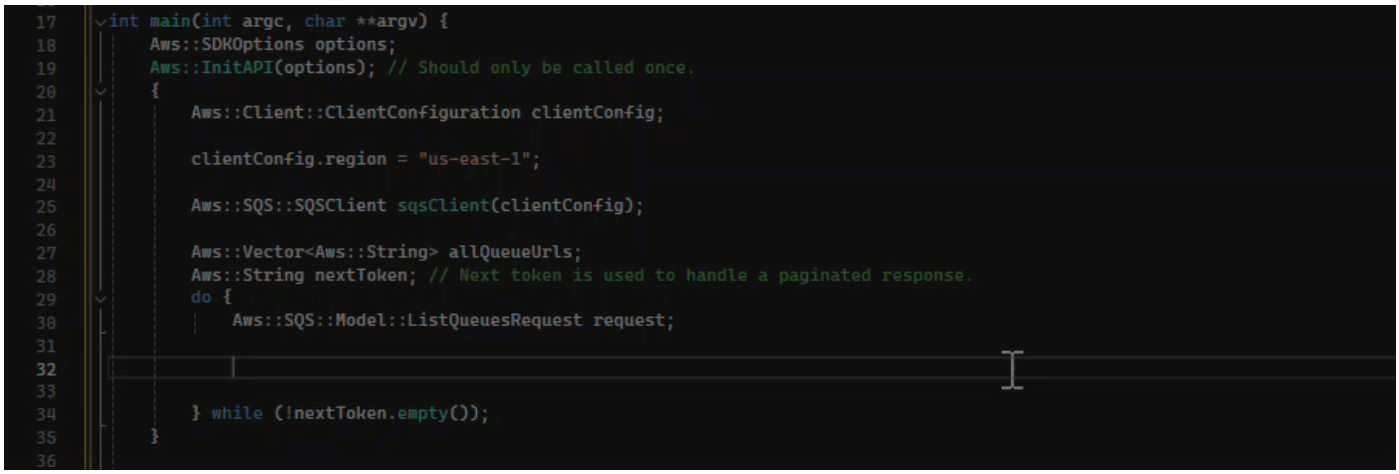
- Using Amazon Q Developer for single-line code completion
- Using Amazon Q Developer for full function generation
- Using Amazon Q Developer for block completion

- Using Amazon Q Developer for Docstring, JSDoc, and Javadoc completion
- Using Amazon Q Developer for line-by-line recommendations

Using Amazon Q Developer for single-line code completion

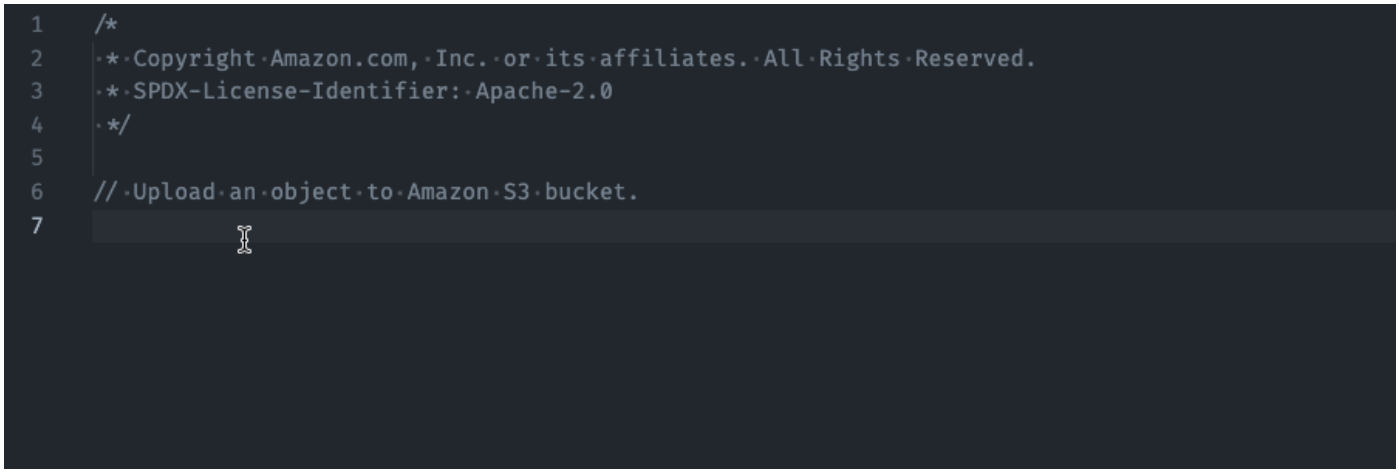
When you start typing out single lines of code, Amazon Q makes suggestions based on your current and previous inputs.

C++



JavaScript

In this example, Amazon Q completes a line of code that the developer begins.



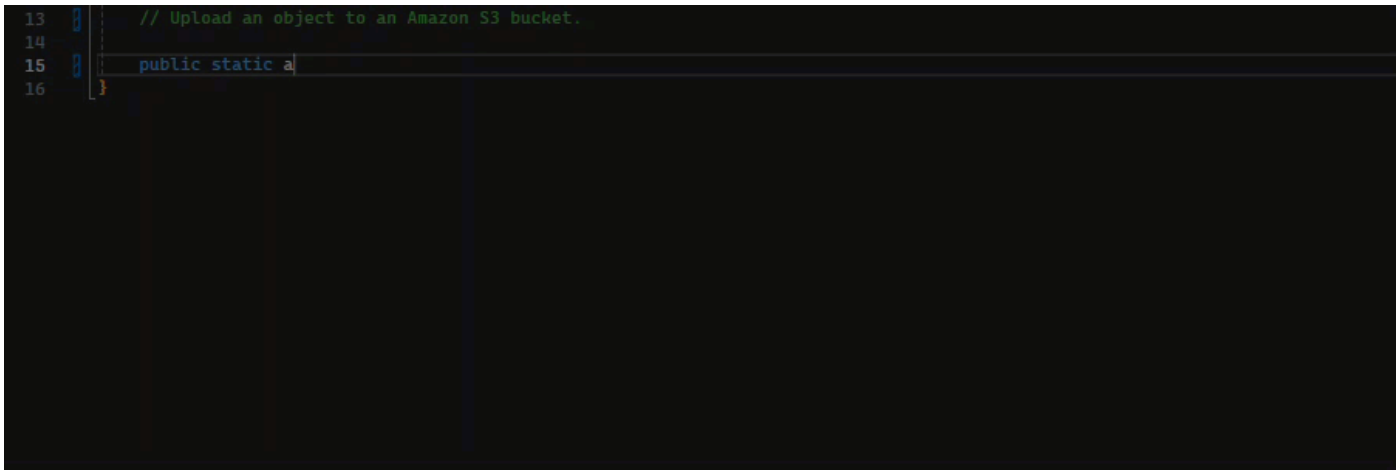
TypeScript

In this example, the user enters a full comment, and then Amazon Q supplies the code that goes with it.



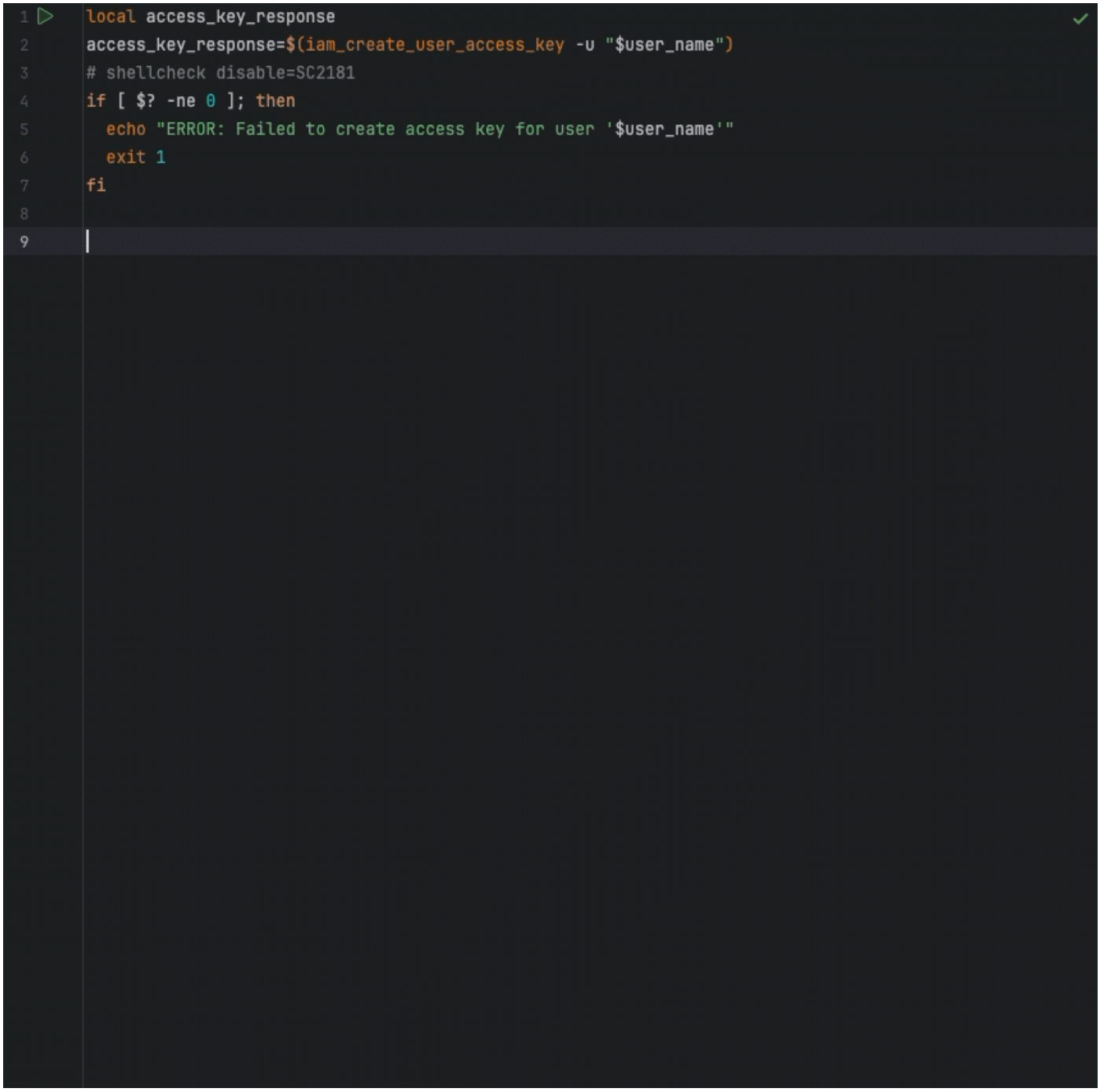
C#

In this example, Amazon Q provides a single-line recommendation based on a comment.



Shell

In the image below, Amazon Q offers recommendations on how to complete a single line of code.



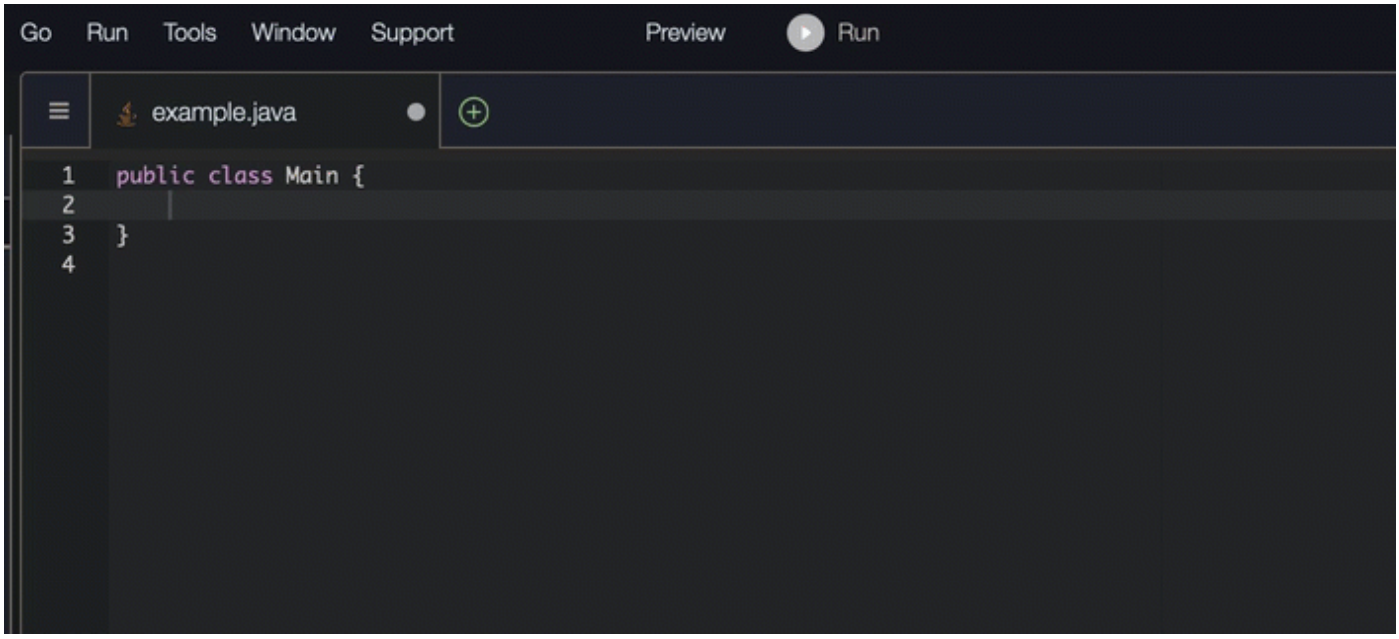
```
local access_key_response
access_key_response=$(iam_create_user_access_key -u "$user_name")
# shellcheck disable=SC2181
if [ $? -ne 0 ]; then
  echo "ERROR: Failed to create access key for user '$user_name'"
  exit 1
fi
```

Java

When you start typing out single lines of code, Amazon Q makes suggestions based on your current and previous inputs.

In the example below, in Java, a user enters the string `public` into an existing class.

Based on the input, Amazon Q generates a suggestion for the signature of the main method.



Python

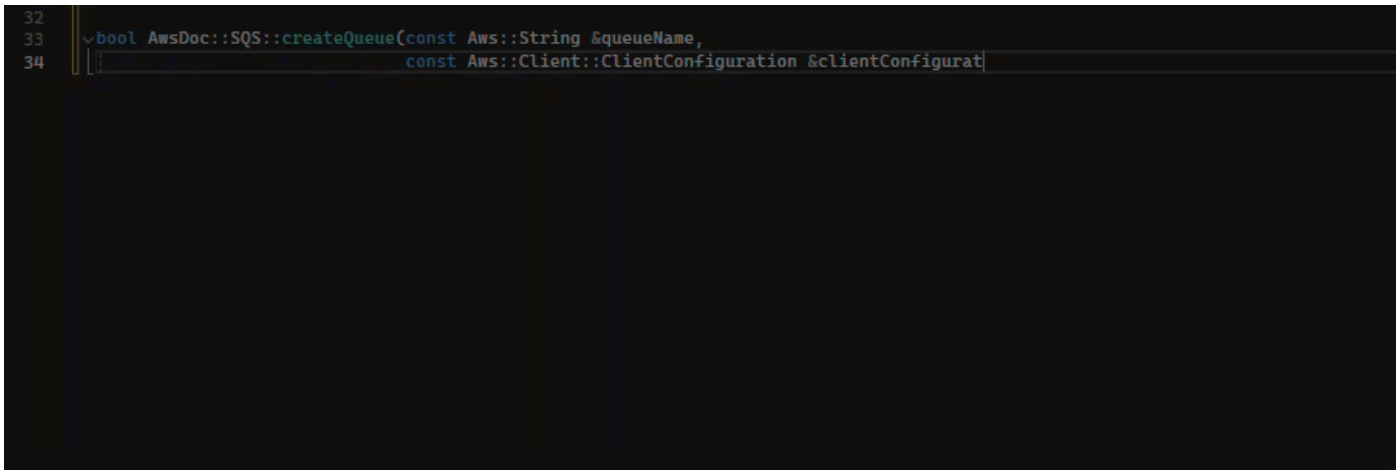
In this example, Amazon Q recommends a single line of code, based on the developer's comment.



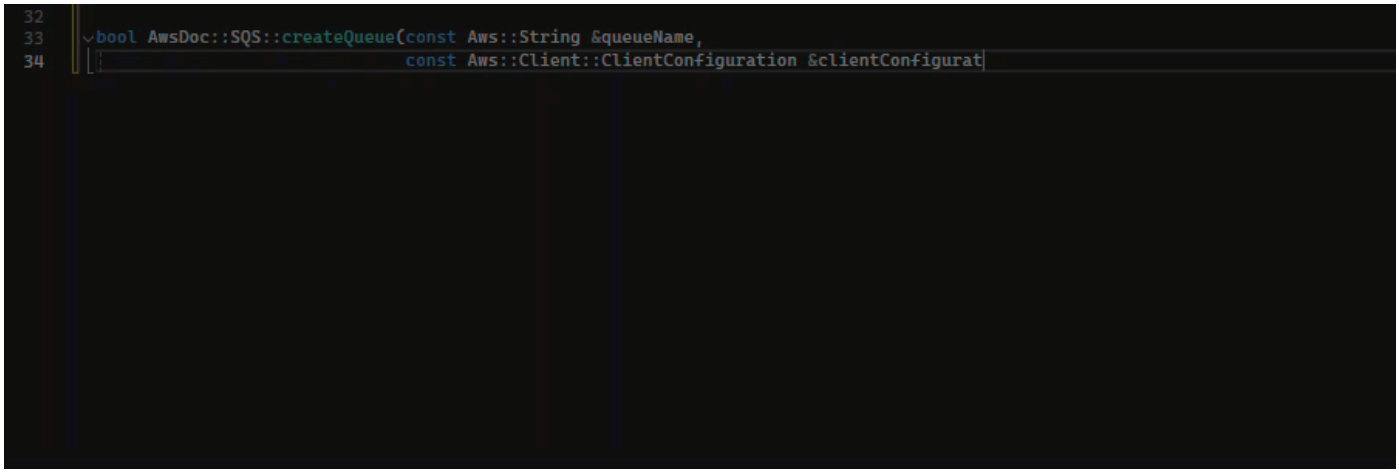
Using Amazon Q Developer for full function generation

Amazon Q can generate an entire function based on a comment that you've written. As you finish your comment Amazon Q will suggest a function signature. If you accept the suggestion, Amazon Q automatically advances your cursor to the next part of the function and makes a suggestion. Even if you enter an additional comment or line of code in between suggestions, Amazon Q will refactor based on your input.

C

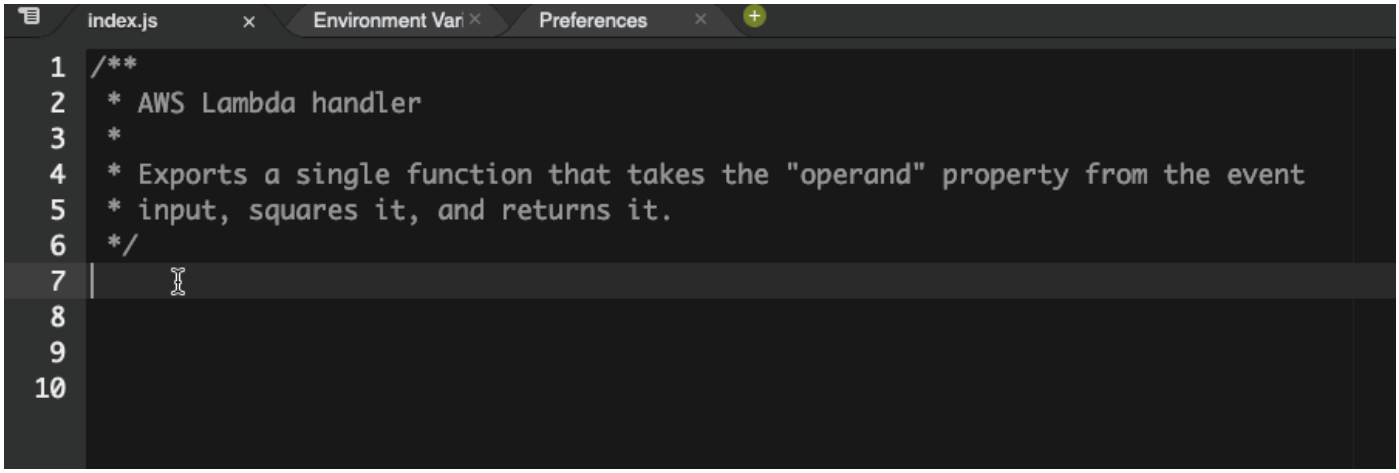


C++

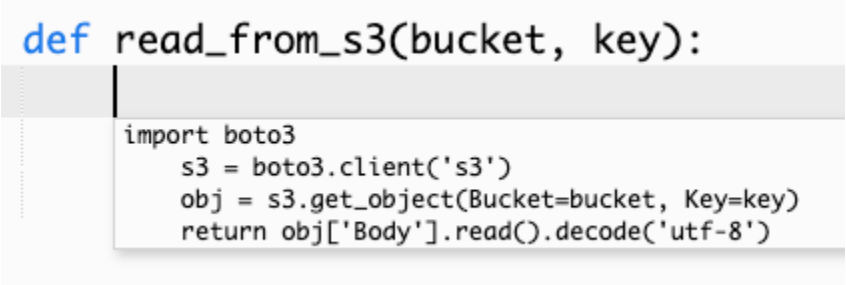


JavaScript

In the following example, the user generates, and then edits, a full function based on a set of comments.

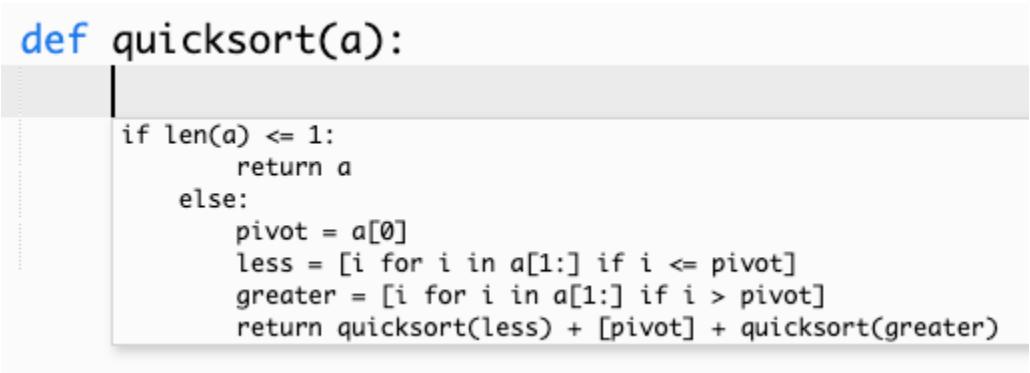


In the following image, a user has written a function signature for reading a file from Amazon S3. Amazon Q then suggests a full implementation of the `read_from_s3` method.



> **Note**
>
> Sometimes, as in the previous example, Amazon Q includes `import` statements as part of its suggestions. As a best practice, manually move these `import` statements to the top of your file.

As another example, in the following image, a user has written a function signature. Amazon Q then suggests a full implementation of the `quicksort` method.



Amazon Q considers past code snippets when making suggestions. In the following image, the user in the previous example has accepted the suggested implementation for `quicksort` above. The user then writes another function signature for a generic `sort` method. Amazon Q then suggests an implementation based on what has already been written.

```
def quicksort(a):
    if len(a) <= 1:
        return a
    else:
        pivot = a[0]
        less = [i for i in a[1:] if i <= pivot]
        greater = [i for i in a[1:] if i > pivot]
        return quicksort(less) + [pivot] + quicksort(greater)

def sort(a):
    return quicksort(a)
```

In the following image, a user has written a comment. Based on this comment, Amazon Q then suggests a function signature.

```
# Binary search function
def binary_search(arr, l, r, x):
```

In the following image, the user in the previous example has accepted the suggested function signature. Amazon Q can then suggest a complete implementation of the `binary_search` function.

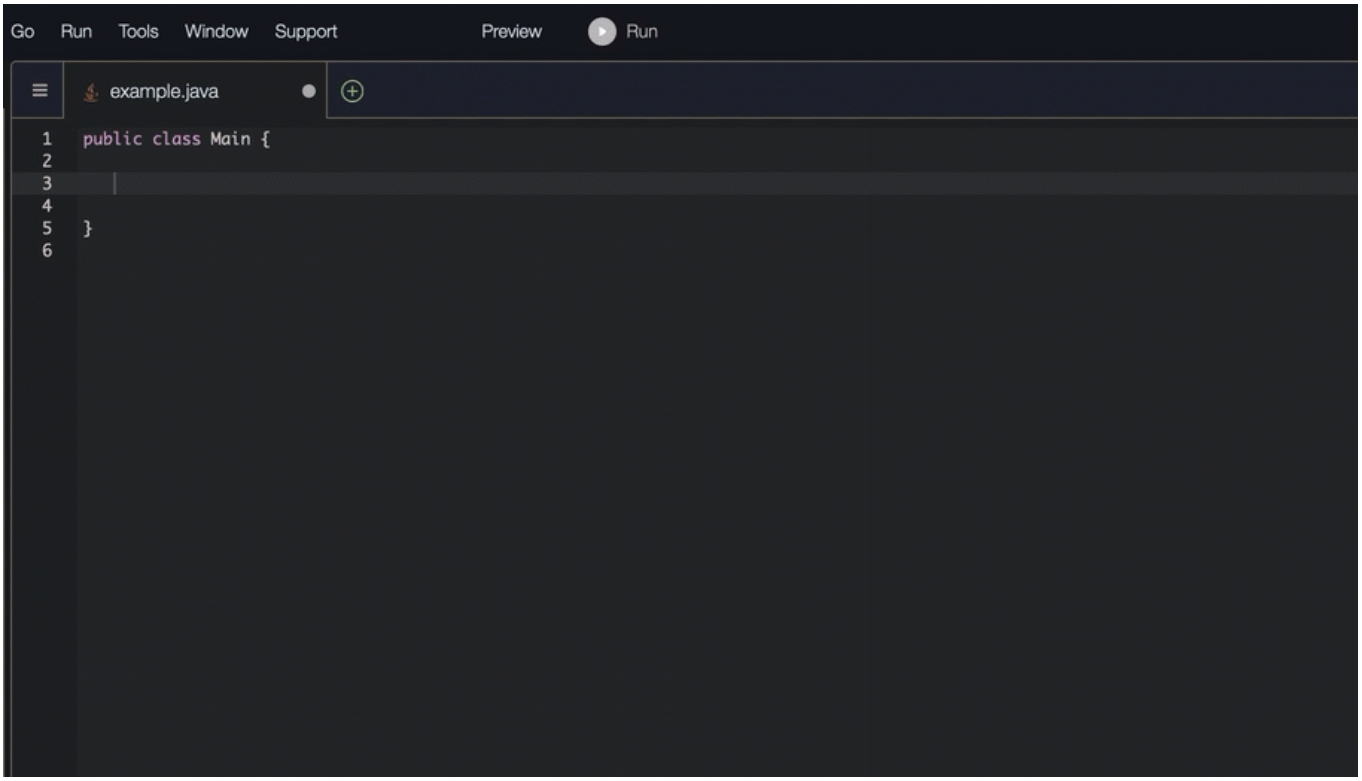
```
# Binary search function
def binary_search(arr, l, r, x):
    while l <= r:
        mid = l + (r - l) // 2
        if arr[mid] == x:
            return mid
        elif arr[mid] < x:
            l = mid + 1
        else:
            r = mid - 1
```

Java

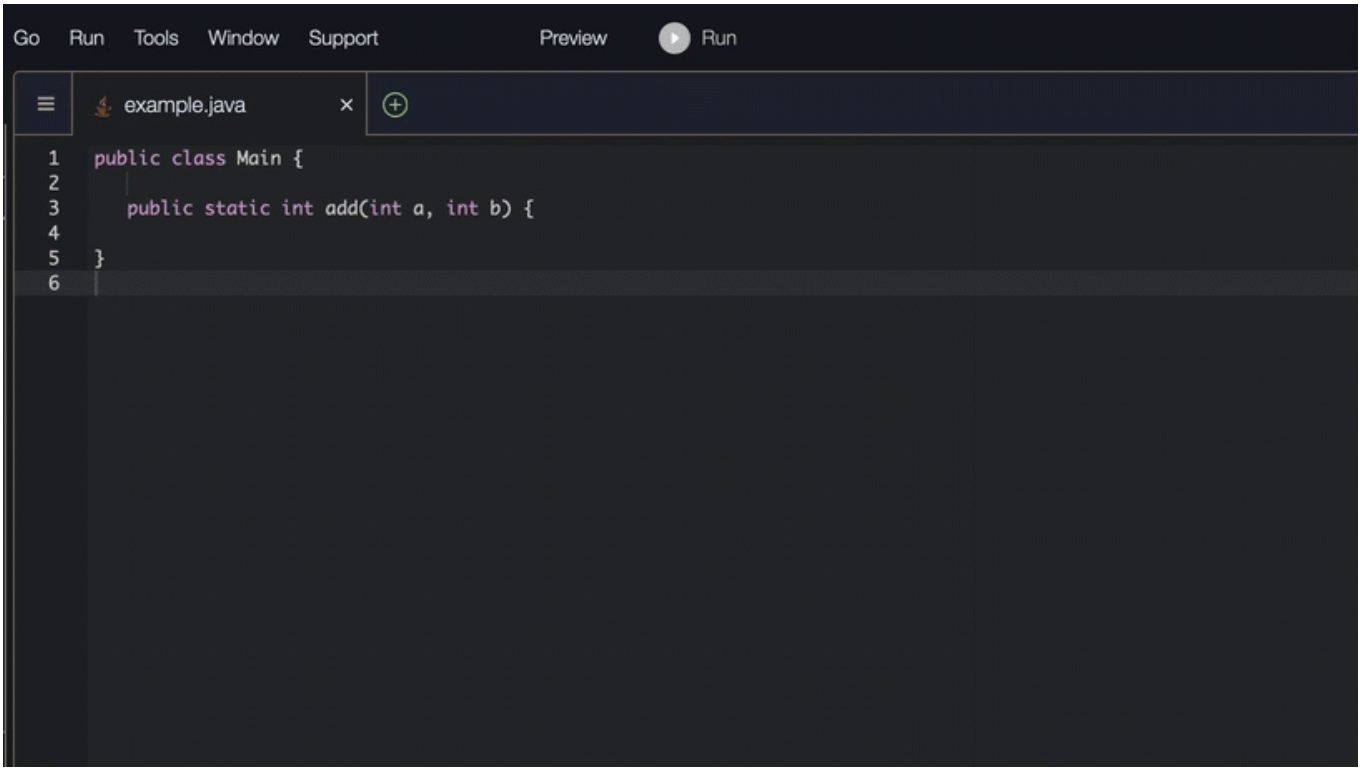
The following list contains examples of how Amazon Q makes suggestions and advances you through the entire process of creating a function.

1. In the following example, a user inputs a comment. Amazon Q suggests a function signature.

   After the user accepts that suggestion, Amazon Q suggests a function body.

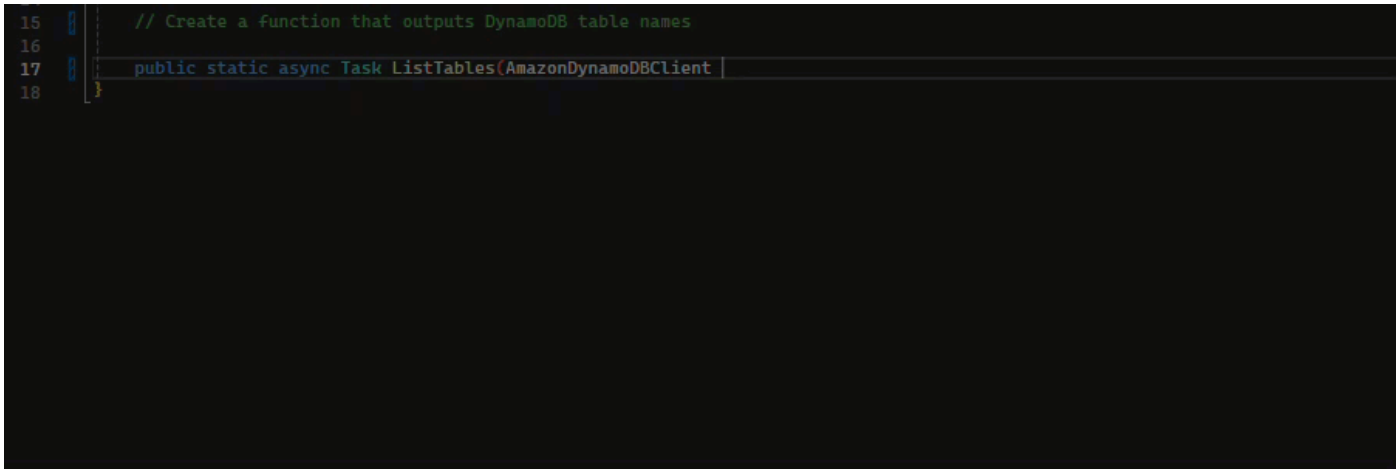


2. In the image below, a user inputs a comment in the body of the function prior to accepting a suggestion from Amazon Q. On the following line, Amazon Q generates a suggestion based on the comment.



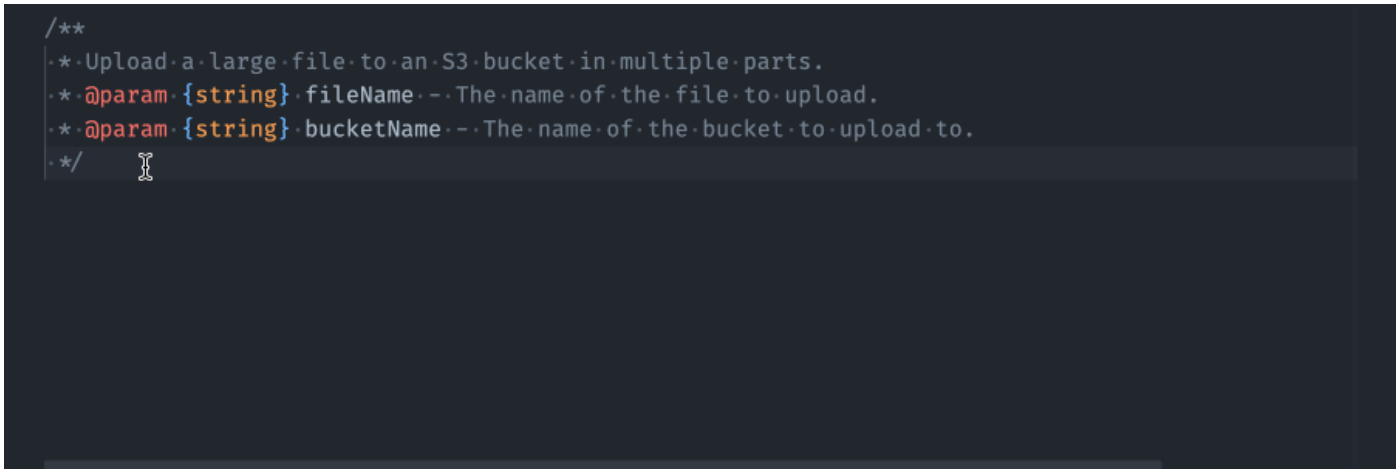
C#

In the following example, Amazon Q recommends a full function.



TypeScript

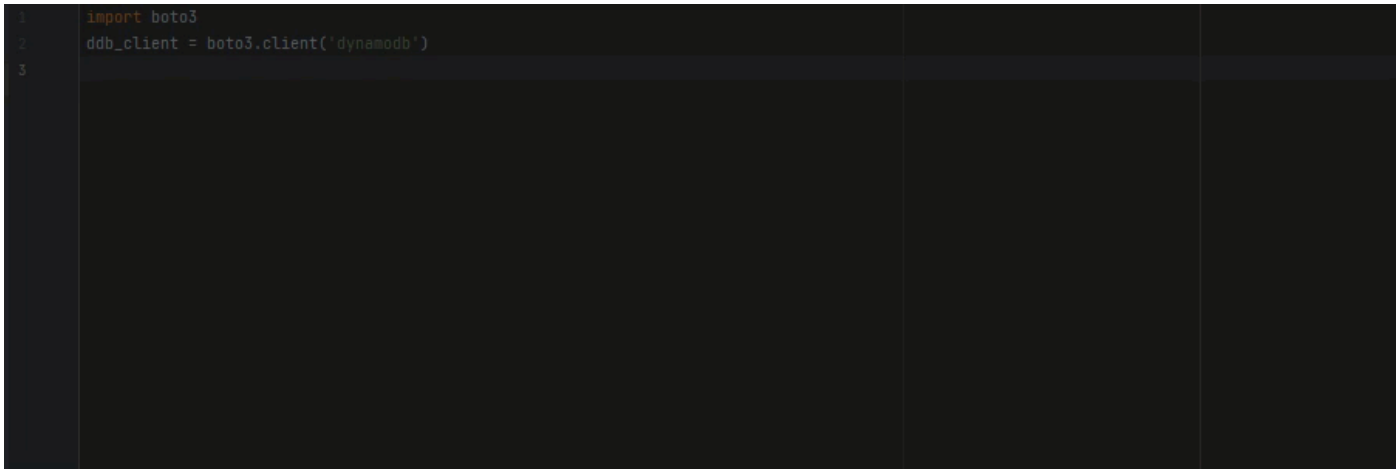
In the following example, Amazon Q generates a function based on the user's docstrings.



Python

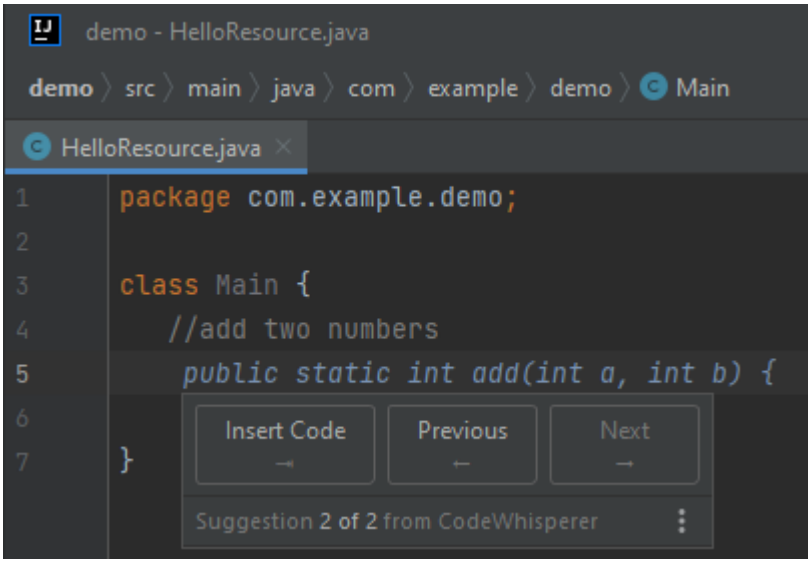
Amazon Q can generate an entire function based on a comment that you've written. As you finish your comment, Amazon Q will suggest a function signature. If you accept the suggestion, Amazon Q automatically advances your cursor to the next part of the function and makes a suggestion. Even if you enter an additional comment or line of code in between suggestions, Amazon Q will refactor based on your input.

In the following example, Amazon Q generates both a full function and the corresponding unit test.



The following list contains examples of how Amazon Q makes suggestions and advances you through the entire process of creating a function.

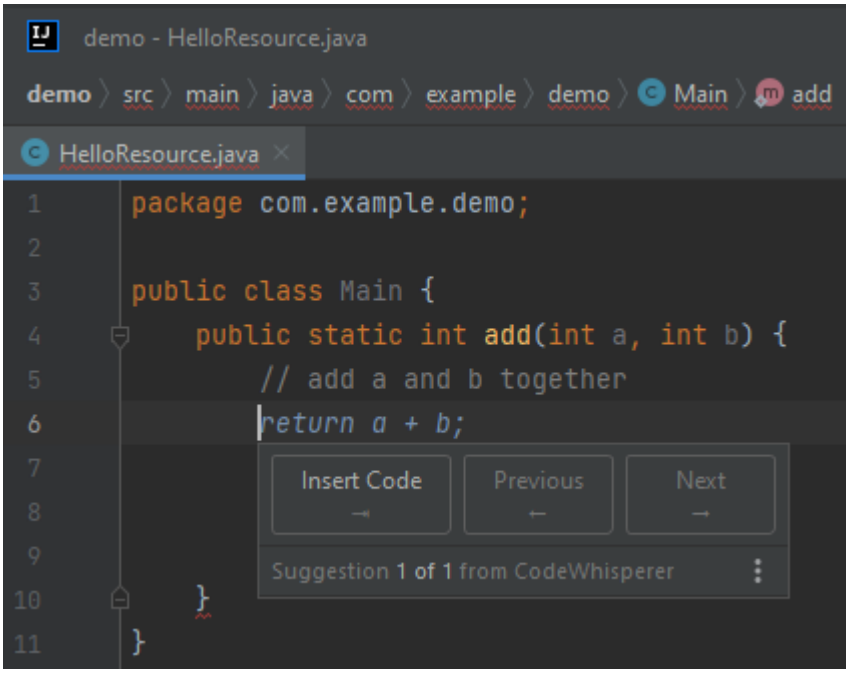
1. In the image below, a user has input a comment. The function signature, located below the comment, is a suggestion from Amazon Q.



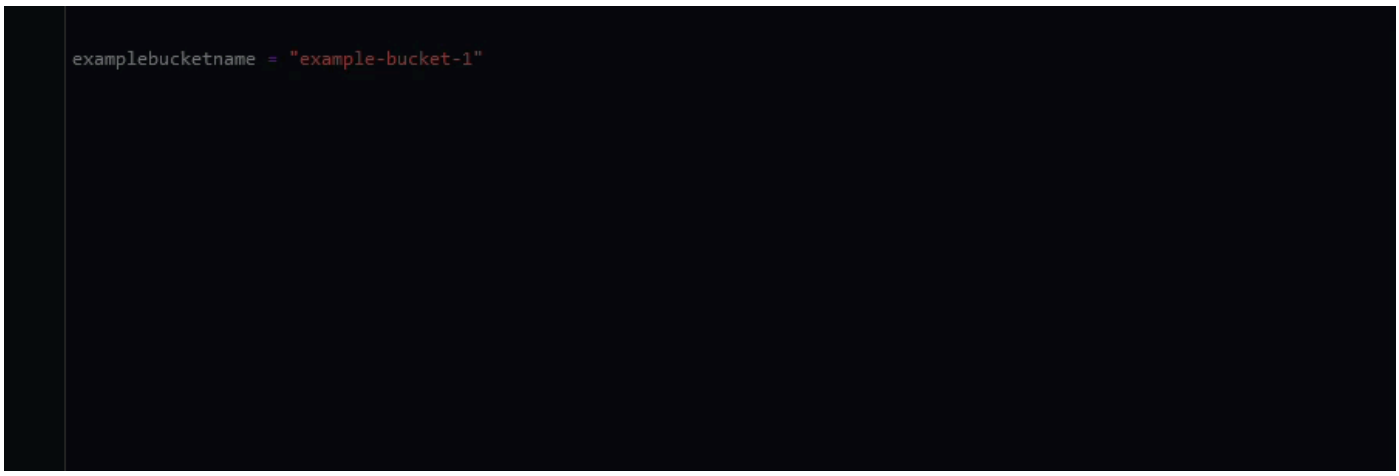
2. In the image below, the user has accepted the Amazon Q suggestion for a function signature. Accepting the suggestion automatically advanced the cursor and Amazon Q has made a new suggestion for the function body.



3. In the image below, a user input a comment in the body of the function prior to accepting a suggestion from Amazon Q. On the following line, Amazon Q has generated a new suggestion based on the content of the comment.



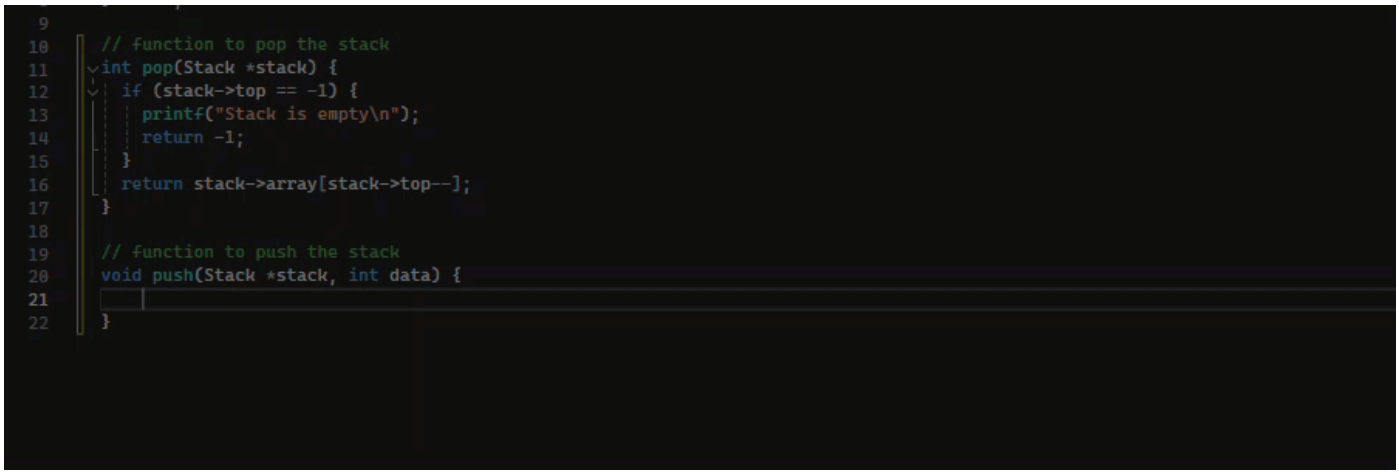
In this example, Amazon Q recommends a full function after the user types part of the signature.



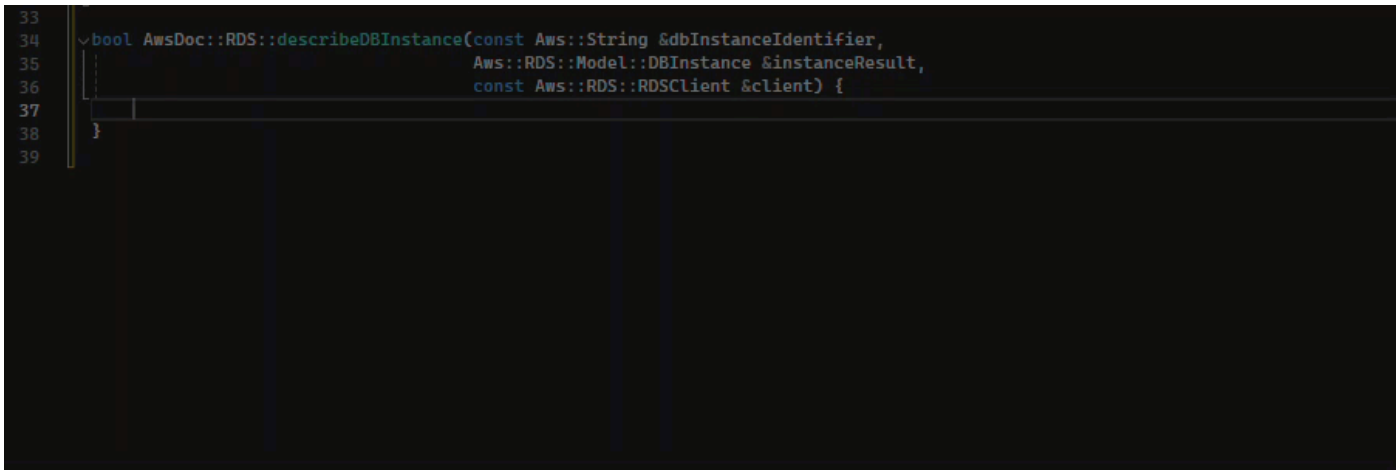
Using Amazon Q Developer for block completion

Block completion is used to complete your `if/for/while/try` code blocks.

C

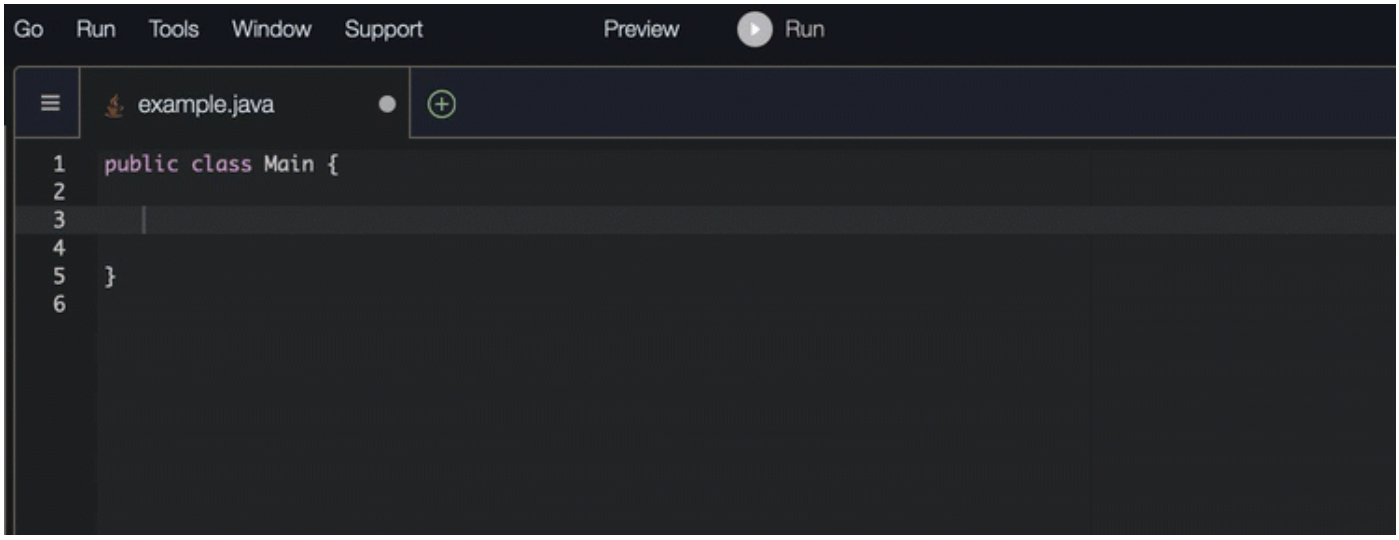


C++



Java

In the example below, a user enters the signature of an `if` statement. The body of the statement is a suggestion from Amazon Q.



C#

In the image below, Amazon Q recommends a way to complete the function.



TypeScript

In the image below, Amazon Q recommends a way to complete the function.



Python

In this example, Amazon Q recommends a block of code, based on the context.



Using Amazon Q Developer for Docstring, JSDoc, and Javadoc completion

Amazon Q can help you generate or complete documentation inside your code.

C++



Javascript

In this example, Amazon Q fills in JSDoc parameters based on existing constants.



C#

In this example, Amazon Q fills in JSDoc parameters based on existing constants.

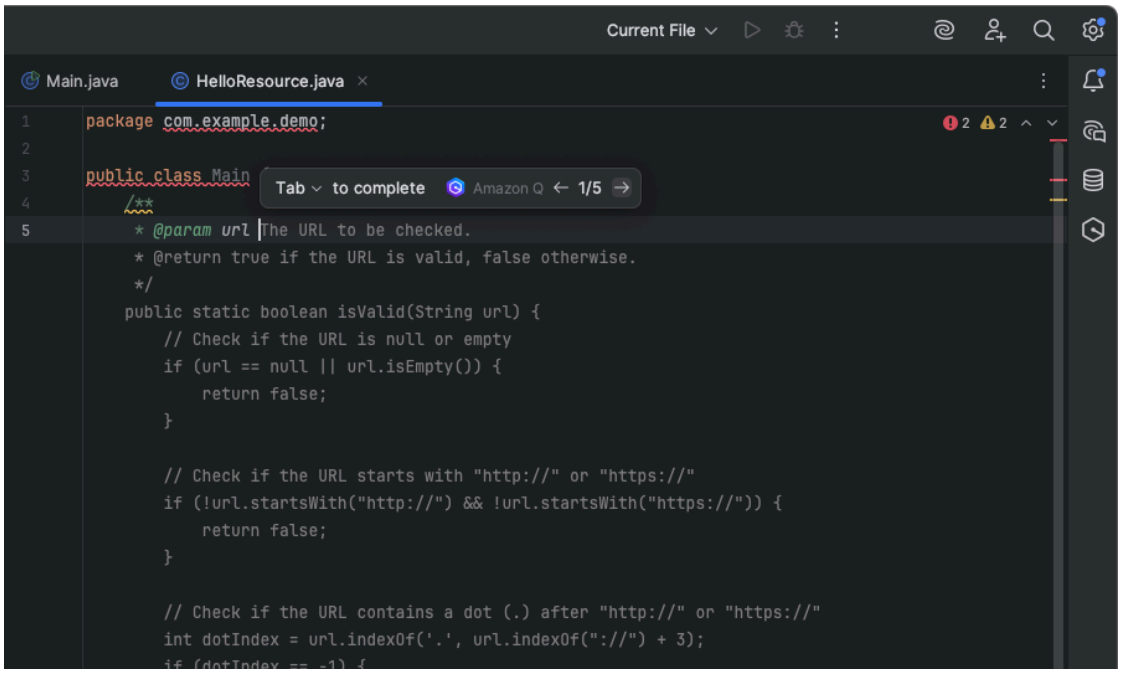
```
/// <summary>
/// Shows how to create a new Amazon S3 bucket.
/// </summary>
public static async Task<bool> CreateBucketAsync(IAmazonS3 client, string bucketName)
{
    try
    {
        var request = new PutBucketRequest
        {
            BucketName = bucketName,
            UseClientRegion = true,
        };

        var response = await client.PutBucketAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
    catch (AmazonS3Exception ex)
```

Java

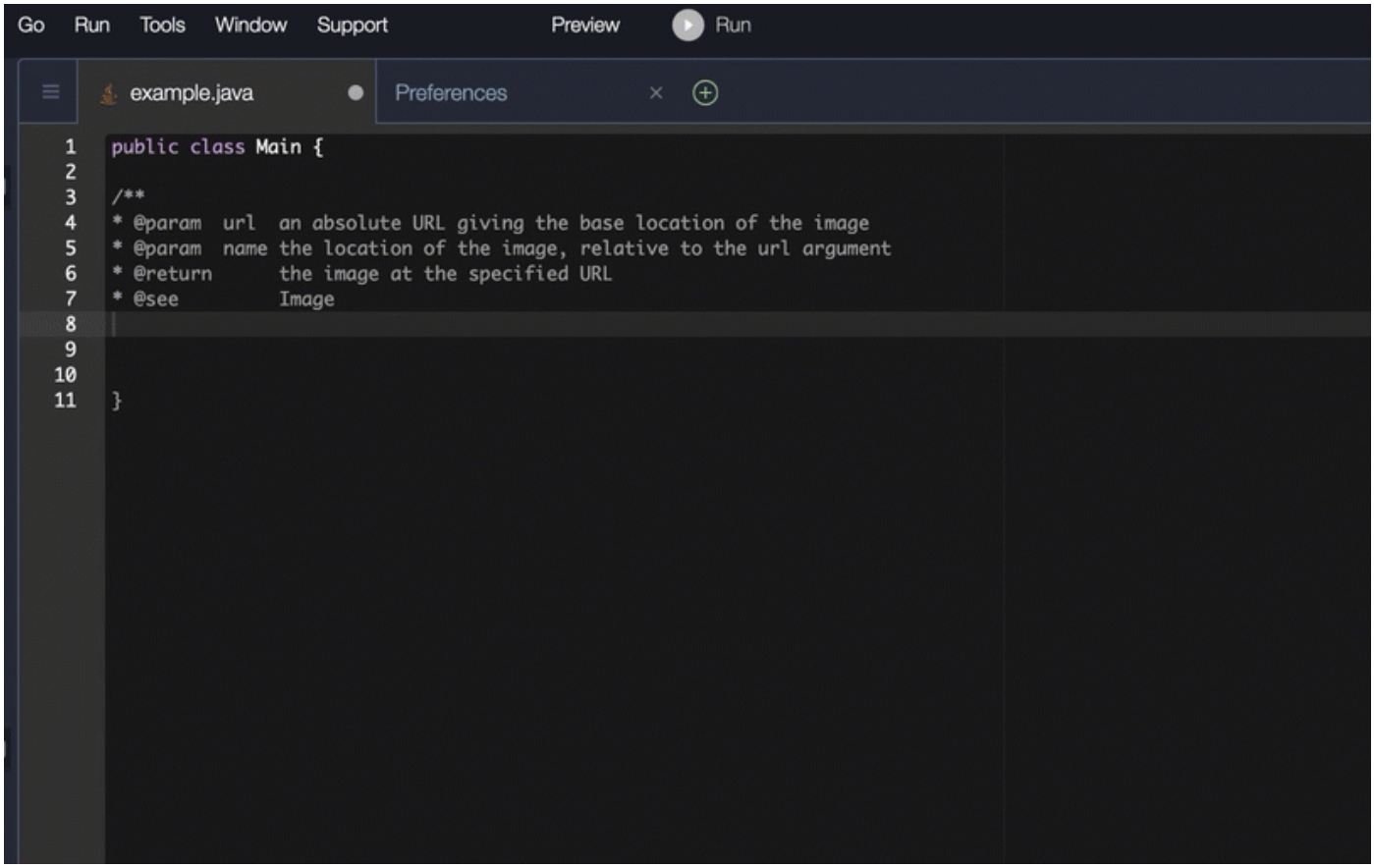
The following example is adapted from an example on the Oracle website.

In the image below, the user has started entering a docstring. Amazon Q has suggested words to add to the docstring.



The following example is adapted from an example on the Oracle website.

In the example below, in Java, the user enters a docstring. Amazon Q suggests a function to process the docstring.



Python

In this example, Amazon Q recommends a Docstring, based on the surrounding context.

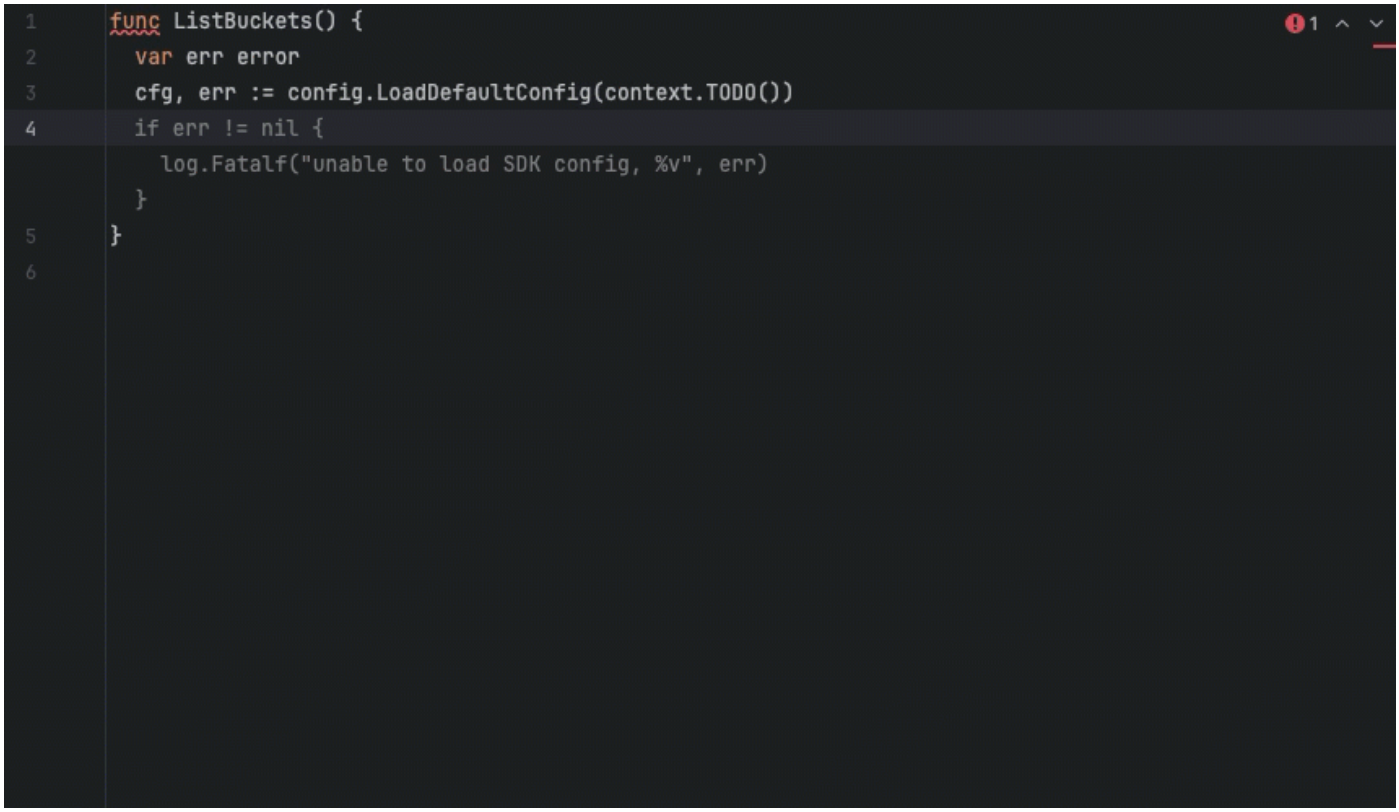


Using Amazon Q Developer for line-by-line recommendations

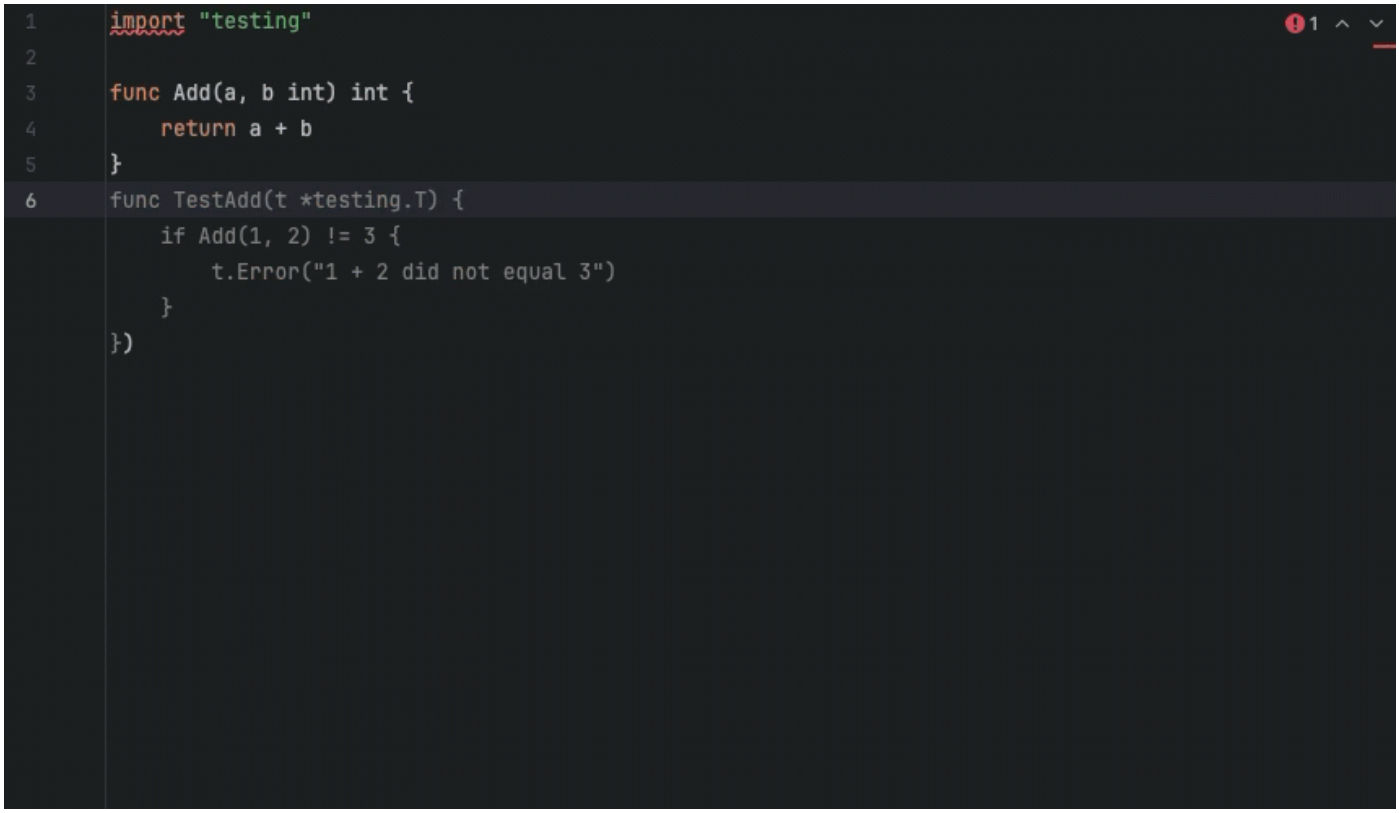
Depending on your use case, Amazon Q may not be able to generate an entire function block in one recommendation. However, Amazon Q can still provide line-by-line recommendations.

Go and GoLand

In this example, Amazon Q provides line-by-line recommendations.

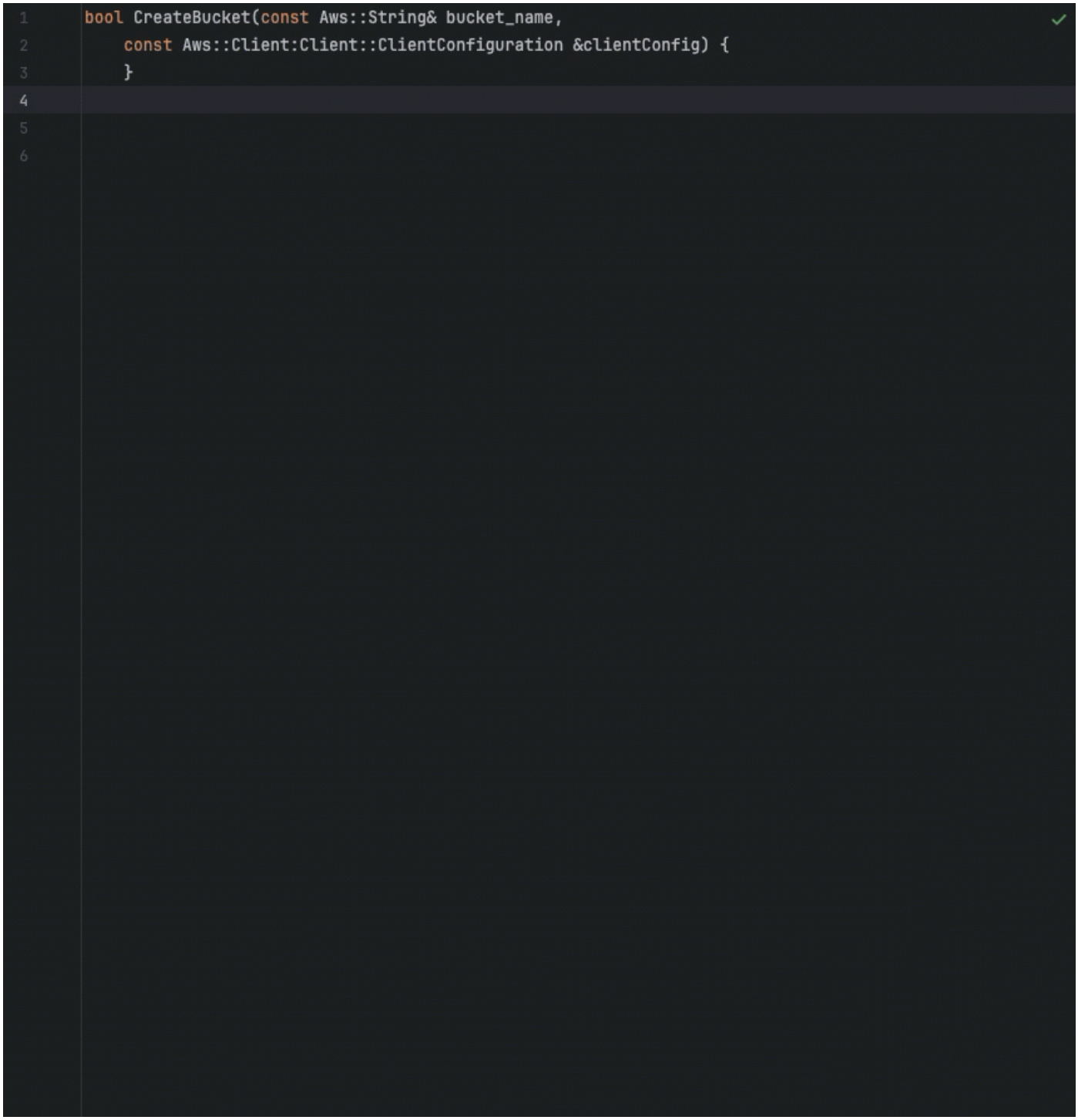


Here is another example of line-by-line recommendations, this time with a unit test.



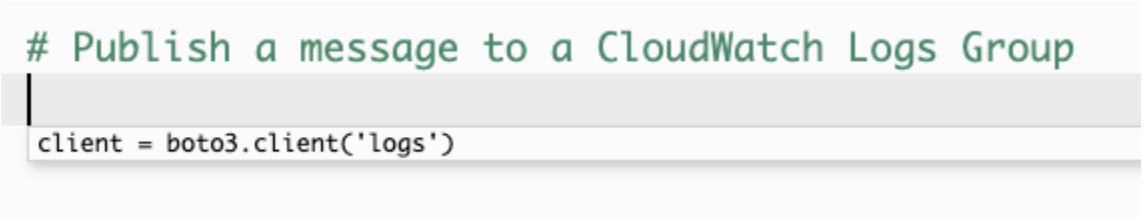
C++ and CLion

In this example, Amazon Q provides line-by-line recommendations.

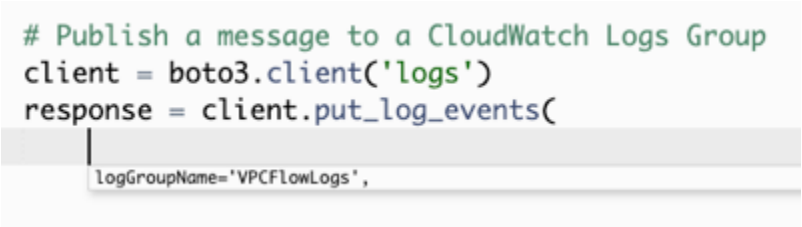


Python

In the following image, the customer has written an initial comment indicating that they want to publish a message to an Amazon CloudWatch Logs group. Given this context, Amazon Q is only able to suggest the client initialization code in its first recommendation, as shown in the following image.



However, if the user continues to request line-by-line recommendations, Amazon Q also continues to suggest lines of code based on what's already been written.



> **Note**
>
> In the example above, `VPCFlowLogs` may not be the correct constant value. As Amazon Q makes suggestions, remember to rename any constants as required.

Amazon Q can eventually complete the entire code block as shown in the following image.



In this example, Amazon Q provides recommendations, one line at at time.



Supported languages for Amazon Q Developer in the IDE

End of support notice

On April 30, 2027, AWS will discontinue support for Amazon Q Developer IDE plugins. For capabilities similar to Amazon Q Developer IDE plugins, explore Kiro to access the latest models and features, including agentic coding, chat and MCP support. For more information, see Amazon Q Developer IDE plugins end of support.

You can use the following features of Amazon Q Developer in the IDE with any programming language:

- Chat
- Inline chat

The quality of outputs while using these features varies based on the popularity of the language.

For the remaining features of Amazon Q in the IDE, the supported languages are listed in the following sections.

Language support for inline suggestions

Amazon Q supports inline code suggestions for multiple programming languages. The accuracy and quality of the code generation for a programming language depends on the size and quality of the training data.

In terms of the quality of the training data, the programming languages with the most support are:

- C
- C++
- C#
- Dart
- Go
- Java
- JavaScript
- Kotlin
- Lua
- PHP
- PowerShell
- Python
- R
- Ruby
- Rust
- Scala
- Shell
- SQL
- Swift
- SystemVerilog
- TypeScript

The Infrastructure as Code (IaC) languages with the most support are:

- CDK (Typescript, Python)
- HCL (Terraform)
- JSON
- YAML

Language support for transformations

The supported languages for transformation depend on the environment where you are transforming code.

In JetBrains IDEs and Visual Studio Code, Amazon Q can transform code in the following languages:

- Java
- Embedded SQL conversion for Oracle to PostgreSQL database migration

In Visual Studio, Amazon Q can transform code in the following languages:

- C# in .NET applications

For more information about supported languages and other prerequisites for transformation, see the topic for the type of transformation you're performing.

Language support for code reviews

Amazon Q can create code reviews and provide automatic code fixes for files and projects written in the following languages:

- Java - Java 17 and earlier
- JavaScript - ECMAScript 2021 and earlier
- Python - Python 3.11 and earlier, within the Python 3 series
- C# - All versions (.NET 6.0 and later recommended)
- TypeScript - All versions
- Ruby - Ruby 2.7 and 3.2
- Go - Go 1.18
- C - C11 and earlier
- C++ - C++17 and earlier
- PHP - PHP 8.2 and earlier
- Kotlin - Kotlin 2.0.0 and earlier
- Scala - Scala 3.2.2 and earlier

- JSX - React 17 and earlier
- Infrastructure as Code (IaC) languages
  - CloudFormation - 2010-09-09
  - Terraform - 1.6.2 and earlier
  - AWS CDK - TypeScript and Python

Language support for customizations

Amazon Q supports for the following languages, and uses the listed file types to create customizations:

- Bash/Shell (.sh, .zsh, .bash)
- C (.c, .h)
- C# (.cs)
- C++ (.cpp, .hpp, .h)
- Dart (.dart)
- Go (.go)
- HCL (.hcl)
- HTML (.html, .htm)
- Java (.java)
- JavaScript (.js, .jsx)
- JSON (.json)
- Kotlin (.kt, .kts)
- Markdown (.md, .mdx)
- PHP (.php)
- Powershell (.ps1, .psm1, .psd1)
- Python (.py)
- reStructuredText (.rst)
- Ruby (.rb)
- Rust (.rs)
- Scala (.scala)
- Terraform (.tf, .tfvars)

- Text (.txt)
- TypeScript (.ts, .tsx)
- YAML (.yaml, .yml)

Using Amazon Q Developer on the command line

The Q CLI has become the Kiro CLI.

For more information, see the Kiro User Guide.

Using MCP with Amazon Q Developer

The Model Context Protocol (MCP) is an open standard that enables AI assistants to interact with external tools and services. Amazon Q Developer CLI now supports MCP, allowing you to extend Q's capabilities by connecting it to custom tools and services.

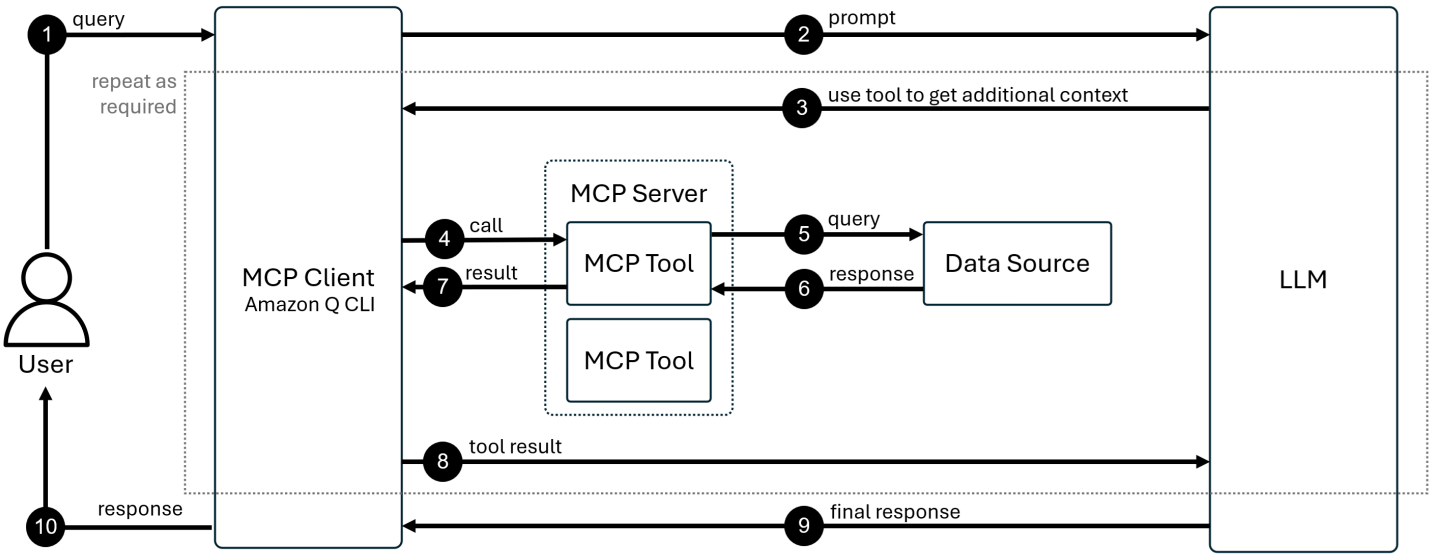
Topics

- MCP overview
- MCP configuration in the CLI
- MCP configuration
- Tools and prompts
- MCP configuration for Q Developer in the IDE
- Key benefits
- MCP architecture
- Core MCP concepts
- MCP security
- MCP governance for Q Developer

MCP overview

MCP (Model Context Protocol) is an open protocol that standardizes how AI assistants communicate with external tools. It defines a structured way for AI models to discover available tools, request tool execution with specific parameters, and receive and process tool results.

Think of MCP like a universal connector for AI models, enabling them to interact with external systems, fetch live data, and integrate with various tools seamlessly. This allows Amazon Q to provide more contextually relevant assistance by accessing the information it needs in real-time.



MCP configuration in the CLI

This page covers CLI-specific options for configuring MCP servers.

Configuration commands

Usage: `qchat mcp [OPTIONS]` *`COMMAND`*

MCP configuration commands

Command	Description
`qchat mcp add`	Add or replace a configured server
`qchat mcp remove`	Remove a server from the MCP configuration
`qchat mcp list`	List configured servers
`qchat mcp import`	Import a server configuration from another file
`qchat mcp status`	Get the status of a configured server
`qchat mcp help`	Print this list of commands or help for the given subcommand(s)

MCP server arguments

The `--args` parameter now supports comma-containing arguments using escaping or JSON array format:

```
# Escaped commas
q mcp add --name server --command cmd --args "arg1,arg2\,with\,commas,arg3"

# JSON array format
q mcp add --name server --command cmd --args '["arg1", "arg2,with,commas", "arg3"]'
```

Remote MCP servers

In addition to local MCP servers that run as processes, Amazon Q Developer CLI supports remote MCP servers that communicate over HTTP. Remote servers can use OAuth authentication or be open (no authentication required).

Configuration

Remote MCP servers are configured in your agent configuration file using the `type` and `url` fields:

```
{
  "mcpServers": {
    "find-a-domain": {
      "type": "http",
      "url": "https://api.findadomain.dev/mcp"
    }
  }
}
```

OAuth authentication flow

When using remote MCP servers that require OAuth authentication:

1. Start your Q CLI session with an agent that includes the remote MCP server
2. The server will initially show as "not yet loaded"
3. Use the `/mcp` command to begin authentication
4. Q CLI will indicate that the server requires authentication and provide a URL
5. Open the provided URL in your browser while keeping the Q CLI session open

6. Follow the authentication instructions in your browser
7. Return to the Q CLI window - you will be signed into the MCP server if authentication was successful

The server's tools will become available once authentication is complete.

MCP configuration

Setting up MCP servers with the Q CLI

The globally defined MCP configuration for Amazon Q CLI is handled at:

```
~/.aws/amazonq/cli-agents
```

Amazon Q Developer CLI supports both local MCP servers (that run as processes) and remote MCP servers (that communicate over HTTP). Remote servers can use OAuth authentication or be open with no authentication required.

For more information, see the custom agent configuration guide on the Q CLI Github repo and Remote MCP servers.

Setting up MCP servers with Q in the IDE

The globally defined MCP configuration for Amazon Q in the IDE is handled at:

```
~/.aws/amazonq/agents/default.json
```

For more information, see MCP configuration for Q Developer in the IDE.

MCP server loading

Amazon Q loads MCP servers in the background, allowing you to start interacting immediately without waiting for all servers to initialize. Tools become available progressively as their respective servers finish loading.

Checking server status

You can use the `/tools` command to see which servers are still loading and which tools are already available.

Configuring server initialization

You can customize the server initialization timeout using:

```
$ q settings mcp.initTimeout [value]
```

Where `[value]` is the timeout in milliseconds. This setting controls how long Amazon Q will wait for servers to initialize before allowing you to start interacting.

Tools and prompts

This section covers how to use MCP tools and prompts with Amazon Q Developer CLI.

Understanding MCP tools

MCP tools are executable functions that MCP servers expose to Amazon Q Developer CLI. They enable Amazon Q Developer to perform actions, process data, and interact with external systems on your behalf.

Each tool in MCP has:

- **Name**: A unique identifier for the tool
- **Description**: A human-readable description of what the tool does
- **Input Schema**: A JSON Schema defining the parameters the tool accepts
- **Annotations**: Optional hints about the tool's behavior and effects

Discovering available tools

To see what tools are available in your Q CLI session:

```
/tools
```

This command displays all available tools, including both built-in tools and those provided by MCP servers.

Tools can have different permission levels that determine how they're used:

- **Auto-approved**: These tools can be used without explicit permission for each invocation

- **Requires approval**: These tools need your explicit permission each time they're used
- **Dangerous**: These tools are marked as potentially risky and require careful consideration before approval

Using tools

You can use MCP tools in two ways:

1. **Natural Language Requests**: Simply describe what you want to do, and Q will determine which tool to use.
2. **Direct Tool Invocation**: You can also explicitly request Q to use a specific tool.

Working with prompts

MCP servers can provide predefined prompts that help guide Q in specific tasks:

- List available prompts: `/prompts`
- Use a prompt:
  - `@ prompt-name arg1 arg2`

Example of using a prompt with arguments:

```
@fetch https://docs.aws.amazon.com/amazonq/latest/qdeveloper-ug/command-line-mcp-configuration.html
```

MCP configuration for Q Developer in the IDE

This page covers IDE-specific options for configuring MCP servers.

Understanding MCP configuration files for Q Developer in the IDE

When you use the GUI to add an MCP server to Q Developer in the IDE, the configuration is stored in one of two files:

- At the global scope: ~/.aws/amazonq/default.json
- At the local scope: .amazonq/default.json

However, for legacy reasons, it is also possible to put MCP configuration information in two other locations:

- At the global scope: ~/.aws/amazonq/mcp.json
- At the local scope: .amazonq/mcp.json

Q Developer gives precedence to workspace level configurations for MCP servers, their permissions, and the settings stored.

> **Note**
>
> If you have already set up an MCP configuration in an mcp.json file, and you are using the MCP configuration GUI for the first time, you will see that configuration in the GUI.

Support for legacy mcp.json files is enabled by the useLegacyMcpJson field in your global default.json config file. By default, this field is set to true. For more information, see UseLegacyMcpJson Field in the Q Developer CLI GitHub repo.

Note that the mcp.json files may also be used by the Q CLI.

For information about how to set granular controls on MCP tooling, see the Built-in tools reference.

Accessing the MCP configuration UI

To access the MCP configuration UI in Q Developer in the IDE:

1. Open your IDE (VS Code, JetBrains, etc.).
2. Open the Q Developer panel.
3. Open the **Chat** panel.
4. Choose the tools icon.



Adding an MCP server

There are two primary transport mechanisms for communication between AI clients and MCP servers: STDIO and HTTP.

Adding an HTTP MCP server

To add an HTTP MCP server to the IDE:

1. Access the MCP configuration UI.
2. Choose the plus (+) symbol.
3. Select the scope: global or local.

   If you select global scope, the MCP server configuration is stored in ~/.aws/amazonq/default.json and available across all your projects. If you select local scope, the configuration is stored in .amazonq/default.json within your current project.
4. In the **Name** field, enter the name of the MCP server.
5. Select `http` as the transport protocol.
6. In the **URL** field, enter the URL that the MCP server will call when it initializes.
7. Under **Headers - optional**, you can enter key-value pairs that must be sent as HTTP request headers. .
8. Enter a **Timeout** value, as applicable.
9. Choose **Save**.

   The configuration panel will be replaced with the tool permissions panel.
10. Follow the procedure under Reviewing and adjusting tool permissions .

> **Note**
>
> If the MCP HTTP endpoint requires authorization, then Amazon Q will automatically open a browser page so that you can authorize Amazon Q to access the MCP server.

Adding an STDIO MCP server

To add an STDIO MCP server to the IDE:

1. Access the MCP configuration UI.
2. Choose the plus (+) symbol.
3. Select the scope: global or local.

If you select global scope, the MCP server configuration is stored in ~/.aws/amazonq/default.json and available across all your projects. If you select local scope, the configuration is stored in .amazonq/default.json within your current project.

4. In the **Name** field, enter the name of the MCP server.

   For example, if we were installing the AWS Documentation MCP server, the name might be *AWSDocMCPServer*.

5. Select `stdio` as the transport protocol.

6. In the **Command** field, enter the shell command that the MCP server will run when it initializes.

   In the case of the AWS Documentation MCP Server, the command is `uvx`. This is an alias for `uv tool run`, which creates an ephemeral Python environment.

7. In the **Arguments** field, enter an argument to be given to the shell command, if applicable.

   In the case of the AWS Documentation MCP Server, the argument is *awslabs.aws-documentation-mcp-server@latest*. This is a Python package identifier that points to a package hosted on PyPI (Python Package Index).

   Add more arguments as necessary.

8. Fill in environment variables as applicable.

   In the case of our example, we first fill in Name: *FASTMCP_LOG_LEVEL* and Value: *ERROR*.

   We will also use the name *AWS_DOCUMENTATION_PARTITION* and the value *aws* to indicate the partition that we'll be working with.

9. Enter a **Timeout** value, as applicable.

   For our example, we'll keep the recommended value of 60 (seconds).

10. Choose **Save**.

    The configuration panel will be replaced with the tool permissions panel.

11. Follow the procedure under Reviewing and adjusting tool permissions .

Troubleshooting your MCP configuration

After you add an MCP server in the IDE, Amazon Q will attempt to connect to it.

If there are connection issues, then an alert appears in the panel. You should not expect the tools from that MCP server to function properly until the alert is resolved.

Choose **Fix Configuration** to return to the MCP configuration screen so that you can make the appropriate changes.

Enabling an MCP server

The following procedure assumes that the MCP server in question is not already enabled.

To enable an MCP server in the IDE:

1. Open the MCP Servers panel.
2. Next to the server that you want to enable, choose **Enable**.

Disabling an MCP server

To disable an MCP server in the IDE:

1. Open the MCP Servers panel.
2. Choose the server you want to disable.
3. Choose the three dots next to **Edit setup**.
4. Choose **Disable MCP Server**.

Deleting an MCP server that is currently enabled

To delete an MCP server that is currently enabled from the IDE:

1. Open the MCP Servers panel.
2. Choose the server you want to delete.

   A panel will open with details about that server.
3. Choose the three dots next to **Edit setup**.
4. Choose **Delete MCP server**.
5. Confirm the deletion when prompted.

Deleting an MCP server that is currently disabled

To delete an MCP server that is currently disabled from the IDE:

1. Open the MCP Servers panel.
2. Next to the server that you want to delete, choose **Delete**.
3. Confirm the deletion when prompted.

Reviewing and adjusting tool permissions

To review and adjusting tool permissions:

1. Open the MCP Servers panel.
2. Choose the MCP server for which you want to review and adjust permissions.
3. For each tool, you can set one of the following permission levels:

    - Ask: Prompt for permission each time the tool is used.
    - Always allow: Allow the tool to run without prompting.
    - Deny: Do not use this tool.

Key benefits

- **Extensibility**: Connect Amazon Q to specialized tools for specific domains or workflows
- **Customization**: Create custom tools tailored to your specific needs
- **Ecosystem Integration**: Leverage the growing ecosystem of MCP-compatible tools
- **Standardization**: Use a consistent protocol supported by multiple AI assistants
- **Flexibility**: MCP allows you to switch between different LLM providers while maintaining the same tool integrations
- **Security**: Keep your data within your infrastructure with local MCP servers

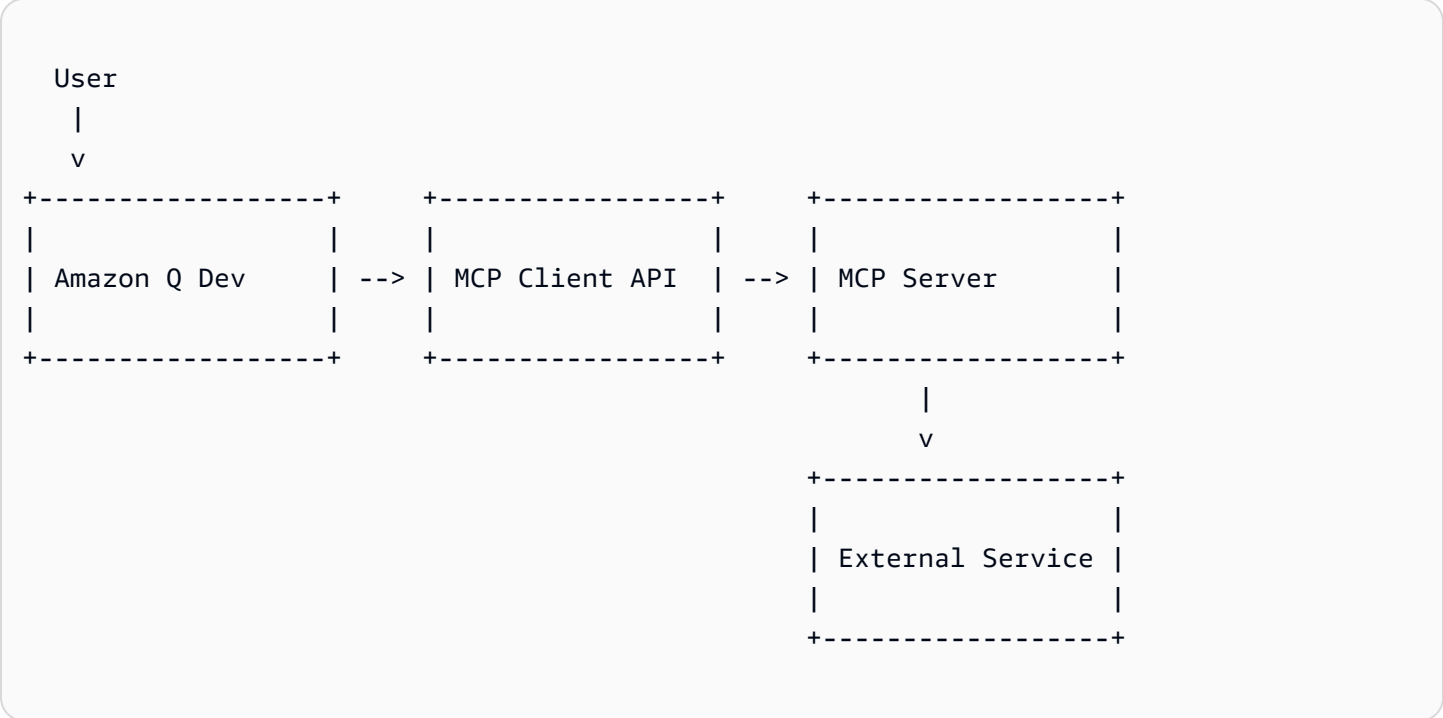
MCP architecture

MCP follows a client-server architecture where:

- **MCP Hosts**: Programs like Amazon Q Developer CLI that want to access data through MCP

- **MCP Clients**: Protocol clients that maintain 1:1 connections with servers
- **MCP Servers**: Lightweight programs that each expose specific capabilities through the standardized Model Context Protocol
- **Local Data Sources**: Your computer's files, databases, and services that MCP servers can securely access
- **Remote Services**: External systems available over the internet (e.g., through APIs) that MCP servers can connect to

Example MCP Communication Flow



<caption>

Communication flow between user, Amazon Q Developer CLI, and external services through MCP
</caption>

Core MCP concepts

Tools

Tools are executable functions that MCP servers expose to clients. They allow Amazon Q to:

- Perform actions in external systems

- Process data in specialized ways
- Interact with APIs and services
- Execute commands on your behalf

Tools are defined with a unique name, a description, an input schema (using JSON Schema), and optional annotations about the tool's behavior.

Prompts

Prompts are predefined templates that help guide Amazon Q in specific tasks. They can:

- Accept dynamic arguments
- Include context from resources
- Chain multiple interactions
- Guide specific workflows
- Surface as UI elements (like slash commands)

Resources

Resources represent data that MCP servers can provide to Amazon Q, such as:

- File contents
- Database records
- API responses
- Documentation
- Configuration data

MCP security

When using MCP servers with Amazon Q Developer CLI, it's important to understand the security implications and best practices.

Security model

The MCP security model in Amazon Q Developer CLI is designed with these principles:

1. **Explicit Permission**: Tools require explicit user permission before execution
2. **Local Execution**: MCP servers run locally on your machine
3. **Isolation**: Each MCP server runs as a separate process
4. **Transparency**: Users can see what tools are available and what they do

Security considerations

Key security considerations when using MCP:

- Only install servers from trusted sources
- Review tool descriptions and annotations before approving
- Use environment variables for sensitive configuration
- Keep MCP servers and the Q CLI updated
- Monitor MCP logs for unexpected activity

MCP governance for Q Developer

Pro-tier customers using IAM Identity Center as the sign-in method can control MCP access for users within their organization. By default, your users can use any MCP server in their Q client. As an administrator, you have the ability to either entirely disable the use of MCP servers by your users, or specify a vetted list of MCP servers that your users are allowed use.

You control these restrictions using an MCP on/off toggle and an MCP registry. The MCP toggle and registry attributes are part of the Q Developer profile, which can be defined at an organization level or at an account level, with the account-level profile superseding the organizational-level profile. You can specify a default MCP policy for your organization and override it for specific accounts; for example, disable MCP for the organization but enable it with an allow-list for certain teams (accounts).

> **Note**
>
> Both the toggle and the registry settings are enforced on the client side. Be aware that your end users could circumvent it.

Disabling MCP for your organization

To disable MCP for your account or organization:

1. Open the Kiro console.
2. Choose **Settings**.
3. Choose the **Q Developer** tab.
4. Toggle **Model Context Protocol (MCP)** to **Off**.

Specifying an MCP allow-list for your organization

To control which MCP servers your users can access, create a JSON file with the allowed servers, serve it over HTTPS, and add the URL to your Q Developer profile. Q Developer clients using this profile allow users to access only the MCP servers in your allow-list.

Specifying the MCP registry URL

1. Open the Kiro console.
2. Choose **Settings**.
3. Choose the **Q Developer** tab.
4. Ensure **Model Context Protocol (MCP)** is **On**.
5. In the **MCP Registry URL** field, choose **Edit**.
6. Enter the URL of an MCP registry JSON file containing the allow-listed MCP servers.
7. Choose **Save**.

The MCP registry URL is encrypted both in transit and at rest in accordance with our data encryption policy.

MCP registry file format

The format of the registry JSON file is a subset of the server schema JSON in the MCP registry standard v0.1. The JSON schema definition for the subset supported by Q Developer is available in the registry schema section at the end of this document.

The following example shows an MCP registry file containing both a remote (HTTP) and a local (stdio) MCP server definition.

```
{
  "servers": [
    {
      "server": {
        "name": "my-remote-server",
        "title": "My server",
        "description": "My server description",
        "version": "1.0.0",
        "remotes": [
          {
            "type": "streamable-http",
            "url": "https://acme.com/my-server",
            "headers": [
              {
                "name": "X-My-Header",
                "value": "SomeValue"
              }
            ]
          }
        ]
      }
    },
    {
      "server": {
        "name": "my-local-server",
        "title": "My server",
        "description": "My server description",
        "version": "1.0.0",
        "packages": [
          {
            "registryType": "npm",
            "registryBaseUrl": "https://npm.acme.com",
            "identifier": "@acme/my-server",
            "transport": {
              "type": "stdio"
            },
            "runtimeArguments": [
              {
                "type": "positional",
                "value": "-q"
              }
            ],
            "packageArguments": [
```

```
              {
                "type": "positional",
                "value": "start"
              }
            ],
            "environmentVariables": [
              {
                "name": "ENV_VAR",
                "value": "ENV_VAR_VALUE"
              }
            ]
          }
        ]
      }
    }
  ]
}
```

The following table lists the properties for the registry JSON file. All properties are mandatory, unless otherwise noted. See the registry schema section for the full JSON schema.

Nested attributes appear indented from their parent. For example, "headers" is a child attribute of "remotes", and "name" and "value" are child attributes of "headers".

Attribute	Description	Optional?	Example value
Common attributes			
name	Server name. Must be unique within a given registry file.		"aws-ccapi-mcp"
title	Human-readable server name.	Yes	"AWS CC API"
description	Description of server.		"Manage AWS infra through natural language."
version	Version of server. Semantic versioning (x.y.z) is strongly recommended.		"1.0.2"

Attribute	Description	Optional?	Example value
Remote (HTTP) server attributes			
remotes	Array with exactly one entry specifying the remote endpoint.		-
type	Must be one of "streamable-http" or "sse".		"streamable-http"
url	MCP server endpoint URL.		"https://mcp.figma.com/mcp"
headers	Array of HTTP headers to include in each request.	Yes	-
name	HTTP header name.		"Authorization"
value	HTTP header value.		"Bearer mF_9.B5f-4.1JqM"
Local (stdio) server attributes			
packages	Array with exactly one entry containing the MCP server definition.		-

Attribute	Description	Optional?	Example value
registryType	Must be one of "npm", "pypi", or "oci". The following package runners are used to download and run the MCP server package: • For registry type "npm", the "npx" runner is used • For "pypi", "uvx" is used • For "oci", "docker" is used Client machines must have the appropriate package runners pre-installed.		“npm”
registryBaseUrl	Package registry URL.	Yes	"https://npm.acme.com"
identifier	Server package identifier.		"@acme/my-server"
transport	Object with exactly one property, "type".		-
type	Must be "stdio".		“stdio”
runtimeArguments	Array of arguments provided to the runtime, that is, to npx, uvx or docker.	Yes	-
type	Must be "positional".		“positional”
value	Runtime argument value.		“-q”
packageArguments	Array of arguments provided to the MCP server.	Yes	-

Attribute	Description	Optional?	Example value
type	Must be "positional".		“positional”
value	Package argument value.		“start”
environmentVariables	Array of env vars to set before starting the server.	Yes	-
name	Environment variable name.		"LOG_LEVEL"
value	Environment variable value.		“INFO”

Serving the MCP registry file

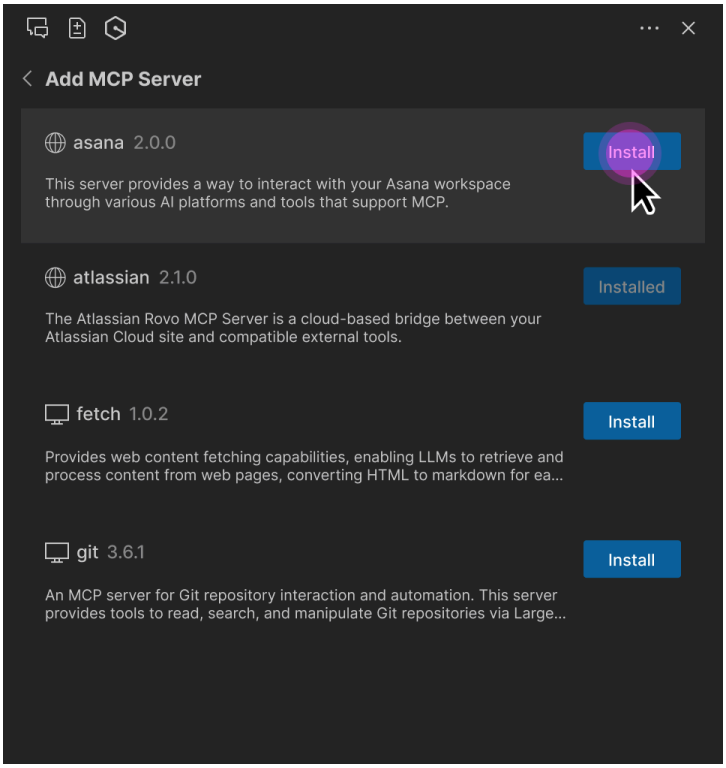
Serve the MCP registry JSON file over HTTPS using any web server, such as Amazon S3, Apache, or nginx. The URL must be accessible to Q Developer clients on your users' computers but can be private to your corporate network.

The HTTPS endpoint must have a valid SSL certificate signed by a trusted Certificate Authority. Self-signed certificates are not supported.

Q Developer fetches the MCP registry at startup and every 24 hours. During periodic synchronization, if a locally installed MCP server is no longer in the registry, Q Developer terminates that server and prevents users from adding it back. If the locally installed server has a different version than the server in the registry, Q Developer relaunches the server with the version defined in the registry.

Q Developer plugins

When users launch Q Developer, it checks whether a registry URL is defined in the profile. If so, it retrieves the registry JSON at that URL and enforces that users can only use the MCP servers defined in the registry. When users add an MCP server, Q Developer displays a list of servers from the registry.



Registry MCP server parameters (URL, package identifier, runtimeArguments, and so forth) are read-only. However, users can:

1. Adjust MCP tool permissions (“Ask to run”, “Always run”, or “Deny”).
2. Select MCP server scope (Global or Workspace).
3. Change the request timeout.
4. Specify additional environment variables for local MCP servers.
5. Specify additional HTTP headers for remote MCP servers.

> **Note**
>
> User-specified environment variables or HTTP headers override registry definitions. This allows users to specify attributes specific to their setup, such as authentication keys or local folder paths.

MCP registry JSON schema

The following JSON schema defines the MCP registry file format supported by Q Developer. You can use this schema to validate any registry files that you create.

```
{
  "$schema": "https://json-schema.org/draft-07/schema",
  "properties": {
    "servers": {
      "type": "array",
      "items": {
        "type": "object",
        "properties": {
          "server": {
            "$ref": "#/definitions/ServerDetail"
          }
        },
        "required": [
          "server"
        ]
      }
    }
  },
  "definitions": {
    "ServerDetail": {
      "properties": {
        "name": {
          "description": "Server name. Must be unique within a given registry file.",
          "example": "weather-mcp",
          "maxLength": 200,
          "minLength": 3,
          "pattern": "^[a-zA-Z0-9._-]+$",
          "type": "string"
        },
        "title": {
          "description": "Optional human-readable title or display name for the MCP
 server. MCP subregistries or clients MAY choose to use this for display purposes.",
          "example": "Weather API",
          "maxLength": 100,
          "minLength": 1,
          "type": "string"
        },
        "description": {
```

```
        "description": "Clear human-readable explanation of server functionality.
Should focus on capabilities, not implementation details.",
        "example": "MCP server providing weather data and forecasts via
OpenWeatherMap API",
        "maxLength": 100,
        "minLength": 1,
        "type": "string"
      },
      "version": {
        "description": "Version string for this server. SHOULD follow semantic
versioning (e.g., '1.0.2', '2.1.0-alpha'). Equivalent of Implementation.version in MCP
specification. Non-semantic versions are allowed but may not sort predictably. Version
ranges are rejected (e.g., '^1.2.3', '~1.2.3', '\u003e=1.2.3', '1.x', '1.*').",
        "example": "1.0.2",
        "maxLength": 255,
        "type": "string"
      },
      "packages": {
        "items": {
          "$ref": "#/definitions/Package"
        },
        "type": "array",
        "minItems": 0,
        "maxItems": 1
      },
      "remotes": {
        "items": {
          "anyOf": [
            {
              "$ref": "#/definitions/StreamableHttpTransport"
            },
            {
              "$ref": "#/definitions/SseTransport"
            }
          ]
        },
        "type": "array",
        "minItems": 0,
        "maxItems": 1
      }
    },
    "required": [
      "name",
      "description",
```

```
        "version"
      ],
      "type": "object"
    },
    "Package": {
      "properties": {
        "registryType": {
          "description": "Registry type indicating how to download packages (e.g.,
'npm', 'pypi', 'oci')",
          "enum": [
            "npm",
            "pypi",
            "oci"
          ],
          "type": "string"
        },
        "registryBaseUrl": {
          "description": "Base URL of the package registry",
          "examples": [
            "https://registry.npmjs.org",
            "https://pypi.org",
            "https://docker.io"
          ],
          "format": "uri",
          "type": "string"
        },
        "identifier": {
          "description": "Package identifier - either a package name (for registries)
or URL (for direct downloads)",
          "examples": [
            "@modelcontextprotocol/server-brave-search",
            "https://github.com/example/releases/download/v1.0.0/package.mcpb"
          ],
          "type": "string"
        },
        "transport": {
          "anyOf": [
            {
              "$ref": "#/definitions/StdioTransport"
            },
            {
              "$ref": "#/definitions/StreamableHttpTransport"
            },
            {
```

```
              "$ref": "#/definitions/SseTransport"
            }
          ],
          "description": "Transport protocol configuration for the package"
        },

        "runtimeArguments": {
          "description": "A list of arguments to be passed to the package's runtime
command (such as docker or npx).",
          "items": {
            "$ref": "#/definitions/PositionalArgument"
          },
          "type": "array"
        },
        "packageArguments": {
          "description": "A list of arguments to be passed to the package's binary.",
          "items": {
            "$ref": "#/definitions/PositionalArgument"
          },
          "type": "array"
        },
        "environmentVariables": {
          "description": "A mapping of environment variables to be set when running the
package.",
          "items": {
            "$ref": "#/definitions/KeyValueInput"
          },
          "type": "array"
        }
      },
      "required": [
        "registryType",
        "identifier",
        "transport"
      ],
      "type": "object"
    },
    "StdioTransport": {
      "properties": {
        "type": {
          "description": "Transport type",
          "enum": [
            "stdio"
          ],
```

```
          "example": "stdio",
          "type": "string"
        }
      },
      "required": [
        "type"
      ],
      "type": "object"
    },
    "StreamableHttpTransport": {
      "properties": {
        "type": {
          "description": "Transport type",
          "enum": [
            "streamable-http"
          ],
          "example": "streamable-http",
          "type": "string"
        },
        "url": {
          "description": "URL template for the streamable-http transport. Variables in
{curly_braces} reference argument valueHints, argument names, or environment variable
names. After variable substitution, this should produce a valid URI.",
          "example": "https://api.example.com/mcp",
          "type": "string"
        },
        "headers": {
          "description": "HTTP headers to include",
          "items": {
            "$ref": "#/definitions/KeyValueInput"
          },
          "type": "array"
        }
      },
      "required": [
        "type",
        "url"
      ],
      "type": "object"
    },
    "SseTransport": {
      "properties": {
        "type": {
          "description": "Transport type",
```

```
          "enum": [
            "sse"
          ],
          "example": "sse",
          "type": "string"
        },
        "url": {
          "description": "Server-Sent Events endpoint URL",
          "example": "https://mcp-fs.example.com/sse",
          "format": "uri",
          "type": "string"
        },
        "headers": {
          "description": "HTTP headers to include",
          "items": {
            "$ref": "#/definitions/KeyValueInput"
          },
          "type": "array"
        }
      },
      "required": [
        "type",
        "url"
      ],
      "type": "object"
    },
    "PositionalArgument": {
      "properties": {
        "type": {
          "enum": [
            "positional"
          ],
          "example": "positional",
          "type": "string"
        },
        "value": {
          "description": "The value for the input.",
          "type": "string"
        }
      },
      "required": [
        "type",
        "value"
      ],
```

```
      "type": "object"
    },
    "KeyValueInput": {
      "properties": {
        "name": {
          "description": "Name of the header or environment variable.",
          "example": "SOME_VARIABLE",
          "type": "string"
        },
        "value": {
          "description": "The value for the input.",
          "type": "string"
        }
      },
      "required": [
        "name"
      ],
      "type": "object"
    }
  },
  "required": [
    "servers"
  ],
  "type": "object"
}
```

Third-party integration with Amazon Q Developer

Amazon Q Developer integrates with popular development platforms to enhance your software development workflows through specialized artificial intelligence (AI) capabilities. Supported integrations include and GitLab Duo and GitHub, providing AI-powered assistance throughout the development lifecycle. These integrations help streamline development by automating routine tasks and improving code quality.

GitLab Duo with Amazon Q Developer

GitLab Duo with Amazon Q Developer provides a comprehensive suite of AI experiences integrated directly into your GitLab workflows. Available for GitLab Self-Managed offering and Ultimate tier subscribers, the integration enables quick actions in GitLab issues and merge requests to trigger AI capabilities. The integration also includes GitLab Duo Chat powered by Amazon Q, providing contextual assistance throughout your development process.

GitLab Duo with Amazon Q provides:

- Feature development of high-level ideas with a quick action in GitLab issues
- Code reviews for code quality, issues, and security concerns with a quick action in merge requests
- Unit test generation with a quick action in merge requests
- Integrated chat support for development tasks

To get started, see Set up GitLab Duo with Amazon Q.

Amazon Q Developer for GitHub (Preview)

The Amazon Q Developer integration with GitHub enables automated feature development and code reviews through specialized AI agents. When you assign a GitHub issue to Amazon Q Developer, it uses the issue and project code as context to generate new code and create a pull request. During the development process, you can provide feedback and Amazon Q Developer iterates on the suggested code, creating a collaborative development workflow.

Amazon Q Developer offers the following key capabilities in GitHub:

- Feature development label that automatically implements new features and bug fixes from idea to pull request

- Automated code reviews of new or reopened pull requests for code quality, issues, and security concerns
- Slash commands to provide alternative ways to initiate feature development from issues, and code reviews after initial automatic review
- Iterative development by providing feedback on generated code and implementing
- Browser extensions to quickly assign feature development tasks to Amazon Q Developer

To get started, see Quickstart: Installing, using features in GitHub, and increasing usage limits.

Project rules for Amazon Q Developer

Amazon Q Developer enables you to create and maintain project-specific rules in GitLab or GitHub, which define coding standards and best practices for your team (such as requiring type hints in Python code or Javadoc comments in Java code). These rules, stored as Markdown files in the *project-root*`/.amazonq/rules` directory, ensure consistency across all developers regardless of experience level, and are automatically incorporated into context for Amazon Q Developer when developers interact with Amazon Q Developer within your project, ensuring all generated responses adhere to your established guidelines. For more information, see Creating project rules for Amazon Q Developer in third-party platforms.

GitLab Duo with Amazon Q

GitLab Duo with Amazon Q provides a suite of artificial intelligence (AI) experiences, such as propose code implementation for your idea, review merge requests for quality and vulnerabilities, and suggest unit tests. Additionally, you can use the GitLab Duo Chat feature that supports Amazon Q to address developmental tasks, such as vulnerability explanation, troubleshoot failed pipelines, and code refactoring. It's available for the GitLab Self-Managed offering and the Ultimate tier subscription (GitLab Duo with Amazon Q subscription add-on). For more information, see the GitLab plans in the *GitLab documentation*.

After configuring GitLab Duo with Amazon Q, you can use quick actions in GitLab issues and merge request comments to trigger the AI capabilities. For more information, see GitLab Duo concepts and Getting started with GitLab Duo with Amazon Q. To learn about all features available on GitLab Duo with Amazon Q, see Additional supported features in the *GitLab documentation*.

Topics

- GitLab Duo concepts
- Getting started with GitLab Duo with Amazon Q
- Troubleshooting issues for GitLab Duo with Amazon Q

GitLab Duo concepts

Here are some concepts and terms to know when using GitLab Duo with Amazon Q.

Topics

- Setting up GitLab Duo with Amazon Q
- Onboarding with AWS resources and permission policies
- GitLab quick actions

Setting up GitLab Duo with Amazon Q

Before you can use Amazon Q artificial intelligence (AI) capabilities in GitLab Duo, you need to complete the prerequisites and create AWS resources. For more information, see Set up GitLab Duo with Amazon Q in the *GitLab documentation*.

Onboarding with AWS resources and permission policies

As part of the GitLab Duo onboarding process, you need to create an Amazon Q Developer profile through the Amazon Q Developer console. The profile allows you to create customization and control settings for all or a subset of users in your identity provider. After creating a profile, you need an OpenID Connect (OIDC) identity provider (IdP), as well as an IAM service role, to establish trust between GitLab Duo and your AWS account. To learn how to create the required resources and set up GitLab Duo with Amazon Q, see Set up GitLab Duo with Amazon Q in the *GitLab documentation*.

When the new IAM role is created, the required trust policy with the necessary permissions is also created. A role trust policy is a required resource-based policy that is attached to a role in IAM.

You need to add a permissions policy, which grants ability to connect with Amazon Q and utilize the features in the GitLab Duo with Amazon Q integration. The policy must be added when creating the IAM role. To learn more about the permissions provided by the permissions policy, see GitLabDuoWithAmazonQPermissionsPolicy.

Alternatively, you can create an inline policy and add the required permissions. You can choose to create an inline policy if you want to custom access control. For more information, see Managed policies and inline policies and Policies and permissions in AWS Identity and Access Management in the *IAM User Guide*.

Trust policy

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Principal": {
                "Federated": "arn:aws-eusc:iam::111122223333:oidc-provider/
auth.token.gitlab.com/cc/oidc/instance-id"
            },
            "Condition": {
                "StringEquals": {
                    "auth.token.gitlab.com/cc/oidc/instance-id:aud": "gitlab-
cc-instance-id"
                }
            }
        }
    ]
}
```

Permissions policy

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "GitLabDuoUsagePermissions",
      "Effect": "Allow",
      "Action": [
```

```
        "q:SendEvent",
        "q:CreateAuthGrant",
        "q:UpdateAuthGrant",
        "q:GenerateCodeRecommendations",
        "q:SendMessage",
        "q:ListPlugins",
        "q:VerifyOAuthAppConnection"
      ],
      "Resource": "*"
    },
    {
      "Sid": "GitLabDuoManagementPermissions",
      "Effect": "Allow",
      "Action": [
        "q:CreateOAuthAppConnection",
        "q:DeleteOAuthAppConnection"
      ],
      "Resource": "*"
    },
    {
      "Sid": "GitLabDuoPluginPermissions",
      "Effect": "Allow",
      "Action": [
        "q:CreatePlugin",
        "q:DeletePlugin",
        "q:GetPlugin"
      ],
      "Resource": "arn:aws-eusc:qdeveloper:*:*:plugin/GitLabDuoWithAmazonQ/*"
    }
  ]
}
```

Optionally, you can also use customer managed keys (CMK) to encrypt your resources if you want full control over the lifecycle and usage of your key. The `kms:ViaService` condition key to limit who can use CMK for encrypting and decrypting content. For more information, see Manage access to Amazon Q Developer for third-party integration.

GitLab quick actions

When invoked, quick actions perform tasks for you in GitLab issues and merge requests. To learn how to invoke quick actions in GitLab, see the GitLab documentation.

Merge request generation and iteration

- `/q dev` – Allows you to go from a high-level idea captured in a GitLab issue to having Amazon Q generate a ready-to-review merge request with the proposed code implementation. This helps streamline the process of turning concepts into working code. The merge request is created in a new branch and Amazon Q assigns the issue creator as a merge request reviewer. You're also provided a merge request summary. For more information, see Turn an idea into a merge request.

Code review

- `/q review` – Allows you to initiate a merge request review in GitLab Duo with Amazon Q. An automatic code review is initiated for new merge requests. As a GitLab administrator, you can also configure Amazon Q to turn off automatic reviews. Automated code reviews identify and fix potential issues as Amazon Q generates and suggests code fixes to your merge request. They provide quality checks, analyzing for issues, logical errors, anti-patterns, code duplication, and more.

  Amazon Q gives you code analysis with comments, with each comment providing a separate finding. This quick action is available for all languages. Automatic code reviews are initiated when you open new merge requests or reopen previously closed ones. However, automatic code reviews won't be triggered by subsequent commits made within an existing merge request. You can manually trigger a code review by using the `/q review` quick action.

  You can configure code reviews to run automatically on every new merge request within your GitLab instance or group. For more information, see Review a merge request.

Chat session in web UI and IDEs

- GitLab Duo Chat and Code Suggestions works with Amazon Q to provide support for CI/CD configuration, error explanations, and addressing questions. You can use slash commands in a chat session to invoke the GitLab Duo with Amazon Q chat capabilities. For more information, see Ask GitLab Duo Chat.

Getting started with GitLab Duo with Amazon Q

GitLab Duo with Amazon Q brings artificial intelligence (AI) capabilities directly into your software development operations and source code management workflows. You can get started with GitLab Duo with Amazon Q using a self-managed GitLab instance and an GitLab Ultimate subscription that's synchronized with GitLab. You need to create an Amazon Q Developer profile, add a connection with an OpenID Connect (OIDC) identity provider, and create an IAM role to access Amazon Q from GitLab. For more information, see Onboarding with AWS resources and permission policies. To learn how to create the required resources and set up GitLab Duo with Amazon Q, see Set up GitLab Duo with Amazon Q in the *GitLab documentation*.

At any time, you can configure the availability of GitLab Duo with Amazon Q by turning it on or off for your instance, group, or project. For more information, see Turn off GitLab Duo with Amazon Q.

Once you set up GitLab Duo with Amazon Q, you can begin using the AI capabilities of Amazon Q in GitLab to review merge requests for quality and vulnerabilities, and suggest unit tests. You can also use the GitLab Duo Chat feature that supports Amazon Q to address developmental tasks, such as vulnerability explanation, troubleshoot failed pipelines, and code refactoring.

To learn more about how to invoke quick actions in GitLab issues and merge requests, see GitLab Duo with Amazon Q.

Troubleshooting issues for GitLab Duo with Amazon Q

Consult the following section to troubleshoot common onboarding problems when using GitLab Duo with Amazon Q.

GitLab instance blocks inbound requests

Problem: My GitLab instance blocks inbound requests, and Amazon Q can't call back into my GitLab instance.

Solution: Identify what's blocking the inbound request, and make modifications to accept inbound requests from Amazon Q, which could be in the form of one of the following:

- A proxy
- A firewall layer
- Denylist or allowlist at any infrastructure layer

You need to reonboard to your GitLab instance to resync. For more information, see Set up GitLab Duo with Amazon Q in the *GitLab documentation*.

Connection between Gitlab and Amazon Q out of sync

Problem: I modified GitLab Duo with Amazon Q application and now the connection between GitLab and Amazon Q is out of sync.

Solution: When GitLab and Amazon Q are out of sync, it can lead to invalid credentials, inability to refresh credentials, and unauthorized response from GitLab when Amazon Q calls into GitLab.

Instance domain name not resolvable

Problem: I modified the GitLab instance URL after onboarding, and now the connection between GitLab and Amazon Q are out of sync. Amazon Q ins't able to call the GitLab instance successfully anymore.

Solution: You need to ensure the domain name is resolvable. Reonboard to your GitLab instance. For more information, see Set up GitLab Duo with Amazon Q in the *GitLab documentation*.

Incorrect IAM role and identity provider (IdP)

Problem: My IAM role doesn't provide the correct permissions to the APIs required by GitLab Duo with Amazon Q.

Solution: Ensure the identity provider (IdP) and IAM roles are set up correctly. For more information, see Set up GitLab Duo with Amazon Q in the *GitLab documentation*.

Amazon Q Developer profile doesn't exist

Problem: I'm trying to onboard to GitLab Duo with Amazon Q, but coming across the following issue: `Application could not be craeted by the AI Gateway: Error 400 - {"detail":"An error occured (ValidationException) when calling the CreateOAuthAppConnection operation: ProfileDoesNotExist"}`

Solution: You need to first create an Amazon Q Developer profile through the Amazon Q Developer console. For more information, see Set up GitLab Duo with Amazon Q in the *GitLab documentation*.

Amazon Q Developer for GitHub (Preview)

> **Note**
>
> Amazon Q Developer for GitHub is in preview release and is subject to change.

Amazon Q Developer for GitHub or GitHub Enterprise Cloud allows you to leverage Amazon Q Developer capabilities for your software development workflows. With specialized development agents, you can implement new ideas, review code for quality issues, and address vulnerabilities with unit tests. Once the agent completes a task, you can provide feedback, and the agent iterates on the previous solution. For more information, see Amazon Q Developer agents.

You can access the Amazon Q Developer integration through GitHub and authorize it to provide access to your organization's repositories. To get started with Amazon Q Developer for GitHub, see Quickstart: Installing, using features in GitHub, and increasing usage limits.

> **Important**
>
> To install the Amazon Q Developer app and authorize access to GitHub repositories, you must meet the requirements for the GitHub organization. For more information, see Requirements to install a GitHub App and Roles in organization in the *GitHub documentation*.

> **Note**
>
> The Amazon Q Developer integration with GitHub processes data in the United States. For more information, see Cross-region processing in Amazon Q Developer.

> **Note**
>
> Amazon Q Developer for GitHub (Preview) does not currently use your content for service improvement. If we enable this in the future, we will provide you with adequate notice and a way for you to opt out of such use.

Topics

- Installing Amazon Q Developer app and authorizing access
- Amazon Q Developer agents
- Registering app installation
- Using browser extensions in GitHub
- Using slash commands in GitHub issues and pull requests
- Quickstart: Installing, using features in GitHub, and increasing usage limits
- Developing features and iterating with Amazon Q Developer in GitHub
- Reviewing code with Amazon Q Developer in GitHub
- Increasing usage limits and configuring details in Amazon Q Developer console
- Configuring registered installation details
- Troubleshooting issues for Amazon Q Developer for GitHub

Installing Amazon Q Developer app and authorizing access

As a GitHub organization administrator, you can install and configure the Amazon Q Developer app from GitHub for free without the need to set up an AWS account to get started. During the installation process, you choose to provide access to all or selected repositories in your GitHub organization. After installing and authorizing, you have access to free usage for the Amazon Q Developer features in GitHub. You can increase free usage by registering the app installation in the Amazon Q Developer console. For more information, see Quickstart: Installing, using features in GitHub, and increasing usage limits.

Important

To install the Amazon Q Developer app and authorize access to GitHub repositories, you must meet the requirements for the GitHub organization. For more information, see Requirements to install a GitHub App and Roles in organization in the *GitHub documentation*.

Note

If your GitHub enterprise organization has enabled IP allowlisting, you must accept the allowed IP addresses on the GitHub app. You can also choose to automatically add the IP

addresses to your allow list. For more information, see Allowing access by GitHub Apps and Enabling allowed IP addresses in the *GitHub documentation*.
The following IP addresses are used to access your GitHub resources:

```
34.228.181.128
44.219.176.187
54.226.244.221
```

Amazon Q Developer agents

Amazon Q Developer agents provide support across the software development lifecycle from coding, testing, and deploying to troubleshooting.

- **Amazon Q development agent** – After creating an issue and adding the feature development label, Amazon Q Developer automatically implements your new features and bug fixes. Amazon Q Developer creates a pull request with the changes and a summary of the changes. Instead of applying a label, you can also initiate feature development with the `/q dev` slash command in a comment of the issue. For more information, see Developing features and iterating with Amazon Q Developer in GitHub.
- **Amazon Q code review agent** – When a new pull request is created or a closed pull request is reopened, Amazon Q Developer automatically performs a code review and provides feedback on code quality, potential issues, and security concerns. Amazon Q Developer also generates fixes for the identified issues, which you can review and choose to commit to the pull request. The code review includes a code review summary with threaded findings. You can interact with Amazon Q Developer by using the `/q` command in pull request comments to ask questions about the code review findings.

  Automatic code reviews are not triggered by subsequent commits made within an existing pull request. You can initiate additional code reviews within pull requests with the `/q review` slash command. For more information, see Reviewing code with Amazon Q Developer in GitHub.

Important

The Amazon Q Developer app attempts to automatically create the **Amazon Q development agent** label in GitHub repositories you authorize access to. If the label is not automatically created, or if it's unintentionally deleted, you can manually create it

in GitHub. The label must be named as **Amazon Q development agent** in order for it to be recognized and processed as a Amazon Q Developer label. For more information, see Creating a label in the *GitHub documentation*.

Registering app installation

The Amazon Q Developer integration for GitHub is available for free without the need to set up an AWS account to get started. You're provided with limited invocations per month for feature development, as well as limited number of lines for code reviews per month. You can increase free usage by registering your Amazon Q Developer app installation with your AWS account. For more information, see Increasing usage limits and configuring details in Amazon Q Developer console.

Important

To register the app installation in the Amazon Q Developer console, you must meet the requirements for the GitHub organization. For more information, see Requirements to install a GitHub App and OAuth apps and organizations in the *GitHub documentation*.

Using browser extensions in GitHub

You can use the Amazon Q Developer extension in a supported browser to quickly add a label for feature development in GitHub issues without having to search through label menus.

The Amazon Q Developer extension is available for the following browsers:

- Google Chrome
- Mozilla Firefox
- Microsoft Edge

Using slash commands in GitHub issues and pull requests

You can use slash commands in comments within GitHub issues or pull requests to invoke Amazon Q Developer to perform development tasks or provide support.

- `/q dev` - Invokes Amazon Q Developer in a GitHub issue to automatically implement new features and bug fixes. Amazon Q Developer creates a pull request with the changes and a summary of the changes.
- `/q review` - Invokes Amazon Q Developer to automatically perform code reviews when pull requests are created or reopened. Code reviews provide feedback on code quality, potential issues, and security concerns, along with suggested fixes and code review summaries with threaded findings. Use `/q` in pull request comments to interact with findings. Automatic reviews are not triggered by subsequent commits to existing pull requests.
- `/q help` - Provides information about Amazon Q Developer for GitHub, including slash comannds, features, customization details, as well as a link to the Amazon Q Developer for GitHub (Preview) documentation in the *Amazon Q Developer Developer Guide*.

Quickstart: Installing, using features in GitHub, and increasing usage limits

> **Note**
>
> Amazon Q Developer for GitHub is in preview release and is subject to change.

This tutorial provides a walkthrough of the following tasks:

1. Install the Amazon Q Developer app from the GitHub Marketplace and provide access to your repositories.
2. Get started with Amazon Q Developer in an issue by adding a label for feature development, or by creating a new pull request for code review. Alternatively, you can use slash commands in issues to initiate feature development. You can also initiate additional code reviews within pull requests with a slash command.
3. (Optional) Register your Amazon Q Developer app installation with your AWS account to increase your usage limits.

Step 1: Install Amazon Q Developer in GitHub and authorize access

You can use Amazon Q Developer in GitHub free without the need to set up an AWS account to get started. The first step to using Amazon Q Developer in GitHub is to install the app from GitHub .

During this process, you can provide Amazon Q Developer access to all your GitHub repositories or selected repositories.

> **Important**
>
> To install the Amazon Q Developer app and authorize access to GitHub repositories, you must meet the requirements for the GitHub organization. For more information, see Requirements to install a GitHub App and Roles in organization in the *GitHub documentation*.

To install Amazon Q Developer and authorize access

1. Navigate to the Amazon Q Developer for GitHub app page.
2. If necessary, sign in to your GitHub account using your GitHub credentials.
3. Review the Amazon Q Developer app's overview and features, and then choose **Install**.
4. Do one of the following to configure access to your GitHub repositories:
   a. To provide access to all current and future repositories, choose **All repositories**.
   b. To provide access to specific repositories, choose **Only select repositories**, choose the **Select repositories** dropdown, and then choose a repository you want to allow to access to.
5. Choose **Install** to complete installing Amazon Q Developer in GitHub and authorizing it to access your repositories.

After installing the app in GitHub and authorizing access, you're redirected to the Amazon Q Developer overview page in GitHub. You can navigate to your GitHub repository to begin using the Amazon Q Developer features.

> **Note**
>
> If your GitHub enterprise organization has enabled IP allowlisting, you must accept the allowed IP addresses on the GitHub app. You can also choose to automatically add the IP addresses to your allow list. For more information, see Allowing access by GitHub Apps and Enabling allowed IP addresses in the *GitHub documentation*.
> The following IP addresses are used to access your GitHub resources:

```
34.228.181.128
44.219.176.187
54.226.244.221
```

Step 2: Using Amazon Q Developer features in GitHub

After installing the Amazon Q Developer app in GitHub and authorizing access to your repositories, you can begin using the Amazon Q Developer agents for support across the software development lifecycle from coding, testing, and deploying to troubleshooting applications. For more information, see one of the following:

> **Important**
>
> The Amazon Q Developer app attempts to automatically create the **Amazon Q development agent** label in GitHub repositories you authorize access to. If the label is not automatically created, or if it's unintentionally deleted, you can manually create it in GitHub. The label must be named as **Amazon Q development agent** in order for it to be recognized and processed as a Amazon Q Developer label. For more information, see Creating a label in the *GitHub documentation*.

- Developing features and iterating with Amazon Q Developer in GitHub
- Reviewing code with Amazon Q Developer in GitHub

Step 3: Increase free usage limits and configure details

You can use Amazon Q Developer agents in GitHub for free without the need to set up an AWS account to get started. You're provided with limited invocations per month for feature development and code review. You can increase your free usage at any time by registering your Amazon Q Developer app installation with your AWS account. Registering also provides with the ability to configure details such as disabling code reviews and adding tags for searching and filtering. For more information, see Increasing usage limits and configuring details in Amazon Q Developer console.

Important

To register the app installation in the Amazon Q Developer console, you must meet the requirements for the GitHub organization. For more information, see Requirements to install a GitHub App and OAuth apps and organizations in the *GitHub documentation*.

Developing features and iterating with Amazon Q Developer in GitHub

Note

Amazon Q Developer for GitHub is in preview release and is subject to change.

You can use Amazon Q Developer in GitHub to streamline development by automatically implementing new features and bug fixes, taking tasks from idea to a completed pull request. When you add the feature development label to an issue or use the `/q dev` slash command, Amazon Q Developer uses the issue, including its title and description, as well as repository code as context to generate new code fixes and create a pull request. On the pull request, you can provide feedback and Amazon Q Developer iterates on the suggested code.

You can have Amazon Q Developer perform feature development a limited number of times per month. You can increase your free usage at any time by registering your Amazon Q Developer app installation with your AWS account. For more information, see Increasing usage limits and configuring details in Amazon Q Developer console.

Important

The Amazon Q Developer app attempts to automatically create the **Amazon Q development agent** label in GitHub repositories you authorize access to. If the label is not automatically created, or if it's unintentionally deleted, you can manually create it in GitHub. The label must be named as **Amazon Q development agent** in order for it to be recognized and processed as a Amazon Q Developer label. For more information, see Creating a label in the *GitHub documentation*.

To use Amazon Q Developer for feature development

1. If necessary, sign in to your GitHub account using your GitHub credentials.
2. Navigate to your GitHub organization, and then navigate to the repository you want to implement new features with Amazon Q Developer.
3. Choose **Issues**, and then create a new issue or choose an existing issue. For more information, see Create an issue in the *GitHub documentation*.

   - For a new issue, in the **Add a title** text input field, enter a title that provides context to Amazon Q Developer for the feature development (example: "Create an image recognition app"). The issue description should also be included as it also provides context.

     For an existing issue, you can edit the issue title and description to provide context to Amazon Q Developer for the feature development. For more informaiton, see Editing an issue in the *GitHub documentation*.
4. When creating an issue or configuring an existing issue, you can apply the feature development Amazon Q Developer label or use the `/q dev` slash command. Do one of the following:

   - To apply the label to the issue, do one of the following:
     - Choose the **Assign to Amazon Q** dropdown menu provided as a browser extension, and then choose the **Amazon Q development agent** label.
     - In the sidebar menu, choose **Labels**, and then choose the **Amazon Q development agent** label.
   - To use the `/q dev` slash command in a comment:
     1. Within the issue, navigate to **Add a comment**, and in the comment text input field, enter `/q dev`.
     2. Choose **Comment**.
5. For a new issue, choose **Create issue** to finish creating the issue with the necessary details for Amazon Q Developer to develop features. If you configure an existing issue, ensure you save the changes. For more informaiton, see Editing an issue in the *GitHub documentation*.

   When Amazon Q Developer finishes generating code changes for the feature development, it comments on the issue and opens a pull request.

6. Navigate to the comment left by Amazon Q Developer (example: "`I finished the proposed code changes, and the pull request is ready for review: [PR link]`"), and then choose the pull request link.

   You can also navigate to the **Pull requests** tab, and then choose the pull request created by Amazon Q Developer.
7. Choose the **Files changed** tab to view the code changes.
8. If you're satisfied with the suggested code changes, you can merge the pull request. For more information, see Merge a pull request.

You can also review the pull request for the feature development and iterate on the suggested code changes by providing feedback to Amazon Q Developer.

To iterate on Amazon Q Developer feature development code

1. Choose the pull request created by Amazon Q Developer, and then choose the **Files changed** tab to view the code changes.
2. Optionally, for specific lines of code you want to provide feedback on, choose **+** to add a comment with feedback.

   In the conversation, you can use the `/q` command followed by your instructions in natural language (for example, `/q implement my suggestions` or `/q refactor this function for better performance`). Amazon Q Developer will respond with a comment describing the changes it will make based on your feedback (for example, "I will implement the following changes based on the feedback: ..."). When the implementation is complete, Amazon Q Developer will post another comment confirming the changes (for example, "I have implemented the suggested changes.") along with a link to the generated commit where you can view the changes.
3. Review the changes made by Amazon Q Developer by following the commit link provided in the conversation. You can continue to provide additional feedback using the `/q` command for further iterations as needed.
4. If you're satisfied with the updated code changes, you can merge the pull request or iterate on the code again with new feedback. For more information, see Merge a pull request.

Reviewing code with Amazon Q Developer in GitHub

> **Note**
>
> Amazon Q Developer for GitHub is in preview release and is subject to change.

Amazon Q Developer enables automated code reviews within GitHub. When you create a new pull request or reopen a closed pull request, Amazon Q Developer automatically performs a code review and provides feedback on code quality, potential issues, and high-severity findings. Each review includes a code review summary with threaded findings. Amazon Q Developer also generates fixes for the identified issues, which you can review and choose to commit to the pull request. You can use the `/q` command in pull request comments to ask questions and interact regarding the code review findings. Automatic code reviews are not triggered by subsequent commits made within an existing pull request.

> **Note**
>
> The Code reviews feature setting in the Amazon Q Developer console controls automated code reviews that run when pull requests are created or reopened. Initiating code reviews using the `/q review` slash command in pull request comments is not affected by this setting.

You can also initiate code reviews within pull requests with the `/q review` slash command. The slash command can be added to a new pull request comment, which initiates a new code review of the pull request in its current state, including any comments and new commits. For more information, see Initiating code reviews within GitHub pull requests.

You can have Amazon Q Developer perform a code review for a limited amount of lines per month. You can increase your free usage at any time by registering your Amazon Q Developer app installation with your AWS account. For more information, see Increasing usage limits and configuring details in Amazon Q Developer console.

> **Note**
>
> If the code review capability was previously disabled, it must be enabled in the Amazon Q Developer console before you can apply the label in GitHub. For more information, see Editing features for Amazon Q Developer in GitHub.

Prerequisites

Before you can initiate code reviews with Amazon Q Developer, you need the appropriate permissions for the target GitHub repository. The supported repository roles are Write, Maintain, or Admin. Users with Read or Triage roles, as well as members without a role, cannot initiate code reviews with Amazon Q Developer.

GitHub users with the Triage role can still review pull requests in a repository. Any user, regardless of role, can review pull requests in public repositories.

For more information, see Repository roles for organizations and About pre-defined organization roles in the *GitHub documentation*.

Initiating code reviews for GitHub pull requests

When you open a new pull request or reopen a previously closed one, Amazon Q Developer automatically runs a code review and delivers feedback on code quality, possible problems, and critical findings.

To use Amazon Q Developer for code reviews and apply fixes

Before you start a review, you can customize a code quality review by defining custom coding standards in simple Markdown files in the `project-root/.amazonq/rules` directory. Amazon Q automatically follows your guidelines, ensuring consistent code quality across your entire project. For more information, see Creating project rules for Amazon Q Developer in third-party platforms.

1. If necessary, sign in to your GitHub account using your GitHub credentials.
2. Navigate to your GitHub organization, and then navigate to the repository you want to perform a code review with Amazon Q Developer.
3. Create a new a pull request for changes made to your source code. For more information, see Creating a pull request in the *GitHub documentation*.

When you create a new pull request, Amazon Q Developer automatically begins a code review to find potential issues. Once Amazon Q Developer completes the review, it provides a code review summary. Each finding appears as a threaded comment under the summary, along with suggested fixes that you can commit to the pull request.

4. Ask the agent to implement changes and create commits directly on your pull request's source branch. You can do this by posting a comment starting with `/q` and followed by your request in natural language for the agent to make changes.
5. (Optional) Ask questions about specific findings. Within the pull request, navigate to **Add a comment**, and in the comment text input field, enter `/q` followed by your question (for example, "`/q explain the importance of this finding`").
6. Review the proposed code changes by Amazon Q Developer, choose **Commit suggestion**, and then choose **Commit changes** to update the pull request.
7. If you're satisfied with the suggested code fixes, you can merge the pull request to apply the code changes suggested by Amazon Q Developer. For more information, see Merging a pull request in the *GitHub documentation*.

Initiating code reviews within GitHub pull requests

After an automatic code review performed by Amazon Q Developer for a new or reopened GitHub pull request, you can initiate additional code reviews to iterate on your code using the `/q review` slash command. The code review is performed on the entire pull request's diff.

> **Note**
>
> You can only initiate a code review within a pull request with a new comment. The `/q review` slash command will not work in an existing comment thread. Initiating a code review using the `/q review` slash command is not affected by the Code reviews feature setting in the Amazon Q Developer console.

To use initiate code reviews in a pull request

1. If necessary, sign in to your GitHub account using your GitHub credentials.
2. Navigate to your GitHub organization, and then navigate to the pull request you want to perform a code review with Amazon Q Developer. For more information, see About pull requests.

3. Within the pull request, navigate to **Add a comment**, and in the comment text input field, enter `/q review`.
4. Choose **Comment** to initiate the code review.

   It can take a few minutes for Amazon Q Developer to complete analyzing the pull request code. After Amazon Q Developer finishes analyzing, it provides a code review summary. Each finding appears as a threaded comment under the summary, along with proposed changes you can choose to commit and update the pull request.
5. (Optional) Ask questions about specific findings. Within the pull request, navigate to **Add a comment**, and in the comment text input field, enter /q followed by your question (for example, "`/q explain the importance of this finding`").
6. If you're satisfied with the suggested code fixes, you can merge the pull request to apply the code changes suggested by Amazon Q Developer. For more information, see Merging a pull request in the *GitHub documentation*.

Increasing usage limits and configuring details in Amazon Q Developer console

> **Note**
>
> Amazon Q Developer for GitHub is in preview release and is subject to change.

You can use Amazon Q Developer agents in GitHub for free without the need to set up an AWS account to get started. You're provided with limited invocations per month for the feature development and code review capabilities. You can increase your free usage at any time by registering your Amazon Q Developer app installation with your AWS account.

By default, the **Code reviews** and **Feature development** features are enabled in GitHub when you install the app. You can disable any of these features when you register. Registering requires an Amazon Q Developer profile to manage the features from the console. The profile stores your settings and code recommendation customization.

> **Important**
>
> To register the app installation in the Amazon Q Developer console, you must meet the requirements for the GitHub organization. For more information, see Requirements to install a GitHub App and OAuth apps and organizations in the *GitHub documentation*.

To register your Amazon Q Developer installation

1. Navigate to the Amazon Q Developer console.
2. Choose **Enable Q Developer**, and follow the prompts to enable Kiro and Amazon Q Developer.

   If you previously enabled Kiro and Amazon Q Developer, skip to step 3.
3. In the navigation pane, choose **Amazon Q Developer in GitHub**.
4. Choose **Register installation**, and then choose **Authorize** to be directed to GitHub.

   If you previously authorized Amazon Q Developer to access your GitHub organization, you'll be redirected back to the Amazon Q Developer console. In this case, skip to step 7.
5. Sign in to your GitHub account using your GitHub credentials. If you have multiple accounts, choose the account where you want to provide access to Amazon Q Developer.
6. Choose **Authorize Amazon Q Developer** to provide access to your GitHub account. You'll be redirected back to Amazon Q Developer console after the authorization.
7. Under **Registration details**, enter your GitHub details, optionally configure the code reviews feature, and add tags.

   a. In the **Registration name** text input filed, enter a name for your app installation.

   b. (Optional) In the **Organization name - *optional*** text input field, enter a name for the organization associated with the app installation.

   c. (Optional) Under **Features**, configure the **Code reviews** feature by choosing the toggle to enable or disable the feature. **Feature development** configuration cannot be modified, and is enabled by default.

   d. (Optional) Under **Tags - optional**, choose **Add new tag** to add a tag that can help to search and filter your AWS resources in GitHub.
8. Choose **Register** to register your app installation in GitHub with your AWS account.

After successfully registering the app installation, you can view the registration details. You can still enable or disable the code reviews feature, as well as add tags at a later time. You can also delete the registration. For more information, see Configuring registered installation details.

Configuring registered installation details

> **Note**
>
> Amazon Q Developer for GitHub is in preview release and is subject to change.

After you create an Amazon Q Developer profile and register the app installation in GitHub, you can do the following from the Amazon Q Developer console:

- Enable or disable the use of code reviews. Feature development configuration cannot currently be modified.
- Find links to browser extensions that provide a way to add Amazon Q Developer feature labels in the GitHub issues.
- Add tags to search and filter your resources or track AWS costs.
- Delete a GitHub app installation registration.

Editing features for Amazon Q Developer in GitHub

The features available for Amazon Q Developer in GitHub are enabled by default when you install the app in GitHub and provide authorization to access your organization. You can choose to disable a feature if you no longer want to use the feature in GitHub.

To enable or disable a feature for Amazon Q Developer in GitHub

1. Navigate to the Amazon Q Developer console.
2. In the navigation pane, under **Amazon Q Developer in GitHub**, choose **Registered installations**.
3. Under the **Name** column, choose the registration name for the installation you want to enable or disable a feature.
4. Under **Features**, choose **Edit** to configure the feature availability.
5. In the modal, choose the toggle for the feature you want to enable or disable, and then choose **Save** to confirm the changes.

After enabling or disabling a feature in the Amazon Q Developer console, the changes are reflected in GitHub. Attempting to assign an issue to a Amazon Q Developer label after disabling the feature will lead to an error. Code reviews will no longer take place in new pull requests if the feature is disabled.

Installing browser extensions

You can install the Amazon Q Developer extension in one of the supported browsers. The extension enables you to quickly assign feature development tasks to Amazon Q Developer in GitHub issues without having to search through label menus.

The Amazon Q Developer extension is available for the following browsers:

- Google Chrome
- Mozilla Firefox
- Microsoft Edge

You can also view the supported browsers on the registration installation details page in the Amazon Q Developer console.

Deleting Amazon Q Developer GitHub app installation registration

You can delete a registration for one or more of your GitHub app installation through the Amazon Q Developer console. After permanently deleting your registration, all data associated with the registration is also deleted. The action can't be undone.

To delete your GitHub app installation

1. Navigate to the Amazon Q Developer console.
2. In the navigation pane, under **Amazon Q Developer in GitHub**, choose **Registered installations**.
3. Do one of the following:

   - Under the **Actions** column, choose **Delete registration** for the installation you want to delete.
   - Under the **Name** column, choose the registration name for the installation you want to delete, and the choose **Delete**.

Under the **Actions** column, choose **Delete registration** for registered installation you want to delete.

4. In the modal, review the details for deleting registration.
5. In the text input field, enter **`confirm`**, and then choose **Delete** to confirm the changes.

Once you delete a GitHub app installation, you can choose to register a new installation.

Troubleshooting issues for Amazon Q Developer for GitHub

Consult the following section to troubleshoot common problems when using Amazon Q Developer for GitHub.

Amazon Q Developer not generating pull requests in repositories with branch protection rules

Problem: Amazon Q Developer isn't able to create a pull request in my GitHub repository, which has branch protection rules.

Solution: The branch protection rules prevents Amazon Q Developer from creating a branch to create a pull request. In order to use Amazon Q Developer for GitHub in repositories with branch protection rules, you need to add the Amazon Q Developer app to your allow list.

To add the Amazon Q Developer app to your allow list

1. If necessary, sign in to your GitHub account using your GitHub credentials.
2. Navigate to your GitHub organization, and then navigate to the repository you want to allow list the Amazon Q Developer for GitHub app.
3. Choose **Settings**, and then choose **Branches** from the navigation pane.
4. Under **Branch protection rules**, choose **Edit** to modify the branch protection rules.
5. Choose **Restrict pushes that create matching branches**, and search for the Amazon Q Developer for GitHub app.
6. Choose **Save changes** to allow Amazon Q Developer to create pull requests for issues with Amazon Q Developer labels.

To learn more about modifying branch protection rules in GitHub, see Creating a branch protection rule.

Amazon Q Developer labels missing in GitHub issues

Problem: I don't see the **Amazon Q development agent** label in GitHub issues.

Solution: If the label isn't automatically created when you installed the Amazon Q Developer for GitHub app, or it was unintentionally deleted, you can manually create it in GitHub. The label must be named as **Amazon Q development agent** in order for it to be recognized and processed as a Amazon Q Developer label. For more information, see Creating a label in the *GitHub documentation*.

Amazon Q Developer not creating code for GitHub issues

Problem: I created a GitHub issue and invoked Amazon Q Developer to perform a task, but I got the following series of messages regarding technical difficulties:

1. # I'm working on generating code for this issue. I'll update this issue with a comment and open a pull request when I'm done.
2. ## I'm having technical difficulties. I couldn't solve the issue. I'm going to try again. This might take some time.
3. ## I'm having technical difficulties. I couldn't solve the issue. I'm going to try again. This might take some time.
4. # I'm having technical difficulties. I couldn't solve the issue.

   Consider reassigning this issue to a user. This will help you stay within the quotas for generative AI feature usage.

Solution: If Amazon Q Developer is not able to process your issue and generate code for it, create a new issue and apply the **Amazon Q development agent** label to the new issue. To learn more about creating an issue and applying an Amazon Q Developer agent label, see Developing features and iterating with Amazon Q Developer in GitHub.

Creating project rules for Amazon Q Developer in third-party platforms

You can build a library of project rules that you can use with Amazon Q Developer in GitLab or GitHub. These rules describe coding standards and best practices across your team. For example, you could have a rule that states that all Python code must use type hints, or that all Java code must use Javadoc comments. By storing these rules in your project, you can ensure consistency across developers, regardless of their experience level.

Project rules are defined in Markdown files in the project's *project-root*`/.amazonq/rules` folder.

Once you've created your project rules, Amazon Q Developer will automatically use them as context within your project, and will make sure to adhere to them when generating code for feature development.

To create a project rule using the file system

1. In your third-party repository, open your project's root folder.
2. In the project root folder, create the following folder:

   project-root`/.amazonq/rules`

   This folder holds all your project rules.
3. In *project-root*`/.amazonq/rules`, create a project rule file. It must be a Markdown file. For example:

   `cdk-rules.md`
4. Open your project rule Markdown file.
5. Add a detailed prompt to the file. For example:

   ```
   All Amazon S3 buckets must have encryption enabled, enforce SSL, and block public
    access.
   All Amazon DynamoDB Streams tables must have encryption enabled.
   All Amazon SNS topics must have encryption enabled and enforce SSL.
   All Amazon SNS queues must enforce SSL.
   ```
6. Commit, review, and merge your changes.
7. (Optional) Add more project rule Markdown files.

You have now created one or more project rules. Amazon Q will use these rules as context automatically within your project.

Chatting with Amazon Q Developer in chat applications

You can chat with Amazon Q Developer in Microsoft Teams and Slack chat applications. In configured channels, Amazon Q can answer questions about best practices for building solutions, troubleshooting issues, and identifying next steps. The following Amazon Q chat features are available in configured chat applications:

- Chatting about AWS
- Chatting about your resources with Amazon Q Developer
- Troubleshooting resource issues
- Chatting about your costs
- Chatting about your telemetry and operations
- Amazon Q network troubleshooting for Reachability Analyzer

For more information about the complete set of features available when you use Amazon Q in chat applications, see What is Amazon Q Developer in chat applications? in the *Amazon Q Developer in chat applications Administrator Guide*.

> **Note**
>
> When you use Amazon Q Developer in chat applications, access is limited to the Amazon Q Developer Free tier.

Enable Amazon Q chat in your channels

To add chat capabilities to a Microsoft Teams or Slack channel that is already configured with Amazon Q Developer, complete the following steps. To set up Amazon Q Developer in your chat applications for the first time and allow users to chat with Amazon Q, see Get started with Microsoft Teams and Get started with Slack in the *Amazon Q Developer in chat applications Administrator Guide*.

Before you can ask Amazon Q questions from a Microsoft Teams or Slack channel, you need to add Amazon Q to the channel. First, update your AWS Identity and Access Management (IAM) role settings to include the AmazonQDeveloperAccess managed policy, and then add the policy as a channel guardrail. If you need administrator access, add the AmazonQFullAccess policy instead.

1. Add the `AmazonQDeveloperAccess` managed policy to your IAM role:
    a. Sign in to the AWS Management Console and open the IAM console.
    b. In the navigation pane of the IAM console, choose **Roles**.
    c. Choose the name of the role that you want to modify.
    d. In **Permissions policies**, choose **Add permissions** and **Attach policies**.
    e. Enter `AmazonQDeveloperAccess` in the search.
    f. Select **AmazonQDeveloperAccess**.
    g. Choose **Add permissions**.
2. Add the `AmazonQDeveloperAccess` managed policy to your channel guardrails:
    a. Open the Amazon Q Developer in chat applications console.
    b. Choose a configured client.
    c. Select a configured channel.
    d. Choose **Set guardrails**.
    e. Enter `AmazonQDeveloperAccess` in the search.
    f. Select **AmazonQDeveloperAccess**.
    g. Choose **Save**.

Ask Amazon Q questions in your channel

To check that your configuration was successful, ask Amazon Q a question. Enter `@Amazon Q` followed by your question.

Following are some examples of questions that you can ask Amazon Q from your configured channel:

- `@Amazon Q how do I troubleshoot lambda concurrency issues?`
- `@Amazon Q what are the best practices for securing S3 buckets?`
- `@Amazon Q what is the maximum zipped file size for a lambda?`
- `@Amazon Q get the configuration for my lambda function` *`name`*`?`
- `@Amazon Q what is the size of the auto scaling group` *`name`* `in us-east-2?`
- `@Amazon Q can you show ec2 instances running in us-east-1?`

Security in Amazon Q Developer

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that is built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The shared responsibility model describes this as security of the cloud and security in the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. Third-party auditors regularly test and verify the effectiveness of our security as part of the AWS Compliance Programs. To learn about the compliance programs that apply to Amazon Q, see AWS Services in Scope by Compliance Program.
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your company's requirements, and applicable laws and regulations

This documentation helps you understand how to apply the shared responsibility model when using Amazon Q Developer. It shows you how to configure Amazon Q to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your Amazon Q resources.

Topics

- Data protection in Amazon Q Developer
- Identity and access management for Amazon Q Developer
- Compliance validation for Amazon Q Developer
- Resilience in Amazon Q Developer
- Infrastructure security in Amazon Q Developer
- Configuring a firewall, proxy server, or data perimeter for Amazon Q Developer
- Amazon Q Developer and interface endpoints (AWS PrivateLink)

Data protection in Amazon Q Developer

The AWS shared responsibility model applies to data protection in Amazon Q Developer. As described in this model, AWS is responsible for protecting the global infrastructure that runs all of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. You are also responsible for the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the Data Privacy FAQ. For information about data protection in Europe, see the AWS Shared Responsibility Model and GDPR blog post on the *AWS Security Blog*.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual users with AWS Identity and Access Management (IAM). That way each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We recommend TLS 1.2 or later.
- Set up API and user activity logging with AWS CloudTrail.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing sensitive data that is stored in Amazon S3.
- If you require FIPS 140-2 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-2.

We strongly recommend that you never put confidential or sensitive information, such as your customers' email addresses, into tags or free-form text fields such as a **Name** field. This includes when you work with Amazon Q or other AWS services using the AWS Management Console, API, AWS Command Line Interface (AWS CLI), or AWS SDKs. Any data that you enter into tags or free-form text fields used for names may be used for billing or diagnostic logs. For more information about how Amazon Q Developer uses content, see Amazon Q Developer service improvement.

Topics

- Data storage in Amazon Q Developer
- Data encryption in Amazon Q Developer
- Amazon Q Developer service improvement

- Opt out of data sharing in the IDE and command line
- Cross-region processing in Amazon Q Developer

Data storage in Amazon Q Developer

Amazon Q stores your questions, its responses, and additional context, such as console metadata and code, to generate responses to your questions and requests. For information about how data is encrypted, see Data encryption in Amazon Q Developer. For information about how AWS may use some questions that you ask Amazon Q and its responses to improve our services, see Amazon Q Developer service improvement.

AWS Regions where content is processed and stored

If you're an IAM Identity Center workforce user, at the Amazon Q Developer Pro tier, your content is stored in the AWS Region where your Amazon Q Developer profile was created only for the following features:

- Amazon Q chat in the AWS Management Console
- Diagnosing AWS console errors with Amazon Q
- Amazon Q in Eclipse, JetBrains IDEs, Visual Studio Code, and Visual Studio
- Amazon Q on the command line

When you use any other feature at the Amazon Q Developer Pro tier, your content may be stored and processed in a US Region. If you are using a Q Developer profile in a non-US Region, you can create a service control policy (SCP) to block access to features that store content and perform inference in the US. For an example SCP, see Manage access to Amazon Q Developer with policies.

For other Amazon Q features and integrations, and when using the Amazon Q Developer Free tier, your content is stored in a Region in the US. Data processed during diagnosing console error sessions is stored in the US West (Oregon) Region. All other data is stored in the US East (N. Virginia) Region. Note the following features that store data differently.

> **Note**
>
> When you use Amazon Q artifacts, your visualizations-related content is stored in a US Region.

When you use Console to Code with Amazon Q, content is stored in your console Region and processed in a US Region.
When you use Amazon Q generative SQL in Amazon Redshift, your content is stored and processed in your console Region. For more information, see Interacting with Amazon Q generative SQL in the *Amazon Redshift Management Guide*.
When you create an investigation with Amazon CloudWatch investigations, your content may be stored and processed in other Regions. For more information, see the Security in CloudWatch investigations topic in the *Amazon CloudWatch User Guide*.

With cross-region inferencing, your requests to Amazon Q Developer may be processed in a different Region within the geography where your content is stored. For more information, see Cross-region inference.

Data encryption in Amazon Q Developer

This topic provides information specific to Amazon Q Developer about encryption in transit and encryption at rest.

Encryption in transit

All communication between customers and Amazon Q and between Amazon Q and its downstream dependencies is protected using TLS 1.2 or higher connections.

Encryption at rest

Amazon Q stores data at rest using Amazon DynamoDB and Amazon Simple Storage Service (Amazon S3). The data at rest is encrypted using AWS encryption solutions by default. Amazon Q encrypts your data using AWS owned encryption keys from AWS Key Management Service (AWS KMS). You don't have to take any action to protect the AWS managed keys that encrypt your data. For more information, see AWS owned keys in the *AWS Key Management Service Developer Guide*.

For IAM Identity Center workforce users subscribed to Amazon Q Developer Pro, administrators can set up encryption with customer managed KMS keys for data at rest for the following features:

- Chat in the AWS console
- Diagnosing AWS console errors
- Customizations

- Agents in the IDE

You can only encrypt data with a customer managed key for the listed features of Amazon Q in the AWS console and the IDE. Your conversations with Amazon Q on the AWS website, AWS Documentation pages, and in chat applications are only encrypted with AWS-owned keys.

Customer managed keys are KMS keys in your AWS account that you create, own, and manage to directly control access to your data by controlling access to the KMS key. Only symmetric keys are supported. For information on creating your own KMS key, see Creating keys in the *AWS Key Management Service Developer Guide*.

When you use a customer managed key, Amazon Q Developer makes use of KMS grants, allowing authorized users, roles, or applications to use a KMS key. When an Amazon Q Developer administrator chooses to use a customer managed key for encryption during configuration, a grant is created for them. This grant is what allows the end user to use the encryption key for data encryption at rest. For more information on grants, see Grants in AWS KMS.

If you change the KMS key used to encrypt chats with Amazon Q in the AWS console, you must start a new conversation to begin using the new key to encrypt your data. Any conversations that were encrypted with the previous key won't be retained, and only future conversations will be encrypted with the updated key. If you want to maintain your conversations from a previous encryption method, you can revert to the key you were using during those conversations. If you change the KMS key used to encrypt diagnosing console error sessions, you must start a new diagnose session to being using the new key to encrypt your data.

Using customer managed KMS keys

After creating a customer managed KMS key, an Amazon Q Developer administrator must provide the key in the Amazon Q Developer console to use it to encrypt data. For information on adding the key in the Amazon Q Developer console, see Managing the encryption method in Amazon Q Developer.

To set up a customer managed key to encrypt data in Amazon Q Developer, administrators need permissions to use AWS KMS. The required KMS permissions are included in the example IAM policy, Allow administrators to use the Amazon Q Developer console.

To use features that are encrypted with a customer managed key, users need permissions to allow Amazon Q to access the customer managed key. For a policy that grants the needed permissions, see Allow Amazon Q access to customer managed keys.

If you see an error related to KMS grants while using Amazon Q Developer, you likely need to update your permissions to allow Amazon Q to create grants. To automatically configure the needed permissions, go to the Amazon Q Developer console and choose **Update permissions** in the permissions banner.

Amazon Q Developer service improvement

To help Amazon Q Developer provide the most relevant information, we may use certain content from Amazon Q, such as questions that you ask Amazon Q and its responses, for service improvement. This page explains what content we use and how to opt out.

Amazon Q Developer Free tier content used for service improvement

We may use certain content from Amazon Q Developer Free tier for service improvement. Amazon Q may use this content, for example, to provide better responses to common questions, fix Amazon Q operational issues, for de-bugging, or for model training.

Content that AWS may use for service improvement includes, for example, your questions to Amazon Q and the responses and code that Amazon Q generates.

We do not use content from Amazon Q Developer Pro or Amazon Q Business for service improvement.

> **Note**
>
> Amazon Q Developer for GitHub (Preview) does not currently use your content for service improvement. If we enable this in the future, we will provide you with adequate notice and a way for you to opt out of such use.

How to opt out

The way you opt out of Amazon Q Developer Free Tier using content for service improvement depends on the environment where you use Amazon Q.

For the AWS Management Console, AWS Console Mobile Application, AWS websites, and in chat applications, configure an AI services opt-out policy in AWS Organizations. For more information, see AI services opt-out policies in the *AWS Organizations User Guide*.

In the IDE, for Amazon Q Developer Free Tier, adjust your settings in the IDE. For more information, see Opt out of data sharing in the IDE and command line.

Opt out of data sharing in the IDE and command line

This page explains how to opt out of sharing your data in the IDE or command line where you use Amazon Q, including third-party IDEs and AWS coding environments. For information on how Amazon Q uses this data, see Amazon Q Developer service improvement.

Opting out of sharing your client-side telemetry

Your client-side telemetry quantifies your usage of the service. For example, AWS may track whether you accept or reject a recommendation. Your client-side telemetry does not contain actual code.

Telemetry collected in IDEs

To learn more about the telemetry data collected by Amazon Q in the IDE, see the commonDefinitions.json document in the `aws-toolkit-common` Github repository.

For detailed information about the telemetry data collected by each IDE where you use Amazon Q, reference the resource documents in the following GitHub repositories:

- Amazon Q extension for VS Code
- Amazon Q plugin for JetBrains
- Amazon Q plugin for Eclipse
- AWS Visual Studio Toolkit with Amazon Q

Telemetry collected in the Q CLI

To learn more about the telemetry data collected by the Q CLI, see the telemetry_definitions.json document in the `amazon-q-developer-cli` Github repository.

Telemetry collected in the command line tool for transformations

Telemetry collection helps AWS understand how the Q command line transformation tool is performing, learn how features are used, and improve our services. For transformations on the command line, we collect telemetry on your tool version and Maven plugin version.

> **Note**
>
> Don't add personally identifiable information (PII) or other confidential or sensitive information in free text fields.

Choose your IDE for instructions on opting out of sharing your client-side telemetry.

Visual Studio Code

To opt out of sharing your telemetry data in VS Code, use this procedure:

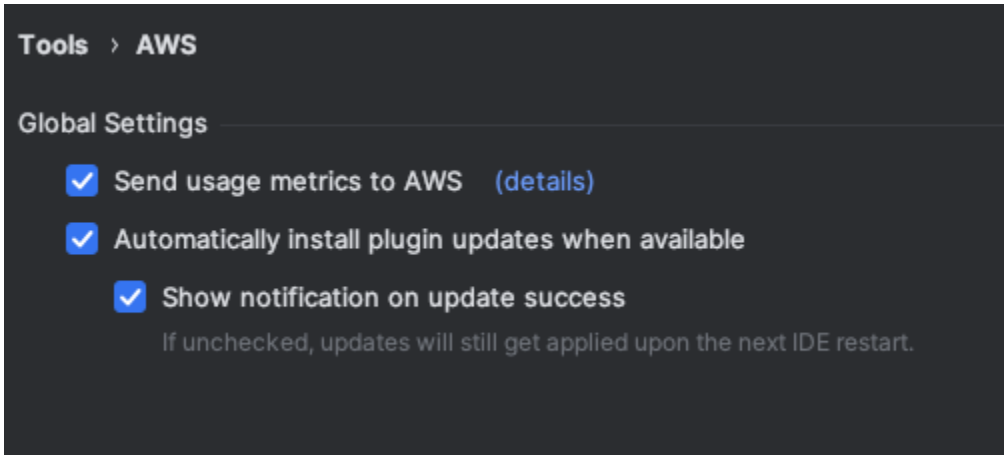
1. Open **Settings** in VS Code.
2. If you are using VS Code workspaces, switch to the **Workspace** sub-tab. In VS Code, workspace settings override user settings.
3. In the Settings search bar, enter `Amazon Q: Telemetry`.
4. Deselect the box.

> **Note**
>
> This is a decision for each developer to make inside their own IDE. If you are using Amazon Q as part of an enterprise, your administrator will not be able to change this setting for you.

JetBrains

To opt out of sharing your telemetry data in JetBrains, use this procedure:

1. In your JetBrains IDE, open **Preferences** (on a Mac, this will be under **Settings**).
2. In the left navigation bar, choose **Tools**, and then choose **AWS**.
3. Deselect **Send usage metrics to AWS**.



> **Note**
>
> This is a decision for each developer to make inside their own IDE. If you are using Amazon Q as part of an enterprise, your administrator will not be able to change this setting for you.

Eclipse

To opt out of sharing your telemetry data in Eclipse IDEs, use this procedure:

1. Open **Settings** in your Eclipse IDE.
2. Choose **Amazon Q** from the left navigation bar.
3. Deselect the box next to **Send usage metrics to AWS**.
4. Choose **Apply** to save your changes.

> **Note**
>
> This is a decision for each developer to make inside their own IDE. If you are using Amazon Q as part of an enterprise, your administrator will not be able to change this setting for you.

Visual Studio

To opt out of sharing your telemetry data in the AWS Toolkit for Visual Studio, use this procedure:

1. Under **Tools**, choose **Options**.
2. In the **Options** pane, choose **AWS Toolkit**, and then choose **General**.
3. Deselect **Allow AWS Toolkit to collect usage information**.

> **Note**
>
> This is a decision for each developer to make inside their own IDE. If you are using Amazon Q as part of an enterprise, your administrator will not be able to change this setting for you.

AWS Cloud9

1. From inside your AWS Cloud9 IDE, choose the Cloud9 logo in the menu bar, then choose **Preferences**.
2. On the **Preferences** tab choose **AWS Toolkit**.
3. Next to **AWS: client-side telemetry**, toggle the switch to the off position.

> **Note**
>
> This setting affects whether or not you share your AWS Cloud9 client-side telemetry in general, not just for Amazon Q.

Lambda

When you use Amazon Q with Lambda, Amazon Q does not share your client-side telemetry with AWS.

SageMaker AI Studio

1. From the top of the SageMaker AI Studio window choose **Settings**.
2. From the **Settings** dropdown, choose **Advanced Settings Editor**.
3. In the Amazon Q dropdown, select or deselect the box next to **Share usage data with Amazon Q**.

JupyterLab

1. From the top of the JupyterLab window choose **Settings**.
2. From the **Settings** dropdown, choose **Advanced Settings Editor**.
3. In the Amazon Q dropdown, select or deselect the box next to **Share usage data with Amazon Q**.

AWS Glue Studio Notebook

1. From the bottom of the AWS Glue Studio Notebook window choose **Amazon Q**.
2. From the pop-up menu, toggle the switch next to **Share telemetry with AWS**.

> **Note**
>
> Pausing the sharing of client-side telemetry will be valid only for the duration of the current AWS Glue Studio Notebook.

Command line

In the command line tool, under **Preferences**, toggle **Telemetry**.

Transformations on the command line

Telemetry collection is enabled by default with the command line tool for transformations. To disable it, complete the following procedure.

To update telemetry preferences

1. Run `qct configure` and provide the requested configuration details, or press enter to use the existing configuration.
2. When prompted whether you want to allow telemetry collection, enter `N` to prevent AWS from collecting telemetry data.
3. If you'd like to re-enable telemetry collection, run `qct configure` again and enter `Y` when prompted.

Opting out of sharing your content

For information on content AWS uses, see Amazon Q Developer service improvement.

Visual Studio Code

At the Amazon Q Developer Pro Tier, Amazon Q does not collect your content.

At the Amazon Q Developer Free Tier, to opt out of sharing your content in VS Code, use the following procedure.

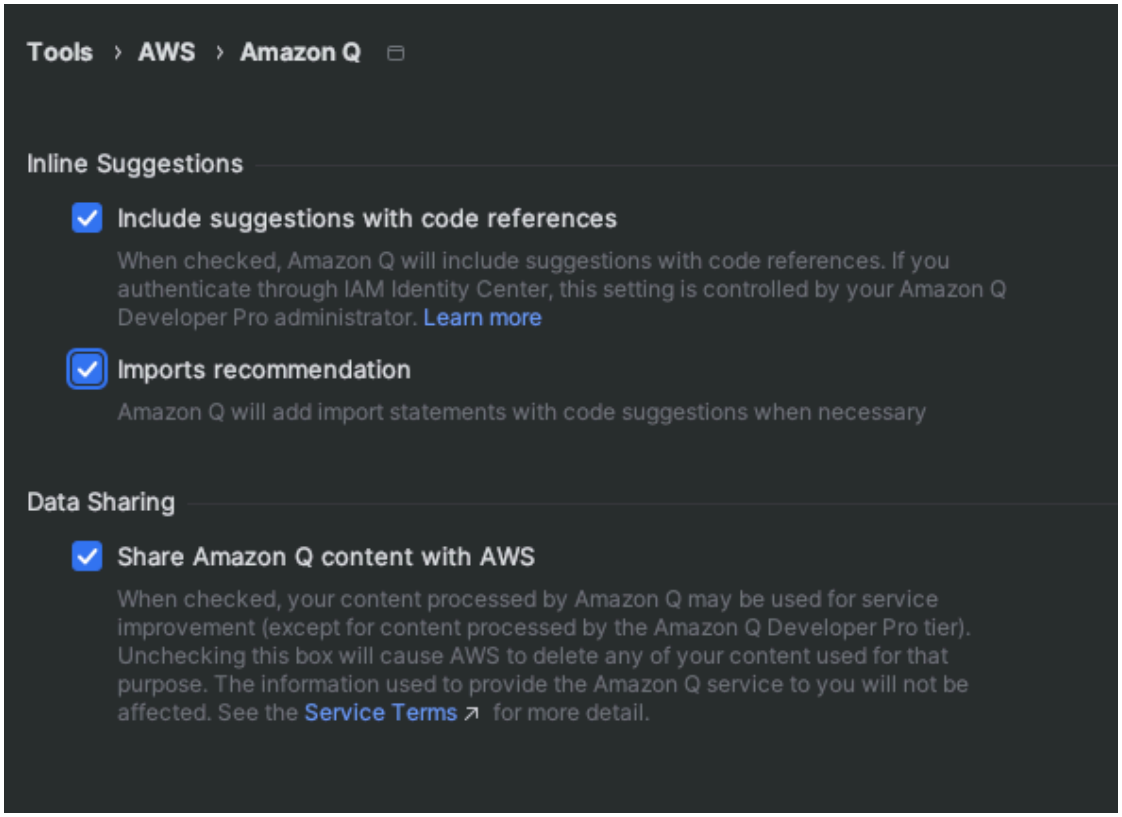
1. Open **Settings** in VS Code.
2. If you are using VS Code workspaces, switch to the **Workspace** sub-tab. In VS Code, workspace settings override user settings.
3. In the Settings search bar, enter `Amazon Q: Share Content`.
4. Deselect the box.

JetBrains

At the Amazon Q Developer Pro Tier, Amazon Q does not collect your content.

At the Amazon Q Developer Free Tier, to opt out of sharing Amazon Q data in JetBrains, use the following procedure.

1. Make sure you are using the latest version of JetBrains.
2. In your JetBrains IDE, open **Preferences** (on a Mac, this will be under **Settings**).
3. In the left navigation bar, choose **Tools** --> **AWS** --> **Amazon Q**.
4. Under **Data sharing**, deselect **Share Amazon Q content with AWS**.



Eclipse

At the Amazon Q Developer Pro tier, Amazon Q does not collect your content.

At the Amazon Q Developer Free tier, to opt out of sharing Amazon Q data in Eclipse IDEs, use the following procedure.

1. Make sure you are using the latest version of your Eclipse IDE.
2. In your Eclipse IDE, open **Settings**.
3. In the left navigation bar, choose **Amazon Q**.
4. Deselect the box next to **Share Amazon Q content with AWS**.
5. Choose **Apply** to save your changes.

Visual Studio

At the Amazon Q Developer Pro Tier, Amazon Q does not collect your content.

At the Amazon Q Developer Free Tier, to opt out of sharing your content in Visual Studio, use the following procedure.

Go to **Tools** -> **Options** -> **AWS Toolkit** -> **Amazon Q**

Toggle **Share Amazon Q Content with AWS** to **True** or **False**.

AWS Cloud9

When you use Amazon Q with AWS Cloud9, Amazon Q does not share your content with AWS.

> **Note**
>
> The AWS Cloud9 settings do contain a toggle switch for sharing Amazon Q content with AWS, but that switch is non-functional.

Lambda

When you use Amazon Q with Lambda, Amazon Q does not share your content with AWS.

> **Note**
>
> The Lambda settings do contain a toggle switch for sharing Amazon Q content with AWS, but that switch is non-functional.

SageMaker AI Studio

When you use Amazon Q with SageMaker AI Studio, Amazon Q does not share your content with AWS.

JupyterLab

1. From the top of the JupyterLab window choose **Settings**.
2. From the **Settings** dropdown, choose **Advanced Settings Editor**.
3. In the Amazon Q dropdown, select or deselect the box next to **Share content with Amazon Q**.

AWS Glue Studio Notebook

When you use Amazon Q with AWS Glue Studio Notebook, Amazon Q does not share your content with AWS.

Command line

In the command line tool, under **Preferences**, toggle **Share Amazon Q content with AWS**.

Transformations on the command line

When you use the Amazon Q command line tool for transformation, Amazon Q does not share your content with AWS.

Cross-region processing in Amazon Q Developer

The following sections describe how cross-region inference and cross-region calls are used to provide the Amazon Q Developer service.

Cross-region inference

Amazon Q Developer is powered by Amazon Bedrock, and uses cross-region inference to distribute traffic across different AWS Regions to enhance large language model (LLM) inference performance and reliability. With cross-region inference, you get:

- Increased throughput and resilience during high demand periods
- Improved performance
- Access to newly launched Amazon Q Developer capabilities and features that rely on the most powerful LLMs hosted on Amazon Bedrock

Cross-region inference requests are kept within the AWS Regions that are part of the geography where the data originally resides. For example, a request made from a Amazon Q Developer profile created in the US is kept within the AWS Regions in the US. Some Amazon Q Developer features and integrations may perform inference in Regions other than where your Q Developer profile was created. For more information, see Supported regions for Amazon Q Developer cross-region inference.

Although cross-region inferencing doesn't change where your data is stored, your requests and output results may move outside of the Region where the data originally resides. All data is encrypted while transmitted across Amazon's secure network. There's no additional cost for using cross-region inference.

Cross region inference doesn't affect where your data is stored. For information on where data is stored when you use Amazon Q Developer, see Data protection in Amazon Q Developer.

Supported regions for Amazon Q Developer cross-region inference

The following table describes what Regions your requests may be routed to depending on the geography where the request originated.

Supported Amazon Q Developer geography	Inference regions
United States	US East (N. Virginia) (us-east-1) US West (Oregon) (us-west-2) US East (Ohio) (us-east-2)
Europe	Europe (Frankfurt) (eu-central-1) Europe (Ireland) (eu-west-1) Europe (Paris) (eu-west-3) Europe (Stockholm) (eu-north-1)
Asia Pacific*	Asia Pacific (Mumbai) (ap-south-1) Asia Pacific (Seoul) (ap-northeast-2) Asia Pacific (Singapore) (ap-southeast-1) Asia Pacific (Sydney) (ap-southeast-2) Asia Pacific (Tokyo) (ap-northeast-1)

*Cross-region inferencing in the Asia Pacific Regions is only supported when you use Amazon Q generative SQL in the Asia Pacific (Seoul) Region.

For a complete list of Regions where you can use Amazon Q Developer, see Supported Regions for Amazon Q Developer.

Cross-region calls

Certain requests that you make to Amazon Q Developer may require cross-region calls. Cross-region calls are API calls made by Amazon Q from one AWS Region to another AWS Region.

Amazon Q makes cross-region calls when your request requires it to retrieve information from a Region different from your current Region. For example, when you ask Amazon Q questions about your AWS resources that are located in different Regions, it will make a cross-region call to access your resources and retrieve the relevant data to respond to your question. In addition, if a response from Amazon Q requires information from a global AWS service endpoint, Amazon Q may make calls outside of the Region where your data is stored. For more information on global services, see Global services in the *AWS Fault Isolation Boundaries AWS Whitepaper*.

If you'd like to disable cross-region calls made by Amazon Q Developer, you can create a policy that prevents Amazon Q from making API calls on your behalf. By doing so, you won't have access to features that require Amazon Q to make API calls on your behalf, even if Amazon Q is making calls within your current Region. For an IAM policy that prevents Amazon Q from making API calls on your behalf, including cross-region calls, see Deny Amazon Q permission to perform actions on your behalf.

Identity and access management for Amazon Q Developer

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be *authenticated* (signed in) and *authorized* (have permissions) to use Amazon Q Developer resources. IAM is an AWS service that you can use with no additional charge.

Topics

- Audience
- Authenticating with identities
- Managing access using policies
- How Amazon Q Developer works with IAM
- Manage access to Amazon Q Developer with policies
- Manage access to Amazon Q Developer for third-party integration
- Amazon Q Developer permissions reference
- AWS managed policies for Amazon Q Developer
- Using service-linked roles for Amazon Q Developer and User Subscriptions

Audience

How you use IAM differs, depending on the work you do in Amazon Q.

Service user – If you use the Amazon Q service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more Amazon Q features to do your work, you might need additional permissions. Understanding how access is managed can help you request the right permissions from your administrator.

Service administrator – If you're in charge of Amazon Q resources at your company, you probably have full access to Amazon Q. It's your job to determine which Amazon Q features and resources your service users should access. You must then submit requests to your IAM administrator to change the permissions of your service users. Review the information on this page to understand the basic concepts of IAM. To learn more about how your company can use IAM with Amazon Q, see How Amazon Q works with IAM.

IAM administrator – If you're an IAM administrator, you might want to learn details about how you can write policies to manage access to Amazon Q. If you're an IAM administrator, consider learning the details about how you can write policies to manage IAM user access to services. For information that's specific to Amazon Q, see AWS Regions managed policies for Amazon Q.

Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. You must be *authenticated* (signed in to AWS) as the AWS account root user, an IAM user, or by assuming an IAM role.

You can sign in to AWS as a federated identity by using credentials provided through an identity source. AWS IAM Identity Center (IAM Identity Center) users, your company's single sign-on authentication, and your Google or Facebook credentials are examples of federated identities. When you sign in as a federated identity, your administrator previously set up identity federation using IAM roles. When you access AWS by using federation, you are indirectly assuming a role.

Depending on the type of user you are, you can sign in to the AWS Management Console or the AWS access portal. For more information about signing in to AWS, see How to sign in to your AWS account in the *AWS Sign-In User Guide*.

Regardless of the authentication method that you use, you might also be required to provide additional security information. For example, AWS recommends that you use multi-factor authentication (MFA) to increase the security of your account. To learn more, see Multi-factor

authentication in the *AWS IAM Identity Center User Guide* and Using multi-factor authentication (MFA) in AWS in the *IAM User Guide*.

AWS account root user

When you first create an AWS account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account root user and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you don't use the root user for your everyday tasks. Safeguard your root user credentials and use them to perform tasks that only the root user can perform. For the complete list of tasks that require you to sign in as the root user, see Tasks that require root user credentials in the *IAM User Guide*.

Federated identity

As a best practice, require human users, including users that require administrator access, to use federation with an identity provider to access AWS services by using temporary credentials.

A federated identity is a user from your enterprise user directory, a web identity provider, the AWS Directory Service, the Identity Center directory, or any user that accesses AWS services by using credentials provided through an identity source. When federated identities access AWS accounts, they assume roles, and the roles provide temporary credentials.

For centralized access management, we recommend that you use AWS IAM Identity Center. You can create users and groups in IAM Identity Center, or you can connect and synchronize to a set of users and groups in your own identity source for use across all your AWS accounts and applications. For information about IAM Identity Center, see What is IAM Identity Center? in the *AWS IAM Identity Center User Guide*.

IAM users and groups

An *IAM user* is an identity within your AWS account that has specific permissions for a single person or application. Where possible, we recommend relying on temporary credentials instead of creating IAM users who have long-term credentials such as passwords and access keys. However, if you have specific use cases that require long-term credentials with IAM users, we recommend that you rotate access keys. For more information, see Rotate access keys regularly for use cases that require long-term credentials in the *IAM User Guide*.

An IAM group is an identity that specifies a collection of IAM users. You can't sign in as a group. You can use groups to specify permissions for multiple users at a time. Groups make permissions easier

to manage for large sets of users. For example, you could have a group named *IAMAdmins* and give that group permissions to administer IAM resources.

Users are different from roles. A user is uniquely associated with one person or application, but a role is intended to be assumable by anyone who needs it. Users have permanent long-term credentials, but roles provide temporary credentials. For more information, see When to create an IAM user (instead of a role) in the *IAM User Guide*.

IAM roles

An *IAM role* is an identity within your AWS account that has specific permissions. An IAM role is similar to an IAM user but is not associated with a specific person. You can temporarily assume an IAM role in the AWS Management Console by switching roles. You can assume a role by calling an AWS Command Line Interface (AWS CLI) or AWS API operation or by using a custom URL. For more information about methods for using roles, see Using IAM roles in the *IAM User Guide*.

IAM roles with temporary credentials are useful in the following situations:

- **Federated user access** – To assign permissions to a federated identity, you create a role and define permissions for the role. When a federated identity authenticates, the identity is associated with the role and is granted the permissions that are defined by the role. For information about roles for federation, see Creating a role for a third-party Identity Provider in the *IAM User Guide*. If you use IAM Identity Center, you configure a permission set. To control what your identities can access after they authenticate, IAM Identity Center correlates the permission set to a role in IAM. For information about permissions sets, see Permission sets in the *AWS IAM Identity Center User Guide*.
- **Temporary IAM user permissions** – An IAM user can assume an IAM role to temporarily take on different permissions for a specific task.
- **Cross-account access** – You can use an IAM role to allow someone (a trusted principal) in a different account to access resources in your account. Roles are the primary way to grant cross-account access. However, with some AWS services, you can attach a policy directly to a resource (instead of using a role as a proxy). For more information about the difference between roles and resource-based policies for cross-account access, see How IAM roles differ from resource-based policies in the *IAM User Guide*.
- **Cross-service access** – Some AWS services use features in other AWS services. A service might do this using the calling principal's permissions, using a service role, or using a service-linked role.
  - **Principal permissions** – When you use an IAM user or role to perform actions in AWS, you are considered a principal. Policies grant permissions to a principal. When you use some services,

you might perform an action that then triggers another action in a different service. In this case, you must have permissions to perform both actions.

- **Service role** – A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Creating a role to delegate permissions to an AWS service in the *IAM User Guide*.
- **Service-linked role** – A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your AWS account and are owned by the service. An IAM administrator can view but not edit the permissions for service-linked roles.

- **Applications running on Amazon EC2** – You can use an IAM role to manage temporary credentials for applications that are running on an Amazon EC2 instance and making AWS CLI or AWS API requests. This is preferable to storing access keys within the Amazon EC2 instance. To assign an IAM role to an Amazon EC2 instance and make it available to all of its applications, you create an instance profile that is attached to the instance. An instance profile contains the role and enables programs that are running on the Amazon EC2 instance to get temporary credentials. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances in the *IAM User Guide*.

For more information about whether to use IAM roles, see When to create an IAM role (instead of a user) in the *IAM User Guide*.

Managing access using policies

You control access in AWS by creating policies and attaching them to AWS identities or resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when a principal (user, root user, or role session) makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. For more information about the structure and contents of JSON policy documents, see Overview of JSON policies in the *IAM User Guide*.

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

Every IAM entity (user or role) starts with no permissions. By default, users can do nothing, not even change their own password. To give a user permission to do something, an administrator must attach a permissions policy to a user. Or the administrator can add the user to a group that

has the intended permissions. When an administrator gives permissions to a group, all users in that group are granted those permissions.

IAM policies define permissions for an action regardless of the method that you use to perform the operation. For example, suppose that you have a policy that allows the `iam:GetRole` action. A user with that policy can get role information from the AWS Management Console, the AWS CLI, or the AWS API.

Identity-based policies

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, role, or group. These policies control what actions users and roles can perform, on which resources, and under what conditions. For more information about how to create an identity-based policy, see Creating IAM policies in the *IAM User Guide*.

Identity-based policies can be further categorized as *inline policies* or *managed policies*. Inline policies are embedded directly into a single user, group, or role. Managed policies are standalone policies that you can attach to multiple users, groups, and roles in your AWS account. Managed policies include AWS managed policies and customer managed policies. For more information about how to choose between a managed policy or an inline policy, see Choosing between managed policies and inline policies in the *IAM User Guide*.

Resource-based policies

Resource-based policies are JSON policy documents that you attach to a resource such as an Amazon S3 bucket. Service administrators can use these policies to define what actions a specified principal (account member, user, or role) can perform on that resource and under what conditions. Resource-based policies are inline policies. There are no managed resource-based policies.

Access control lists (ACLs)

Access control lists (ACLs) are a type of policy that controls which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format. Amazon S3, AWS WAF, and Amazon VPC are examples of services that support ACLs. For more information about ACLs, see Access Control List (ACL) overview in the *Amazon S3 User Guide*.

Other policy types

AWS supports additional, less-common policy types. These policy types can set the maximum permissions granted to you by the more common policy types.

- **Permissions boundaries** – A permissions boundary is an advanced feature in which you set the maximum permissions that an identity-based policy can grant to an IAM entity (IAM user or role). You can set a permissions boundary for an entity. The resulting permissions are the intersection of an entity's identity-based policies and its permissions boundaries. Resource-based policies that specify the user or role in the `Principal` field are not limited by the permissions boundary. An explicit deny in any of these policies overrides the allow. For more information about permissions boundaries, see Permissions boundaries for IAM entities in the *IAM User Guide*.
- **Service control policies (SCPs)** – SCPs are JSON policies that specify the maximum permissions for an organization or organizational unit (OU) in AWS Organizations. AWS Organizations is a service for grouping and centrally managing multiple AWS accounts that your business owns. If you enable all features in an organization, then you can apply SCPs to any or all of your accounts. The SCP limits permissions for entities in member accounts, including each AWS account root user. For more information about Organizations and SCPs, see How SCPs work in the *AWS Organizations User Guide*.
- **Session policies** – Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary session for a role or federated user. The resulting session's permissions are the intersection of the user or role's identity-based policies and the session policies. Permissions can also come from a resource-based policy. An explicit deny in any of these policies overrides the allow. For more information, see Session policies in the *IAM User Guide*.

Multiple policy types

When multiple types of policies apply to a request, the resulting permissions are more complicated to understand. To learn how AWS determines whether to allow a request when multiple policy types are involved, see Policy evaluation logic in the *IAM User Guide*.

How Amazon Q Developer works with IAM

Before you use IAM to manage access to Amazon Q Developer, learn what IAM features are available to use with Amazon Q Developer.

IAM features you can use with Amazon Q Developer

IAM feature	Amazon Q support
Identity-based policies	Yes
Resource-based policies	No
Policy actions	Yes
Policy resources	No
Policy condition keys	No
ACLs	No
ABAC (tags in policies)	No
Temporary credentials	Yes
Principal permissions	Yes
Service roles	No
Service-linked roles	Yes

To get a high-level view of how Amazon Q and other AWS services work with most IAM features, see AWS services that work with IAM in the *IAM User Guide*.

Identity-based policies for Amazon Q

Supports identity-based policies: Yes

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Define custom IAM permissions with customer managed policies in the *IAM User Guide*.

With IAM identity-based policies, you can specify allowed or denied actions and resources as well as the conditions under which actions are allowed or denied. To learn about all of the elements that you can use in a JSON policy, see IAM JSON policy elements reference in the *IAM User Guide*.

Identity-based policy examples for Amazon Q

To view examples of Amazon Q Developer identity-based policies, see Identity-based policy examples for Amazon Q Developer.

Resource-based policies within Amazon Q

Supports resource-based policies: No

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must specify a principal in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

To enable cross-account access, you can specify an entire account or IAM entities in another account as the principal in a resource-based policy. For more information, see Cross account resource access in IAM in the *IAM User Guide*.

Policy actions for Amazon Q

Supports policy actions: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Action` element of a JSON policy describes the actions that you can use to allow or deny access in a policy. Include actions in a policy to grant permissions to perform the associated operation.

To see a list of Amazon Q actions, see Manage access to Amazon Q Developer with policies.

Policy actions in Amazon Q use the following prefix before the action:

```
q
```

To specify multiple actions in a single statement, separate the actions with commas.

```
"Action": [
```

```
      "q:action1",
      "q:action2"
         ]
```

You can specify multiple actions using wildcards (*). For example, to specify all actions that begin with the word `Get`, include the following action:

```
"Action": "q:Get*"
```

To view examples of Amazon Q Developer identity-based policies, see Identity-based policy examples for Amazon Q Developer.

Policy resources for Amazon Q

Supports policy resources: No

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Resource` JSON policy element specifies the object or objects to which the action applies. As a best practice, specify a resource using its Amazon Resource Name (ARN). For actions that don't support resource-level permissions, use a wildcard (*) to indicate that the statement applies to all resources.

```
"Resource": "*"
```

To view examples of Amazon Q Developer identity-based policies, see Identity-based policy examples for Amazon Q Developer.

Policy condition keys for Amazon Q

Supports service-specific policy condition keys: No

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Condition` element specifies when statements execute based on defined criteria. You can create conditional expressions that use condition operators, such as equals or less than, to match

the condition in the policy with values in the request. To see all AWS global condition keys, see AWS global condition context keys in the *IAM User Guide*.

To view examples of Amazon Q Developer identity-based policies, see Identity-based policy examples for Amazon Q Developer.

ACLs in Amazon Q

Supports ACLs: No

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

ABAC with Amazon Q

Supports ABAC (tags in policies): No

Attribute-based access control (ABAC) is an authorization strategy that defines permissions based on attributes called tags. You can attach tags to IAM entities and AWS resources, then design ABAC policies to allow operations when the principal's tag matches the tag on the resource.

To control access based on tags, you provide tag information in the condition element of a policy using the `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, or `aws:TagKeys` condition keys.

If a service supports all three condition keys for every resource type, then the value is **Yes** for the service. If a service supports all three condition keys for only some resource types, then the value is **Partial**.

For more information about ABAC, see Define permissions with ABAC authorization in the *IAM User Guide*. To view a tutorial with steps for setting up ABAC, see Use attribute-based access control (ABAC) in the *IAM User Guide*.

Using temporary credentials with Amazon Q

Supports temporary credentials: Yes

Temporary credentials provide short-term access to AWS resources and are automatically created when you use federation or switch roles. AWS recommends that you dynamically generate temporary credentials instead of using long-term access keys. For more information, see Temporary security credentials in IAM and AWS services that work with IAM in the *IAM User Guide*.

Cross-service principal permissions for Amazon Q

Supports forward access sessions (FAS): Yes

Forward access sessions (FAS) use the permissions of the principal calling an AWS service, combined with the requesting AWS service to make requests to downstream services. For policy details when making FAS requests, see Forward access sessions.

Service roles for Amazon Q

Supports service roles: No

A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Create a role to delegate permissions to an AWS service in the *IAM User Guide*.

> **Warning**
>
> Changing the permissions for a service role might break Amazon Q functionality. Edit service roles only when Amazon Q provides guidance to do so.

Service-linked roles for Amazon Q

Supports service-linked roles: Yes

A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your AWS account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.

For details about creating or managing Amazon Q service-linked roles, see Using service-linked roles for Amazon Q Developer and User Subscriptions.

Manage access to Amazon Q Developer with policies

> **Note**
>
> The information on this page pertains to accessing Amazon Q Developer. For information about managing access to Amazon Q Business, see Identity-based policy examples for Amazon Q Business in the *Amazon Q Business User Guide*.

> The policies and examples in this topic are specific to Amazon Q in the AWS Management Console, AWS Console Mobile Application, AWS website, AWS Documentation, and in chat applications. Other services integrated with Amazon Q might require different policies or settings. End users of Amazon Q in third-party IDEs are not required to use IAM policies. For more information, see the documentation for the service that contains an Amazon Q feature or integration.

By default, users and roles don't have permission to use Amazon Q. IAM administrators can manage access to Amazon Q Developer and its features by granting permissions to IAM identities.

The quickest way for an administrator to grant access to users is through an AWS managed policy. The `AmazonQFullAccess` policy can be attached to IAM identities to grant full access to Amazon Q Developer and its features. For more information about this policy, see AWS managed policies for Amazon Q Developer.

To manage specific actions that IAM identities can perform with Amazon Q Developer, administrators can create custom policies that define what permissions a user, group, or role has. You can also use service control policies (SCPs) to control what Amazon Q features are available in your organization.

For a list of all Amazon Q permissions you can control with policies, see the see the Amazon Q Developer permissions reference.

Topics

- Policy best practices
- Assign permissions
- Manage access with service control policies (SCPs)
- Identity-based policy examples for Amazon Q Developer

Policy best practices

Identity-based policies determine whether someone can create, access, or delete Amazon Q Developer resources in your account. These actions can incur costs for your AWS account. When you create or edit identity-based policies, follow these guidelines and recommendations:

- **Get started with AWS managed policies and move toward least-privilege permissions** – To get started granting permissions to your users and workloads, use the *AWS managed policies*

that grant permissions for many common use cases. They are available in your AWS account. We recommend that you reduce permissions further by defining AWS customer managed policies that are specific to your use cases. For more information, see AWS managed policies or AWS managed policies for job functions in the *IAM User Guide*.

- **Apply least-privilege permissions** – When you set permissions with IAM policies, grant only the permissions required to perform a task. You do this by defining the actions that can be taken on specific resources under specific conditions, also known as *least-privilege permissions*. For more information about using IAM to apply permissions, see Policies and permissions in IAM in the *IAM User Guide*.
- **Use conditions in IAM policies to further restrict access** – You can add a condition to your policies to limit access to actions and resources. For example, you can write a policy condition to specify that all requests must be sent using SSL. You can also use conditions to grant access to service actions if they are used through a specific AWS service, such as CloudFormation. For more information, see IAM JSON policy elements: Condition in the *IAM User Guide*.
- **Use IAM Access Analyzer to validate your IAM policies to ensure secure and functional permissions** – IAM Access Analyzer validates new and existing policies so that the policies adhere to the IAM policy language (JSON) and IAM best practices. IAM Access Analyzer provides more than 100 policy checks and actionable recommendations to help you author secure and functional policies. For more information, see Validate policies with IAM Access Analyzer in the *IAM User Guide*.
- **Require multi-factor authentication (MFA)** – If you have a scenario that requires IAM users or a root user in your AWS account, turn on MFA for additional security. To require MFA when API operations are called, add MFA conditions to your policies. For more information, see Secure API access with MFA in the *IAM User Guide*.

For more information about best practices in IAM, see Security best practices in IAM in the *IAM User Guide*.

Assign permissions

To provide access, add permissions to your users, groups, or roles:

- Users and groups in AWS IAM Identity Center:

  Create a permission set. Follow the instructions in Create a permission set in the *AWS IAM Identity Center User Guide*.
- Users managed in IAM through an identity provider:

Create a role for identity federation. Follow the instructions in Create a role for a third-party identity provider (federation) in the *IAM User Guide*.

- IAM users:
  - Create a role that your user can assume. Follow the instructions in Create a role for an IAM user in the *IAM User Guide*.
  - (Not recommended) Attach a policy directly to a user or add a user to a user group. Follow the instructions in Adding permissions to a user (console) in the *IAM User Guide*.

Manage access with service control policies (SCPs)

Service control policies (SCPs) are a type of organization policy that you can use to manage permissions in your organization. You can control what Amazon Q Developer features are available in your organization by creating an SCP that specifies permissions for some or all Amazon Q actions.

For more information about using SCPs to control access in your organization, see Creating, updating, and deleting service control policies and Attaching and detaching service control policies in the *AWS Organizations User Guide*.

Example SCP: Deny access to Amazon Q outside EU Regions

The following SCP denies access to any use of Amazon Q Developer outside of the Europe (Frankfurt) Region (eu-central-1).

> **Note**
>
> The `codewhisperer` prefix is a legacy name from a service that merged with Amazon Q Developer. For more information, see Amazon Q Developer rename - Summary of changes.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAmazonQDeveloperOutsideEU",
      "Effect": "Deny",
```

```
      "Action": [
         "codewhisperer:GenerateRecommendations",
         "q:SendMessage",
         "q:GenerateCodeFromCommands",
         "sqlworkbench:GetQSqlRecommendations"
         ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals":
        {"aws:RequestedRegion": [ "eu-central-1"] }
      }
    }
  ]
}
```

Example SCP: Deny access to Amazon Q

The following SCP denies access to Amazon Q Developer.

> **Note**
>
> Denying access to Amazon Q will not disable the Amazon Q icon or chat panel in the AWS console, AWS website, AWS documentation pages, or AWS Console Mobile Application.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAmazonQFullAccess",
      "Effect": "Deny",
      "Action": [
        "q:*"
      ],
      "Resource": "*"
    }
  ]
}
```

Identity-based policy examples for Amazon Q Developer

The following example IAM policies control permissions for various Amazon Q Developer actions. Use them to allow or deny Amazon Q Developer access for your users, roles, or groups.

> **Note**
>
> The following example policies grant permissions for features of Amazon Q Developer, but users might need additional permissions to access Amazon Q with an Amazon Q Developer Pro subscription. For more information, see Allow users to access Amazon Q with an Amazon Q Developer Pro subscription.

You can use these policies as written, or you can add permissions for the individual Amazon Q features you want to use. For more information about configuring IAM permissions with Amazon Q, see Manage access to Amazon Q Developer with policies.

For a list of all Amazon Q permissions you can control with policies, see the Amazon Q Developer permissions reference.

Topics

- Administrator permissions
- User permissions

Administrator permissions

The following policies allow Amazon Q Developer administrators to perform administrative tasks in the Amazon Q subscription management console and Amazon Q Developer console.

For policies that enable the use of Amazon Q Developer features, see User permissions.

Allow administrators to use the Amazon Q console

The following example policy grants permissions for a user to perform actions in the Amazon Q console. The Amazon Q console is where you configure Amazon Q's integration with AWS IAM Identity Center and AWS Organizations. Most other Amazon Q Developer-related tasks must be completed in the Amazon Q Developer console. For more information, see Allow administrators to use the Amazon Q Developer console.

Note

The `codewhisperer` prefix is a legacy name from a service that merged with Amazon Q Developer. For more information, see Amazon Q Developer rename - Summary of changes.

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "organizations:ListAWSServiceAccessForOrganization",
            "organizations:DisableAWSServiceAccess",
            "organizations:EnableAWSServiceAccess",
            "organizations:DescribeOrganization"
         ],
         "Resource":[
            "*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "sso:ListApplications",
            "sso:ListInstances",
            "sso:DescribeRegisteredRegions",
            "sso:GetSharedSsoConfiguration",
            "sso:DescribeInstance",
            "sso:CreateInstance",
            "sso:CreateApplication",
            "sso:PutApplicationAuthenticationMethod",
            "sso:PutApplicationAssignmentConfiguration",
            "sso:PutApplicationGrant",
            "sso:PutApplicationAccessScope",
            "sso:DescribeApplication",
            "sso:DeleteApplication",
            "sso:GetSSOStatus",
            "sso:CreateApplicationAssignment",
            "sso:DeleteApplicationAssignment",
```

```
            "sso:UpdateApplication"
         ],
         "Resource":[
            "*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "sso-directory:DescribeUsers",
            "sso-directory:DescribeGroups",
            "sso-directory:SearchGroups",
            "sso-directory:SearchUsers",
            "sso-directory:DescribeGroup",
            "sso-directory:DescribeUser",
            "sso-directory:DescribeDirectory"
         ],
         "Resource":[
            "*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "signin:ListTrustedIdentityPropagationApplicationsForConsole",
            "signin:CreateTrustedIdentityPropagationApplicationForConsole"
         ],
         "Resource":[
            "*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "codewhisperer:ListProfiles",
            "codewhisperer:CreateProfile",
            "codewhisperer:DeleteProfile"
         ],
         "Resource":[
            "*"
         ]
      },
      {
         "Effect":"Allow",
```

```
         "Action":[
            "user-subscriptions:ListClaims",
            "user-subscriptions:ListUserSubscriptions",
            "user-subscriptions:CreateClaim",
            "user-subscriptions:DeleteClaim",
            "user-subscriptions:UpdateClaim"
         ],
         "Resource":[
            "*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "q:CreateAssignment",
            "q:DeleteAssignment"
         ],
         "Resource":[
            "*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "iam:CreateServiceLinkedRole"
         ],
         "Resource":[
            "arn:aws-eusc:iam::*:role/aws-service-role/user-
subscriptions.amazonaws.com/AWSServiceRoleForUserSubscriptions"
         ]
      }
   ]
}
```

Allow administrators to use the Amazon Q Developer console

The following example policy grants permissions for a user to access the Amazon Q Developer console. In the Amazon Q Developer console, administrators perform most Amazon Q Developer-related configuration tasks, including tasks related to subscriptions, code references, customizations, and chat plugins. This policy also includes permissions to create and configure customer managed KMS keys.

There are a few Amazon Q Developer Pro tasks that administrators must complete through the Amazon Q console (instead of the Amazon Q Developer console). For more information, see Allow administrators to use the Amazon Q console.

> **Note**
>
> To create customizations or plugins, your Amazon Q Developer Pro administrator will require additional permissions.
>
> - For permissions needed for customizations, see the Prerequisites for customizations section.
> - For permissions needed for plugins, see Allow administrators to configure plugins.

You will need one of two policies to use the Amazon Q Developer console. The policy you need depends on if you're setting up Amazon Q Developer for the first time or if you're configuring a legacy Amazon CodeWhisperer profile.

> **Note**
>
> The `codewhisperer` prefix is a legacy name from a service that merged with Amazon Q Developer. For more information, see Amazon Q Developer rename - Summary of changes.

For new administrators of Amazon Q Developer, use the following policy:

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sso:ListInstances",
        "sso:CreateInstance",
        "sso:CreateApplication",
        "sso:PutApplicationAuthenticationMethod",
        "sso:PutApplicationGrant",
        "sso:PutApplicationAssignmentConfiguration",
        "sso:ListApplications",
        "sso:GetSharedSsoConfiguration",
```

```
        "sso:DescribeInstance",
        "sso:PutApplicationAccessScope",
        "sso:DescribeApplication",
        "sso:DeleteApplication",
        "sso:CreateApplicationAssignment",
        "sso:DeleteApplicationAssignment",
        "sso:UpdateApplication",
        "sso:DescribeRegisteredRegions",
        "sso:GetSSOStatus"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "iam:ListRoles"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "identitystore:DescribeUser"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "sso-directory:GetUserPoolInfo",
        "sso-directory:DescribeUsers",
        "sso-directory:DescribeGroups",
        "sso-directory:SearchGroups",
        "sso-directory:SearchUsers",
        "sso-directory:DescribeDirectory"
      ],
      "Resource": [
        "*"
```

```
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "signin:ListTrustedIdentityPropagationApplicationsForConsole",
        "signin:CreateTrustedIdentityPropagationApplicationForConsole"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "user-subscriptions:ListClaims",
        "user-subscriptions:ListApplicationClaims",
        "user-subscriptions:ListUserSubscriptions",
        "user-subscriptions:CreateClaim",
        "user-subscriptions:DeleteClaim",
        "user-subscriptions:UpdateClaim"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:DisableAWSServiceAccess",
        "organizations:EnableAWSServiceAccess"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:ListAliases",
        "kms:CreateGrant",
```

```
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:GenerateDataKey*",
        "kms:RetireGrant",
        "kms:DescribeKey"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "codeguru-security:UpdateAccountConfiguration"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": [
        "arn:aws-eusc:iam::*:role/aws-service-role/q.amazonaws.com/
AWSServiceRoleForAmazonQDeveloper"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "codewhisperer:UpdateProfile",
        "codewhisperer:ListProfiles",
        "codewhisperer:TagResource",
        "codewhisperer:UnTagResource",
        "codewhisperer:ListTagsForResource",
        "codewhisperer:CreateProfile"
      ],
      "Resource": [
        "*"
      ]
    },
    {
```

```
      "Effect": "Allow",
      "Action": [
        "q:ListDashboardMetrics",
        "q:CreateAssignment",
        "q:DeleteAssignment"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "cloudwatch:GetMetricData",
        "cloudwatch:ListMetrics"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

For legacy Amazon CodeWhisperer profiles, the following policy will enable an IAM principal to administer a CodeWhisperer application.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sso-directory:SearchUsers",
        "sso-directory:SearchGroups",
        "sso-directory:GetUserPoolInfo",
        "sso-directory:DescribeDirectory",
        "sso-directory:ListMembersInGroup"
      ],
      "Resource": [
        "*"
      ]
```

```
    },
    {
      "Effect": "Allow",
      "Action": [
        "iam:ListRoles"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "pricing:GetProducts"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "sso:AssociateProfile",
        "sso:DisassociateProfile",
        "sso:GetProfile",
        "sso:ListProfiles",
        "sso:ListApplicationInstances",
        "sso:GetApplicationInstance",
        "sso:CreateManagedApplicationInstance",
        "sso:GetManagedApplicationInstance",
        "sso:ListProfileAssociations",
        "sso:GetSharedSsoConfiguration",
        "sso:ListDirectoryAssociations",
        "sso:DescribeRegisteredRegions",
        "sso:GetSsoConfiguration",
        "sso:GetSSOStatus"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
```

```
        "identitystore:ListUsers",
        "identitystore:ListGroups"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:ListAliases",
        "kms:CreateGrant",
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:GenerateDataKey*",
        "kms:RetireGrant",
        "kms:DescribeKey"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "codeguru-security:UpdateAccountConfiguration"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
```

```
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": [
        "arn:aws-eusc:iam::*:role/aws-service-role/q.amazonaws.com/
AWSServiceRoleForAmazonQDeveloper"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "codewhisperer:UpdateProfile",
        "codewhisperer:ListProfiles",
        "codewhisperer:TagResource",
        "codewhisperer:UnTagResource",
        "codewhisperer:ListTagsForResource",
        "codewhisperer:CreateProfile"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "q:ListDashboardMetrics",
        "cloudwatch:GetMetricData",
        "cloudwatch:ListMetrics"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

Allow administrators to create customizations

The following policy grants administrators permission to create and manage customizations in Amazon Q Developer.

To configure customizations in the Amazon Q Developer console, your Amazon Q Developer administrator will require access to the Amazon Q Developer console. For more information, see Allow administrators to use the Amazon Q Developer console.

> **Note**
>
> In the following policy, the IAM service will report errors on the `codeconnections:ListOwners` and `codeconnections:ListRepositories` permissions. Create the policy with these permissions anyway. The permissions are required, and the policy will work despite the errors.

> **Note**
>
> The `codewhisperer` prefix is a legacy name from a service that merged with Amazon Q Developer. For more information, see Amazon Q Developer rename - Summary of changes.

In the following example, replace *`account number`* with your AWS account number.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sso-directory:DescribeUsers"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:CreateGrant"
            ],
            "Resource": [
```

```
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "codewhisperer:CreateCustomization",
                "codewhisperer:DeleteCustomization",
                "codewhisperer:ListCustomizations",
                "codewhisperer:ListCustomizationVersions",
                "codewhisperer:UpdateCustomization",
                "codewhisperer:GetCustomization",
                "codewhisperer:ListCustomizationPermissions",
                "codewhisperer:AssociateCustomizationPermission",
                "codewhisperer:DisassociateCustomizationPermission"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "codeconnections:ListOwners",
                "codeconnections:ListRepositories",
                "codeconnections:ListConnections",
                "codeconnections:GetConnection"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "codeconnections:UseConnection",
            "Resource": [
                "*"
            ],
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "codeconnections:ProviderAction": [
                        "GitPull",
                        "ListRepositories",
                        "ListOwners"
```

```
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject*",
                "s3:GetBucket*",
                "s3:ListBucket*"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Allow administrators to configure plugins

The following example policy grants administrators permissions to view and configure third party plugins in the Amazon Q Developer console.

> **Note**
>
> In order to access the Amazon Q Developer console, administrators also need the permissions defined in Allow administrators to use the Amazon Q Developer console.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "q:CreatePlugin",
        "q:GetPlugin",
        "q:DeletePlugin",
```

```
        "q:ListPlugins",
        "q:ListPluginProviders",
        "q:UpdatePlugin",
        "q:CreateAuthGrant",
        "q:CreateOAuthAppConnection",
        "q:SendEvent",
        "q:UpdateAuthGrant",
        "q:UpdateOAuthAppConnection",
        "q:UpdatePlugin",
        "iam:CreateRole",
        "secretsmanager:CreateSecret"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "iam:PassRole"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "iam:PassedToService": [
            "q.amazonaws.com"
          ]
        }
      }
    }
  ]
}
```

Allow administrators to configure plugins from one provider

The following example policy grants an administrator permission to configure plugins from one provider, specified by the plugin ARN with the name of the plugin provider and a wildcard character (*). To use this policy, replace the following in the ARN in the Resource field:

- *AWS-region* – The AWS Region where the plugin will be created.
- *AWS-account-ID* – The AWS account ID of the account where your plugin is configured.
- *plugin-provider* – The name of the plugin provider that you want to allow configuration for, like `CloudZero`, `Datadog`, or `Wiz`. The plugin provider field is case sensitive.

> **Note**
>
> In order to access the Amazon Q Developer console, administrators also need the permissions defined in Allow administrators to use the Amazon Q Developer console.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCreateProviderPlugin",
            "Effect": "Allow",
            "Action": [
                "q:CreatePlugin",
                "q:GetPlugin",
                "q:DeletePlugin"
            ],
            "Resource": "arn:aws-eusc:qdeveloper:eusc-de-
east-1:111122223333:plugin/plugin-provider/*"
        }
    ]
}
```

Allow migration of more than one network or more than one subnet

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
            "Sid": "MGNNetworkMigrationAnalyzerEC2ResourceSgTag",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateSecurityGroup"
            ],
            "Resource": [
                "arn:aws-eusc:ec2:eusc-de-east-1:111122223333:vpc/*"
            ],
```

```
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/CreatedBy": "AWSApplicationMigrationService"
                }
            }
        },
        {
            "Sid": "MGNNetworkMigrationAnalyzerEC2RequestSgTag",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateSecurityGroup"
            ],
            "Resource": [
                "arn:aws-eusc:ec2:eusc-de-east-1:111122223333:security-group/*",
                "arn:aws-eusc:ec2:eusc-de-east-1:111122223333:security-group-
rule/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": "AWSApplicationMigrationService"
                }
            }
        },

        {
            "Sid": "MGNNetworkMigrationAnalyzerEC2SecurityGroupTags",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws-eusc:ec2:eusc-de-east-1:111122223333:security-group/*",
                "arn:aws-eusc:ec2:eusc-de-east-1:111122223333:security-group-
rule/*",
                "arn:aws-eusc:ec2:eusc-de-east-1:111122223333:network-interface/
*",
                "arn:aws-eusc:ec2:eusc-de-east-1:111122223333:network-insights-
path/*",
                "arn:aws-eusc:ec2:eusc-de-east-1:111122223333:network-insights-
analysis/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": "AWSApplicationMigrationService",
```

```
                    "ec2:CreateAction": [
                        "CreateSecurityGroup",
                        "CreateNetworkInterface",
                        "CreateNetworkInsightsPath",
                        "StartNetworkInsightsAnalysis"
                    ]
                }
            }
        },
        {
            "Sid": "MGNNetworkMigrationAnalyzerENIResourceTag",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface"
            ],
            "Resource": [
                "arn:aws-eusc:ec2:eusc-de-east-1:111122223333:subnet/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/CreatedBy": "AWSApplicationMigrationService"
                }
            }
        },
        {
            "Sid": "MGNNetworkMigrationAnalyzerENISG",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface"
            ],
            "Resource": [
                "arn:aws-eusc:ec2:eusc-de-east-1:111122223333:security-group/*"
            ]
        },
        {
            "Sid": "MGNNetworkMigrationAnalyzerEC2ResourceTag",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInsightsPath"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
```

```
                "StringEquals": {
                    "aws:ResourceTag/CreatedBy": "AWSApplicationMigrationService"
                }
            }
        },
        {
            "Sid": "MGNNetworkMigAnalyzerEC2RequestTag",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface",
                "ec2:CreateNetworkInsightsPath",
                "ec2:StartNetworkInsightsAnalysis"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": "AWSApplicationMigrationService"
                }
            }
        },
        {
            "Sid": "MGNNetworkMigrationAnalyzeNetwork",
            "Effect": "Allow",
            "Action": [
                "ec2:StartNetworkInsightsAnalysis"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

User permissions

The following policies allow users to access features of Amazon Q Developer on AWS apps and websites, including the AWS Management Console, AWS Console Mobile Application, and AWS Documentation site.

For policies that enable administrative access to Amazon Q Developer, see Administrator permissions.

> **Note**
>
> Users accessing Amazon Q in the IDE or Amazon Q on the command line don't require IAM permissions.

Allow users to access Amazon Q with an Amazon Q Developer Pro subscription

The following example policy grants permission to use Amazon Q with an Amazon Q Developer Pro subscription. Without these permissions, users can only access the Free tier of Amazon Q. To chat with Amazon Q or use other Amazon Q features, users need additional permissions, such as those granted by the example policies in this section.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowGetIdentity",
            "Effect": "Allow",
            "Action": [
                "q:GetIdentityMetaData"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowSetTrustedIdentity",
            "Effect": "Allow",
            "Action": [
                "sts:SetContext"
            ],
            "Resource": "arn:aws-eusc:sts::*:self"
        }
    ]
}
```

Allow Amazon Q access to customer managed keys

The following example policy grants users permissions to access features encrypted with a customer managed key by allowing Amazon Q access to the key. This policy is required to use Amazon Q if an administrator has set up a customer managed key for encryption.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "QKMSDecryptGenerateDataKeyPermissions",
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "kms:GenerateDataKey",
                "kms:GenerateDataKeyWithoutPlaintext",
                "kms:ReEncryptFrom",
                "kms:ReEncryptTo"
            ],
            "Resource": [
            "arn:aws-eusc:kms:eusc-de-east-1:111122223333:key/key_id"
            ],
            "Condition": {
                "StringLike": {
                    "kms:ViaService": [
                    "q.eusc-de-east-1.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

Allow users to chat with Amazon Q

The following example policy grants permissions to chat with Amazon Q in the console.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAmazonQConversationAccess",
      "Effect": "Allow",
      "Action": [
        "q:StartConversation",
        "q:SendMessage",
        "q:GetConversation",
        "q:ListConversations",
        "q:UpdateConversation",
        "q:DeleteConversation"
      ],
      "Resource": "*"
    }
  ]
}
```

Allow users to use Amazon Q CLI with AWS CloudShell

The following example policy grants permissions to use Amazon Q CLI with AWS CloudShell.

> **Note**
>
> The `codewhisperer` prefix is a legacy name from a service that merged with Amazon Q Developer. For more information, see Amazon Q Developer rename - Summary of changes.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "codewhisperer:GenerateRecommendations",
                "codewhisperer:ListCustomizations"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "q:StartConversation",
                "q:SendMessage"
            ],
            "Resource": "*"
        }
    ]
}
```

Allow users to run transformations on the command line

The following example policy grants permissions to transform code with the Amazon Q command line tool for transformations. This policy does not affect access to Amazon Q on the command line.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "qdeveloper:StartAgentSession",
              "qdeveloper:ImportArtifact",
              "qdeveloper:ExportArtifact",
              "qdeveloper:TransformCode"
            ],
            "Resource": "*"
        }
    ]
}
```

Allow users to diagnose console errors with Amazon Q

The following example policy grants permissions to diagnose console errors with Amazon Q.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAmazonQTroubleshooting",
      "Effect": "Allow",
      "Action": [
        "q:StartTroubleshootingAnalysis",
        "q:GetTroubleshootingResults",
        "q:StartTroubleshootingResolutionExplanation",
        "q:UpdateTroubleshootingCommandResult",
        "q:PassRequest",
        "cloudformation:GetResource"
      ],
      "Resource": "*"
    }
  ]
}
```

Allow users to generate code from CLI commands with Amazon Q

The following example policy grants permissions to generate code from recorded CLI commands with Amazon Q, which enables the use of the Console-to-Code feature.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
       {
         "Sid": "AllowAmazonQConsoleToCode",
         "Effect": "Allow",
         "Action": "q:GenerateCodeFromCommands",
         "Resource": "*"
```

```
        }
    ]
}
```

Allow users to chat about resources with Amazon Q

The following example policy grants permission to chat with Amazon Q about resources, and allows Amazon Q to retrieve resource information on your behalf. Amazon Q only has permission to access resources that your IAM identity has permissions for.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAmazonQPassRequest",
      "Effect": "Allow",
      "Action": [
        "q:StartConversation",
        "q:SendMessage",
        "q:GetConversation",
        "q:ListConversations",
        "q:UpdateConversation",
        "q:DeleteConversation",
        "q:PassRequest"
      ],
      "Resource": "*"
    },
    {
      "Sid": "AllowCloudControlReadAccess",
      "Effect": "Allow",
      "Action": [
         "cloudformation:GetResource",
         "cloudformation:ListResources"
      ],
      "Resource": "*"
    }
  ]
}
```

Allow Amazon Q to perform actions on your behalf in chat

The following example policy grants permission to chat with Amazon Q, and allows Amazon Q to perform actions on your behalf. Amazon Q only has permission to perform actions that your IAM identity has permission to perform.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAmazonQPassRequest",
      "Effect": "Allow",
      "Action": [
        "q:StartConversation",
        "q:SendMessage",
        "q:GetConversation",
        "q:ListConversations",
        "q:UpdateConversation",
        "q:DeleteConversation",
        "q:PassRequest"
      ],
      "Resource": "*"
    }
  ]
}
```

Allow Amazon Q to access cost data and provide cost optimization recommendations

The following example policy grants permission to chat with Amazon Q about your costs and allows Amazon Q to access your cost data and provide cost analysis and optimization recommendations. This policy includes permissions for AWS Cost Explorer, AWS Cost Optimization Hub, AWS Compute Optimizer, AWS Budgets, AWS Free Tier, AWS Pricing, and Savings Plans and reservation recommendations.

JSON

```
{
  "Version":"2012-10-17",
```

```
  "Statement": [
    {
      "Sid": "AllowAmazonQChatAndPassRequest",
      "Effect": "Allow",
      "Action": [
        "q:StartConversation",
        "q:SendMessage",
        "q:GetConversation",
        "q:ListConversations",
        "q:UpdateConversation",
        "q:DeleteConversation",
        "q:PassRequest"
      ],
      "Resource": "*"
   },
    {
      "Sid": "AllowCostExplorerAccess",
      "Effect": "Allow",
      "Action": [
        "ce:GetCostAndUsage",
        "ce:GetCostAndUsageWithResources",
        "ce:GetCostForecast",
        "ce:GetUsageForecast",
        "ce:GetTags",
        "ce:GetCostCategories",
        "ce:GetDimensionValues",
        "ce:GetSavingsPlansUtilization",
        "ce:GetSavingsPlansCoverage",
        "ce:GetSavingsPlansUtilizationDetails",
        "ce:GetReservationUtilization",
        "ce:GetReservationCoverage",
        "ce:GetSavingsPlansPurchaseRecommendation",
        "ce:GetReservationPurchaseRecommendation",
        "ce:GetRightsizingRecommendation",
        "ce:GetAnomalies",
       "ce:GetCostAndUsageComparisons",
       "ce:GetCostComparisonDrivers"
      ],
      "Resource": "*"
   },
    {
      "Sid": "AllowCostOptimizationHubAccess",
      "Effect": "Allow",
      "Action": [
```

```
        "cost-optimization-hub:GetRecommendation",
        "cost-optimization-hub:ListRecommendations",
        "cost-optimization-hub:ListRecommendationSummaries"
      ],
      "Resource": "*"
    },
     {
      "Sid": "AllowComputeOptimizerAccess",
      "Effect": "Allow",
      "Action": [
        "compute-optimizer:GetAutoScalingGroupRecommendations",
        "compute-optimizer:GetEBSVolumeRecommendations",
        "compute-optimizer:GetEC2InstanceRecommendations",
        "compute-optimizer:GetECSServiceRecommendations",
        "compute-optimizer:GetRDSDatabaseRecommendations",
        "compute-optimizer:GetLambdaFunctionRecommendations",
        "compute-optimizer:GetIdleRecommendations",
        "compute-optimizer:GetLicenseRecommendations",
        "compute-optimizer:GetEffectiveRecommendationPreferences"
      ],
      "Resource": "*"
    },
     {
      "Sid": "AllowBudgetsAccess",
      "Effect": "Allow",
      "Action": [
        "budgets:ViewBudget"
      ],
      "Resource": "*"
    },
     {
      "Sid": "AllowFreeTierAccess",
      "Effect": "Allow",
      "Action": [
        "freetier:GetFreeTierUsage",
        "freetier:GetAccountPlanState",
        "freetier:ListAccountActivities",
       "freetier:GetAccountActivity"
      ],
      "Resource": "*"
    },
     {
      "Sid": "AllowPricingAccess",
      "Effect": "Allow",
```

```
      "Action": [
        "pricing:GetProducts",
        "pricing:GetAttributeValues",
        "pricing:DescribeServices"
      ],
      "Resource": "*"
    }
  ]
}
```

Deny Amazon Q permission to perform specific actions on your behalf

The following example policy grants permission to chat with Amazon Q, and allows Amazon Q to perform any action on your behalf that your IAM identity has permission to perform, except for Amazon EC2 actions. This policy uses the aws:CalledVia global condition key to specify that Amazon EC2 actions are only denied when Amazon Q calls them.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "q:StartConversation",
        "q:SendMessage",
        "q:GetConversation",
        "q:ListConversations",
        "q:UpdateConversation",
        "q:DeleteConversation",
        "q:PassRequest"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Deny",
      "Action": [
        "ec2:*"
      ],
      "Resource": "*",
```

```
      "Condition": {
            "ForAnyValue:StringEquals": {
               "aws:CalledVia": ["q.amazonaws.com"]
            }
       }
    }
  ]
}
```

Allow Amazon Q permission to perform specific actions on your behalf

The following example policy grants permission to chat with Amazon Q, and allows Amazon Q to perform any action on your behalf that your IAM identity has permission to perform, with the exception of Amazon EC2 actions. This policy grants your IAM identity permission to perform any Amazon EC2 action, but only allows Amazon Q to perform the `ec2:describeInstances` action. This policy uses the [`aws:CalledVia` global condition key] to specify that Amazon Q is only allowed to call `ec2:describeInstances`, and not any other Amazon EC2 actions.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "q:StartConversation",
        "q:SendMessage",
        "q:GetConversation",
        "q:ListConversations",
        "q:UpdateConversation",
        "q:DeleteConversation",
        "q:PassRequest"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "ec2:*"
      ],
```

```
      "Resource": "*",
      "Condition": {
            "ForAnyValue:StringNotEquals": {
               "aws:CalledVia": ["q.amazonaws.com"]
            }
       }
    },
    {
      "Effect": "Allow",
      "Action": [
        "ec2:describeInstances"
      ],
      "Resource": "*",
       "Condition": {
            "ForAnyValue:StringEquals": {
               "aws:CalledVia": ["q.amazonaws.com"]
            }
       }
    }
  ]
}
```

Allow Amazon Q permission to perform actions on your behalf in specific regions

The following example policy grants permission to chat with Amazon Q, and allows Amazon Q to make calls to only the `us-east-1` and `us-west-2` Regions when performing actions on your behalf. Amazon Q can't make calls to any other Region. For more information on how to specify what Regions you can make calls to, see aws:RequestedRegion in the *AWS Identity and Access Management User Guide*.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "q:StartConversation",
        "q:SendMessage",
        "q:GetConversation",
```

```
        "q:ListConversations",
        "q:UpdateConversation",
        "q:DeleteConversation",
        "q:PassRequest"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
            "aws:RequestedRegion": [
                "eusc-de-east-1",
                "us-west-2"
            ]
        }
      }
    }
  ]
}
```

Deny Amazon Q permission to perform actions on your behalf

The following example policy prevents Amazon Q from performing actions on your behalf.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAmazonQPassRequest",
      "Effect": "Deny",
      "Action": [
        "q:PassRequest"
      ],
      "Resource": "*"
    }
  ]
}
```

Allow users to chat with plugins from one provider

The following example policy grants permission to chat with any plugin from a given provider that an administrator configures, specified by the plugin ARN with the name of the plugin provider and a wildcard character (*). If the plugin is deleted and re-configured, a user with these permissions will retain access to the newly configured plugin. To use this policy, replace the following in the ARN in the `Resource` field:

- *`AWS-region`* – The AWS Region where the plugin was created.
- *`AWS-account-ID`* – The AWS account ID of the account where your plugin is configured.
- *`plugin-provider`* – The name of the plugin provider that you want to allow access to, like `CloudZero`, `Datadog`, or `Wiz`. The plugin provider field is case sensitive.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAmazonQConversationAccess",
            "Effect": "Allow",
            "Action": [
                "q:StartConversation",
                "q:SendMessage",
                "q:GetConversation",
                "q:ListConversations",
                "q:UpdateConversation",
                "q:DeleteConversation"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowAmazonQPluginAccess",
            "Effect": "Allow",
            "Action": [
                "q:UsePlugin"
            ],
            "Resource": "arn:aws-eusc:qdeveloper:eusc-de-
east-1:111122223333:plugin/plugin-provider/*"
        }
```

```
    ]
}
```

Allow users to chat with a specific plugin

The following example policy grants permission to chat with a specific plugin, specified by the plugin ARN. If the plugin is deleted and re-configured, a user will not have access to the new plugin unless the plugin ARN is updated in this policy. To use this policy, replace the following in the ARN in the `Resource` field:

- *`AWS-region`* – The AWS Region where the plugin was created.
- *`AWS-account-ID`* – The AWS account ID of the account where your plugin is configured.
- *`plugin-provider`* – The name of the plugin provider that you want to allow access to, like `CloudZero`, `Datadog`, or `Wiz`. The plugin provider field is case sensitive.
- *`plugin-ARN`* – The ARN of the plugin you want to allow access to.

Deny access to Amazon Q

The following example policy denies all permissions to use Amazon Q.

> **Note**
>
> When you deny access to Amazon Q, the Amazon Q icon and chat panel will still appear in the AWS console, AWS website, AWS documentation pages, or AWS Console Mobile Application.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAmazonQFullAccess",
      "Effect": "Deny",
      "Action": [
        "q:*"
```

```
      ],
      "Resource": "*"
    }
  ]
}
```

Allow users to view their permissions

This example shows how you might create a policy that allows IAM users to view the inline and managed policies that are attached to their user identity. This policy includes permissions to complete this action on the console or programmatically using the AWS CLI or AWS API.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws-eusc:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
```

```
    ]
}
```

Manage access to Amazon Q Developer for third-party integration

For third-party integrations, you must use the AWS Key Management Service (KMS) to manage access to Amazon Q Developer instead of IAM policies that are neither identity-based or resource-based.

Allow administrators to use customer managed keys to update role policies

The following example key policy grants permission to use customer managed keys (CMK) when creating your key policy on a configured role in the KMS Console. When configuring the CMK, you must provide the IAM role ARN, an identifier, used by your integration to call Amazon Q. If you already onboarded an integration such as a GitLab instance, you must reonboard the instance for all resources to be encrypted with CMK.

The `kms:ViaService` condition key limits the use of a KMS key to requests from specified AWS services. Additionally, it's used to deny permission to use a KMS key when the request comes from particular services. With the condition key, you can limit who can use CMK for encrypting or decrypting content. For more information, see kms:ViaService in the *AWS Key Management Service Developer Guide*.

With KMS encryption context, you have an optional set of key-value pairs that can be included in cryptographic operations with symmetric encryption KMS keys to enhance authorization and auditability. The encryption context can be used to verify the integrity and authenticity of encrypted data, control access to symmetric encryption KMS keys in key policies and IAM policies, and identify and categorize cryptographic operations in AWS CloudTrail logs. For more information, see Encryption context in the *AWS Key Management Service Developer Guide*.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Sid0",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws-eusc:iam::111122223333:role/rolename"
```

```
            },
            "Action": [
                "kms:GenerateDataKeyWithoutPlaintext",
                "kms:Decrypt",
                "kms:ReEncryptFrom",
                "kms:ReEncryptTo",
                "kms:GenerateDataKey",
                "kms:Encrypt"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": "q.eusc-de-east-1.amazonaws.com",
                    "kms:EncryptionContext:aws-crypto-
ec:aws:qdeveloper:accountId": "111122223333"
                }
            }
        },
        {
            "Sid": "Sid1",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws-eusc:iam::111122223333:role/rolename"
            },
            "Action": "kms:DescribeKey",
            "Resource": "*"
        }
    ]
}
```

Amazon Q Developer permissions reference

Amazon Q Developer uses two types of APIs to provide the service:

- User and administrator permissions, which can be used in policies to control usage of Amazon Q
- Other APIs used to provide the service, which can't be used in policies to control usage of Amazon Q

This section provides information about the APIs used by Amazon Q Developer, and what they do.

Topics

- Amazon Q Developer permissions
- Amazon Q User Subscriptions permissions
- Other Amazon Q Developer APIs

Amazon Q Developer permissions

You can use the following permissions as a reference when you are setting up Authenticating with identities in Amazon Q and writing permissions policies that you can attach to an IAM identity (identity-based policies).

The following table shows the Amazon Q Developer permissions that you can allow or deny access to in policies.

Important

To chat with Amazon Q, an IAM identity needs permissions for the following actions:

- `StartConversation`
- `SendMessage`
- `GetConversation` (console only)
- `ListConversations` (console only)

If one of these actions isn't explicitly allowed by an attached policy, an IAM permissions error is returned when you try to chat with Amazon Q.

Note

The `codewhisperer` prefix is a legacy name from a service that merged with Amazon Q Developer. For more information, see Amazon Q Developer rename - Summary of changes.

Using q:PassRequest

`q:PassRequest` is an Amazon Q permission that allows Amazon Q to call AWS APIs on your behalf. When you add the `q:PassRequest` permission to an IAM identity, Amazon Q gains

permission to call any API that the IAM identity has permission to call. For example, if an IAM role has the `s3:ListAllMyBuckets` permission and the `q:PassRequest` permission, Amazon Q is able to call the `ListAllMyBuckets` API when a user assuming that IAM role asks Amazon Q to list their Amazon S3 buckets.

You can create IAM policies that restrict the scope of the `q:PassRequest` permission. For example, you can prevent Amazon Q from performing a specific action, or only permit Amazon Q to perform a subset of actions for a service. You can also specify what regions Amazon Q can make calls to when performing actions on your behalf.

For examples of IAM policies that control the use of `q:PassRequest`, see the following identity-based policy examples:

- Allow Amazon Q to perform actions on your behalf in chat
- Deny Amazon Q permission to perform specific actions on your behalf
- Allow Amazon Q permission to perform specific actions on your behalf
- Allow Amazon Q permission to perform actions on your behalf in specific regions
- Deny Amazon Q permission to perform actions on your behalf

Amazon Q User Subscriptions permissions

Amazon Q Developer administrators must have the following permissions to create and manage subscriptions for users and groups in their organization.

The following terminology is useful in understanding what subscriptions permissions do:

User

An individual user, represented within AWS IAM Identity Center by a unique user ID.

Group

A collection of users, represented within AWS IAM Identity Center by a unique group ID.

Subscription

A subscription is tied to a single Identity Center user, and entitles them to use Amazon Q features. A subscription does not authorize a user to use Amazon Q features. For example, if Adam is subscribed to Amazon Q Developer Pro, they are entitled to used Amazon Q Developer

features, but they don't have access to those features until their administrator grants them the needed permissions.

Other Amazon Q Developer APIs

The following table shows the APIs that are used by features of Amazon Q in the IDE. These APIs aren't used to control access to features of Amazon Q, but they will appear in AWS CloudTrail logs in management accounts when users access the associated feature.

> **Note**
>
> The `codewhisperer` prefix is a legacy name from a service that merged with Amazon Q Developer. For more information, see Amazon Q Developer rename - Summary of changes.

AWS managed policies for Amazon Q Developer

An AWS managed policy is a standalone policy that is created and administered by AWS. AWS managed policies are designed to provide permissions for many common use cases so that you can start assigning permissions to users, groups, and roles.

The quickest way for an administrator to grant access to users is through an AWS managed policy. The following AWS managed policies for Amazon Q Developer can be attached to IAM identities:

- `AmazonQFullAccess` provides full access to enable interactions with Amazon Q Developer, including administrator access.
- `AmazonQDeveloperAccess` provides full access to enable interactions with Amazon Q Developer, without administrator access.

> **Note**
>
> Users accessing Amazon Q in the IDE or on the command line don't require IAM permissions.

Keep in mind that AWS managed policies might not grant least-privilege permissions for your specific use cases because they're available for all AWS customers to use. We recommend that you

reduce permissions further by defining customer managed policies that are specific to your use cases.

You cannot change the permissions defined in AWS managed policies. If AWS updates the permissions defined in an AWS managed policy, the update affects all principal identities (users, groups, and roles) that the policy is attached to. AWS is most likely to update an AWS managed policy when a new AWS service is launched or new API operations become available for existing services.

For more information, see AWS managed policies in the *IAM User Guide*.

AmazonQFullAccess

The `AmazonQFullAccess` managed policy provides administrator access to allow users in your organization to access Amazon Q Developer. It also provides full access to enable interactions with Amazon Q Developer, including logging in with IAM Identity Center to access Amazon Q through an Amazon Q Developer Pro subscription.

> **Note**
>
> To enable full access to complete administrative tasks in the Amazon Q subscription management console and Amazon Q Developer Pro console, additional permissions are needed. For more information, see Administrator permissions.

To view the permissions for this policy, see AmazonQFullAccess in the *AWS Managed Policy Reference*.

AmazonQDeveloperAccess

The `AmazonQDeveloperAccess` managed policy provides full access to enable interactions with Amazon Q Developer, without administrator access. It includes access to log in with IAM Identity Center to access Amazon Q through an Amazon Q Developer Pro subscription.

To use some features of Amazon Q, you might need additional permissions. See the topic for the feature you want to use for information on permissions.

To view the permissions for this policy, see AmazonQDeveloperAccess in the *AWS Managed Policy Reference*.

AWSServiceRoleForAmazonQDeveloper

This AWS managed policy grants permissions commonly needed to use Amazon Q Developer. The policy is added to the AWSServiceRoleForAmazonQDeveloper service linked role that is created when you onboard to Amazon Q.

You can't attach AWSServiceRoleForAmazonQDeveloper to your IAM entities. This policy is attached to a service-linked role that allows Amazon Q to perform actions on your behalf. For more information, see Using service-linked roles for Amazon Q Developer and User Subscriptions.

This policy grants *`administrator`* permissions that allows metrics to be published for Billing / Usage.

Permissions details

This policy includes the following permissions.

- `cloudwatch` – Allows principals to publish usage metrics to CloudWatch for Billing / Usage. This is required so that you can track your usage of Amazon Q in CloudWatch.

To view the permissions for this policy, see AWSServiceRoleForAmazonQDeveloper in the *AWS Managed Policy Reference*.

AWSServiceRoleForUserSubscriptions

This AWS managed policy grants permissions commonly needed to use Amazon Q Developer. The policy is added to the AWSServiceRoleForUserSubscriptions service-linked role that is created when you create Amazon Q subscriptions.

You can't attach AWSServiceRoleForUserSubscriptions to your IAM entities. This policy is attached to a service-linked role that allows Amazon Q to perform actions on your behalf. For more information, see Using service-linked roles for Amazon Q Developer and User Subscriptions.

This policy provides access for Amazon Q Subscriptions to your Identity Center resources to automatically update your subscriptions.

Permissions details

This policy includes the following permissions.

- `identitystore` – Allows principals to track Identity Center directory changes so that subscriptions can be automatically updated.
- `organizations` – Allows principals to track AWS Organizations changes so that subscriptions can be automatically updated.
- `sso` – Allows principals to track Identity Center instance changes so that subscriptions can be automatically updated.
- `kms` – Allows principals to access to KMS keys to authorize with Identity Center.

To view the permissions for this policy, see AWSServiceRoleForUserSubscriptions in the *AWS Managed Policy Reference*.

GitLabDuoWithAmazonQPermissionsPolicy

This policy grants permission to connect with Amazon Q and utilize the features in the GitLab Duo with Amazon Q integration. The policy is added to the IAM role created from the Amazon Q Developer console to access Amazon Q. You need to manually provide the IAM role to GitLab as an Amazon Resource Name (ARN). The policy allows the following:

- **GitLab Duo usage permissions** - Allows basic operations such as sending events and messages, creating and updating auth grants, generating code recommendations, listing plugins, and verifying OAuth app connections.
- **GitLab Duo management permissions** - Enables the creation and deletion of OAuth app connections, providing control over the integration setup.
- **GitLab Duo plugin permissions** - Grants specific permissions to create, delete, and retrieve plugins related to the GitLab Duo integration with Amazon Q.

To view the permissions for this policy, see GitLabDuoWithAmazonQPermissionsPolicy in the *AWS Managed Policy Reference*.

Policy updates

View details about updates to AWS managed policies for Amazon Q Developer since this service began tracking these changes. For automatic alerts about changes to this page, subscribe to the RSS feed on the Document history for Amazon Q Developer User Guide page.

Change	Description	Date
AmazonQDeveloperAccess - Updated policy	Additional permissions have been added to enable access to KMS keys to authorize with Identity Center.	October 29, 2025
AmazonQFullAccess - Updated policy	Additional permissions have been added to enable access to KMS keys to authorize with Identity Center.	October 29, 2025
AWSServiceRoleForUserSubscriptions - Updated policy	Additional permissions have been added to enable access to KMS keys to authorize with Identity Center.	October 29, 2025
AmazonQDeveloperAccess - Updated policy	Additional permissions have been added to manage conversation history in Amazon Q chat.	May 14, 2025
AmazonQFullAccess - Updated policy	Additional permissions have been added to manage conversation history in Amazon Q chat.	May 14, 2025
AmazonQFullAccess - Updated policy	Additional permission has been added to update feature enablement controls for third-party integration plugin.	May 2, 2025
AmazonQFullAccess - Updated policy	Additional permissions have been added to allow access and enable interactions with Amazon Q Developer for third-party plugins.	April 30, 2025
GitLabDuoWithAmazonQPermissionsPolicy - Updated policy	Additional permission has been added to allow updates to a third-party OAuth application with Amazon Q Developer.	April 30, 2025
GitLabDuoWithAmazo	Allows GitLab to connect with Amazon Q Developer to use GitLab Duo with Amazon Q integration features.	April 17, 2025

Change	Description	Date
nQPermissionsPolicy - New policy		
AWSServiceRoleForUserSubscriptions - Updated policy	Allows Amazon Q to discover the email verification status of end users.	February 17, 2025
AmazonQDeveloperAccess - Updated policy	Additional permissions have been added to enable the use of Amazon Q Developer plugins.	November 13, 2024
AmazonQFullAccess - Updated policy	Additional permissions have been added to configure and use Amazon Q Developer plugins and to create and manage tags for Amazon Q Developer resources.	November 13, 2024
AmazonQDeveloperAccess - Updated policy	Additional permissions have been added to enable code generation from CLI commands with Amazon Q.	October 28, 2024
AmazonQFullAccess - Updated policy	Additional permissions have been added to enable code generation from CLI commands with Amazon Q.	October 28, 2024
AmazonQFullAccess - Updated policy	Additional permissions have been added to enable Amazon Q to access downstream resources.	July 9, 2024
AmazonQDeveloperAccess - New policy	Provides full access to enable interactions with Amazon Q Developer, without administrator access.	July 9, 2024
AmazonQFullAccess - Updated policy	Additional permissions have been added to enable subscriptions checks for Amazon Q Developer.	April 30, 2024

Change	Description	Date
AWSServiceRoleForUserSubscriptions - New policy	Allows Amazon Q Subscriptions to automatically update subscriptions from changes in AWS IAM Identity Center, AWS IAM Identity Center directory and AWS Organizations on your behalf.	April 30, 2024
AWSServiceRoleForAmazonQDeveloper - New policy	Allows Amazon Q to call Amazon CloudWatch and Amazon CodeGuru on your behalf.	April 30, 2024
AmazonQFullAccess - New policy	Provides full access to enable interactions with Amazon Q Developer.	November 28, 2023
Amazon Q Developer started tracking changes	Amazon Q Developer started tracking changes to AWS managed policies.	November 28, 2023

Using service-linked roles for Amazon Q Developer and User Subscriptions

Amazon Q Developer uses AWS Identity and Access Management (IAM) service-linked roles. A service-linked role is a unique type of IAM role that is linked directly to Amazon Q Developer. Service-linked roles are predefined by Amazon Q Developer and include all the permissions that the service requires to call other AWS services on your behalf.

Topics

- Using service-linked roles for Amazon Q Developer
- Using service-linked-roles for User Subscriptions

Using service-linked roles for Amazon Q Developer

Amazon Q Developer uses AWS Identity and Access Management (IAM) service-linked roles. A service-linked role is a unique type of IAM role that is linked directly to Amazon Q Developer. Service-linked roles are predefined by Amazon Q Developer and include all the permissions that the service requires to call other AWS services on your behalf.

A service-linked role makes setting up Amazon Q Developer easier because you don't have to manually add the necessary permissions. Amazon Q Developer defines the permissions of its service-linked roles, and unless defined otherwise, only Amazon Q Developer can assume its roles. The defined permissions include the trust policy and the permissions policy, and that permissions policy cannot be attached to any other IAM entity.

You can delete a service-linked role only after first deleting their related resources. This protects your Amazon Q Developer resources because you can't inadvertently remove permission to access the resources.

For information about other services that support service-linked roles, see AWS services that work with IAM and look for the services that have **Yes** in the **Service-linked roles** column. Choose a **Yes** with a link to view the service-linked role documentation for that service.

Learn about AWS managed policies for Amazon Q Developer.

Service-linked role permissions for Amazon Q Developer

Amazon Q Developer uses the service-linked role named **AWSServiceRoleForAmazonQDeveloper** – This role grants permissions to Amazon Q to access data in your account to calculate billing, provides access to create and access security reports in Amazon CodeGuru, and emit data to CloudWatch.

The AWSServiceRoleForAmazonQDeveloper service-linked role trusts the following services to assume the role:

- `q.amazonaws.com`

The role permissions policy named AWSServiceRoleForAmazonQDeveloper allows Amazon Q Developer to complete the following actions on the specified resources:

- Action: `cloudwatch:PutMetricData` on `AWS/Q CloudWatch namespace`

You must configure permissions to allow your users, groups, or roles to create, edit, or delete a service-linked role. For more information, see Service-linked role permissions in the *IAM User Guide*.

Creating a service-linked role for Amazon Q Developer

You don't need to manually create a service-linked role. When you create a profile for Amazon Q in the AWS Management Console, Amazon Q Developer creates the service-linked role for you.

If you delete this service-linked role, and then need to create it again, you can use the same process to recreate the role in your account. When you update the settings, Amazon Q creates the service-linked role for you again.

You can also use the IAM console or AWS CLI to create a service-linked role with the `q.amazonaws.com` service name. For more information, see Creating a service-linked role in the *IAM User Guide*. If you delete this service-linked role, you can use this same process to create the role again.

Editing a service-linked role for Amazon Q Developer

Amazon Q Developer does not allow you to edit the AWSServiceRoleForAmazonQDeveloper service-linked role. After you create a service-linked role, you cannot change the name of the role because various entities might reference the role. However, you can edit the description of the role using IAM. For more information, see Editing a service-linked role in the *IAM User Guide*.

Deleting a service-linked role for Amazon Q Developer

If you no longer need to use a feature or service that requires a service-linked role, we recommend that you delete that role. That way you don’t have an unused entity that is not actively monitored or maintained. However, you must clean up the resources for your service-linked role before you can manually delete it.

> **Note**
>
> If the Amazon Q Developer service is using the role when you try to delete the resources, then the deletion might fail. If that happens, wait for a few minutes and try the operation again.

To manually delete the service-linked role using IAM

Use the IAM console, the AWS CLI, or the AWS API to delete the AWSServiceRoleForAmazonQDeveloper service-linked role. For more information, see Deleting a service-linked role in the *IAM User Guide*.

Supported Regions for Amazon Q Developer service-linked roles

Amazon Q Developer does not support using service-linked roles in every Region where the service is available. You can use the AWSServiceRoleForAmazonQDeveloper role in the following Regions. For more information, see AWS Regions and endpoints.

Region name	Region identity	Support in Amazon Q Developer
US East (N. Virginia)	eusc-de-east-1	Yes
	eusc-de-east-1	No
US West (N. California)	us-west-1	No
US West (Oregon)	us-west-2	No
Africa (Cape Town)	af-south-1	No
Asia Pacific (Hong Kong)	ap-east-1	No
Asia Pacific (Jakarta)	ap-southeast-3	No
Asia Pacific (Mumbai)	ap-south-1	No
Asia Pacific (Osaka)	ap-northeast-3	No
Asia Pacific (Seoul)	ap-northeast-2	No
Asia Pacific (Singapore)	ap-southeast-1	No
Asia Pacific (Sydney)	ap-southeast-2	No
Asia Pacific (Tokyo)	ap-northeast-1	No
Canada (Central)	ca-central-1	No

Region name	Region identity	Support in Amazon Q Developer
Europe (Frankfurt)	eu-central-1	No
Europe (Ireland)	eu-west-1	No
Europe (London)	eu-west-2	No
Europe (Milan)	eu-south-1	No
Europe (Paris)	eu-west-3	No
Europe (Stockholm)	eu-north-1	No
Middle East (Bahrain)	me-south-1	No
Middle East (UAE)	me-central-1	No
South America (São Paulo)	sa-east-1	No
AWS GovCloud (US-East)	us-gov-east-1	No
AWS GovCloud (US-West)	us-gov-west-1	No

Using service-linked-roles for User Subscriptions

User Subscriptions uses AWS Identity and Access Management (IAM) service-linked roles. A service-linked role is a unique type of IAM role that is linked directly to User Subscriptions. Service-linked roles are predefined by User Subscriptions and include all the permissions that the service requires to call other AWS services on your behalf.

A service-linked role makes setting up User Subscriptions easier because you don't have to manually add the necessary permissions. User Subscriptions defines the permissions of its service-linked roles, and unless defined otherwise, only User Subscriptions can assume its roles. The defined permissions include the trust policy and the permissions policy, and that permissions policy cannot be attached to any other IAM entity.

You can delete a service-linked role only after first deleting their related resources. This protects your User Subscriptions because you can't inadvertently remove permissions required by the resources.

For information about other services that support service-linked roles, see AWS services that work with IAM and look for the services that have **Yes** in the **Service-linked roles** column. Choose a **Yes** with a link to view the service-linked role documentation for that service.

Service-linked role permissions for User Subscriptions

User Subscriptions uses the service-linked role named **AWSServiceRoleForUserSubscriptions**. This role provides access for User Subscriptions to your IAM Identity Center resources in order to automatically update your subscriptions.

The AWSServiceRoleForUserSubscriptions service-linked role trusts the following services to assume the role:

- `user-subscriptions.amazonaws.com`

The role permissions policy named AWSServiceRoleForUserSubscriptions allows User Subscriptions to complete the following actions on the specified resources:

- Action: `identitystore:DescribeGroup` on *

  Action: `identitystore:DescribeUser` on *

  Action: `identitystore:IsMemberInGroups` on *

  Action: `identitystore:ListGroupMemberships` on *

  Action: `organizations:DescribeOrganization` on *

  Action: `sso:DescribeApplication` on *

  Action: `sso:DescribeInstance` on *

  Action: `sso:ListInstances` on *

  Action: `sso-directory:DescribeUser` on *

You must configure permissions to allow your users, groups, or roles to create, edit, or delete a service-linked role. For more information, see Service-linked role permissions in the *IAM User Guide*.

Creating a service-linked role for User Subscriptions

You don't need to manually create a service-linked role. When you create a User Subscription in the AWS Management Console, User Subscriptions creates the service-linked role for you.

If you delete this service-linked role, and then need to create it again, you can use the same process to recreate the role in your account. When you update the settings, User Subscriptions creates the service-linked role for you again.

You can also use the IAM console or AWS CLI to create a service-linked role with the `q.amazonaws.com` service name. For more information, see Creating a service-linked role in the *IAM User Guide*. If you delete this service-linked role, you can use this same process to create the role again.

Editing a service-linked role for User Subscriptions

User Subscriptions does not allow you to edit the AWSServiceRoleForUserSubscriptions service-linked role. After you create a service-linked role, you cannot change the name of the role because various entities might reference the role. However, you can edit the description of the role using IAM. For more information, see Editing a service-linked role in the *IAM User Guide*.

Deleting a service-linked role for User Subscriptions

If you no longer need to use a feature or service that requires a service-linked role, we recommend that you delete that role. That way you don’t have an unused entity that is not actively monitored or maintained. However, you must clean up the resources for your service-linked role before you can manually delete it.

> **Note**
>
> If the User Subscriptions service is using the role when you try to delete the resources, then the deletion might fail. If that happens, wait for a few minutes and try the operation again.

To manually delete the service-linked role using IAM

Use the IAM console, the AWS CLI, or the AWS API to delete the AWSServiceRoleForUserSubscriptions service-linked role. For more information, see Deleting a service-linked role in the *IAM User Guide*.

Supported Regions for User Subscriptions service-linked roles

Amazon Q Developer Subscriptions supports using service-linked roles in all of the Regions where the service is available. For more information, see AWS Regions and endpoints.

Amazon Q Developer Subscriptions does not support using service-linked roles in every Region where the service is available. You can use the AWSServiceRoleForUserSubscriptions role in the following Regions.

Region name	Region identity	Support in User Subscriptions
US East (N. Virginia)	eusc-de-east-1	Yes
US West (Oregon)	us-west-2	Yes
US East (N. Virginia)	us-east-1	Yes
US East (Ohio)	us-east-2	Yes
US East (Ohio)	us-east-2	Yes
US West (N. California)	us-west-1	Yes
Asia Pacific (Mumbai)	ap-south-1	Yes
Asia Pacific (Osaka)	ap-northeast-3	Yes
Asia Pacific (Seoul)	ap-northeast-2	Yes
Asia Pacific (Singapore)	ap-southeast-1	Yes
Asia Pacific (Sydney)	ap-southeast-2	Yes
Asia Pacific (Tokyo)	ap-northeast-1	Yes
Canada (Central)	ca-central-1	Yes

Region name	Region identity	Support in User Subscriptions
Europe (Frankfurt)	eu-central-1	Yes
Europe (Ireland)	eu-west-1	Yes
Europe (London)	eu-west-2	Yes
Europe (Paris)	eu-west-3	Yes
Europe (Stockholm)	eu-north-1	Yes
South America (São Paulo)	sa-east-1	Yes

Compliance validation for Amazon Q Developer

To learn whether an AWS service is within the scope of specific compliance programs, see AWS services in Scope by Compliance Program and choose the compliance program that you are interested in. For general information, see AWS Compliance Programs.

You can download third-party audit reports using AWS Artifact. For more information, see Downloading Reports in AWS Artifact.

Your compliance responsibility when using AWS services is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. For more information about your compliance responsibility when using AWS services, see AWS Security Documentation.

Resilience in Amazon Q Developer

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

For more information about AWS Regions and Availability Zones, see AWS Global Infrastructure.

Infrastructure security in Amazon Q Developer

As a managed service, Amazon Q is protected by AWS global network security. For information about AWS security services and how AWS protects infrastructure, see AWS Cloud Security. To design your AWS environment using the best practices for infrastructure security, see Infrastructure Protection in *Security Pillar AWS Well-Architected Framework*.

You use AWS published API calls to access Amazon Q Developer through the network. Clients must support the following:

- Transport Layer Security (TLS). We require TLS 1.2 and recommend TLS 1.3.
- Cipher suites with perfect forward secrecy (PFS) such as DHE (Ephemeral Diffie-Hellman) or ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Most modern systems such as Java 7 and later support these modes.

Configuring a firewall, proxy server, or data perimeter for Amazon Q Developer

If you're using a firewall, proxy server, or data perimeter, make sure to allowlist traffic to the following URLs and Amazon Resource Names (ARNs) so that Amazon Q works as expected.

General URLs to allowlist

In the following URLs, replace:

- *`idc-directory-id-or-alias`* with your IAM Identity Center instance's directory ID or alias. For more information about IAM Identity Center, see What is IAM Identity Center? in the *AWS IAM Identity Center User Guide*.
- *`sso-region`* with the AWS Region where your IAM Identity Center instance is enabled. For more information, see IAM Identity Center Regions supported by Amazon Q Developer.
- *`profile-region`* with the AWS Region where your Amazon Q Developer profile is installed. For more information about the Amazon Q Developer profile, see What is the Amazon Q Developer profile? and Supported Regions for the Q Developer console and Q Developer profile.

URL	Purpose
idc-directory-id-or-alias.awsapps.com	Authentication
oidc.*sso-region* .amazonaws.com	Authentication
*.sso.*sso-region* .amazonaws.com	Authentication
*.sso-portal. *sso-region* .amazonaws.com	Authentication
*.aws.dev	Authentication
*.awsstatic.com	Authentication
*.console.aws.a2z.com	Authentication
*.sso.amazonaws.com	Authentication
https://codewhisperer.us-east-1.amazonaws.com	Amazon Q Developer features
https://q. *profile-region* .amazonaws.com	Amazon Q Developer features
https://idetoolkits-hostedfiles.amazonaws.com/*	Amazon Q Developer in the IDE, configuration
https://idetoolkits.amazonwebservices.com/*	Amazon Q Developer in the IDE, endpoints
q-developer-integration.us-east-1.api.aws	Amazon Q Developer in the IDE, endpoints
https://aws-toolkit-language-servers.amazonaws.com/*	Amazon Q Developer in the IDE, language processing
https://aws-language-servers.us-east-1.amazonaws.com/*	Amazon Q Developer in the IDE, language processing

URL	Purpose
`https://client-telemetry.us-east-1.amazonaws.com`	Amazon Q Developer in the IDE, telemetry
`cognito-identity.us-east-1.amazonaws.com`	Amazon Q Developer in the IDE, telemetry

Amazon S3 bucket URLs and ARNs to allowlist

For some features, Amazon Q uploads artifacts to AWS service-owned Amazon S3 buckets. If you are using data perimeters to control access to Amazon S3 in your environment, you might need to explicitly allow access to these buckets to use the corresponding Amazon Q features.

The following table lists the URL and ARN of each of the Amazon S3 buckets that Amazon Q requires access to, and the features that use each bucket. You can use the bucket URL or bucket ARN to allowlist these buckets, depending on how you control access to Amazon S3.

You only need to allowlist the bucket in the AWS Region where the Amazon Q Developer profile is installed. For more information about the Amazon Q Developer profile, see What is the Amazon Q Developer profile?.

Amazon S3 bucket URL and ARN	Purpose
US East (N. Virginia): • `https://amazonq-code-scan-us-east-1-29121b44f7b.s3.amazonaws.com/` • `arn:aws:s3:::amazonq-code-scan-us-east-1-29121b44f7b` Europe (Frankfurt): • `https://amazonq-code-scan-eu-central-1-9374e402cc5.s3.amazonaws.com/`	An Amazon S3 bucket used to upload artifacts for Amazon Q code reviews

Amazon S3 bucket URL and ARN	Purpose
• `arn:aws:s3:::amazonq-code-scan-eu-central-1-9374e402cc5`	
US East (N. Virginia): • `https://amazonq-code-transformation-us-east-1-c6160f047e0.s3.amazonaws.com/` • `arn:aws:s3:::amazonq-code-transformation-us-east-1-c6160f047e0` Europe (Frankfurt): • `https://amazonq-code-transformation-eu-central-1-a0a89cc2b94.s3.amazonaws.com/` • `arn:aws:s3:::amazonq-code-transformation-eu-central-1-a0a89cc2b94`	An Amazon S3 bucket used to upload artifacts for Amazon Q code transformations

Configuring a corporate proxy in Amazon Q

If your end users are working behind a corporate proxy, have them complete the following steps to successfully connect to Amazon Q.

Step 1: Configuring proxy settings in your IDE

Specify your proxy server's URL in your IDE.

> **Note**
>
> SOCKS proxies and Proxy Auto-Configuration (PAC) files are not supported. You must manually configure an HTTP or HTTPS proxy using the instructions that follow.

Eclipse

1. In Eclipse, open **Preferences** as follows:
   - On Windows or Ubuntu:
     - From the Eclipse menu bar, choose **Window**, and then choose **Preferences**.
   - On macOS:
     - From the menu bar, choose **Eclipse**, and then choose **Settings** or **Preferences** depending on your macOS version.
2. In the search bar, enter **`Amazon Q`** and open Amazon Q.
3. Under **Proxy Settings**, set **HTTPS Proxy URL** to your corporate proxy URL.

   Examples: `http://proxy.company.com:8080`, `https://proxy.company.com:8443`
4. Leave the Amazon Q settings open and go to the next step.

JetBrains

In JetBrains, manually configure your proxy server's host name and port following the guidance in the HTTP Proxy topic of the IntelliJ IDEA documentation.

Visual Studio

1. From the Visual Studio main menu, choose **Tools**, then choose **Options**.
2. From the **Options** menu, expand **AWS Toolkit**, then choose **Proxy**.
3. From the **Proxy** menu, set **Host** and **Port** to your corporate proxy host and port.

   Examples: `http://proxy.company.com:8080`, `https://proxy.company.com:8443`

Visual Studio Code

1. From VS Code, open VS Code **Settings** by pressing **`CMD + ,`** (Mac) or **`Ctrl + ,`** (Windows/Linux).
2. From the **Settings** search bar, enter **`Http: Proxy`**, then locate it in the search results.
3. Enter your corporate proxy URL.

   Examples: `http://proxy.company.com:8080`, `https://proxy.company.com:8443`
4. (Optional) in the **Settings** search bar, enter **`HTTP: No Proxy`**, then locate it in the results.

5. Choose the **Add Item** button, then add domains that bypass the proxy, separated by commas.

Step 2: Configuring SSL certificate handling

Amazon Q automatically detects and uses the trusted certificates installed on your system. If you are experiencing certificate errors, you must manually specify a certificate bundle by completing the following procedure.

> **Note**
>
> The following are situations where manual configuration is required.
>
> - You are encountering certificate-related errors after configuring the proxy.
> - Your corporate proxy uses certificates that aren't in your system's trust store.
> - Amazon Q fails to automatically detect your corporate certificates.

Eclipse

- In the Amazon Q settings in Eclipse, under **Proxy Settings**, set **CA Cert PEM** to the path of your corporate certificate file. The file must have a `.pem` file extension. (You cannot use a `.crt` file.)

  An example path resembles the following:

  `/path/to/corporate-ca-bundle.pem`

  For instructions on obtaining this file, see Getting your corporate certificate.

JetBrains

In JetBrains, manually install your corporate certificate following the guidance in the Trusted root certificates topic of the IntelliJ IDEA documentation.

For instructions on obtaining the certificate, see Getting your corporate certificate.

Visual Studio

- Configure the following Windows Environment Variables:

- `NODE_OPTIONS = --use-openssl-ca`
- `NODE_EXTRA_CA_CERTS = cert-path`

Replace *`cert-path`* with the path of your corporate certificate file. The file must have a `.pem` file extension. (You cannot use a `.crt` file.)

An example path resembles the following:

```
/path/to/corporate-ca-bundle.pem
```

For instructions on obtaining the corporate certificate file, see Getting your corporate certificate. For detailed information about the Windows Environment Variables, see the Node.js documentation.

Visual Studio Code

1. From VS Code, open VS Code **Settings** by pressing **`CMD + ,`** (Mac) or **`Ctrl + ,`** (Windows/Linux).
2. From the **Settings** search bar, enter **`Amazon Q › Proxy: Certificate Authority`**, then locate it in the search results.
3. Enter the path of your corporate certificate file. It will have a `.pem` or `.crt` file extension.

   An example path resembles the following:

   ```
   /path/to/corporate-ca-bundle.pem
   ```

   For instructions on obtaining this file, see Getting your corporate certificate.

Step 3: Restart your IDE

You must restart your IDE in order to update Amazon Q with your changes.

Troubleshooting

If you completed the procedures in the previous sections and you are still experiencing issues, use the following instructions to troubleshoot.

Eclipse

1. Make sure your proxy URL format includes `http://` or `https://`.
2. Make sure your certificate file path is correct and accessible.
3. Review your Amazon Q logs in the Eclipse Error Log. To navigate to the Error Log, do one of the following:

   - Sign in to Amazon Q in Eclipse, choose the down-arrow next to the Q gear icon, and then choose **Help**, **View Logs**.
   - From the Eclipse menu, choose **Window**, **Show View**, **Error Log**.

   Note

   If you encounter the following error messages:

   - `unable to verify the first certificate`, make sure you followed the instructions in Step 2: Configuring SSL certificate handling to specify a certificate manually.
   - `self signed certificate`, make sure you followed the instructions in Step 2: Configuring SSL certificate handling to specify a certificate manually.
   - `ECONNREFUSED`, check your internet connection and proxy information.

JetBrains

1. Make sure your proxy URL format includes `http://` or `https://`.
2. Make sure your certificate file path is correct and accessible.
3. Review your Amazon Q logs in the JetBrains log file. For help on finding the log file, see Locating IDE log files on the JetBrains IDEs Support page.

   Note

   If you encounter the following error messages:

   - `unable to verify the first certificate`, make sure you followed the instructions in Step 2: Configuring SSL certificate handling to specify a certificate manually.

- `self signed certificate`, make sure you followed the instructions in Step 2: Configuring SSL certificate handling to specify a certificate manually.
- `ECONNREFUSED`, check your internet connection and proxy information.

Visual Studio

1. Make sure your proxy URL format includes `http://` or `https://`.
2. Make sure your certificate file path is correct and accessible.
3. Review the AWS Toolkit extension's logs as follows:
   - From the Visual Studio main menu, expand **Extensions**.
   - Choose **AWS Toolkit** to expand the AWS Toolkit menu, then choose **View Toolkit Logs**.
   - When the AWS Toolkit logs folder opens in your Operating System, sort the files by date and locate any log file that contains information relevant to your current issue.
4. Review your Amazon Q logs in the Visual Studio Activity Log. For more information, see Troubleshooting Extensions with the Activity Log.

> **Note**
>
> If you encounter the following error messages:
>
> - `unable to verify the first certificate`, make sure you followed the instructions in Step 2: Configuring SSL certificate handling to specify a certificate manually.
> - `self signed certificate`, make sure you followed the instructions in Step 2: Configuring SSL certificate handling to specify a certificate manually.
> - `ECONNREFUSED`, check your internet connection and proxy information.

Visual Studio Code

1. Make sure your proxy URL format includes `http://` or `https://`.
2. Make sure your certificate file path is correct and accessible.
3. Review your Amazon Q logs in the **Output panel** of VS Code.

> **Note**
>
> If you encounter the following error messages:
>
> - `unable to verify the first certificate`, make sure you followed the instructions in Step 2: Configuring SSL certificate handling to specify a certificate manually.
> - `self signed certificate`, make sure you followed the instructions in Step 2: Configuring SSL certificate handling to specify a certificate manually.
> - `ECONNREFUSED`, check your internet connection and proxy information.

Getting your corporate certificate

To obtain your corporate certificate, ask your IT team for the following information:

- Your corporate certificate bundle, which is typically a `.pem` or `.crt` file.
- Your proxy server details, including your host name, port, and authentication method.

Or, export the certificate from your browser:

1. Visit any HTTPS site on your corporate domain.
2. Near the address bar, choose the lock icon or a similar icon. (The icon differs depending your browser vendor.)
3. Export the root certificate to a file. Make sure you include the whole certificate chain. The steps to export the root certificate will be slightly different depending on the browser you're using. Consult your browser's documentation for detailed steps.

Amazon Q Developer and interface endpoints (AWS PrivateLink)

> **Note**
>
> Amazon Q Developer supports interface endpoints for features available in your IDE. Chatting with Amazon Q on AWS apps and websites is not supported for VPC endpoints. Neither is the Amazon Q Developer transformation web experience.

You can establish a private connection between your VPC and Amazon Q Developer by creating an *interface VPC endpoint*. Interface endpoints are powered by AWS PrivateLink, a technology that enables you to privately access Amazon Q APIs without an internet gateway, NAT device, VPN connection, or AWS Direct Connect connection. Instances in your VPC don't need public IP addresses to communicate with Amazon Q APIs. Traffic between your VPC and Amazon Q does not leave the Amazon network.

Each interface endpoint is represented by one or more Elastic Network Interfaces in your subnets.

For more information, see Interface VPC endpoints (AWS PrivateLink) in the *Amazon VPC User Guide*.

Considerations for Amazon Q VPC endpoints

Before you set up an interface VPC endpoint for Amazon Q, ensure that you review Interface endpoint properties and limitations in the *Amazon VPC User Guide*.

Amazon Q supports making calls to all of its API actions from your VPC, in the context of services that are configured to work with Amazon Q.

Prerequisites

Before you begin any of the procedures below, ensure that you have the following:

- An AWS account with appropriate permissions to create and configure resources.
- A VPC already created in your AWS account.
- Familiarity with AWS services, especially Amazon VPC and Amazon Q.

Creating an interface VPC endpoint for Amazon Q

You can create a VPC endpoint for the Amazon Q service using either the Amazon VPC console or the AWS Command Line Interface (AWS CLI). For more information, see Creating an interface endpoint in the *Amazon VPC User Guide*.

Create the following VPC endpoints for Amazon Q using the following service names:

- com.amazonaws.*region*.q
- com.amazonaws.us-east-1.codewhisperer

Replace *region* with AWS Region where your Amazon Q Developer profile is installed. For more information, see Supported Regions for the Q Developer console and Q Developer profile.

> **Note**
>
> The Amazon CodeWhisperer endpoint (com.amazonaws.us-east-1.codewhisperer) is only supported in the US East (N. Virginia) Region.

If you enable private DNS for the endpoint, you can make API requests to Amazon Q using its default DNS name for the Region, for example, `q.us-east-1.amazonaws.com`.

For more information, see Accessing a service through an interface endpoint in the *Amazon VPC User Guide*.

Using an on-premises computer to connect to a Amazon Q endpoint

This section describes the process of using an on-premises computer to connect to Amazon Q through a AWS PrivateLink endpoint in your AWS VPC.

1. Create a VPN connection between your on-premises device and your VPC.
2. Create an interface VPC endpoint for Amazon Q.
3. Set up an inbound Amazon Route 53 endpoint. This will enable you to use the DNS name of your Amazon Q endpoint from your on-premises device.

Using an in-console coding environment to connect to a Amazon Q endpoint

This section describes the process of using an in-console coding environment to connect to a Amazon Q endpoint.

In this context, an in-console IDE is an IDE that you access inside the AWS console, and authenticate to with IAM. Examples include SageMaker AI Studio and AWS Glue Studio.

1. Create an interface VPC endpoint for Amazon Q.
2. Set up Amazon Q with the in-console coding environment
   - SageMaker AI Studio
   - AWS Glue Studio
3. Configure the coding environment to use the Amazon Q endpoint.
   - SageMaker AI Studio
   - AWS Glue Studio

Connecting to Amazon Q through AWS PrivateLink from a third-Party IDE on an Amazon EC2 instance

This section will walk you through the process of installing a third-party Integrated Development Environment (IDE) like Visual Studio Code or JetBrains on an Amazon EC2 instance, and configuring it to connect to Amazon Q using AWS PrivateLink.

1. Create an interface VPC endpoint for Amazon Q.
2. Launch an Amazon EC2 instance in your desired subnet within your VPC. You can choose an Amazon Machine Image (AMI) that is compatible with your third-party IDE. For example, you can select an Amazon Linux 2 AMI.
3. Connect to the Amazon EC2 instance.
4. Install and Configure the IDE (Visual Studio Code or JetBrains).
5. Install the Amazon Q extension or plugin.
6. Configure the IDE to connect via AWS PrivateLink.
   - Network connections in Visual Studio Code

- JetBrains remote development

Monitoring and tracking the use of Amazon Q Developer

Monitoring is an important part of maintaining the reliability, availability, and performance of Amazon Q Developer and your other AWS solutions. AWS provides the following monitoring tools and features to monitor and record Amazon Q Developer activity:

- *AWS CloudTrail* captures API calls and related events made by or on behalf of your AWS account and delivers the log files to an Amazon Simple Storage Service (Amazon S3) bucket that you specify. You can identify which users and accounts called AWS, the source IP address from which the calls were made, and when the calls occurred. For more information, see Logging Amazon Q Developer API calls using AWS CloudTrail.
- *Amazon CloudWatch* monitors your AWS resources and the applications you run on AWS in real time. You can collect and track metrics, create customized dashboards, and set alarms that notify you or take actions when a specified metric reaches a threshold that you specify. For example, you can have CloudWatch track the number of times that Amazon Q has been invoked on your account, or the number of daily active users. For more information, see Monitoring Amazon Q Developer with Amazon CloudWatch.

Amazon Q Developer also includes the following features to help you track and record user activity in Amazon Q:

- *A dashboard* shows you aggregate user activity metrics of Amazon Q Developer Pro subscribers. For more information, see Viewing Amazon Q Developer user activity on the dashboard.
- *User activity reports* show you what individual users are up to in Amazon Q. For more information, see Viewing the activity of specific users in Amazon Q Developer.
- *Prompt logs* provide you with a record of all the prompts that users enter into the Amazon Q chat in their integrated development environment (IDE). For more information, see Logging users' prompts in Amazon Q Developer.

Logging Amazon Q Developer API calls using AWS CloudTrail

Amazon Q Developer Pro is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service in Amazon Q. CloudTrail captures all API calls for Amazon Q as events. The calls captured include calls from the Amazon Q console and code calls to the Amazon Q API operations. If you create a trail, you can enable continuous delivery of CloudTrail

events to an Amazon S3 bucket, including events for Amazon Q. If you don't configure a trail, you can still view the most recent events in the CloudTrail console in **Event history**. Using the information collected by CloudTrail, you can determine the request that was made to Amazon Q, the IP address from which the request was made, who made the request, when it was made, and additional details.

For more information about CloudTrail, see the AWS CloudTrail User Guide.

Amazon Q Developer information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in Amazon Q Developer, that activity is recorded in a CloudTrail event along with other AWS service events in **Event history**. You can view, search, and download recent events in your AWS account. For more information, see Viewing Events with CloudTrail Event History in the *AWS CloudTrail User Guide*.

For an ongoing record of events in your AWS account, including events for Amazon Q, create a trail. A *trail* enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create a trail in the console, the trail applies to all AWS Regions. The trail logs events from all Regions in the AWS partition and delivers the log files to the Amazon S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see the following topics in the *AWS CloudTrail User Guide*:

- Overview for creating a trail
- CloudTrail supported services and integrations
- Configuring Amazon SNS notifications for CloudTrail
- Receiving CloudTrail log files from multiple Regions
- Receiving CloudTrail log files from multiple accounts

All Amazon Q Developer actions are logged by CloudTrail and generate entries in the CloudTrail log files.

Every event or log entry contains information about who generated the request. The identity information helps you determine the following:

- Whether the request was made with root or AWS Identity and Access Management (IAM) user credentials
- Whether the request was made with temporary security credentials for a role or federated user

- Whether the request was made by another AWS service

For more information, see CloudTrail userIdentity element in the *AWS CloudTrail User Guide*.

Understanding Amazon Q Developer log file entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An event represents a single request from any source and includes information about the requested action, the date and time of the action, request parameters, and so on. CloudTrail log files aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.

Amazon Q Developer also makes API calls with a `dryRun` parameter to verify that you have the necessary permissions for the action, without actually making the request. Calls to Amazon Q Developer APIs with the `dryRun` parameter are captured as events and recorded in a CloudTrail log with `"dryRun" : true` in the `requestParameters` field.

The following example shows a CloudTrail log entry that demonstrates the `SendMessage` action.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAXD12ABCDEF3G4HI5J:aws-user",
        "arn": "arn:aws:sts::123456789012:assumed-role/PowerUser/aws-user",
        "accountId": "123456789012",
        "accessKeyId": "ASIAAB12CDEFG34HIJK",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAXD12ABCDEF3G4HI5J",
                "arn": "arn:aws:iam::123456789012:role/PowerUser",
                "accountId": "123456789012",
                "userName": "PowerUser"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2023-11-28T10:00:00Z",
                "mfaAuthenticated": "false"
            }
        }
    },
```

```
    "eventTime": "2023-11-28T10:00:00Z",
    "eventSource": "q.amazonaws.com",
    "eventName": "SendMessage",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "123.456.789.012",
    "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101
 Firefox/115.0",
    "requestParameters": {
        "Origin": "https://conversational-experience-
worker.widget.console.aws.amazon.com",
        "conversationId": "a298ec0d-0a49-4d2e-92bd-7d6e629b4619",
        "source": "CONSOLE",
        "conversationToken": "***",
        "utterance": "***"
    },
    "responseElements": {
        "result": {
            "content": {
                "text": {
                    "body": "***",
                    "references": []
                }
            },
            "format": "PLAINTEXT",
            "intents": {},
            "type": "TEXT"
        },
        "Access-Control-Expose-Headers": "x-amzn-RequestId,x-amzn-ErrorType,x-amzn-
ErrorMessage,Date",
        "metadata": {
            "conversationExpirationTime": "2024-02-25T19:31:38Z",
            "conversationId": "a298ec0d-0a49-4d2e-92bd-7d6e629b4619",
            "conversationToken": "***",
            "utteranceId": "3b87b46f-04a9-41ef-b8fe-8abf52d2c053"
        },
        "resultCode": "LLM"
    },
    "additionalEventData": {
        "quickAction": "dev"
    },
    "requestID": "19b3c30e-906e-4b7f-b5c3-509f67248655",
    "eventID": "a552c487-7d97-403a-8ec4-d49539c7a03d",
    "readOnly": false,
    "eventType": "AwsApiCall",
```

```
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management"
}
```

The following example shows a CloudTrail log entry that demonstrates the `PassRequest` action.

```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDA6ON6E4XEGIEXAMPLE",
        "arn": "arn:aws:iam::555555555555:user/Mary",
        "accountId": "555555555555",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
            "type": "Role",
            "principalId": "AIDA6ON6E4XEGIEXAMPLE",
            "arn": "arn:aws:iam::555555555555:user/Mary",
            "accountId": "555555555555",
            "userName": "Mary"

        },
        "attributes": {
            "creationDate": "2024-04-10T20:03:01Z",
            "mfaAuthenticated": "false"
        },
        "invokedBy": "q.amazonaws.com"
    },
    "eventTime": "2024-04-10T20:04:42Z",
    "eventSource": "q.amazonaws.com",
    "eventName": "PassRequest",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "q.amazonaws.com",
    "userAgent": "q.amazonaws.com",
    "requestParameters": null,
    "responseElements": null,
    "requestID": "2d528c76-329e-410b-9516-EXAMPLE565dc",
    "eventID": "ba0801a1-87ec-4d26-be87-EXAMPLE75bbb",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
```

```
    "recipientAccountId": "555555555555",
    "eventCategory": "Management"
}
```

The following example shows a CloudTrail log entry that demonstrates Amazon Q calling the `s3:ListBuckets` action on your behalf.

```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDA6ON6E4XEGIEXAMPLE",
        "arn": "arn:aws:iam::555555555555:user/Paulo",
        "accountId": "555555555555",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDA6ON6E4XEGIEXAMPLE",
                "arn": "arn:aws:iam::555555555555:user/Paulo",
                "accountId": "555555555555",
                "userName": "Paulo"
            },
            "attributes": {
                "creationDate": "2024-04-10T14:06:08Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "q.amazonaws.com"
    },
    "eventTime": "2024-04-10T14:07:55Z",
    "eventSource": "s3.amazonaws.com",
    "eventName": "ListBuckets",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "q.amazonaws.com",
    "userAgent": "q.amazonaws.com",
    "requestParameters": {
        "Host": "s3.amazonaws.com"
    },
    "responseElements": null,
    "additionalEventData": {
        "SignatureVersion": "SigV4",
        "CipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
```

```
        "bytesTransferredIn": 0,
        "AuthenticationMethod": "AuthHeader",
        "x-amz-id-2": "ExampleRequestId123456789",
        "bytesTransferredOut": 4054
    },
    "requestID": "ecd94349-b36f-44bf-b6f5-EXAMPLE9c463",
    "eventID": "2939ba50-1d26-4a5a-83bd-EXAMPLE85850",
    "readOnly": true,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "555555555555",
    "vpcEndpointId": "vpce-EXAMPLE1234",
    "eventCategory": "Management"
}
```


Monitoring Amazon Q Developer with Amazon CloudWatch

> **Note**
>
> The metrics discussed here only pertain to using Amazon Q in your IDE.

You can monitor Amazon Q Developer using CloudWatch, which collects raw data and processes it into readable, near real-time metrics. These statistics are kept for 15 months, so that you can access historical information and gain a better perspective on how Amazon Q is performing. You can also set alarms that watch for certain thresholds, and send notifications or take actions when those thresholds are met. For more information, see the Amazon CloudWatch User Guide.

The Amazon Q Developer service reports the following metrics in the `AWS/Q` namespace.

Dimensions	Metric	Use case or explanation
Count	Invocations	You want to determine how many invocations have been counted over time.
UserCount	DailyActiveUserTrend	You want to determine the number of active users per day.

Dimensions	Metric	Use case or explanation
SubscriptionUserCount	SubscriptionCount	You want to determine the number of users with paying subscriptions.
UniqueUserCount	MonthlyActiveUniqueUsers	You want to determine the number of users who are active in a given month.
ProgrammingLanguage, SuggestionState, CompletionType	GeneratedLineCount	You want to determine the number of lines generated by Amazon Q Developer.
ProgrammingLanguage, SuggestionState, CompletionType	SuggestionReferenceCount	You want to determine the number of recommendation triggers with references that have taken place.
ProgrammingLanguage	CodeScanCount	You want to determine the number of code scans that have taken place.
ProgrammingLanguage	TotalCharacterCount	The number of characters in your file, including all suggestions from Amazon Q Developer.
ProgrammingLanguage	CodeWhispererCharacterCount	The number of characters generated by Amazon Q Developer.

To aggregate Invocations, use the Sum statistic.

To aggregate DailyActiveUserTrend, use the Sum statistic, and use "1 Day" as the period.

To aggregate SubscriptionCount, use the Sum statistic.

To aggregate MonthlyActiveUniqueUsers use the Sum statistic, and use "30 Days" as the period.

Identifying actions by specific users with Amazon CloudWatch Logs

It's possible to get user-level metrics on your Amazon Q Developer usage. To figure out which user has taken a particular action, look for the events called SendTelemetryEvent, and examine the JSON object of type SendTelemetryEventRequest that they contain. Within that object, the schema appears as follows.

Tip

You can also output individual users' activity in Amazon Q Developer to a report in CSV format. For more information, see Viewing the activity of specific users in Amazon Q Developer.

```
http://json-schema.org/draft-07/schema#",
    "definitions": {
        "ProgrammingLanguage": {
            "type": "object",
            "properties": {
                "languageName": {
                    "type": "string",
                    "enum": [
                        "python",
                        "javascript",
                        "java",
                        "csharp",
                        "typescript",
                        "c",
                        "cpp",
                        "go",
                        "kotlin",
                        "php",
                        "ruby",
                        "rust",
```

```
                        "scala",
                        "shell",
                        "sql",
                        "json",
                        "yaml",
                        "vue",
                        "tf",
                        "tsx",
                        "jsx",
                        "plaintext"
                    ],
                    "description": "Programming Languages supported by Q"
                }
            }
        },
        "Dimension": {
            "type": "object",
            "properties": {
                "name": {
                    "type": "string",
                    "description": "must match ^[-a-zA-Z0-9._]*$ and be between 1 and
255 characters"
                },
                "value": {
                    "type": "string",
                    "description": "must match ^[-a-zA-Z0-9._]*$ and be between 1 and
1024 characters"
                }
            }
        }
    },
    "telemetryEvents": {
        "UserTriggerDecisionEvent": {
            "type": "object",
            "properties": {
                "sessionId": {
                    "type": "string",
                    "description": "UUID for the session"
                },
                "requestId": {
                    "type": "string",
                    "description": "UUID for the request"
                },
                "customizationArn": {
```

```
                    "type": "string",
                    "description": "ARN of the customization matching pattern: ^arn:
[-.a-z0-9]{1,63}:codewhisperer:([-.a-z0-9]{0,63}:){2}([a-zA-Z0-9-_:/]){1,1023}$"
                },
                "programmingLanguage": {
                    "$ref": "#/definitions/ProgrammingLanguage"
                },
                "completionType": {
                    "type": "string",
                    "enum": [
                        "BLOCK",
                        "LINE"
                    ]
                },
                "suggestionState": {
                    "type": "string",
                    "enum": [
                        "ACCEPT",
                        "REJECT",
                        "DISCARD",
                        "EMPTY"
                    ]
                },
                "recommendationLatencyMilliseconds": {
                    "type": "number"
                },
                "timestamp": {
                    "type": "string",
                    "description": "datetime, example: Jul 23, 2024, 12:11:02 AM"
                },
                "triggerToResponseLatencyMilliseconds": {
                    "type": "number"
                },
                "suggestionReferenceCount": {
                    "type": "integer"
                },
                "generatedLine": {
                    "type": "integer"
                },
                "numberOfRecommendations": {
                    "type": "integer"
                }
            },
            "required": [
```

```
                "sessionId",
                "requestId",
                "programmingLanguage",
                "completionType",
                "suggestionState",
                "recommendationLatencyMilliseconds",
                "timestamp"
            ]
        },
        "CodeCoverageEvent": {
            "type": "object",
            "properties": {
                "customizationArn": {
                    "type": "string",
                    "description": "ARN of the customization matching pattern: ^arn:
[-.a-z0-9]{1,63}:codewhisperer:([-.a-z0-9]{0,63}:){2}([a-zA-Z0-9-_:/]){1,1023}$"
                },
                "programmingLanguage": {
                    "$ref": "#/definitions/ProgrammingLanguage"
                },
                "acceptedCharacterCount": {
                    "type": "integer"
                },
                "totalCharacterCount": {
                    "type": "integer"
                },
                "timestamp": {
                    "type": "string",
                    "description": "datetime, example: Jul 23, 2024, 12:11:02 AM"
                },
                "unmodifiedAcceptedCharacterCount": {
                    "type": "integer"
                }
            },
            "required": [
                "programmingLanguage",
                "acceptedCharacterCount",
                "totalCharacterCount",
                "timestamp"
            ]
        },
        "UserModificationEvent": {
            "type": "object",
            "properties": {
```

```
                "sessionId": {
                    "type": "string",
                    "description": "UUID for the session"
                },
                "requestId": {
                    "type": "string",
                    "description": "UUID for the request"
                },
                "programmingLanguage": {
                    "$ref": "#/definitions/ProgrammingLanguage"
                },
                "modificationPercentage": {
                    "type": "number",
                    "description": "This is the percentage of AI generated code which
 has been modified by the user"
                },
                "customizationArn": {
                    "type": "string",
                    "description": "ARN of the customization matching pattern: ^arn:
[-.a-z0-9]{1,63}:codewhisperer:([-.a-z0-9]{0,63}:){2}([a-zA-Z0-9-_:/]){1,1023}$"
                },
                "timestamp": {
                    "type": "string",
                    "description": "datetime, example: Jul 23, 2024, 12:11:02 AM"
                }
            },
            "required": [
                "sessionId",
                "requestId",
                "programmingLanguage",
                "modificationPercentage",
                "timestamp"
            ]
        },
        "CodeScanEvent": {
            "type": "object",
            "properties": {
                "programmingLanguage": {
                    "$ref": "#/definitions/ProgrammingLanguage"
                },
                "codeScanJobId": {
                    "type": "string"
                },
                "timestamp": {
```

```
                    "type": "string",
                    "description": "datetime, example: Jul 23, 2024, 12:11:02 AM"
                },
                "codeAnalysisScope": {
                    "type": "string",
                    "enum": [
                        "FILE",
                        "PROJECT"
                    ]
                }
            },
            "required": [
                "programmingLanguage",
                "codeScanJobId",
                "timestamp"
            ]
        },
        "CodeScanRemediationsEvent": {
            "type": "object",
            "properties": {
                "programmingLanguage": {
                    "$ref": "#/definitions/ProgrammingLanguage"
                },
                "CodeScanRemediationsEventType": {
                    "type": "string",
                    "enum": [
                        "CODESCAN_ISSUE_HOVER",
                        "CODESCAN_ISSUE_APPLY_FIX",
                        "CODESCAN_ISSUE_VIEW_DETAILS"
                    ]
                },
                "timestamp": {
                    "type": "string",
                    "description": "datetime, example: Jul 23, 2024, 12:11:02 AM"
                },
                "detectorId": {
                    "type": "string"
                },
                "findingId": {
                    "type": "string"
                },
                "ruleId": {
                    "type": "string"
                },
```

```
                "component": {
                    "type": "string"
                },
                "reason": {
                    "type": "string"
                },
                "result": {
                    "type": "string"
                },
                "includesFix": {
                    "type": "boolean"
                }
            }
        },
        "MetricData": {
            "type": "object",
            "properties": {
                "metricName": {
                    "type": "string",
                    "description": "must match pattern ^[-a-zA-Z0-9._]*$ and be between
1 and 1024 characters"
                },
                "metricValue": {
                    "type": "number"
                },
                "timestamp": {
                    "type": "string",
                    "description": "datetime, example: Jul 23, 2024, 12:11:02 AM"
                },
                "product": {
                    "type": "string",
                    "description": "must match pattern ^[-a-zA-Z0-9._]*$ and be between
1 and 128 characters"
                },
                "dimensions": {
                    "type": "array",
                    "items": {
                        "$ref": "#/definitions/Dimension"
                    },
                    "description": "maximum size of 30"
                }
            },
            "required": [
                "metricName",
```

```
                "metricValue",
                "timestamp",
                "product"
            ]
        },
        "ChatAddMessageEvent": {
            "type": "object",
            "properties": {
                "conversationId": {
                    "type": "string",
                    "description": "ID which represents a multi-turn conversation,
 length between 1 and 128"
                },
                "messageId": {
                    "type": "string",
                    "description": "Unique identifier for the chat message"
                },
                "customizationArn": {
                    "type": "string",
                    "description": "ARN of the customization matching pattern: ^arn:
[-.a-z0-9]{1,63}:codewhisperer:([-.a-z0-9]{0,63}:){2}([a-zA-Z0-9-_:/]){1,1023}$"
                },
                "userIntent": {
                    "type": "string",
                    "enum": [
                        "SUGGEST_ALTERNATE_IMPLEMENTATION",
                        "APPLY_COMMON_BEST_PRACTICES",
                        "IMPROVE_CODE",
                        "SHOW_EXAMPLES",
                        "CITE_SOURCES",
                        "EXPLAIN_LINE_BY_LINE",
                        "EXPLAIN_CODE_SELECTION",
                        "GENERATE_CLOUDFORMATION_TEMPLATE"
                    ]
                },
                "hasCodeSnippet": {
                    "type": "boolean"
                },
                "programmingLanguage": {
                    "$ref": "#/definitions/ProgrammingLanguage"
                },
                "activeEditorTotalCharacters": {
                    "type": "integer"
                },
```

```
                "timeToFirstChunkMilliseconds": {
                    "type": "number"
                },
                "timeBetweenChunks": {
                    "type": "array",
                    "items": {
                        "type": "number"
                    },
                    "description": "maximum size of 100"
                },
                "fullResponselatency": {
                    "type": "number"
                },
                "requestLength": {
                    "type": "integer"
                },
                "responseLength": {
                    "type": "integer"
                },
                "numberOfCodeBlocks": {
                    "type": "integer"
                },
                "hasProjectLevelContext": {
                    "type": "boolean"
                }
            },
            "required": [
                "conversationId",
                "messageId"
            ]
        },
        "ChatInteractWithMessageEvent": {
            "type": "object",
            "properties": {
                "conversationId": {
                    "type": "string",
                    "description": "ID which represents a multi-turn conversation,
length between 1 and 128"
                },
                "messageId": {
                    "type": "string",
                    "description": "Unique identifier for the chat message"
                },
                "customizationArn": {
```

```
                    "type": "string",
                    "description": "ARN of the customization matching pattern: ^arn:
[-.a-z0-9]{1,63}:codewhisperer:([-.a-z0-9]{0,63}:){2}([a-zA-Z0-9-_:/]){1,1023}$"
                },
                "interactionType": {
                    "type": "string",
                    "enum": [
                        "INSERT_AT_CURSOR",
                        "COPY_SNIPPET",
                        "COPY",
                        "CLICK_LINK",
                        "CLICK_BODY_LINK",
                        "CLICK_FOLLOW_UP",
                        "HOVER_REFERENCE",
                        "UPVOTE",
                        "DOWNVOTE"
                    ],
                    "description": "Chat Message Interaction Type"
                },
                "interactionTarget": {
                    "type": "string",
                    "description": "Target of message interaction"
                },
                "acceptedCharacterCount": {
                    "type": "integer"
                },
                "acceptedLineCount": {
                    "type": "integer"
                },
                "acceptedSnippetHasReference": {
                    "type": "boolean"
                },
                "hasProjectLevelContext": {
                    "type": "boolean"
                }
            },
            "required": [
                "conversationId",
                "messageId"
            ]
        },
        "ChatUserModificationEvent": {
            "type": "object",
            "properties": {
```

```
                "conversationId": {
                    "type": "string",
                    "description": "ID which represents a multi-turn conversation,
 length between 1 and 128"
                },
                "customizationArn": {
                    "type": "string",
                    "description": "ARN of the customization matching pattern: ^arn:
[-.a-z0-9]{1,63}:codewhisperer:([-.a-z0-9]{0,63}:){2}([a-zA-Z0-9-_:/]){1,1023}$"
                },
                "messageId": {
                    "type": "string",
                    "description": "Unique identifier for the chat message"
                },
                "programmingLanguage": {
                    "$ref": "#/definitions/ProgrammingLanguage"
                },
                "modificationPercentage": {
                    "type": "number",
                    "description": "This is the percentage of AI generated code which
 has been modified by the user"
                },
                "hasProjectLevelContext": {
                    "type": "boolean"
                }
            },
            "required": [
                "conversationId",
                "messageId",
                "modificationPercentage"
            ]
        },
        "SuggestionState": {
            "type": "string",
            "enum": [
                "ACCEPT",
                "REJECT",
                "DISCARD",
                "EMPTY"
            ]
        },
        "TerminalUserInteractionEvent": {
            "type": "object",
            "properties": {
```

```
                "terminalUserInteractionEventType": {
                    "type": "string",
                    "enum": [
                        "CODEWHISPERER_TERMINAL_TRANSLATION_ACTION",
                        "CODEWHISPERER_TERMINAL_COMPLETION_INSERTED"
                    ],
                    "description": "Terminal User Interaction Event Type"
                },
                "terminal": {
                    "type": "string"
                },
                "terminalVersion": {
                    "type": "string"
                },
                "shell": {
                    "type": "string"
                },
                "shellVersion": {
                    "type": "string"
                },
                "duration": {
                    "type": "integer"
                },
                "timeToSuggestion": {
                    "type": "integer"
                },
                "isCompletionAccepted": {
                    "type": "boolean"
                },
                "cliToolCommand": {
                    "type": "string"
                }
            }
        },
        "FeatureDevEvent": {
            "type": "object",
            "properties": {
                "conversationId": {
                    "type": "string",
                    "description": "ID which represents a multi-turn conversation,
length between 1 and 128"
                }
            },
            "required": [
```

```
                "conversationId"
            ]
        }
    },
    "SendTelemetryEventRequest": {
        "type": "object",
        "properties": {
            "clientToken": {
                "type": "string",
                "description": "The client's authentication token"
            },
            "telemetryEvent": {
                "properties": {
                    "oneOf": [
                        {
                            "_comment": "This event is emitted when a user accepts or
rejects an inline code suggestion",
                            "$ref": "#/definitions/userTriggerDecisionEvent"
                        },
                        {
                            "_comment": "This event is emitted every five minutes. It
details how much code is written by inline code suggestion and in total during that
period",
                            "$ref": "#/definitions/codeCoverageEvent"
                        },
                        {
                            "_comment": "This event is emitted when a code snippet from
inline code suggestion has been edited by a user. It details the percentage of that
code snippet modified by the user",
                            "$ref": "#/definitions/userModificationEvent"
                        },
                        {
                            "_comment": "This field is emitted when a security scan is
requested by a user",
                            "$ref": "#/definitions/codeScanEvent"
                        },
                        {
                            "_comment": "This field is emitted when a security scan
recommended remediation is accepted by a user",
                            "$ref": "#/definitions/codeScanRemediationsEvent"
                        },
                        {
                            "_comment": "This event is deprecated but may still occur
in telemetry. Do not use this.",
```

```
                            "$ref": "#/definitions/metricData"
                        },
                        {
                            "_comment": "This event is emitted when Q adds an AI
generated message to the chat window",
                            "$ref": "#/definitions/chatAddMessageEvent"
                        },
                        {
                            "_comment": "This event is emitted when a user interacts
with a chat message",
                            "$ref": "#/definitions/chatInteractWithMessageEvent"
                        },
                        {
                            "_comment": "This event is emitted when a user modifies a
code snippet sourced from chat. It gives a percentage of the code snippet which has
been modified",
                            "$ref": "#/definitions/chatUserModificationEvent"
                        },
                        {
                            "_comment": "This event is emitted when a user interacts
with a terminal suggestion",
                            "$ref": "#/definitions/terminalUserInteractionEvent"
                        },
                        {
                            "_comment": "This event is emitted when a user first
prompts the /dev feature.",
                            "$ref": "#/definitions/featureDevEvent"
                        }
                    ]
                }
            },
            "optOutPreference": {
                "type": "string",
                "enum": [
                    "OPTIN",
                    "OPTOUT"
                ],
                "description": "OPTOUT and telemetry is only provided to the account of
purchasing enterprise, OPTIN and telemetry may also be used for product improvement"
            },
            "userContext": {
                "type": "object",
                "properties": {
                    "ideCategory": {
```

```
                        "type": "string",
                        "enum": [
                            "JETBRAINS",
                            "VSCODE",
                            "CLI",
                            "JUPYTER_MD",
                            "JUPYTER_SM"
                        ]
                    },
                    "operatingSystem": {
                        "type": "string",
                        "description": "The operating system being used"
                    },
                    "product": {
                        "type": "string",
                        "description": "The name of the product being used"
                    },
                    "clientId": {
                        "type": "string",
                        "description": "A UUID representing the individual client being
used"
                    },
                    "ideVersion": {
                        "type": "string",
                        "description": "The version of the Q plugin"
                    }
                },
                "required": [
                    "ideCategory",
                    "operatingSystem",
                    "product",
                    "clientId",
                    "ideVersion"
                ]
            },
            "profileArn": {
                "type": "string",
                "description": "The arn of the Q Profile used to configure individual
user accounts."
```

Observe that a SendTelemetryEvent may contain one of a number of telemetry events. Each of these describes a specific interaction between the development environment.

A more detailed description of each event appears in the following sections.

UserTriggerDecisionEvent

This event is triggered when a user interacts with a suggestion made by Amazon Q. It captures whether the suggestion was accepted, rejected, or modified, along with relevant metadata.

- `completionType`: Whether the completion was a block or a line.
- `suggestionState`: Whether the user accepted, rejected, or discarded the suggestion.

CodeScanEvent

This event is logged when a code scan is performed. It helps track the scope and result of the scan, providing insights into security and code quality checks.

- `codeScanJobId`: The unique identifier for the code scan job.
- `codeAnalysisScope`: Whether the scan was performed at the file level or the project level.
- `programmingLanguage`: The language being scanned.

CodeScanRemediationsEvent

This event captures user interactions with Amazon Q's remediation suggestions, such as applying fixes or viewing issue details.

- `CodeScanRemediationsEventType`: The type of remediation action taken (e.g., viewing details or applying a fix).
- `includesFix`: A boolean indicating whether the code issue includes a suggested fix.

ChatAddMessageEvent

This event is triggered when a new message is added to an ongoing chat conversation. It captures the user's intent and any code snippets involved.

- `conversationId`: The unique identifier for the conversation.
- `messageId`: The unique identifier for the chat message.
- `userIntent`: The user's intent, such as improving code or explaining code.
- `programmingLanguage`: The language related to the chat message.

ChatInteractWithMessageEvent

This event captures when users interact with chat messages, such as copying code snippets, clicking links, or hovering over references.

- `interactionType`: The type of interaction (for example, copy, hover, click).
- `interactionTarget`: The target of the interaction (for example, a code snippet or a link).
- `acceptedCharacterCount`: The number of characters from the message that were accepted.
- `acceptedSnippetHasReference`: A boolean indicating if the accepted snippet included a reference.

TerminalUserInteractionEvent

This event logs user interactions with terminal commands or completions in the terminal environment.

- `terminalUserInteractionEventType`: The type of interaction (for example, terminal translation or code completion).
- `isCompletionAccepted`: A boolean indicating whether the completion was accepted by the user.
- `duration`: The time taken for the interaction.

Accessing customization-related messages in Amazon CloudWatch Logs

You can store information about the creation of your customizations in Amazon CloudWatch Logs.

You can authorize your Amazon Q Developer administrator to view those logs with the following permission set.

To learn more about the permissions required to delivery logs to multiple resources, see Logging that requires additional permissions [V2] in the *Amazon CloudWatch Logs User Guide*.

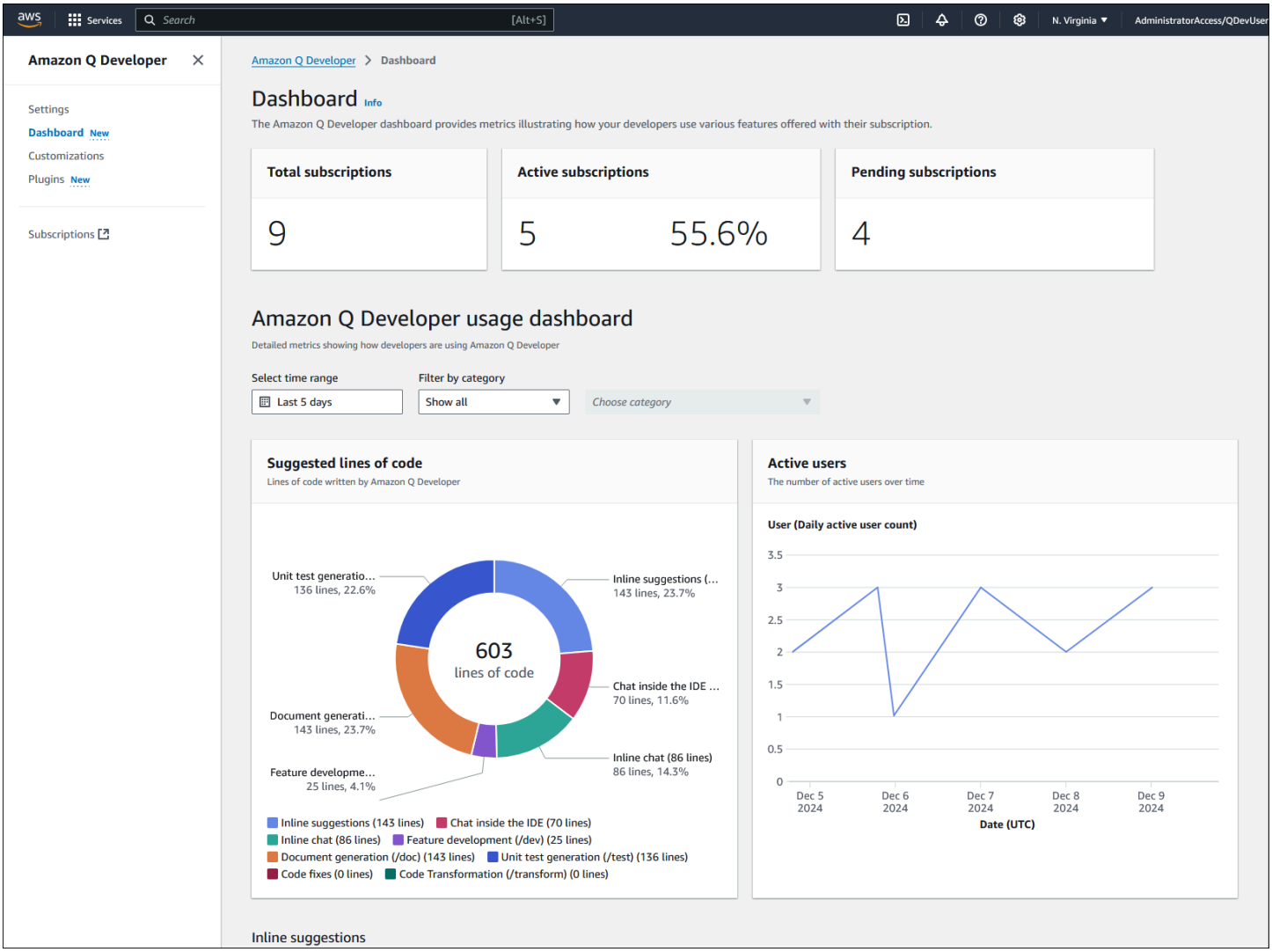
JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "AllowLogDeliveryActions",
            "Effect": "Allow",
            "Action": [
                "logs:PutDeliverySource",
                "logs:GetDeliverySource",
                "logs:DeleteDeliverySource",
                "logs:DescribeDeliverySources",
                "logs:PutDeliveryDestination",
                "logs:GetDeliveryDestination",
                "logs:DeleteDeliveryDestination",
                "logs:DescribeDeliveryDestinations",
                "logs:CreateDelivery",
                "logs:GetDelivery",
                "logs:DeleteDelivery",
                "logs:DescribeDeliveries",
                "firehose:ListDeliveryStreams",
                "firehose:DescribeDeliveryStream",
                "s3:ListAllMyBuckets",
                "s3:ListBucket",
                "s3:GetBucketLocation"
            ],
            "Resource": [
                "arn:aws-eusc:logs:eusc-de-east-1:111122223333:log-group:*",
                "arn:aws-eusc:firehose:eusc-de-
east-1:111122223333:deliverystream/*",
                "arn:aws-eusc:s3:::*"
            ]
        }
    ]
}
```

Viewing Amazon Q Developer user activity on the dashboard

Available only for Amazon Q Developer administrators, the Amazon Q Developer dashboard summarizes useful data about how your Pro tier subscribers use the service.



Amazon Q generates and displays new metrics on an hourly basis for the most part. The only section that is not updated hourly is the **Active users** widget, which is updated daily according to the coordinated universal time (UTC) clock.

The dashboard shows metrics collected from users who are subscribed in:

- the AWS account that you're currently signed into

  and

- member accounts, if you're signed in to a management account for which organization-wide visibility of subscriptions has been enabled.

> **Note**
>
> The **Active users** widget only displays information from the account that you're currently signed into.

To view and filter the dashboard

1. Sign in to the AWS Management Console.
2. Switch to the Amazon Q Developer console.
3. From the navigation pane, choose **Dashboard**.
4. (Optional) Filter the information by date range, programming language, customization, or integrated development environment (IDE) vendor.

Notes:

- If the **Dashboard** link is not available in the navigation pane, see Troubleshooting the dashboard.
- If you'd like to send user metrics to a daily report with a per-user breakdown of their Amazon Q Developer usage, see Viewing the activity of specific users in Amazon Q Developer.
- For information about specific metrics, see Descriptions of Amazon Q Developer dashboard usage metrics or choose the help link (ⓘ) on the dashboard page.

Descriptions of Amazon Q Developer dashboard usage metrics

The following table describes the metrics shown in the Amazon Q Developer dashboard.

For more information about the dashboard, see Viewing Amazon Q Developer user activity on the dashboard.

Metric name	Description
Total subscriptions	Shows the total subscriptions in the current AWS account, as well as subscriptions in

Metric name	Description
	member accounts, if you're signed in to a management account for which organization-wide visibility of subscriptions has been enabled.
Active subscriptions	Shows the total active subscriptions in the current AWS account, as well as subscriptions in member accounts, if you're signed in to a management account for which organization-wide visibility of subscriptions has been enabled. *Active subscriptions* are those belonging to users who have started using Amazon Q in their integrated development environment (IDE). You are being charged for these subscriptions. For more information about active subscriptions, see Amazon Q Developer subscription statuses.
Pending subscriptions	Shows the total pending subscriptions in the current AWS account, as well as subscriptions in member accounts, if you're signed in to a management account for which organization-wide visibility of subscriptions has been enabled. *Pending subscriptions* are those belonging to users who have not yet started using Amazon Q in their IDE. You are not being charged for these subscriptions. For more information about pending subscriptions, see Amazon Q Developer subscription statuses.

Metric name	Description
Accepted lines of code	Shows a pie chart that indicates the lines of code accepted by users, broken down by Amazon Q feature (feature development, documentation generation, unit test generation, and so on).
Active users	Shows a line graph that indicates the number of subscribers who were actively using Amazon Q in the IDE during a specified date range.
Inline suggestions	Shows the total number of suggestions and accepted suggestions for the inline suggestions feature. The percentage of accepted suggestions is calculated by taking the number of suggestions accepted by users, and dividing it by the total suggestions generated by Amazon Q. The total suggestions count includes suggestions that were accepted and actively rejected; it does not include suggestions that were dismissed because the user continued to type or started performing other operations in their IDE.
Inline chat	Shows the total number of suggestions and number of accepted suggestions for the inline chat feature. The percentage of accepted suggestions is calculated by taking the number of suggestions accepted by users, and dividing it by the total suggestions generated by Amazon Q.
Chat in the IDE – Total messages sent	Shows the total number of responses from Amazon Q in the Amazon Q chat window of the user's IDE.

Metric name	Description
Feature development – Acceptance rate	Shows the acceptance rate for the Feature development feature. The acceptance rate is calculated by taking the number of lines of code accepted by users, and dividing it by the total lines of code suggested by Amazon Q.
Document generation	Shows the total number of documenta tion files (such as READMEs and supporting files) created and updated by the Document generation feature. The acceptance rates equal the number of file updates or creations accepted by users, divided by the total number of files updates or creations suggested by Amazon Q.
Unit test generation	Shows the total number of unit tests generated by the Unit test generation feature and the number of unit tests accepted by users. The acceptance rate is calculated by taking the number of unit tests accepted by users, and dividing it by the total number of unit tests generated by Amazon Q.
Code reviews	Shows the total number of code reviews and findings reports generated by the code review feature. The **Total code reviews (manual only)** and **Findings report (manual only)** refer to the code reviews and finding reports that are *not* auto-generated.

Metric name	Description
Code fixes	Shows the total number of code fixes generated by Amazon Q. The acceptance rate is calculated by taking the number of code fixes accepted by users, and dividing it by the total number of code fixes suggested by Amazon Q.
Code transformation	Shows the total number code transformations performed by the transformation feature, and the number of lines of code processed.

Disabling the Amazon Q Developer dashboard

You might want to disable the Amazon Q Developer dashboard if you have concerns about data privacy, page load times, or other potential issues. When you disable the dashboard, the dashboard page (and any links to it) will no longer be available in the Amazon Q Developer console.

For more information about the dashboard, see Viewing usage metrics (dashboard).

To disable the dashboard

1. Open the Amazon Q Developer console:
   - If you set up Amazon Q Developer with an organization instance of AWS IAM Identity Center, then sign in using a management account or member account.
   - If you set up Amazon Q Developer with an account instance of IAM Identity Center, then sign in using the account associated with that instance.
2. Choose **Settings**, and in the **Amazon Q Developer user activity** section, choose **Edit**.
3. Disable **Amazon Q Developer usage dashboard**.

Troubleshooting the Amazon Q Developer dashboard

If the Amazon Q Developer dashboard page is not available, do the following:

- **Verify your permissions**. To view the dashboard, you need the following permissions:

- `q:ListDashboardMetrics`
- `codewhisperer:ListProfiles`
- `sso:ListInstances`
- `user-subscriptions:ListUserSubscriptions`
- To see metrics generated before November 22, 2024, you also need: `cloudwatch:GetMetricData` and `cloudwatch:ListMetrics`

  For more information about permissions, see Allow administrators to use the Amazon Q Developer console.

- **Verify your settings**. In the Amazon Q Developer console, choose **Settings** and make sure that the **Amazon Q Developer usage dashboard** toggle is enabled.

For more information about the dashboard, see Viewing usage metrics (dashboard).

Viewing the activity of specific users in Amazon Q Developer

You can configure Amazon Q to collect user activity telemetry of individual Amazon Q Developer subscribers in your organization and present that information in a report. The report gives you insights into how specific users are using Amazon Q.

Amazon Q generates the report every day at midnight (00:00) Coordinated Universal Time (UTC), and saves it in a CSV file at the following path:

`s3://`*`bucketName`*`/`*`prefix`*`/AWSLogs/`*`accountId`*`/QDeveloperLogs/by_user_analytic/`*`region`*`/`*`year`*`/`*`month`*`/`*`day`*`/00/`*`accountId`*`_by_user_analytic_`*`timestamp`*`.c`

The CSV file is laid out as follows:

- Each row shows a user who interacted with Amazon Q that day.
- Each column shows a metric, as described in User activity report metrics. Metrics are calculated based on the user telemetry collected over the course of the day.

If more than 1,000 users interact with Amazon Q during the day, Amazon Q splits the data into several CSV files containing 1,000 users each, and having suffixes of `part_1`, `part_2`, and so on.

> **Note**
>
> When you enable user activity reports, Amazon Q collects telemetry regardless of how a developer has set the **Enable Amazon Q to send usage data to AWS** setting in their IDE. That setting controls whether telemetry can be used by the *AWS corporation*, not your organization. For more information about this setting, see Opting out of sharing your client-side telemetry.

Use the following instructions to enable user activity reports.

Prerequisite

Create an Amazon S3 bucket to hold the user activity report CSV file. The bucket must:

- Be in the AWS Region where the Amazon Q Developer profile was installed. This profile was installed when you subscribed IAM Identity Center workforce users to Amazon Q Developer Pro for the first time. For more information about this profile and the Regions where it's supported, see What is the Amazon Q Developer profile?, and Supported Regions for the Q Developer console and Q Developer profile.
- Be in the AWS account where users are subscribed. If users are subscribed in multiple AWS accounts, then you must create buckets in each of those accounts. Cross-account buckets are not supported.
- (Optional but recommended) Be different from the bucket you might be using for prompt logging.
- Include a prefix, also known as a subfolder, where Amazon Q will save the CSV file. The CSV file cannot be saved in the root of the bucket.
- Have a bucket policy like the one that follows. Replace *`bucketName`*, *`region`*, *`accountId`*, and *`prefix`* with your own information.

  JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "QDeveloperLogsWrite",
            "Effect": "Allow",
            "Principal": {
```

```
                "Service": "q.amazonaws.com"
            },
            "Action": [
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws-eusc:s3:::bucketName/prefix/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "111122223333"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws-eusc:codewhisperer:eusc-de-
east-1:111122223333:*"
                }
            }
        }
    ]
}
```

If you're configuring SSE-KMS on the bucket, add the below policy on the KMS key:

```
{
    "Effect": "Allow",
    "Principal": {
        "Service": "q.amazonaws.com"
    },
    "Action": "kms:GenerateDataKey",
    "Resource": "*",
    "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "accountId"
        },
        "ArnLike": {
           "aws:SourceArn": "arn:aws:codewhisperer:region:accountId:*"
        }
    }
}
```

To learn about protecting the data in your Amazon S3 bucket, see Protecting data with encryption in the *Amazon Simple Storage Service User Guide*.

To enable user activity reports

1. Open the Amazon Q Developer console.

   To use the Amazon Q Developer console, you must have the permissions defined in Allow administrators to use the Amazon Q Developer console.
2. Choose **Settings**.
3. Under **Q Developer user activity reports**, choose **Edit**.
4. Toggle **Collect granular metrics per user**.
5. Under **S3 location**, enter the Amazon S3 URI that you will use to hold the CSV reports. Example: `s3://amzn-s3-demo-bucket/user-activity-reports/`

User activity report metrics

The following table describes the metrics that are included in the user activity reports generated by Amazon Q Developer.

For more information about these reports, see Viewing the activity of specific users in Amazon Q Developer.

Metric name	Description
Chat_AICodeLines	Lines of code suggested by Amazon Q and accepted by the user. This metric includes code that was generated through the Amazon Q chat (not inline chat) and inserted into the IDE.
Chat_MessagesInteracted	Number of chat messages where the user has interacted positively with Amazon Q. Examples of positive interactions: clicking a link, inserting a suggestion, and upvoting a response from Amazon Q. This metric includes messages that were generated by Amazon Q chat (not inline chat).
Chat_MessagesSent	Number of messages sent to and from Amazon Q. This metric includes the user

Metric name	Description
	prompts and Amazon Q responses in the Amazon Q chat (not inline chat).
CodeFix_AcceptanceEventCount	Number of code fixes suggested by Amazon Q and accepted by the user. This metric applies to code fixes generated through the code review feature.
CodeFix_AcceptedLines	Lines of code suggested by Amazon Q and accepted by the user. This metric applies to lines of code generated through the code review feature.
CodeFix_GeneratedLines	Lines of code suggested by Amazon Q. This metric applies to lines of code generated through the code review feature.
CodeFix_GenerationEventCount	Number of code fixes suggested by Amazon Q. This metric applies to code fixes generated through the code review feature.
CodeReview_FailedEventCount	Number of code issues that were found but for which Amazon Q could not suggest a code fix. This metric applies to code issues generated using the code review feature.
CodeReview_FindingsCount	Number of code issues found by Amazon Q. This metric applies to code issues found using the code review feature.
CodeReview_SucceededEventCount	Number of code issues that were found and for which Amazon Q was able to generate a suggested code fix. This metric applies to code issues found using the code review feature.

Metric name	Description
Dev_AcceptanceEventCount	Number of code features suggested by Amazon Q and accepted by the user. This metric applies to code features generated through the /dev command.
Dev_AcceptedLines	Lines of code suggested by Amazon Q and accepted by the user. This metric applies to lines of code generated through the /dev command.
Dev_GeneratedLines	Lines of code suggested by Amazon Q. This metric applies to lines of code generated through the /dev command.
Dev_GenerationEventCount	Number of code features suggested by Amazon Q. This metric applies to code features generated through the /dev command.
DocGeneration_AcceptedFileUpdates	Number of file updates suggested by Amazon Q and accepted by the user. This metric applies to file updates generated through the /doc command.
DocGeneration_AcceptedFilesCreations	Number of file creations suggested by Amazon Q and accepted by the user. This metric applies to file creations generated through the /doc command.
DocGeneration_AcceptedLineAdditions	Lines of documentation additions suggested by Amazon Q and accepted by the user. This metric applies to documentation generated through the /doc command.

Metric name	Description
DocGeneration_AcceptedLineUpdates	Lines of documentation updates suggested by Amazon Q and accepted by the user. This metric applies to documentation generated using the /doc command.
DocGeneration_EventCount	Number of times the user engaged with Amazon Q using the /doc command.
DocGeneration_RejectedFileCreations	Number of file creations suggested by Amazon Q and rejected by the user. This metric applies to file creations generated through the /doc command.
DocGeneration_RejectedFileUpdates	Number of file updates suggested by Amazon Q and rejected by the user. This metric applies to file updates generated through the /doc command.
DocGeneration_RejectedLineAdditions	Lines of documentation additions suggested by Amazon Q and rejected by the user. This metric applies to documentation generated through the /doc command.
DocGeneration_RejectedLineUpdates	Lines of documentation updates suggested by Amazon Q and rejected by the user. This metric applies to documentation generated using the /doc command.
InlineChat_AcceptedLineAdditions	Lines of code additions suggested by Amazon Q and accepted by the user. This metric includes code additions generated through the inline chat (not Amazon Q chat).

Metric name	Description
InlineChat_AcceptedLineDeletions	Lines of code deletions suggested by Amazon Q and accepted by the user. This metric includes code deletions suggested through the inline chat (not Amazon Q chat).
InlineChat_AcceptanceEventCount	Number of inline chat (not Amazon Q chat) suggestions that were accepted by the user.
InlineChat_DismissalEventCount	Number of inline chat (not Amazon Q chat) suggestions that were abandoned by the user. By 'abandoned', we mean that the code suggestion was displayed but the user kept typing or performing other operations in their IDE, and did not explicitly accept or reject the code suggestion.
InlineChat_DismissedLineAdditions	Lines of code additions suggested by Amazon Q and abandoned by the user. By 'abandoned', we mean that the code suggestion was displayed but the user kept typing or performing other operations in their IDE, and did not explicitly accept or reject the code suggestion. This metric includes code additions generated through the inline chat (not Amazon Q chat).
InlineChat_DismissedLineDeletions	Lines of code deletions suggested by Amazon Q and abandoned by the user. By 'abandoned', we mean that the code suggestion was displayed but the user kept typing or performing other operations in their IDE, and did not explicitly accept or reject the code suggestion. This metric includes code deletions suggested through the inline chat (not Amazon Q chat).

Metric name	Description
InlineChat_EventCount	Number of inline chat (not Amazon Q chat) sessions that the user engaged in.
InlineChat_RejectedLineAdditions	Lines of code additions suggested by Amazon Q and rejected by the user. This metric includes code additions generated through the inline chat (not Amazon Q chat).
InlineChat_RejectedLineDeletions	Lines of code deletions suggested by Amazon Q and rejected by the user. This metric includes code deletions suggested through the inline chat (not Amazon Q chat).
InlineChat_RejectionEventCount	Number of inline chat (not Amazon Q chat) suggestions that were rejected by the user.
Inline_AICodeLines	Lines of code suggested by Amazon Q and accepted by the user. This metric includes code that was accepted as inline suggestions.
Inline_AcceptanceCount	Number of inline suggestions accepted by the user.
Inline_SuggestionsCount	Number of inline suggestions displayed to the user.
TestGeneration_AcceptedLines	Lines of code suggested by Amazon Q and accepted by the user. This metric applies to lines of code generated through the /test command.
TestGeneration_AcceptedTests	Number of unit tests suggested by Amazon Q and accepted by the user. This metric applies to unit tests generated through the /test command.

Metric name	Description
TestGeneration_EventCount	Number of times the user engaged with Amazon Q through the /test command.
TestGeneration_GeneratedLines	Lines of code suggested by Amazon Q. This metric applies to lines of code generated through the /test command.
TestGeneration_GeneratedTests	Number of unit tests suggested by Amazon Q. This metric applies to unit tests generated through the /test command.
Transformation_EventCount	Number of times the user engaged with Amazon Q through the /transform command, excluding the times when the when the user transformed code on the command line.
Transformation_LinesGenerated	Lines of code suggested by Amazon Q. This metric applies to code generated through the /transform command, excluding code transformed on the command line.
Transformation_LinesIngested	Lines of code provided to Amazon Q for transformation. This metric applies to code that is provided through the /transform command, excluding code provided for transformation on the command line, or for an SQL conversion.

Logging users' prompts in Amazon Q Developer

Administrators can enable the logging of all inline suggestions and chat conversations that users have with Amazon Q in their integrated development environment (IDE). These logs can help with auditing, debugging, analytics, and ensuring compliance.

When developers use inline suggestions, Amazon Q will log the accepted and actively rejected suggestions. When developers chat with Amazon Q, Amazon Q will log both the developers'

prompts and Amazon Q's responses. When developers chat with the Amazon Q Agent for software development using the **/dev** command, only the prompts will be logged.

Amazon Q stores the logs in an Amazon S3 bucket that you create, at the following path:

bucketName/*prefix*/AWSLogs/*accountId*/QDeveloperLogs/*log-type*/*region*/*year*/*month*/*day*/*utc-hour*/*zipFile*.gz/*logFile*.json

At the previous path, *log-type* is one of the following:

- `GenerateAssistantResponse` — holds chat logs
- `GenerateCompletions` — holds inline completion logs
- `StartTaskAssistCodeGeneration` — holds **/dev** logs

For examples and explanations of log file contents, see Prompt log examples in Amazon Q Developer.

There is no charge for the prompt logging feature other than the storage cost of the Amazon S3 bucket used to hold the logs, and a small fee for the optional KMS key used to encrypt the bucket.

Use the following instructions to enable prompt logging.

Prerequisites

- Make sure users are subscribed in a standalone account or, if you're using AWS Organizations, a management account. Currently, Q Developer does not support logging the prompts of users who are subscribed in member accounts in AWS Organizations.
- Create an Amazon S3 bucket to hold the prompt logs. The bucket must:
  - Be in the AWS Region where the Amazon Q Developer profile was installed. This profile was installed when you subscribed users to Amazon Q Developer Pro for the first time. For more information about this profile and the Regions where it's supported, see What is the Amazon Q Developer profile?, and Supported Regions for the Q Developer console and Q Developer profile.
  - Be in the AWS account where users are subscribed.
  - Have a bucket policy like the one that follows. Replace *bucketName*, *region*, *accountId*, and *prefix* with your own information.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "QDeveloperLogsWrite",
            "Effect": "Allow",
            "Principal": {
                "Service": "q.amazonaws.com"
            },
            "Action": [
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws-eusc:s3:::bucketName/prefix/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "111122223333"
                },
                "ArnLike": {
                "aws:SourceArn": "arn:aws-eusc:codewhisperer:eusc-de-
east-1:111122223333:*"
                }
            }
        }
    ]
}
```

If you're configuring SSE-KMS on the bucket, add the following policy on the KMS key:

```
{
    "Effect": "Allow",
    "Principal": {
        "Service": "q.amazonaws.com"
    },
    "Action": "kms:GenerateDataKey",
    "Resource": "*",
    "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "accountId"
```

```
        },
        "ArnLike": {
           "aws:SourceArn": "arn:aws:codewhisperer:region:accountId:*"
        }
    }
}
```

To learn about protecting the data in your Amazon S3 bucket, see Protecting data with encryption in the *Amazon Simple Storage Service User Guide*.

To enable prompt logging

1. Open the Amazon Q Developer console.

   To use the Amazon Q Developer console, you must have the permissions defined in Allow administrators to use the Amazon Q Developer console.

   > **Note**
   >
   > You must sign in as a standalone account administrator, or management account administrator. Member account administrators cannot enable prompt logging because prompt logging is not supported for users subscribed in member accounts.

2. Choose **Settings**.
3. Under **Preferences**, choose **Edit**.
4. In the Edit preferences window, toggle **Q Developer prompt logging**.
5. Under Amazon S3 location, enter the Amazon S3 URI that you will use to receive the logs. Example: `s3://amzn-s3-demo-bucket/qdev-prompt-logs/`

Prompt log examples in Amazon Q Developer

This section provides examples of prompt logs generated by Amazon Q Developer.

Following each example is a table that describes the log file's fields.

For more information about prompt logs, see Logging users' prompts in Amazon Q Developer.

Topics

- Inline suggestions log example
- Chat log example
- /dev log example

Inline suggestions log example

The following example shows a log file that is generated when a user accepts an inline suggestion.


```
{
    "records": [
        {
            "generateCompletionsEventRequest": {
                "leftContext": "import * cdk from 'aws-cdk-lib';\r\nimport * s3
 from 'aws-cdk-lib/aws-s3';\r\nimport { Stack, StackProps } from 'constructs';\r
\nexport class MyStack extends Stack {\r\n  constructor(scope: cdk.App, id: string,
 props?: StackProps) {\r\n    super(scope, id, props);\r\n\r\n    new s3.Bucket(this,
 'XXXXXXXX', {\r\n      versioned: true\r\n    });\r\n  }\r\n  ",
                "rightContext": "",
                "fileName": "cdk-modified.ts",
                "customizationArn": null,
                "userId": "d-92675051d5.b8f1f340-9081-70ad-5fc5-0f37151937a6",
                "timeStamp": "2025-01-06T15:09:16.412719Z"
            },
            "generateCompletionsEventResponse": {
                "completions": ["synth() {\n    return
 cdk.App.prototype.synth.apply(this, arguments);\n  }"],
                "requestId": "797c70ee-abc9-4cc7-a148-b9df17f6ce48"
            }
        }
    ]
}
```


The following table describes the fields in the log file for inline suggestions.

Field name	Description
`records`	Top-level field that contains a set of inline suggestions, also known as inline completions.

Field name	Description
generateCompletionsEventRequest	Describes the request for an inline code suggestion. The request is made by Amazon Q, on behalf of the user.
leftContext	Indicates the code before the cursor that Amazon Q is using for context to generate an inline suggestion.
rightContext	Indicates the code after the cursor that Amazon Q is using for context to generate an inline suggestion.
fileName	Indicates the name of the file in which an inline suggestion was requested.
customizationArn	Indicates the Amazon Resource Name (ARN) of the customization applied to inline suggestio ns. In this example, there is no customization.
userId	Indicates the user ID of the user who requested an inline suggestion.
timeStamp	Indicates the date and time that the inline suggestion was requested. The timestamp is in Coordinated Universal Time (UTC).
generateCompletionsEventResponse	Describes the inline suggestion generated by Amazon Q.
completions	Indicates the inline suggestion that Amazon Q provided and that the user accepted.
requestId	The ID of the inline request that uniquely identifies the event.

Chat log example

The following example shows a log file that is generated when a user chats using Amazon Q in their IDE.


```
{
    "records": [
        {
            "generateAssistantResponseEventRequest": {
                "prompt": "Can you explain how to set up object-level access control in
 an S3 bucket using AWS CDK?",
                "chatTriggerType": "MANUAL",
                "customizationArn": null,
                "userId": "d-92675051d5.b8f1f340-9081-70ad-5fc5-0f37151937a6",
                "timeStamp": "2025-01-06T15:05:56.313070Z"
            },
            "generateAssistantResponseEventResponse": {
                "assistantResponse": "I'll explain how to set up object-level access
 control for an S3 bucket using AWS CDK. Here's how you can implement different
 access control configurations: [[1]](https://docs.aws.amazon.com/AmazonS3/latest/
API/s3_example_s3_Scenario_ManageACLs_section.html)\n\n1. Using Bucket Owner Enforced
 (Recommended Modern Approach):\n```typescript\nimport * as cdk from 'aws-cdk-lib';
\nimport * as s3 from 'aws-cdk-lib/aws-s3';\n\nnew s3.Bucket(this, 'XXXXXXXX', {\n
  objectOwnership: s3.ObjectOwnership.BUCKET_OWNER_ENFORCED,\n  blockPublicAccess:
 s3.BlockPublicAccess.BLOCK_ALL\n});\n```\n\n2. Using Public Access Block
 Configuration:\n```typescript\nimport * as cdk from 'aws-cdk-lib';\nimport * as s3
 from 'aws-cdk-lib/aws-s3';\n\nnew s3.Bucket(this, 'XXXXXXXX', {\n  blockPublicAccess:
 new s3.BlockPublicAccess({\n    blockPublicAcls: true,\n    blockPublicPolicy:
 true,\n    ignorePublicAcls: true,\n    restrictPublicBuckets: true\n  })\n});\n```
\n\n3. Setting Access Control for Specific Use Cases:\n```typescript\nimport * as
 cdk from 'aws-cdk-lib';\nimport * as s3 from 'aws-cdk-lib/aws-s3';\nimport * as
 iam from 'aws-cdk-lib/aws-iam';\n\nconst bucket = new s3.Bucket(this, 'XXXXXXXX',
 {\n  // Enable versioning for object-level recovery\n  versioned: true,\n  \n  //
 Enforce encryption\n  encryption: s3.BucketEncryption.S3_MANAGED,\n  \n  // Block
 all public access\n  blockPublicAccess: s3.BlockPublicAccess.BLOCK_ALL,\n  \n  //
 Enforce object ownership\n  objectOwnership: s3.ObjectOwnership.BUCKET_OWNER_ENFORCED,
\n  \n  // Enable server access logs\n  serverAccessLogsPrefix: 'logs/'\n});\n
\n// Grant read access to specific IAM role\nconst role = new iam.Role(this,
 'MyRole', {\n  assumedBy: new iam.ServicePrincipal('lambda.amazonaws.com'),
\n});\nbucket.grantRead(role);\n\n// Grant write access to specific objects
 with prefix\nbucket.addToResourcePolicy(new iam.PolicyStatement({\n  actions:
 ['s3:PutObject'],\n  resources: [bucket.arnForObjects('uploads/*')],\n  principals:
 [new iam.AccountRootPrincipal()]\n}));\n```\n\n4. Setting up Lifecycle Rules
```

```
 for Objects:\n```typescript\nimport * as cdk from 'aws-cdk-lib';\nimport
 * as s3 from 'aws-cdk-lib/aws-s3';\n\nnew s3.Bucket(this, 'XXXXXXXX', {\n
 lifecycle: [\n    {\n      transitions: [\n        {\n          storageClass:
 s3.StorageClass.INFREQUENT_ACCESS,\n          transitionAfter: cdk.Duration.days(30)\n
       },\n        {\n          storageClass: s3.StorageClass.GLACIER,\n
   transitionAfter: cdk.Duration.days(90)\n        }\n      ],\n      expiration:
 cdk.Duration.days(365)\n    }\n  ]\n});\n```\n\nKey points to remember:\n- Always use
 the principle of least privilege\n- Block public access by default unless explicitly
 required\n- Use BUCKET_OWNER_ENFORCED for object ownership when possible\n- Consider
 implementing lifecycle rules for cost optimization\n- Use versioning for sensitive
 data\n- Implement server-side encryption\n- Use resource policies and IAM roles
 for access control instead of ACLs\n- Monitor access patterns using server access
 logging\n\nThese configurations help create a secure and well-managed S3 bucket with
 appropriate access controls at both bucket and object levels.",
               "followupPrompts": "How can you implement multi-factor authentication
 for S3 bucket access using AWS CDK?What are the best practices for setting up object-
level versioning in AWS S3 using CDK?How can you use AWS CDK to create a custom error
 message for unauthorized access attempts on an S3 bucket?",
               "messageMetadata": {
                   "conversationId": "491bd3cd-d70d-40c0-a452-5ef3f0878422",
                   "utteranceId": null
               },
               "codeReferenceEvents": [],
               "supplementaryWebLinksEvent": [
                   {
                       "uri": "https://docs.aws.amazon.com/AmazonS3/latest/API/
s3_example_s3_Scenario_ManageACLs_section.html",
                       "title": "[1] Manage access control lists (ACLs) for Amazon S3
 buckets using an AWS SDK - Amazon Simple Storage Service",
                       "snippet": "The following code example shows how to manage
 access control lists (ACLs) for Amazon S3 buckets.\n\n.NET\n\n**AWS SDK for .NET**
\n\n```\n    using System;\n    using System.Collections.Generic;\n    using
 System.Threading.Tasks;\n    using Amazon.S3;\n    using Amazon.S3.Model;\n\n
   /// <summary>\n    /// This example shows how to manage Amazon Simple Storage
 Service\n    /// (Amazon S3) access control lists (ACLs) to control Amazon S3
 bucket\n    /// access.\n    /// </summary>\n    public class ManageACLs\n    {\n
       public static async Task Main()\n        {\n            string bucketName
 = \"amzn-s3-demo-bucket1\";\n            string newBucketName = \"amzn-s3-demo-
bucket2\";\n            string keyName = \"sample-object.txt\";\n            string
 emailAddress = \"someone@example.com\";\n\n            // If the AWS Region where
 your bucket is located is different from\n            // the Region defined for
 the default user, pass the Amazon S3 bucket's\n            // name to the client
 constructor. It should look like this:\n            // RegionEndpoint bucketRegion =
 RegionEndpoint.USEast1;\n            IAmazonS3 client = new AmazonS3Client();\n\n
```

```
          await TestBucketObjectACLsAsync(client, bucketName, newBucketName, keyName,
 emailAddress);\n        }\n\n        /// <summary>\n        /// Creates a new Amazon
 S3 bucket with a canned ACL, then retrieves the ACL\n        /// information and then
 adds a new ACL to one of the objects in the\n        /// Amazon S3 bucket.\n
  /// </summary>\n        /// <param name=\"client\">The initialized Amazon S3 client
 object used to call\n        /// methods to create a bucket, get an ACL, and add a
 different ACL to\n        /// one of the objects.</param>\n        /// <param name=
\"bucketName\">A string representing the original Amazon S3\n        /// bucket name.</
param>\n        /// <param name=\"newBucketName\">A string representing the name of
 the\n        /// new bucket that will be created.</param>\n        /// <param name=
\"keyName\">A string representing the key name of an Amazon S3\n        /// object
 for which we will change the ACL.</param>\n        /// <param name=\"emailAddress\">A
 string representing the email address\n        /// belonging to the person to whom
 access to the Amazon S3 bucket will be\n        /// granted.</param>\n        public
 static async Task TestBucketObjectACLsAsync(\n            IAmazonS3 client,\n
    string bucketName,\n            string newBucketName,\n            string keyName,
\n            string emailAddress)\n        {\n            try\n            {\n
          // Create a new Amazon S3 bucket and specify canned ACL.\n
 var success = await CreateBucketWithCannedACLAsync(client, newBucketName);\n\n
         // Get the ACL on a bucket.\n                await GetBucketACLAsync(client,
 bucketName);\n\n                // Add (replace) the ACL on an object in a
 bucket.\n                await AddACLToExistingObjectAsync(client, bucketName,
 keyName, emailAddress);\n            }\n            catch (AmazonS3Exception
 amazonS3Exception)\n            {\n                Console.WriteLine($\"Exception:
 {amazonS3Exception.Message}\");\n            }\n        }\n\n        /// <summary>\n
       /// Creates a new Amazon S3 bucket with a canned ACL attached.\n        ///
 </summary>\n        /// <param name=\"client\">The initialized client object used to
 call\n        /// PutBucketAsync.</param>\n        /// <param name=\"newBucketName\">A
 string representing the name of the\n        /// new Amazon S3 bucket.</param>\n
    /// <returns>Returns a boolean value indicating success or failure.</returns>\n
      public static async Task<bool> CreateBucketWithCannedACLAsync(IAmazonS3 client,
 string newBucketName)\n        {\n            var request = new PutBucketRequest()\n
          {\n                BucketName = newBucketName,\n                BucketRegion
 = S3Region.EUWest1,\n\n                // Add a canned ACL.\n                CannedACL
 = S3CannedACL.LogDeliveryWrite,\n            };\n\n            var response =
 await client.PutBucketAsync(request);\n            return response.HttpStatusCode
 == System.Net.HttpStatusCode.OK;\n        }\n\n\n        /// <summary>\n
   /// Retrieves the ACL associated with the Amazon S3 bucket name in the\n
    /// bucketName parameter.\n        /// </summary>\n        /// <param name=
\"client\">The initialized client object used to call\n        /// PutBucketAsync.</
param>\n        /// <param name=\"bucketName\">The Amazon S3 bucket for which we
 want to get the\n        /// ACL list.</param>\n        /// <returns>Returns an
 S3AccessControlList returned from the call to\n        /// GetACLAsync.</returns>
\n        public static async Task<S3AccessControlList> GetBucketACLAsync(IAmazonS3
```

```
 client, string bucketName)\n        {\n            GetACLResponse response = await
 client.GetACLAsync(new GetACLRequest\n            {\n                BucketName =
 bucketName,\n            });\n\n            return response.AccessControlList;\n
    }\n\n\n\n        /// <summary>\n        /// Adds a new ACL to an existing object
 in the Amazon S3 bucket.\n        /// </summary>\n        /// <param name=\"client
\">The initialized client object used to call\n        /// PutBucketAsync.</param>\n
        /// <param name=\"bucketName\">A string representing the name of the Amazon
 S3\n        /// bucket containing the object to which we want to apply a new ACL.</
param>\n        /// <param name=\"keyName\">A string representing the name of the
 object\n        /// to which we want to apply the new ACL.</param>\n        /// <param
 name=\"emailAddress\">The email address of the person to whom\n        /// we will be
 applying to whom access will be granted.</param>\n        public static async Task
 AddACLToExistingObjectAsync(IAmazonS3 client, string bucketName, string keyName,
 string emailAddress)\n        {\n            // Retrieve the ACL for an object.\n
        GetACLResponse aclResponse = await client.GetACLAsync(new GetACLRequest\n
        {\n                BucketName = bucketName,\n                Key = keyName,\n
           });\n\n            S3AccessControlList acl = aclResponse.AccessControlList;
\n\n            // Retrieve the owner.\n            Owner owner = acl.Owner;\n\n
          // Clear existing grants.\n            acl.Grants.Clear();\n\n
  // Add a grant to reset the owner's full permission\n            // (the previous
 clear statement removed all permissions).\n            var fullControlGrant = new
 S3Grant\n            {\n                Grantee = new S3Grantee { CanonicalUser =
 acl.Owner.Id },\n            };\n            acl.AddGrant(fullControlGrant.Grantee,
 S3Permission.FULL_CONTROL);\n\n            // Specify email to identify grantee
 for granting permissions.\n            var grantUsingEmail = new S3Grant\n
    {\n                Grantee = new S3Grantee { EmailAddress = emailAddress },\n
                Permission = S3Permission.WRITE_ACP,\n            };\n\n
  // Specify log delivery group as grantee.\n            var grantLogDeliveryGroup
 = new S3Grant\n            {\n                Grantee = new S3Grantee { URI =
 \"http://acs.amazonaws.com/groups/s3/LogDelivery\" },\n                Permission =
 S3Permission.WRITE,\n            };\n\n            // Create a new ACL.\n
   var newAcl = new S3AccessControlList\n            {\n                Grants = new
 List<S3Grant> { grantUsingEmail, grantLogDeliveryGroup },\n                Owner =
 owner,\n            };\n\n            // Set the new ACL. We're throwing away the
 response here.\n            _ = await client.PutACLAsync(new PutACLRequest\n
   {\n                BucketName = bucketName,\n                Key = keyName,\n
         AccessControlList = newAcl,\n            });\n        }\n\n    }\n\n\n```\n\n*
 For API details, see the following topics in _AWS SDK for .NET API Reference_.\n  \n
 * GetBucketAcl\n\n* GetObjectAcl\n\n* PutBucketAcl\n\n* PutObjectAcl\n\nFor a complete
 list of AWS SDK developer guides and code examples, see Developing with Amazon S3
 using the AWS SDKs. This topic also includes information about getting started and
 details about previous SDK versions.\n"
                    }
                ],
```

```
                "requestId": "dad38fc0-815c-45f7-970a-db916cb7f131"
            }
        }
    ]
}
```

Field name	Description
`records`	Top-level field that contains a set of prompts and responses.
`generateAssistantResponseEventRequest`	Describes the prompt entered by the user in the chat window in their IDE.
`prompt`	Indicates the prompt the user entered into the chat window.
`chatTriggerType`	`MANUAL` indicates that the user entered a prompt into the chat window, or clicked on one of the suggested questions in the chat window. `INLINE_CHAT` indicates that the user entered a prompt into the small input screen in the main coding window. For more information about chatting inline, see Chatting inline with Amazon Q Developer.
`customizationArn`	Indicates the Amazon Resource Name (ARN) of the customization applied to the chat. In this example, there is no customization. For
`userId`	Indicates the user ID of the user who entered the prompt.
`timeStamp`	Indicates the date and time that the user entered the prompt. The timestamp is in universal time (UTC).
`generateAssistantResponseEventResponse`	Describes the response generated by Amazon Q.

Field name	Description
`assistantResponse`	Indicates the response that Amazon Q provided to the user's prompt.
`followupPrompts`	Indicates the follow-up example prompts that were displayed to the user at the end of the response.
`messageMetadata`	Describes metadata associated with the response.
`conversationId`	Indicates the conversation ID of the response. The conversation ID groups together messages in a chat session.
`utteranceId`	Indicates the utterance ID of the response. An utterance ID is a label that distinguishes one prompt from another within a dialogue or data set.
`codeReferenceEvents`	Describes links to code references included in the response.
`supplementaryWebLinksEvent`	Indicates the links that were displayed to the user at the end of the response.
`requestId`	The ID of the response that uniquely identifies the event.

/dev log example

The following example shows a log file that is generated with a user enters a **`/dev`** command in the Amazon Q chat in their IDE.

```
{
    "records": [
        {
            "startTaskAssistCodeGenerationEventRequest": {
```

```
                "prompt": "write a python application that prints 'hello world!' text
 to the screen and format it in red bold text",
                "chatTriggerType": "MANUAL",
                "conversationId": "da1c95b6-84e1-46a2-9ef9-fe92f5ee169e",
                "customizationArn": null,
                "userId": "d-92675051d5.b8f1f340-9081-70ad-5fc5-0f37151937a6",
                "timeStamp": "2025-01-13T15:40:27.808027101Z"
            },
            "startTaskAssistCodeGenerationEventResponse": {
                "requestId": "e504f126-7197-4e3c-a046-1a10d5a3f3e0"
            }
        }
    ]
}
```


Field name	Description
`records`	Top-level field that contains a set of prompts and responses.
`startTaskAssistCodeGenerationEventRequest`	Describes the **`/dev`** prompt entered by the user in the chat window in their IDE.
`prompt`	Indicates the **`/dev`** prompt the user entered into the chat window.
`chatTriggerType`	`MANUAL` indicates that the user entered a prompt into the chat window, or clicked on one of the suggested questions in the chat window. `INLINE_CHAT` indicates that the user entered a prompt into the small input screen in the main coding window. For more information about chatting inline, see Chatting inline with Amazon Q Developer.
`conversationId`	Indicates the conversation ID of the response. The conversation ID groups together messages in a chat session.

Field name	Description
`customizationArn`	Indicates the Amazon Resource Name (ARN) of the customization applied to the chat. In this example, there is no customization. For
`userId`	Indicates the user ID of the user who entered the prompt.
`timeStamp`	Indicates the date and time that the user entered the prompt. The timestamp is in universal time (UTC).
`startTaskAssistCodeGenerationEventResponse`	Describes the response generated by Amazon Q. Currently, recording the responses to **`/dev`** commands is not supported, so the field won't include a response.
`assistantResponse`	Indicates the response that Amazon Q provided to the user's prompt.
`requestId`	The ID of the response that uniquely identifies the event.

Supported Regions for Amazon Q Developer

> **Note**
>
> If you make a request that requires Amazon Q Developer to retrieve information from an opt-in Region not listed on this page, Amazon Q can make calls to that Region. To manage access to Regions Amazon Q can make calls to, see Allow Amazon Q permission to perform actions on your behalf in specific regions.

This topic describes the AWS Regions where you can use Amazon Q Developer. For more information about AWS Regions, see Specify which AWS Regions your account can use in the *AWS Account Management Reference Guide*.

Your data may be processed in a different Region from the Region where you use Amazon Q Developer. For information on cross-region processing in Amazon Q Developer, see Cross-region processing. For information on where data is stored during processing, see Data protection.

Supported Regions (enabled by default)

Amazon Q Developer is available in the AWS Management Console, AWS Console Mobile Application, AWS website, AWS Documentation website, and integrated chat applications in the following AWS Regions. These Regions are enabled by default, meaning you don't need to enable them before use. For more information, see Regions that are enabled by default.

> **Note**
>
> The Amazon Q Developer service within the AWS Management Console is only available in the Regions outlined in Supported Regions for the Q Developer console and Q Developer profile. To manage Amazon Q Developer settings as an administrator, you must go to the Amazon Q Developer service and then use the Region selector to switch to a supported Region.
>
> You can chat and use other Amazon Q console features in the following Regions. Certain features of Amazon Q might not be available in all of these Regions. Check the topic for the feature you're using to verify availability.

- US East (Ohio)
- US East (N. Virginia)
- US West (N. California)
- US West (Oregon)
- Asia Pacific (Mumbai)
- Asia Pacific (Osaka)
- Asia Pacific (Seoul)
- Asia Pacific (Singapore)
- Asia Pacific (Sydney)
- Asia Pacific (Tokyo)
- Canada (Central)
- Europe (Frankfurt)
- Europe (Ireland)
- Europe (London)
- Europe (Paris)
- Europe (Stockholm)
- South America (São Paulo)

Supported opt-in Regions

To use an opt-in Region with Amazon Q Developer, you must enable the Region manually. For more information, see Opt-in Regions.

Opt-in Regions are only supported in the Amazon Q Developer Free tier. The following opt-in Regions are supported.

- Africa (Cape Town)
- Asia Pacific (Hong Kong)
- Asia Pacific (Hyderabad)
- Asia Pacific (Jakarta)
- Asia Pacific (Malaysia)
- Asia Pacific (Melbourne)

- Asia Pacific (Thailand)
- Canada West (Calgary)
- Europe (Milan)
- Europe (Spain)
- Europe (Zurich)
- Mexico (Central)
- Middle East (Bahrain)
- Middle East (UAE)
- Israel (Tel Aviv)

Troubleshoot Amazon Q Developer

This troubleshooting guide helps you resolve common issues when using Amazon Q Developer. Each section provides clear problem descriptions, possible causes, step-by-step solutions, and verification steps to get you back on track quickly.

Before diving into specific issues, try these general troubleshooting steps:

- Verify your internet connection is AWS stable
- Check that you're signed in to Amazon Q with valid credentials
- Ensure Amazon Q is up to date in your IDE
- Restart your IDE or type `/quit` to exit and restart the Amazon Q command line if issues persist

If these steps don't resolve your issue, refer to the specific troubleshooting sections below.

Topics

- Accessing and using Amazon Q Developer logs

Accessing and using Amazon Q Developer logs

Amazon Q Developer generates detailed logs that can help you diagnose and resolve issues. This guide shows you how to access logs for different Amazon Q interfaces and configure logging levels to get the information you need for troubleshooting.

Quick Navigation:

- Log Access Overview
- IDE Extension Logs
- Amazon Q CLI Logs
- Common Log Patterns and Solutions
- Getting Help with Log Analysis

Log access overview

There are two main ways to access Amazon Q Developer logs, depending on how you're using the service:

- **IDE Extensions** - VS Code and JetBrains IDEs have a "Show Logs" button for accessing Amazon Q specific logs
- **Command Line Interface (Amazon Q CLI)** - Logs are stored locally in temporary directories with configurable detail levels

> **Important**
>
> Log files may contain sensitive information from your conversations and interactions with Amazon Q, including file paths, code snippets, command outputs, account IDs, and resource names. Exercise caution when sharing log files with others.

IDE extension logs

Accessing logs through the IDE interface

1. Open the Amazon Q chat panel in your IDE (VS Code or JetBrains)
2. Click the **Show Logs** button in the chat panel header
3. Acknowledge the sensitivity warning that appears
4. The log file location will open in your system's file manager for review

Analyzing IDE extension logs

When reviewing IDE extension logs, look for:

- **Error messages** - Lines containing "ERROR" or "FATAL" indicate critical issues
- **Authentication issues** - Look for authentication or credential-related errors
- **Network connectivity** - Connection timeouts or network-related errors
- **Feature-specific errors** - Issues related to code suggestions, chat, or other specific features

Amazon Q CLI Logs

The Amazon Q CLI automatically generates comprehensive logs for all operations, regardless of verbosity settings. Logs are always written to files, while verbosity flags only control what appears in the terminal output.

Amazon Q CLI log locations and files

Amazon Q CLI logs are automatically stored in the following locations:

Operating System	Log Location
macOS	`$TMPDIR/qlog/` (typically `/var/folders/.../qlog/`)
Linux/WSL	`$XDG_RUNTIME_DIR/qlog/` or `$TMPDIR/qlog/` or `/tmp/qlog/`
Windows	`%TEMP%\qlog\`

The Amazon Q CLI creates multiple specialized log files automatically:

`chat.log` - **Main Amazon Q CLI wrapper logs** including:

- Amazon Q CLI initialization and startup operations
- AWS SDK calls (Cognito Identity, authentication flows)
- Network operations (HTTP/TLS connections, certificate handling)
- Low-level system operations (telemetry, socket connections)
- AWS service endpoint resolution and connection pooling
- Detailed debugging information for infrastructure components

`qchat.log` - **Chat application-specific logs** including:

- Chat application errors and state processing issues
- MCP (Model Context Protocol) server management and connection errors
- Application-level migration issues
- User interaction interruptions and chat processing failures
- Higher-level application logic errors

`mcp.log` - Model Context Protocol server logs (populated when using MCP servers)

`translate.log` - Natural language to shell translation logs (populated when using translate feature)

Key differences between log files

Scope and detail differences:

- `chat.log`: Comprehensive system-level logging covering the entire Q CLI infrastructure
- `qchat.log`: Focused application-level logging specific to chat functionality

Content focus differences:

- `chat.log`: AWS SDK internals, networking protocols, authentication flows, system operations
- `qchat.log`: Chat logic, MCP server lifecycle, user experience issues, application errors

> **Note**
>
> Log files are stored only on your local machine and are not sent to AWS. All log files are created automatically when you first use the CLI, even without verbose flags.

Amazon Q CLI troubleshooting workflow

Follow this approach to gather diagnostic information from the logs.

1. Identify the log directory for your system:

   On Linux/WSL:

   ```
   echo $XDG_RUNTIME_DIR/qlog/
   ```

   On macOS:

   ```
   echo $TMPDIR/qlog/
   ```

   On Windows:

```
echo %TEMP%\qlog\
```

2. Run the Amazon Q CLI command with maximum verbosity to see detailed output in your terminal:

```
q -vvv chat
```

3. Reproduce the issue you're experiencing
4. Exit the Amazon Q CLI and examine the relevant log files. For most issues, check both main log files:

   On macOS/Linux:

```
less -r $XDG_RUNTIME_DIR/qlog/qchat.log
less -r $XDG_RUNTIME_DIR/qlog/chat.log
```

   Alternative on macOS:

```
less -r $TMPDIR/qlog/qchat.log
less -r $TMPDIR/qlog/chat.log
```

   On Windows:

```
type %TEMP%\qlog\qchat.log
type %TEMP%\qlog\chat.log
```

5. For real-time log monitoring during troubleshooting, use:

   Monitor all log files simultaneously:

```
tail -f $XDG_RUNTIME_DIR/qlog/*.log
```

   Monitor specific files:

```
tail -f $XDG_RUNTIME_DIR/qlog/qchat.log
```

```
tail -f $XDG_RUNTIME_DIR/qlog/chat.log
```

Analyzing Amazon Q CLI logs

Amazon Q CLI logs use standard logging levels to categorize information by importance:

ERROR

Critical issues that prevent normal operation - start here when troubleshooting

WARN

Potential problems that don't stop functionality but should be noted

INFO

General operational messages about what the application is doing

DEBUG

Detailed technical information useful for deeper investigation

When examining Amazon Q CLI logs, focus on these key areas in the different log files:

`qchat.log` analysis - Application-level issues including:

- **ERROR chat_cli::cli::chat** - Chat processing and state management errors
- **ERROR chat_cli::cli::agent** - Migration and agent-related problems
- **ERROR chat_cli::telemetry** - Telemetry validation and transmission failures

`chat.log` analysis - Runtime operational details including:

- **DEBUG q_cli::cli** - Amazon Q CLI command execution and initialization
- **DEBUG aws_sdk_*** - AWS SDK operations and service calls
- **DEBUG rustls::*** - TLS/SSL connection establishment and certificate handling
- **DEBUG hyper_*** - HTTP connection management and network operations
- **ERROR fig_telemetry** - System telemetry and socket connection issues

General analysis tips:

- **Timestamps** - Correlate log entries with when issues occurred
- **Error patterns** - Look for repeated errors or error cascades
- **Request IDs** - Track specific API calls and their outcomes

- **Connection states** - Monitor network connectivity and authentication status

> **Tip**
>
> Use tools like **grep**, **awk**, or text editors with search functionality to filter logs for specific error messages or patterns. For example: **grep -i error $XDG_RUNTIME_DIR/qlog/*.log**

Common log patterns and solutions

Here are some common issues you might find in logs and their typical solutions:

MCP server connection errors

Log pattern (in qchat.log): "Background listening thread for client [server-name]: RecvError(Closed)" or "All senders dropped for transport layer"

Solution: MCP server processes have stopped running. This is typically expected behavior when exiting the Amazon Q CLI or when servers shut down normally.

Chat processing interruptions

Log pattern (in qchat.log): "An error occurred processing the current state err=Interrupted { tool_uses: None }"

Solution: This occurs when chat operations are cancelled by the user (e.g., Ctrl+C) and is expected behavior.

Telemetry validation errors

Log pattern (in qchat.log): "Failed to send cw telemetry event err=ValidationError [ValidationException]: Improperly formed request"

Solution: These are typically non-critical telemetry transmission issues that don't affect core functionality.

Migration warnings

Log pattern (in qchat.log): "Migration did not happen for the following reason: Aborting migration"

Solution: This is typically a non-critical warning related to configuration migration and can usually be ignored.

Authentication failures

Log pattern (in chat.log): Authentication-related errors in AWS SDK operations

Solution: Run **q login** to re-authenticate or check your AWS credentials

Network connectivity issues

Log pattern (in chat.log): "Connection timeout", "Network unreachable", or failed HTTP connections

Solution: Check your internet connection and firewall settings

AWS SDK operation failures

Log pattern (in chat.log): Failed Cognito Identity operations or credential retrieval errors

Solution: Check your AWS credentials and network connectivity. May require re-authentication

Getting help with log analysis

If you need assistance analyzing logs or resolving issues:

- When contacting support, include relevant log excerpts (with sensitive information removed)
- Provide context about when the issue occurs and steps to reproduce it

Amazon Q Developer rename - Summary of changes

On April 30, 2024, Amazon CodeWhisperer became a part of Amazon Q Developer. This section points you to the parts of this guide where you can find documentation for features that you are accustomed to using through CodeWhisperer.

As you transition from using CodeWhisperer to using Amazon Q Developer, you may consider the following changes to be most significant:

- The administrative setup at the professional tier (Amazon Q Developer Pro) is different than it was for CodeWhisperer Professional.
- You can chat with Amazon Q Developer in the AWS Management Console, and on the AWS documentation and marketing websites.

The following familiar features of CodeWhisperer are available as part of Amazon Q Developer, with a few changes:

- Coding suggestions in a third-party IDE
- Coding suggestions in the context of another AWS service
- Suggestions at the command line
-
- Code reviews
- Dashboard

Document history for Amazon Q Developer User Guide

The following table describes the document history for the *Amazon Q Developer User Guide*. For notifications about updates to this documentation, you can subscribe to the RSS feed.

Change	Description	Date
End of support notice	End of support notice: On April 30, 2027, AWS will discontinue support for Amazon Q Developer IDE plugins. For capabilities similar to Amazon Q Developer IDE plugins, explore Kiro to access the latest models and features, including agentic coding, chat and MCP support. For more information, see Amazon Q Developer IDE plugins end of support.	August 17, 2026
Updated Builder ID unsubscribe instructions	Added guidance for unsubscribing a personal account (Builder ID) when the Amazon Q Developer Profile has been deleted, including steps to re-enable via the console and a link to create a Kiro Profile.	March 19, 2026
Updated account limit for subscriptions	You can now subscribe users in a maximum of 20 accounts per AWS Region, per organization, increased from 10.	February 18, 2026

Deprecated code transformation for GitLab	Code transformation capabilities for Java modernization have been deprecated for GitLab Duo with Amazon Q Developer. The `/q transform` quick action and CI/CD pipeline customization documentation have been removed.	January 16, 2026
Removed deprecated GitLab quick actions	Removed the `/q dev (revise)` bullet point and the entire Unit test generation section with `/q test` command from the GitLab quick actions documentation.	January 13, 2026
Updated GitHub feature development workflow	Updated the GitHub feature development workflow to use conversation-based feedback with the `/q` command instead of the traditional GitHub review process. Users can now provide feedback through natural language commands in pull request conversations.	December 29, 2025
Updated GitHub app installation registration steps	Updated the Increasing usage limits and configuring details in Amazon Q Developer console topic to include steps for navigating the Kiro and Amazon Q Developer shared console.	December 18, 2025

Deprecated code transformation for GitHub	Code transformation capabilities for Java modernization have been deprecated for Amazon Q Developer for GitHub.	December 15, 2025
Q CLI becomes Kiro CLI	The Q CLI has become the Kiro CLI.	November 17, 2025
Updated managed policies: AmazonQFullAccess, AmazonQDeveloperAccess, and AWSServiceRoleForUserSubscriptions	Additional permissions have been added to the AmazonQFullAccess policy, AmazonQDeveloperAccess policy, and AWSServiceRoleForUserSubscriptions.	October 29, 2025
Removed IDE agent documentation	Agentic chat capabilities have replaced the /dev, /doc/, /test, and /review agents in the IDE. There are updated steps for code reviews in the IDE.	October 21, 2025
Placeholder title	Placeholder content.	October 21, 2025
Added IDE memory bank documentation	Added documentation for generating a memory bank for Amazon Q chat.	October 21, 2025
Added Q CLI feature documentation	Added documentation for creating agents with AI assistance, prompt management, responding to messages, and project rules.	October 3, 2025

Enhanced context management documentation	Added comprehensive guidance for choosing the right context approach with comparison table, decision flowchart, and best practices for agent resources, session context, and knowledge bases.	September 23, 2025
Added Q CLI v1.16.0 feature documentation	Added documentation for agent default behavior with priority-based selection , legacy MCP configuration support, new chat commands (`/tangent tail`, `/changelog`), and agent-related settings.	September 18, 2025
Context hooks deprecated	Context hooks are deprecated in favor of agent hooks. Existing configurations are automatically migrated to agent files.	September 17, 2025
Added remote MCP server support and built-in tools documentation	Added documentation for remote MCP servers with OAuth authentication and comprehensive built-in tools configuration.	September 17, 2025
Removed Amazon S3 buckets from allowlist table	You no longer need to allowlist Amazon S3 buckets for software development or unit test generation in the IDE.	September 9, 2025

New command line tool for transformation version	The most recent command line tool for transformation version includes bug fixes.	September 9, 2025
Updated Amazon Q Developer for GitHub review agent content	Amazon Q Developer for GitHub code reviews include a code review summary with threaded findings. You can interact with Amazon Q Developer in pull request comments about the findings.	September 2, 2025
Added prerequisites to Amazon Q Developer for GitHub review section	Amazon Q Developer for GitHub code reviews can only be initiated with Write, Maintain, or Admin role in GitHub.	September 2, 2025
Disabling MCP	You can disable MCP for your organization.	August 21, 2025
Updated proxy topic	Updated the Configuring a corporate proxy in Amazon Q topic to indicate that Proxy Auto-Configuration (PAC) files are not supported.	August 19, 2025
Updated the deployment options topic	Updated the Deployment options topic to indicate that you can subscribe users in a maximum of 10 accounts per AWS Region, per organization.	August 15, 2025

Added troubleshooting section and troubleshooting logs section	Added comprehensive troubleshooting documentation and detailed log access and analysis guide to help users diagnose and resolve issues with Amazon Q Developer features and functionality.	August 15, 2025
Added command line transformation job history section	Added documentation for the `qct history` command that allows users to view their transformation job history from the command line.	August 7, 2025
Updated a subscription topic	Updated the Viewing an aggregated list of Amazon Q Developer subscriptions topic with new instructions and corrections.	August 6, 2025
Added IDE transformation job history topic	Added documentation for the transformation job history feature, which allows users to view, manage, and retrieve artifacts from their recent Java transformation jobs in Visual Studio Code.	August 6, 2025
GitLab self-managed instance update	GitLab Duo with Amazon Q configuration requires self-managed instances with GitLab 17.11.0 or later.	August 5, 2025
Added a troubleshooting topic	Added a Can't see subscribed users troubleshooting topic.	August 1, 2025

Added steps for code reviews with agentic chat in VSC	Added procedures for running a code review and addressing code issues using agentic chat in Visual Studio Code.	July 31, 2025
Added model selection for IDE chat	Added a Selecting models topic with available models and instructions for selecting a model for chat in the IDE.	July 31, 2025
Added keyboard shortcuts topic	Added a Keyboard shortcuts for Amazon Q Developer topic that provides a comprehen sive list of keyboard shortcuts available across different IDEs and platforms.	July 31, 2025
Added custom agents feature	Added custom agents feature that allows you to customize Amazon Q Developer CLI for specific workflows by configuring which tools, permissions, and context are available for different use cases.	July 31, 2025
Added chat history compactio n topic	Added a Chat history compaction in Amazon Q Developer topic that explains how to manage context window limits when chatting with Amazon Q Developer in your IDE.	July 30, 2025
Added a profile deletion topic	Added a Deleting the Amazon Q Developer profile topic.	July 24, 2025

Added opt out steps for telemetry collection	Added instructions for how to opt out of telemetry collection while using the Amazon Q command line tool for transformation.	July 21, 2025
Added GitHub workflow for first-party dependencies content	Updated Customizing a workflow for code transformation section with new GitHub workflow for handling first-party dependencies page.	July 16, 2025
Updated the user activity metrics topic	Updated the User activity report metrics topic with additional metrics related to the inline chat feature.	July 15, 2025
Updated the proxy configuration section	Updated the Configuring a corporate proxy in Amazon Q to include instructions for Eclipse, JetBrains, and Visual Studio.	July 15, 2025
Image context support added to IDE chat	Amazon Q now supports using images as context in the IDE chat panel. This feature allows users to include images when prompting, enabling scenarios such as generating code from UI mockups or sequence diagrams.	July 9, 2025

Updated the Builder ID upgrade section	Added a note to the Upgrading a personal account (Builder ID) section to indicate that you cannot link multiple Builder IDs to a single AWS account.	July 7, 2025
Updated the profile sharing section	The Enabling profile sharing section incorrectly indicated that the profile settings in the management account would override those in member accounts when profile sharing was enabled. This statement has been removed, and the purpose of profile sharing has been clarified.	July 2, 2025
Added slash commands details to Amazon Q Developer for GitHub content	Amazon Q Developer for GitHub can be invoked with slash commands to perform feature development, code reviews, and code transform ation, as well invoke for help, which provides a link to to the Amazon Q Developer for GitHub content.	June 30, 2025

Updated details for QCT CLI in IDEs	Added details about maintaining security, pausing or canceling code transformation, new parameter for `qct transform` command, and LLM summary output in the IDE code transformation content. QCT CLI also now supports `https://q.eu-central-1.amazonaws.com/` .	June 26, 2025
New content for creating project rules in third-party platforms	You can create project rules in GitLab and GitHub for standards and best practices.	June 26, 2025
Updated KMS policy for third-party integration	Added `"kms:Decrypt"` , `"kms:DescribeKey"` , `"kms:Encrypt"` , `"kms:GenerateDataKey"` to KMS policy for managed access to Amazon Q Developer for third-party integration.	June 25, 2025
Added proxy configuration instructions	Added a Configuring a corporate proxy in Amazon Q for Visual Studio Code section.	June 25, 2025
Updated user activity reports	User activity reports are now generated at midnight UTC, and broken into several files with 1,000 users in each.	June 24, 2025

Updated the Amazon S3 bucket section	Updated the Amazon S3 bucket URLs and ARNs to allowlist section to indicate version 1.72.0 and later of the Visual Studio Code plugin for Amazon Q does not require allowlisting of buckets.	June 24, 2025
Rebranded Amazon Q Developer operational investigations	Updated various sections to reflect the rebrandin g of *Amazon Q Developer operational investigations* to *CloudWatch investigations*.	June 24, 2025
Updated GitHub feature development and code transformation details	Amazon Q Developer for GitHub can now be invoked to perform feature developme nt and transform code using slash commands in comments within an issue.	June 19, 2025
Rules button added to VS Code IDE chat	Added Rules button to the chat interface for creating and managing project rules through the UI.	June 18, 2025
Pinned context support added to VS Code IDE chat	Added context pinning functionality in VS Code to maintain selected context items across chat interacti ons, with support for manual pinning and system-added context management.	June 18, 2025

Updated details for code transformation in IDEs	Added code transformation details about source code version and target code version, and additional transformation requirements to upgrade project libraries and dependencies. Removed details about multiple diffs.	June 17, 2025
Chat with Amazon Q about network security	You can chat about network security with Amazon Q in the console and in chat applicati ons.	June 17, 2025
Added new Data storage topic	Added the Data storage in Amazon Q Developer topic with information about where content is stored.	June 13, 2025
MCP in the IDE	MCP servers can now be used with Amazon Q Developer in the IDE.	June 12, 2025
Personal account (Builder ID) upgrade	Users with personal accounts (Builder IDs) can now upgrade to the Pro tier.	June 11, 2025
Updates to subscription statuses	Updated the description of the **Unavailable** subscription status in Amazon Q Developer subscription statuses.	June 9, 2025
Support for specifying dependencies for transform ations on the CLI	You can provide a dependenc y upgrade file and modify the transformation plan for Java upgrades with the command line tool for transformations.	June 9, 2025

Expanded command line reference	A new command reference section provides more information on the arguments you can use to invoke the Q CLI.	June 9, 2025
Model selection for chat on the command line	You can now select the model you want Amazon Q to use for chat sessions on the command line.	June 5, 2025
Chat context support expanded to JetBrains, Visual Studio, and Eclipse IDEs	Amazon Q now supports contexts in the JetBrains, Visual Studio, and Eclipse IDE chat panel.	June 5, 2025
Agentic chat in all supported IDEs	Agentic chat is available in all supported IDEs.	June 5, 2025
Updated GitHub code review details	Amazon Q Developer for GitHub can now perform code reviews within GitHub pull requests. They can be initiated with the `/q review` slash command in a new comment.	June 2, 2025
Cost optimization capabilities in chat	In addition to cost analysis, you can chat with Amazon Q to get AWS cost optimization insights.	June 2, 2025
Updated the firewall page	Updated and added URLs on the Configuring a firewall, proxy server, or data perimeter for Amazon Q Developer Pro page.	May 22, 2025

MCP configuration commands	Commands that you can give directly from your non-Q command line to configure MCP servers for Amazon Q.	May 21, 2025
Updated the customization policy topic	The Allow administrators to create customizations topic now describes permission errors that may occur when you create the customization policy.	May 16, 2025
Updated troubleshooting tasks	The Troubleshooting Amazon Q Developer Pro subscriptions topic now includes more information.	May 15, 2025
Updated new code review behavior for GitLab Duo and GitHub	Automatic code reviews have been updated so they are triggered in new or reopened GitLab merge requests and GitHub pull requests. Code reviews are not triggered by subsequent commits.	May 15, 2025
Updated VPC endpoints	Updated the VPC endpoints on the Amazon Q Developer and interface endpoints (AWS PrivateLink) page.	May 15, 2025
Improved MCP server loading	MCP servers now load in the background, allowing you to start interacting with Amazon Q immediately without waiting for all servers to initialize.	May 15, 2025

Added image support in chat	Amazon Q can now analyze and discuss images directly in your chat session using the `fs_read` tool with the Image mode.	May 15, 2025
Added conversation persistence	Amazon Q automatically remembers your conversations based on the directory where they take place, and provides `/export` and `/import` commands for manually managing conversation state.	May 15, 2025
Updated managed policies: AmazonQFullAccess and AmazonQDeveloperAccess	Additional permissions have been added to the AmazonQFullAccess policy and AmazonQDeveloperAccess policy to manage conversation history.	May 14, 2025
Support for multiple conversations in console chat	You can save and switch between conversations with Amazon Q in the AWS console.	May 14, 2025
Added subscription details	The Subscribe users to Amazon Q Developer Pro across accounts topic now includes information about where to install IAM Identity Center and the Amazon Q Developer profile.	May 13, 2025

Transformation website removed	The Amazon Q Developer transformation website has been taken down, and with it, the related documentation.	May 12, 2025
Updated terminology	Updated the term *identity-aware console sessions* to *identity-enhanced console sessions* throughout this guide.	May 10, 2025
Example SCP added	This service control policy (SCP) denies access to Amazon Q outside of EU regions.	May 8, 2025
Amazon Q Developer for GitHub	Information about Amazon Q Developer for GitHub, including concepts and procedures focused on setting up, key features, and configuration.	May 5, 2025
Added context hooks	Added support for context hooks.	May 3, 2025
Managed policy updated	Permission has been added to AmazonQFullAccess.	May 2, 2025
Agentic chat in the IDE	Agentic chat functionality is available in the IDE.	May 1, 2025
Updates to customizations	Customizations now support additional languages.	April 30, 2025

Managed and example policies update	Permissions have been added to AmazonQFullAccess, GitLabDuoWithAmazonQPermissionsPolicy, and Allow administrators to configure plugins.	April 30, 2025
Added MCP support	Added support for MCP in the CLI.	April 29, 2025
Added upgrade details	Added information about how to upgrade from the Free tier to the Pro tier.	April 28, 2025
Support for conversation history	Your conversation history is now saved when you chat with Amazon Q in the IDE.	April 21, 2025
Support for code as context	You can now specify classes, functions, and global variables as context when you chat with Amazon Q in the IDE.	April 21, 2025
Updates to GitLab Duo with Amazon Q onboarding and policy	GitLab Duo with Amazon Q has been updated with changes to onboarding and permissions policy (GitLabDuoWithAmazonQPermissionsPolicy).	April 16, 2025
Updated dashboard permissions	Updated the list of permissions required to view the Amazon Q Developer dashboard.	April 15, 2025

Improved security documentation for command line	Reorganized and enhanced security documentation with comprehensive guidance on security considerations, best practices, and safe usage of tool permissions.	April 13, 2025
Enhanced command-line security and settings	Added a new command-line settings section with configuration options. Enhanced tool permissions documentation with security best practices for sensitive environments.	April 12, 2025
Update to the subscription experience	The workflow to subscribe users to Amazon Q Developer Pro and install the Amazon Q Developer profile has been moved from the Amazon Q console to the Amazon Q Developer console.	April 10, 2025
Inline chat is available in Eclipse	You can chat inline with Amazon Q in Eclipse.	April 10, 2025
Amazon Q Developer profiles are available in Europe (Frankfurt)	When you subscribe to Amazon Q Developer, you can create profiles in the Europe (Frankfurt) Region.	April 10, 2025
/tools feature added to CLI	You can use the `/tools` command to manage permissions for tools that Amazon Q uses to perform actions on your system.	April 10, 2025

Support for natural languages other than English	You can chat with Amazon Q in the IDE and on the command line.	April 9, 2025
Updates to GitLab Duo with Amazon Q	GitLab Duo with Amazon Q has been updated regarding changes to inline policy, and you can optionally create a CMK policy. The `/fix` feature has been removed.	April 8, 2025
Email notifications for transformations	You may receive email notifications for updates related to your transform ations.	April 8, 2025
New context, prompt, and project rules topics	The Adding context to the chat, Saving prompts, and Creating project rules topics have been added.	April 4, 2025
Updates to subscriptions topics	The Understanding subscript ions, Viewing aggregated subscriptions, and Enabling profile sharing topics have been corrected.	March 25, 2025
Example policy update	The example policies in Allow administrators to use the Amazon Q console and Allow administrators to use the Amazon Q Developer console have been updated with the `sso:CreateInstance` permission.	March 24, 2025
Support for C++ and C# in customizations	Customizations now support C++ and C#.	March 20, 2025

Updates to chatting about resources	You can chat with Amazon Q about multiple AWS resources and services to get answers about your AWS infrastru cture and configurations.	March 13, 2025
Additional language support for documentation generation	The agent for documentation generation now supports C++ and C#.	March 12, 2025
New subscription-related limit	Updated the Subscribing users to Amazon Q Developer Pro topic to indicate that you can enable Amazon Q Developer in a maximum of 50 AWS accounts within an organization managed by AWS Organizations.	March 6, 2025
Context integration to CLI chat	Amazon Q CLI now has context integration, giving Amazon Q enhanced understanding of use cases and enabling it to provide more relevant and context-a ware responses.	March 6, 2025
Policy correction	A JSON sytax error has been corrected in the policy described in Allow administr ators to use the Amazon Q console.	February 28, 2025

New version of the command line tool for transformation	The latest version of the command line tool for transformation includes support for authenticating with IAM through the AWS CLI.	February 28, 2025
Upgrading to the Pro tier	Added information about how to upgrade to the Pro tier in the Amazon Q Developer Free tier topic.	February 25, 2025
Customizations policy update	A permission has been added to the customizations policy.	February 25, 2025
New dashboard topic	The following topic has been added: Descriptions of Amazon Q Developer dashboard usage metrics.	February 21, 2025
New cross-region processing topic	The cross-region processing topic describes how Amazon Q Developer processes requests and makes calls across AWS Regions to provide the service.	February 21, 2025
Managed policy update	Permissions have been added to AWSServiceRoleForUserSubscriptions.	February 21, 2025
/doc enhancement	Amazon Q can now generate infrastructure diagrams in response to a **/doc** command.	February 20, 2025

New subscription topics	Two subscription-related topics were added: Amazon Q Developer subscription statuses and Viewing aggregated Amazon Q Developer subscriptions.	February 19, 2025
Amazon Q Developer in chat applications chapter	Amazon Q Developer in chat applications is now Amazon Q Developer in chat applications. A new chapter describes the supported features.	February 19, 2025
Support for Java 21 transformations	You can upgrade Java applications to Java 21 in the IDE and on the command line.	February 14, 2025
New firewall topic	A Configuring a firewall or proxy server for Amazon Q Developer topic has been added.	February 14, 2025
New version of the command line tool for transformation	The latest version of the command line tool for transformation includes support for converting embedded SQL in Java applications.	February 12, 2025
User activity report correction	The path to the user activity report CSV file has been corrected.	February 10, 2025
Update to the retention period of transformed code	Amazon Q now retains transformed code for 30 days, up from 24 hours.	February 7, 2025

New subscription workflow	The steps to subscribe users to Amazon Q Developer have been improved.	February 6, 2025
New version of the command line tool for transformation	The latest version of the command line for transform ation includes the ability to receive your upgraded Java code in multiple commits.	February 3, 2025
/dev enhancement	Amazon Q can now test the code it generates in response to a **/dev** command.	January 31, 2025
Customizations section update	The Creating your customiza tion topic now indicates you can include any number of repositories in your customiza tion.	January 24, 2025
Prompt logging examples	The Enabling prompt logging section now includes example logs.	January 23, 2025
CloudZero plugin	The CloudZero plugin is available in Amazon Q chat.	January 15, 2025
User activity report update	New metrics have been added to User activity reports.	December 16, 2024
Dashboard update	Information about the old dashboard has been removed from the Amazon Q Developer Pro dashboard section. Information about filters and metrics has been added.	December 16, 2024

Troubleshooting with Amazon Q	An Asking Amazon Q to troubleshoot your resources section has been added.	December 13, 2024
Identity-enhanced sessions update	The instructions for enabling identity-enhanced console sessions have been clarified in the Subscribing users to the Amazon Q Developer Pro tier with an organization instance section.	December 6, 2024
New test generation agent	You can use Amazon Q test generation feature to generate unit tests.	December 3, 2024
Large-scale transformation	Amazon Q can transform .NET, mainframe, and VMware workloads in bulk.	December 3, 2024
GitLab Duo with Amazon Q	Information about GitLab Duo with Amazon Q, including concepts, getting started procedures, and troublehs ooting.	December 3, 2024
Documentation generation in the IDE	Amazon Q can generate READMEs for your code in supported IDEs.	December 3, 2024
Code reviews in the IDE	Amazon Q code reviews, previously security scans, can detect and address issues in your code in supported IDEs.	December 3, 2024

.NET transformation in the IDE	Amazon Q can port your .NET applications to Linux-compatible cross-platform applications in Visual Studio, available in preview.	December 3, 2024
Transformation on the command line	You can transform Java applications on the command line, available in preview.	November 27, 2024
Multiple diffs for transformation in the IDE	You can choose to receive transformation changes from Amazon Q in multiple diffs.	November 27, 2024
Amazon Q in Eclipse	The Amazon Q plugin is available in preview in Eclipse.	November 27, 2024
Cost analysis	The cost analysis capability, previously available in preview, is now generally available.	November 26, 2024
Transformation for embedded SQL code	You can convert embedded SQL code in your Java applications with Amazon Q transformation in the IDE.	November 22, 2024
Dashboard update	The Amazon Q Developer Pro dashboard has been update with new metrics.	November 22, 2024
AWS CodeConnections repositories	When creating a customization using a AWS CodeConnections connection, you can now choose the repositories you want to use.	November 22, 2024

Amazon Q command line now supports Linux	Amazon Q command line supports Linux environments. It supports Ubuntu 22 and 24, and may otherwise work with GNOME v42+ or environments where the display server is Xorg and the input method framework is IBus.	November 21, 2024
Subscribing users	The instructions for subscribing users in Setting up access to the Amazon Q Developer Pro tier have been updated to reflect new user interface (UI) elements.	November 20, 2024
Changes to customizations	The Customization in chat feature is now generally available. Also, customizations can now be created with the following file types: `.md`, `.mdx`, `.rst`, and `.txt`.	November 20, 2024
Supported IAM Identity Center Regions	A section has been added with information about the Regions where you can set up IAM Identity Center instances for Amazon Q Developer Pro subscriptions.	November 18, 2024
Languages added	Support has been added for Dart, Lua, R, Swift, SystemVerilog, and Powershell, as well as expanded support for JSON and YAML.	November 18, 2024

Customer managed key support	Information about using customer managed keys and the features that can be encrypted with them has been added to the Data encryption topic.	November 18, 2024
Cross-region inference	A topic on cross-region inference in Amazon Q Developer has been added.	November 18, 2024
Amazon Q Developer Pro quotas	A Pro tier quotas section has been added.	November 18, 2024
Updated managed policy: AmazonQFullAccess	Additional permissions have been added to the AmazonQFullAccess policy.	November 13, 2024
Updated managed policy: AmazonQDeveloperAccess	Additional permissions have been added to the AmazonQDeveloperAccess policy.	November 13, 2024
Amazon Q plugins	Plugins enable users to chat with Amazon Q about metrics provided by third party tools.	November 13, 2024
User activity reports	You can now enable user activity reports.	November 8, 2024
Customizations section update	The Preparing your data section now describes file and directory naming limitations.	November 5, 2024
Clarified the Amazon Q Developer Pro section	The instructions for subscribing users to Amazon Q Developer Pro have been clarified.	November 1, 2024

Inline chat	You can transform code using the new inline chat feature.	October 29, 2024
Updated managed policies: AmazonQFullAccess and AmazonQDeveloperAccess	Additional permissions have been added to the AmazonQFullAccess policy and AmazonQDeveloperAccess policy.	October 28, 2024
Customizations section correction	The Creating your customization section now indicates that your codebase must reside in a folder in Amazon S3, not the bucket's root.	October 28, 2024
Prompt logging section clarification	The Enabling prompt logging section's wording was clarified .	October 24, 2024
Amazon S3 bucket policy fix	The Amazon S3 bucket policy shown in Enabling prompt logging contained a JSON syntax error that was fixed.	October 22, 2024
Expanded features chapter	The chapter describing various Amazon Q Developer features has been significantly expanded.	October 3, 2024
Console-to-Code	Console-to-Code, previously available in preview as a feature of Amazon EC2, is now generally available as a feature of Amazon Q Developer. It integrates with Amazon EC2, Amazon VPC, and Amazon RDS.	October 3, 2024

New policy: Use Amazon Q CLI with AWS CloudShell	Identity-based policy allows users to use Amazon Q CLI with AWS CloudShell.	October 2, 2024
Prompt logging	You can log your users' IDE prompts in an Amazon S3 bucket.	September 16, 2024
Setup content updated	The Getting started chapter has been significantly simplified and restructured.	August 15, 2024
CodeWhisperer endpoint needed for IDE VPC access	Access from a Amazon VPC must include both q and `codewhisperer` endpoints.	July 18, 2024
New endpoint	Endpoints can now use the string q instead of `codewhisperer` .	July 12, 2024
Customizations are GA	The customizations feature is generally available.	July 10, 2024
Chatting about customizations (Preview)	In Preview, you can use the customizations feature to ask questions related to your codebase.	July 10, 2024
Updated managed policy: AmazonQFullAccess	Additional permissions have been added to the AmazonQFullAccess policy.	July 9, 2024
New managed policy: AmazonQDeveloperAccess	The AmazonQDeveloperAccess managed policy provides full access to enable interactions with Amazon Q Developer, without administrator access.	July 9, 2024

Updated Amazon Q Developer admin policy	The policy for empowering Amazon Q Developer administrators has been updated to include `sso:ListProfiles` .	June 19, 2024
Trusted access section	A new section more clearly explains how a Amazon Q Developer administrator can share settings with member accounts.	June 19, 2024
Updated setup procedures	There's an improved Getting started chapter that includes support for account instances.	June 6, 2024
Updated code examples	The code examples now include C and C++, and have improved examples for C#.	June 6, 2024
Updated managed policy: AmazonQFullAccess	Additional permissions have been added to the AmazonQFullAccess policy.	April 30, 2024
New service-linked role: AWSServiceRoleForUserSubscriptions	The AWSServiceRoleForUserSubscriptions service-linked role provides access for User Subscriptions to your IAM Identity Center resources to automatically update your subscriptions.	April 30, 2024
New service-linked role: AWSServiceRoleForAmazonQDeveloper	The AWSServiceRoleForAmazonQDeveloper service-linked role grants permission to access and emit data, and to create reports.	April 30, 2024

New managed policy: AWSServiceRoleForUserSubscriptionPolicy	The AWSServiceRoleForUserSubscriptionPolicy allows principals to track IAM Identity Center directory and AWS Organizations changes.	April 30, 2024
New managed policy: AWSServiceRoleForAmazonQDeveloperPolicy	The AWSServiceRoleForAmazonQDeveloperPolicy allows Amazon Q Developer to call CloudWatch and CodeGuru on your behalf.	April 30, 2024
GA release	Amazon Q Developer is available for general audiences.	April 30, 2024
Amazon CodeWhisperer merge	Amazon CodeWhisperer is now a part of Amazon Q Developer.	April 30, 2024
New guide name	This service and accompanying user guide have been renamed Amazon Q Developer.	March 29, 2024
New permission	The ListConversations action is required to chat with Amazon Q in the console.	March 5, 2024
New data protection topic	Amazon Q now uses content for service improvement purposes.	January 25, 2024

New topic	Added instructions for how to add Amazon Q to Slack and Microsoft Teams channels that are configured with Amazon Q Developer in chat applications.	January 18, 2024
Preview release	This is the initial preview release of the *Amazon Q Developer User Guide*.	November 28, 2023