

Amazon Q Developer

The most capable generative AI-powered assistant for software development



Michael Lin

Sr. Solutions Architect
Amazon Web Services



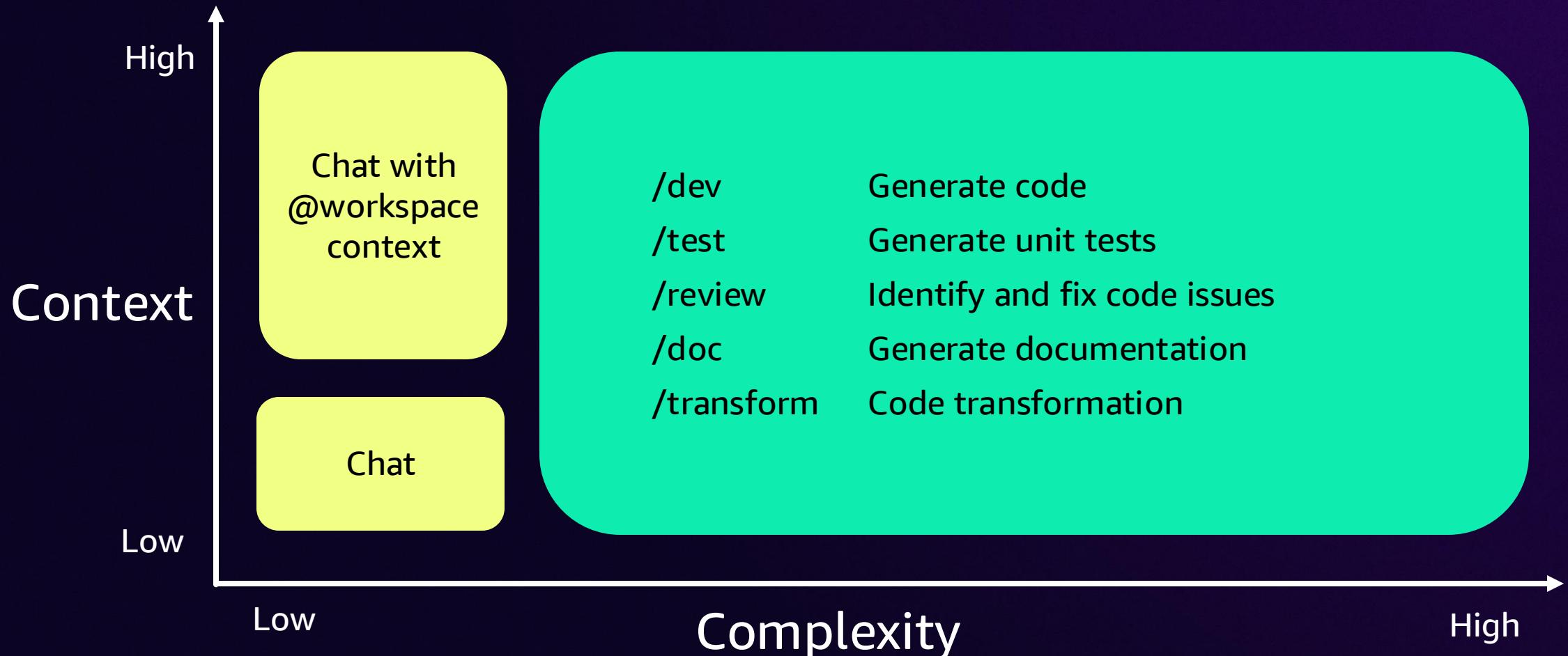
© 2025, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AI coding assistants' adoption

By 2028, 75% of enterprise software engineers will use AI code assistants, up from less than 10% in early 2023.

Gartner® Press Release, Gartner Says 75% of Enterprise Software Engineers Will Use AI Code Assistants by 2028, April 11, 2024,
<https://www.gartner.com/en/newsroom/press-releases/2024-04-11-gartner-says-75-percent-of-enterprise-software-engineers-will-use-ai-code-assistants-by-2028>. GARTNER is a registered trademark and service mark of Gartner, Inc. and/or its affiliates in the U.S. and internationally and is used herein with permission. All rights reserved.

Positioning of Amazon Q Developer and agents



Interactive chat

- You can ask Q for Developer open-ended questions, like this example, "*How do I use React.js with .NET?*"
- Ask questions about AWS services and features
- Refactor and optimize your code
- Get guidance on architectural best practices
- Q shows you the sources it leveraged when generating your answer.



For example, to set up a basic integration using ReactJS.NET in an ASP.NET MVC application, you would:

1. Install the ReactJS.NET NuGet package
2. Configure ReactJS.NET in your application startup
3. Create React components

4. Render the component

Here's a simple example of rendering a React component in a Razor view:

```
@using React.AspNet

@{
    var props = new
    {
        name = "World"
    };
}

<div id="content">
    @Html.React("HelloWorld", props)
</div>

@section scripts {
    <script src="https://cdnjs.cloudflare.com/ajax/libs/react/16.8.0/umd/react.development.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/react-dom/16.8.0/umd/react-dom.development.js"></script>
    <script src="@Url.Content("~/Scripts/HelloWorld.jsx")"></script>
    @Html.ReactInitJavaScript()
}
```

csharp

Insert at cursor

Copy

This example assumes you have a `HelloWorld.jsx` component defined in your project.

Remember that the specific implementation details may vary depending on your project structure, requirements, and the version of .NET and React you're using.

[1] React Server-side Rendering with ReactJS.NET | by INDRAJITH EKANAYAKE | Bits and Pieces

[blog.bitsrc.io > react-server-side-rendering-with-reactjs-net-17177bc0caa0](http://blog.bitsrc.io/react-server-side-rendering-with-reactjs-net-17177bc0caa0)

[2] Ideation of an IDE: The .NET and ReactJS Love Affair | by João Neves | OutSystems Engineering | Medium

[medium.com > @joaompneves > ideation-of-an-ide-the-net-and-reactjs-love-affair-dc29a9dd26b6](https://medium.com/@joaompneves/ideation-of-an-ide-the-net-and-reactjs-love-affair-dc29a9dd26b6)

Inline chat

- Describe issues or ask Q to refactor without leaving your code (using **Ctrl+I** / **⌘ + I**)
- Ask questions about AWS services and features
- Refactor and optimize your code
- As Q to add comments to document your code
- Available now for VS Code and JetBrains Rider, on the roadmap for Visual Studio

The screenshot shows a code editor interface with two panes. The top pane displays a tooltip with the text "Please add comments to document this code". The bottom pane shows the code for the `MoveWishListItemToShoppingCart` method in the `ShoppingCart.cs` file. The code is as follows:

```
public void MoveWishListItemToShoppingCart(int shoppingCartItemID)
{
    var wishListItem = ShoppingCartItems.SingleOrDefault(x => x.Id == shoppingCartItemID);

    if (wishListItem == null) return;

    wishListItem.WantToBuy = true;
    // Moves an item from the wishlist to the shopping cart by changing its WantToBuy flag
    // </summary>
    // <param name="shoppingCartItemID">ID of the wishlist item to move</param>
}

// Try to find the wishlist item with the given ID
var wishListItem = ShoppingCartItems.SingleOrDefault(x => x.Id == shoppingCartItemID);

// If item not found, exit early
if (wishListItem == null) return;

// Mark the item as "want to buy" to move it to shopping cart
wishListItem.WantToBuy = true;
```

The code editor has several icons in the sidebar, including a magnifying glass, a file icon with a blue circle containing the number 3, a link icon, a copy icon, a square icon with a blue circle containing the number 4, a monitor icon, a flask icon, a ship icon, an AWS logo, and a hexagon icon.



Amazon Q dashboard

METRICS WE TRACK

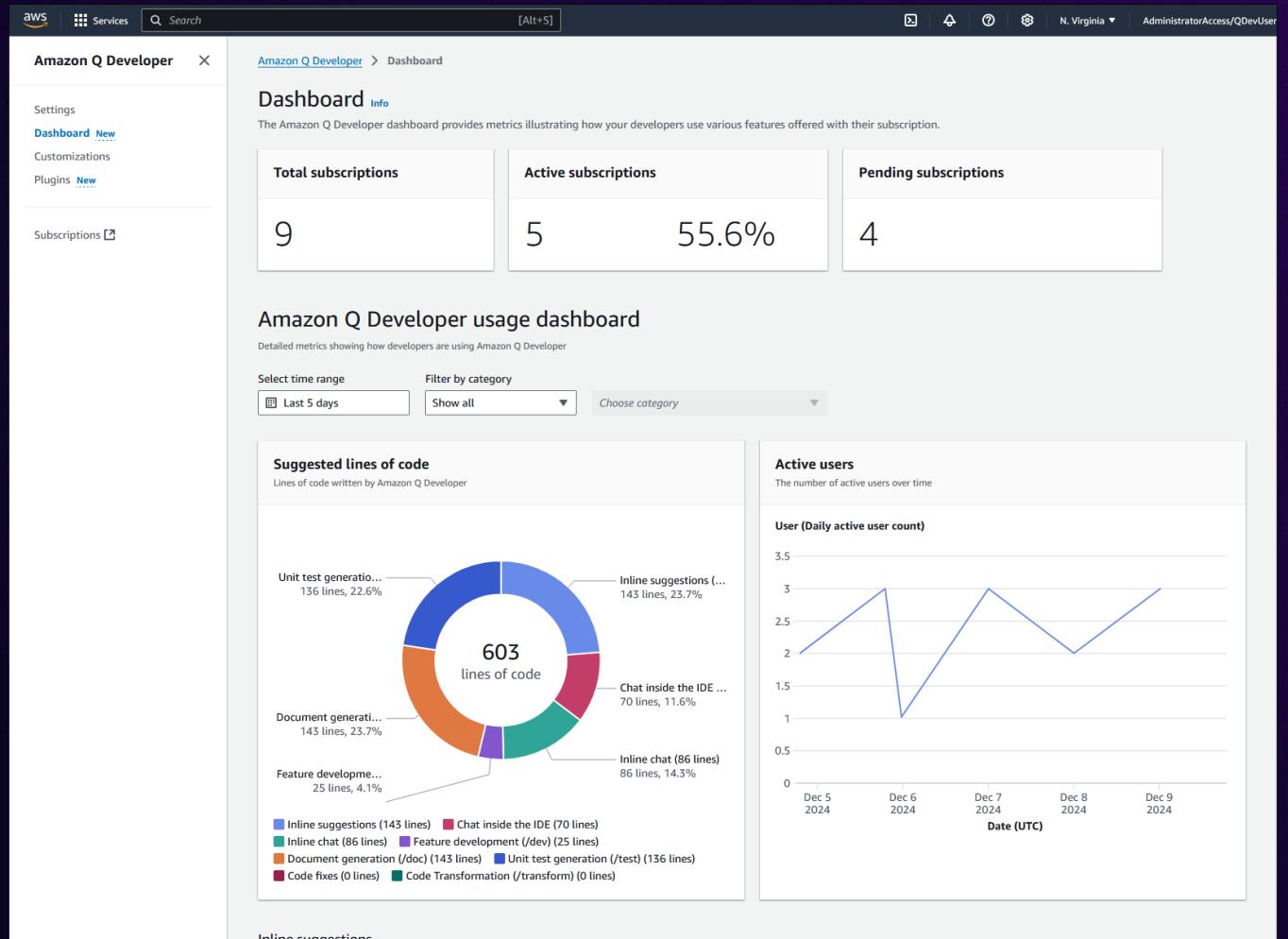
User activity

Lines of code generated

Acceptance rate

Percentage of code written
by Amazon Q Developer

Accepted recommendations
with references



Enhance product development with Amazon Q



BT Group

BT Group, one of the world's leading communications services companies, is using Amazon Q Developer's coding assistant **to improve developer productivity at BT**. Their developers are receiving an **average of 15-20 code suggestions per day, with an acceptance rate of 37 percent**, which is higher than most other published reports of similar statistics. In its first four months, **BT says Amazon Q Developer has generated over 100,000 lines of code and automated approximately 12 percent of repetitive work for an initial group of volunteers**.

Source - <https://www.telecomstechnews.com/news/2024/feb/20/bt-developer-productivity-amazon-generative-ai-coding-assistant/>

Customize code recommendations

The screenshot shows a Java code editor with a file named `FoodDeliveryFulfillment.java`. The code is annotated with several callouts:

- A yellow bar highlights line 2: `// Process a list of unassigned food deliveries around the driver's current location`.
- A green bar highlights line 15: `List<Delivery> unassignedDeliveries = deliveryService.getUnassignedDeliveries();`.
- A blue bar highlights line 22: `boolean isAssigned = deliveryService.assignDeliveryToDriver(delivery, driver);`.
- A red bar highlights line 25: `deliveryService.notifyDelivery(delivery);`.
- A purple bar highlights line 26: `driverLocationService.notifyDriver(driver);`.
- A green bar highlights line 27: `break;`.
- A green bar highlights line 28: `}`.
- A green bar highlights line 29: `}`.
- A green bar highlights line 30: `}`.
- A green bar highlights line 34: `}`.

Annotations on the right side of the code editor:

- Natural language prompt**: A green callout pointing to the highlighted code.
- Generated in-line code based on your company's internal code bases**: A green callout pointing to the highlighted code.

```
1 // Process a list of unassigned food deliveries around the driver's current location
2
3 package anycompany.fooddelivery.fulfillment;
4 import anycompany.delivery.Delivery;
5 import anycompany.delivery.DeliveryService;
6 import anycompany.driver.Driver;
7 import anycompany.driver.DriverLocationService;
8
9 public class FoodDeliveryFulfillment {
10     private DeliveryService deliveryService;
11     private DriverLocationService driverLocationService;
12
13     //Process all the unassigned deliveries
14     public void processUnassignedDeliveries() {
15         List<Delivery> unassignedDeliveries = deliveryService.getUnassignedDeliveries();
16         //Iterate over all the unassigned deliveries and assign them to a driver
17         for (Delivery delivery : unassignedDeliveries) {
18             //Get the nearest drivers for delivery location
19             List<Driver> drivers = driverLocationService.getDriver(delivery.getLocation());
20             for (Driver driver : drivers) {
21                 //Assign delivery to driver and send notification
22                 boolean isAssigned = deliveryService.assignDeliveryToDriver(delivery, driver);
23                 if (isAssigned) {
24                     deliveryService.notifyDelivery(delivery);
25                     driverLocationService.notifyDriver(driver);
26                     break;
27                 }
28             }
29         }
30     }
31
32     ...
33 }
34 }
```

Generate code based on your private repositories

Use Amazon Q Developer chat in the IDE to ask questions about your internal code base

Help understand your internal code base

Customized code suggestions with Amazon Q



National Australia Bank

“

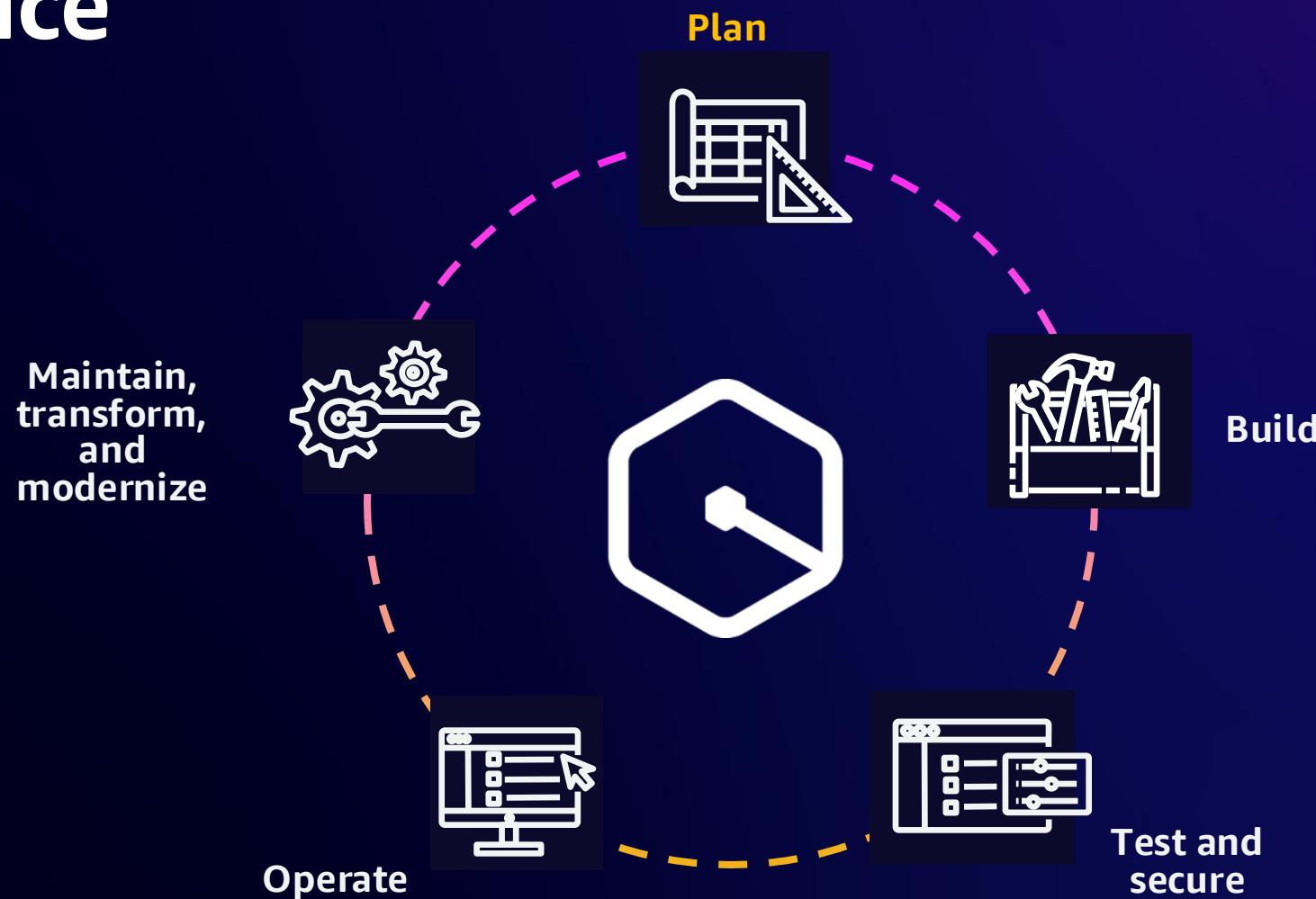
At NAB, our developers use Amazon Q Developer's generative AI-powered assistant to generate real-time coding suggestions. NAB has now added specific suggestions using the Amazon Q customisation capability that are tailored to the NAB coding standards. **We have 500 developers using Amazon Q Developer with customisation enabled. We're seeing increased acceptance rates of 60%** with customisation, and we expect this to grow with Amazon Q Developer chat, which makes it even easier to interact using natural language. ”

Andrew Brydon

Executive for Enterprise Engineering, National Australia Bank



Amazon Q Developer transforms the development experience

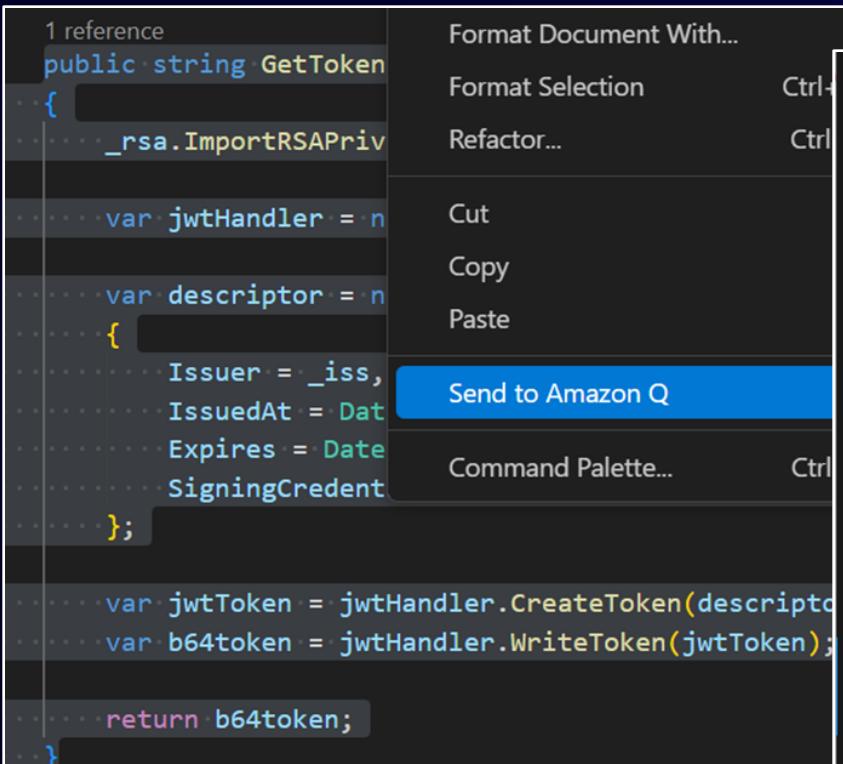


Code understanding



Code explanation

HAVE Q FOR DEVELOPER EXPLAIN LEGACY CODE WRITTEN BY DEVS LONG GONE...



The provided code block is a method named `GetToken()` that generates a JSON Web Token (JWT) using the RSA algorithm. Here's a breakdown of what the code is doing:

1. The method starts by importing the RSA private key into an `RSACryptoServiceProvider` object named `_rsa`.
2. It then creates a `JwtSecurityTokenHandler` object, which is used to create and manipulate JWT tokens.
3. Next, it creates a `SecurityTokenDescriptor` object, which contains the following properties:
 - `Issuer`: The issuer of the JWT token, stored in the `_iss` field.
 - `IssuedAt`: The time the token was issued, set to 30 seconds before the current time.
 - `Expires`: The time the token expires, set to 10 minutes after the current time.
 - `SigningCredentials`: The signing credentials used to sign the JWT token, which include the RSA private key and the SHA256 algorithm.
4. The `JwtSecurityTokenHandler` is then used to create a JWT token based on the `SecurityTokenDescriptor`.
5. Finally, the method converts the JWT token to a base64-encoded string and returns it.

This code is likely used to generate a JWT token that can be used for authentication or authorization purposes, such as in a web application or API. The RSA private key is used to sign the token, ensuring its integrity and authenticity.



© 2025, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Code completion

PROMPT Q FOR DEVELOPER WITH A COMMENT OR CODE SNIPPET

```
57     return order;
58 }
59
60 // PUT: api/Orders/1234-5678-9abc
61 [HttpPost("{id}")]
62 [Authorize(Policy = "IsRegisteredUser")]
63 0 references
64 public async void Put(Guid id)
65 {
66     //Update existing Order
67     var context = new DynamoDBContext(_ddbClient);
68     await context.SaveAsync(id);
69 }
70
71 // DELETE: api/ApiWithActions/{id}
72 [HttpDelete("{id}")]
73 [Authorize(Policy = "IsRegisteredUser")]
74 0 references
75 public async void Delete(Guid id)
```

Based on a comment (VS Code example)

```
53     // POST: api/Products
54     [HttpPost]
55     [Authorize(Policy = "IsSiteAdmin")]
56     - references
57     public async Task<string> Post([FromBody]Product product)
58     {
59         var context = new DynamoDBContext(_ddbClient);
60         await context.SaveAsync(product);
61
62         //TODO save the product.productImage byte array to S3 as image ({productId}.jpg)
63
64         _logger.LogInformation("New product {0} created by user {1}", product.Name, User.Identity.Name);
65
66         return product.ProductId;
67     }
68
69     - references
70     private void SaveImageToAmazonS3(byte[] image, string bucket, string key) ← prompt
71     {
72         Suggestion 1 / 1 | Tab to accept
73         var transferUtility = new TransferUtility(_s3Client);
74
75         var stream = new MemoryStream(image);
76         transferUtility.Upload(stream, bucket, key);
77     }
78 }
```

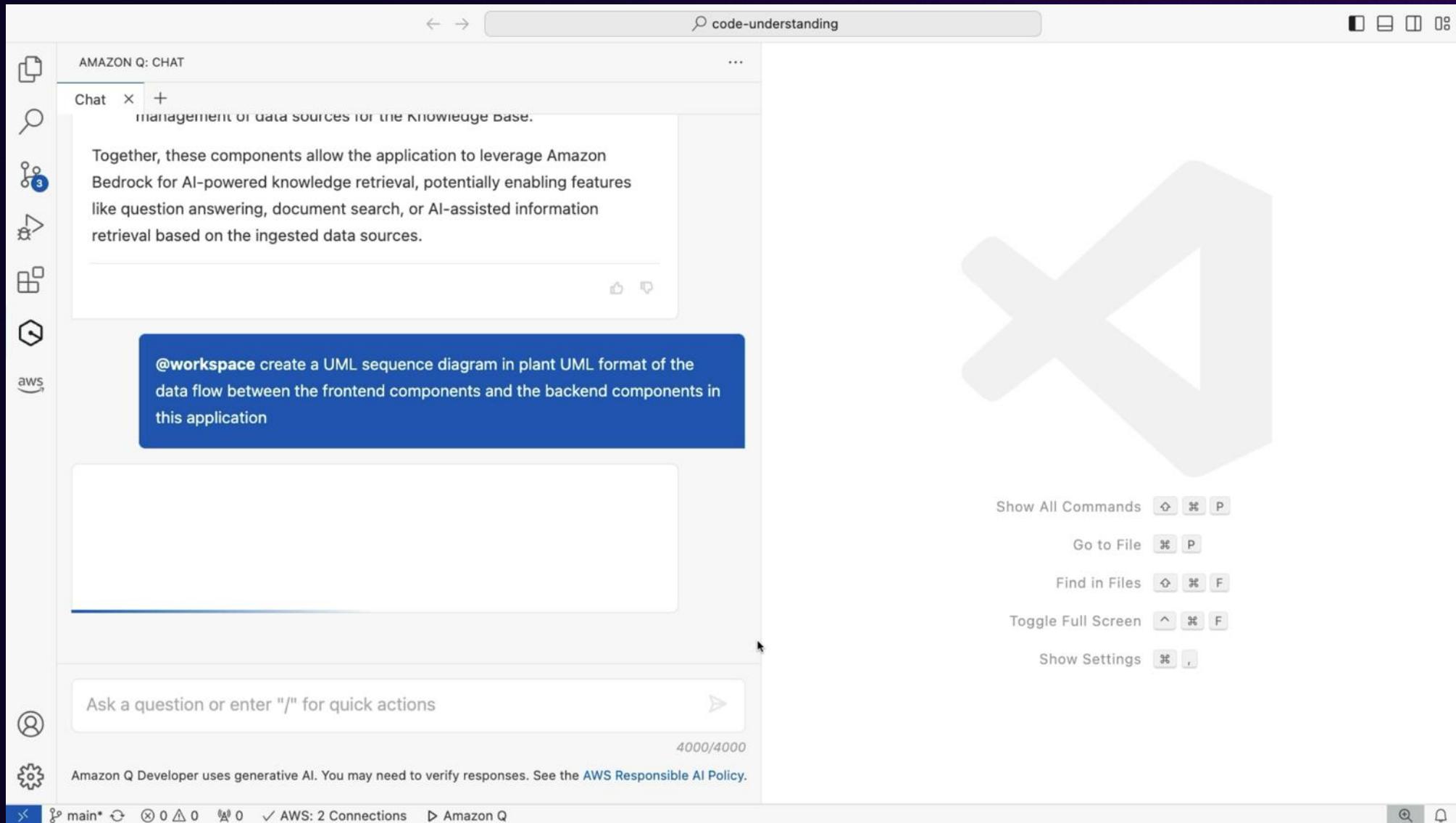
Based on code snippets you've written
(Visual Studio 2022 example)

Visualization

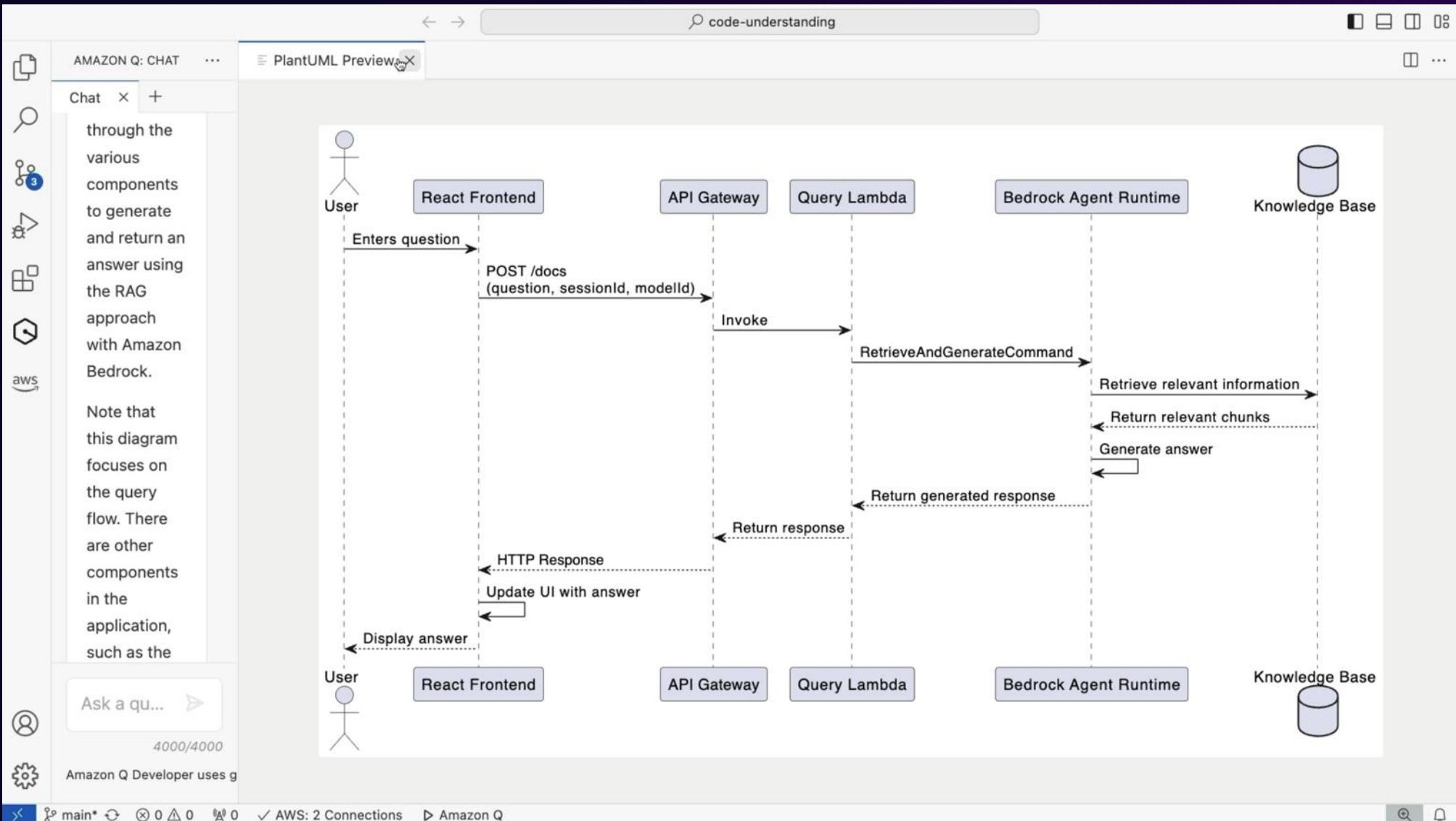


© 2025, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Create a sequence diagram with PlantUML



Create a sequence diagram with PlantUML



Visualize CloudFormation with Mermaid

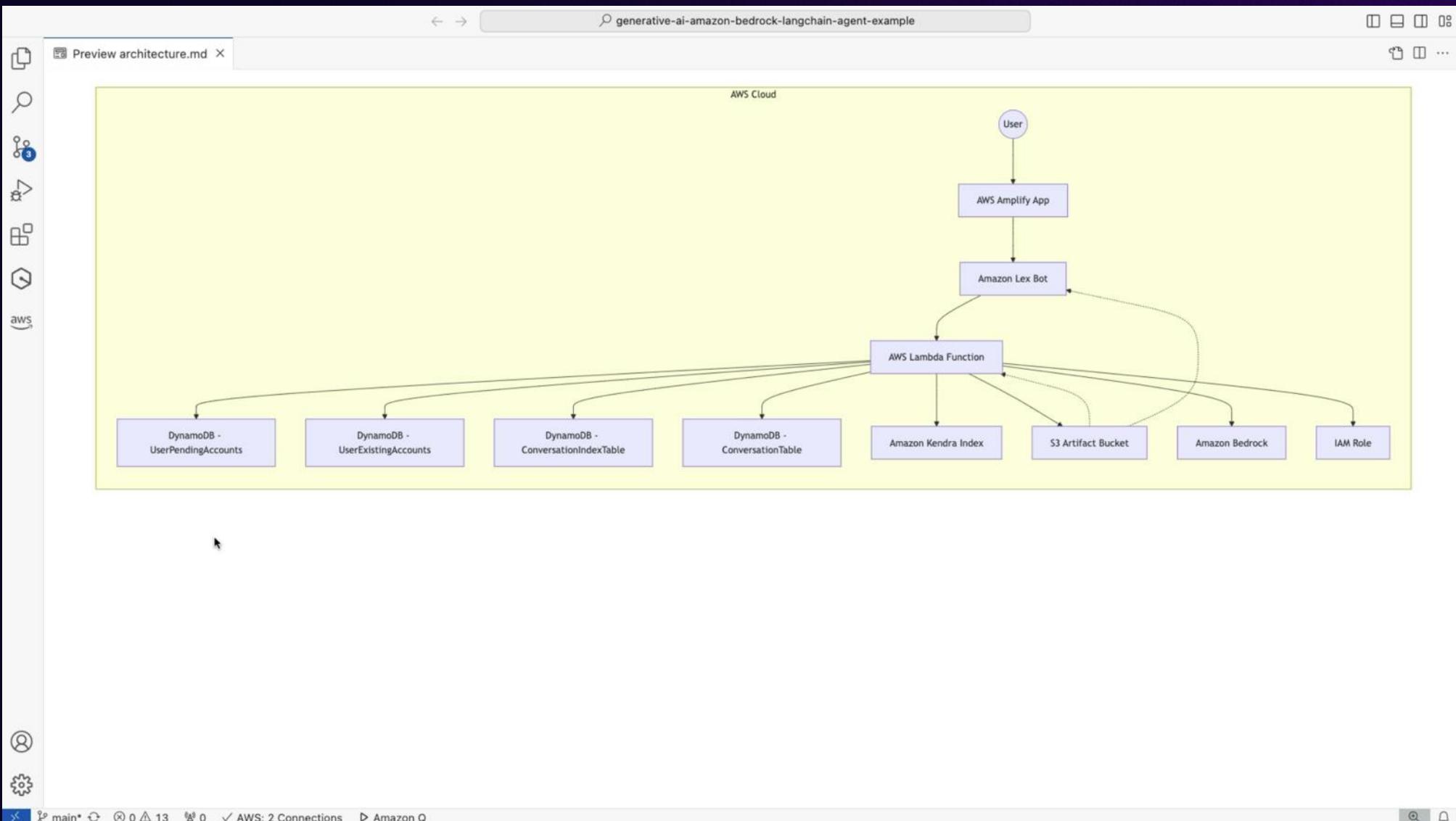
The screenshot shows the AWS CloudFormation console interface. On the left, there's a sidebar with icons for Chat, AWS Lambda, and other services. The main area has tabs for 'CloudFormation' and 'AWS CloudWatch Metrics'. A search bar at the top right says 'generative-ai-amazon-bedrock-langchain-agent-example'. Below it, a file browser shows 'GenAI-FSI-Agent.yml' with a preview of the template content.

The template content is as follows:

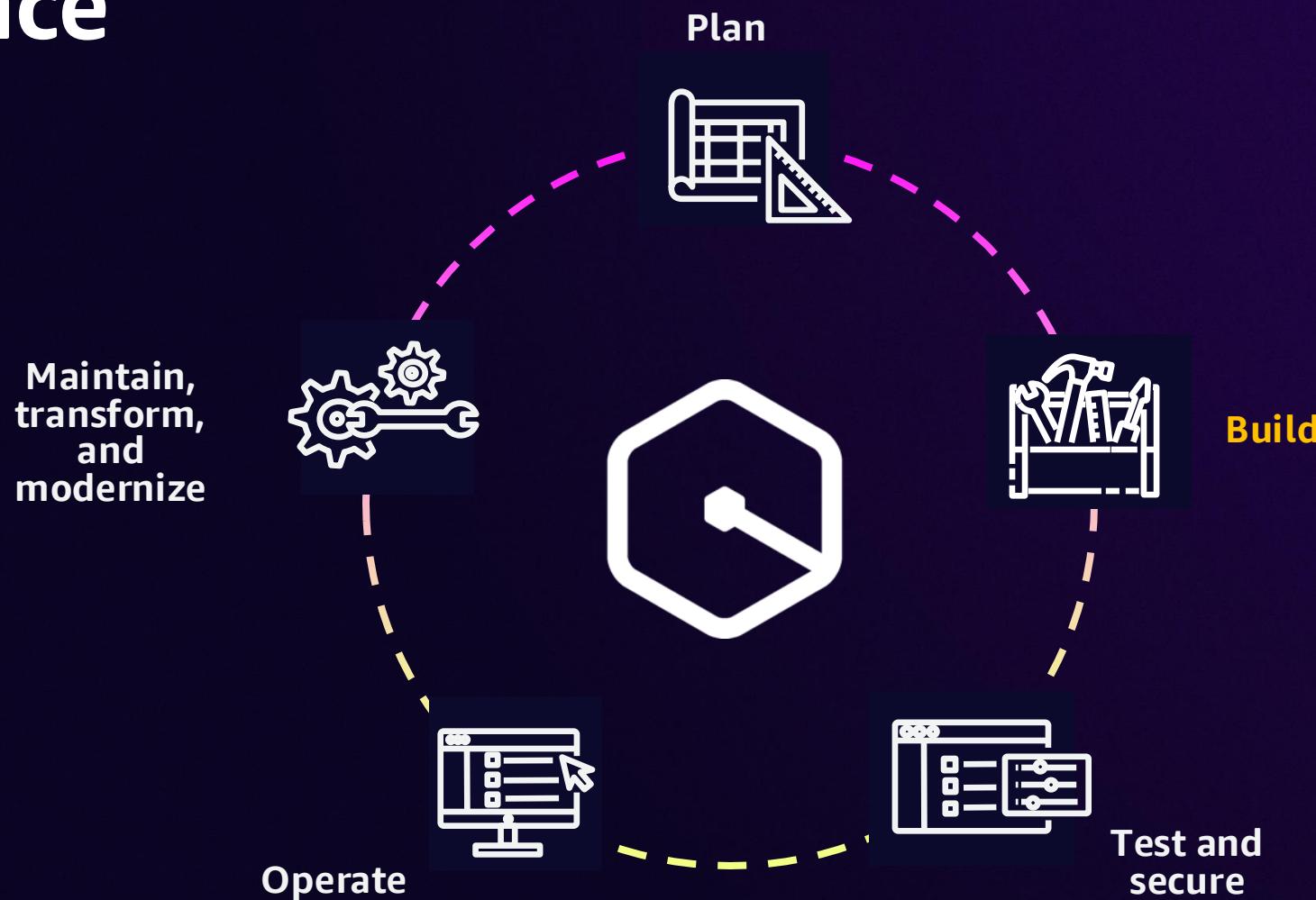
```
! GenAI-FSI-Agent.yml 9+, M X
cfn > ! GenAI-FSI-Agent.yml > AWSTemplateFormatVersion
AWSTemplateFormatVersion: '2010-09-09'
Description: GenAI Financial Services Agent powered by Amazon Bedrock, Amazon
DynamoDB, AWS Lambda, Amazon Lex, and Amazon Kendra
Metadata:
LICENSE: |-
Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES
AWS::CloudFormation::Interface:
ParameterGroups:
- Label:
  default: S3 Bucket, Lambda and Lex Deployment Package Keys, and Lambda Layer
  ARNs
Parameters:
- S3ArtifactBucket
- DataLoaderS3Key
- LambdaHandlerS3Key
- LexBotS3Key
- BedrockLangChainPDFRWLayerArn
- CfresponseLayerArn
- Label:
  default: GitHub Secrets Manager Configuration
Parameters:
- GitHubTokenSecretName
- Label:
  default: Kendra Web Crawler Root Domain
Parameters:
- KendraWebCrawlerUrl
- Label:
  default: Amplify Source Repository URL
Parameters:
- AmplifyRepository
ParameterLabels:
S3ArtifactBucket:
default: your-s3-bucket-name
DataLoaderS3Key:
default: /agent/lambda/data-loader/loader_deployment_package.zip
LambdaHandlerS3Key:
default: /agent/lambda/agent-handler/agent_deployment_package.zip
LexBotS3Key:
default: /agent/bot/lex.zip
BedrockLangChainPDFRWLayerArn:
```

In the bottom-left corner, there's a blue button with white text that says 'Generate an architecture diagram in mermaid format showing all the AWS services in this CloudFormation template with their dependencies and relationships'. Below this button, a message says 'Generating your answer...'. At the bottom, there's a text input field with placeholder text 'Ask a question or enter "/" for quick actions' and a progress bar indicating '4000/4000'. A note at the bottom says 'Amazon Q Developer uses generative AI. You may need to verify responses. See the AWS R'.

Visualize CloudFormation with Mermaid

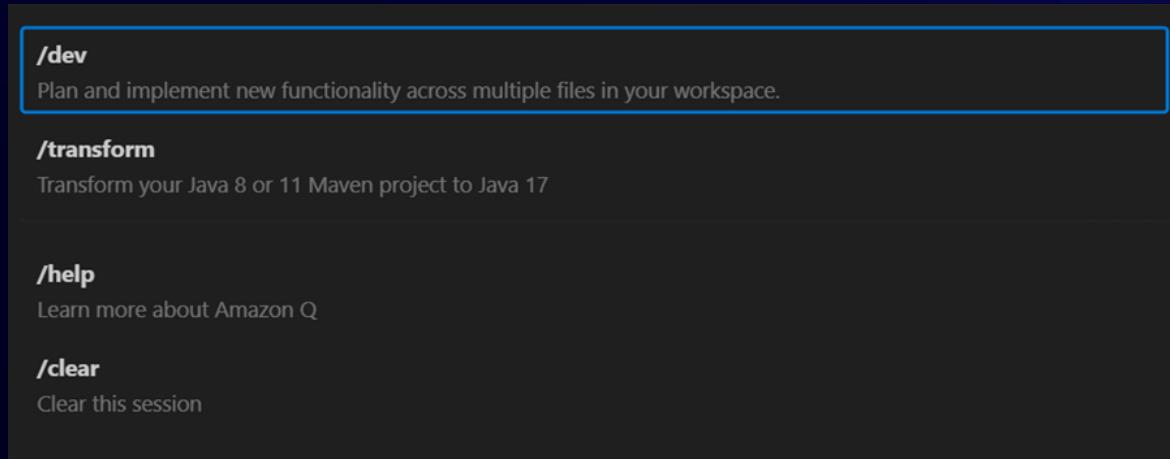


Amazon Q Developer transforms the development experience

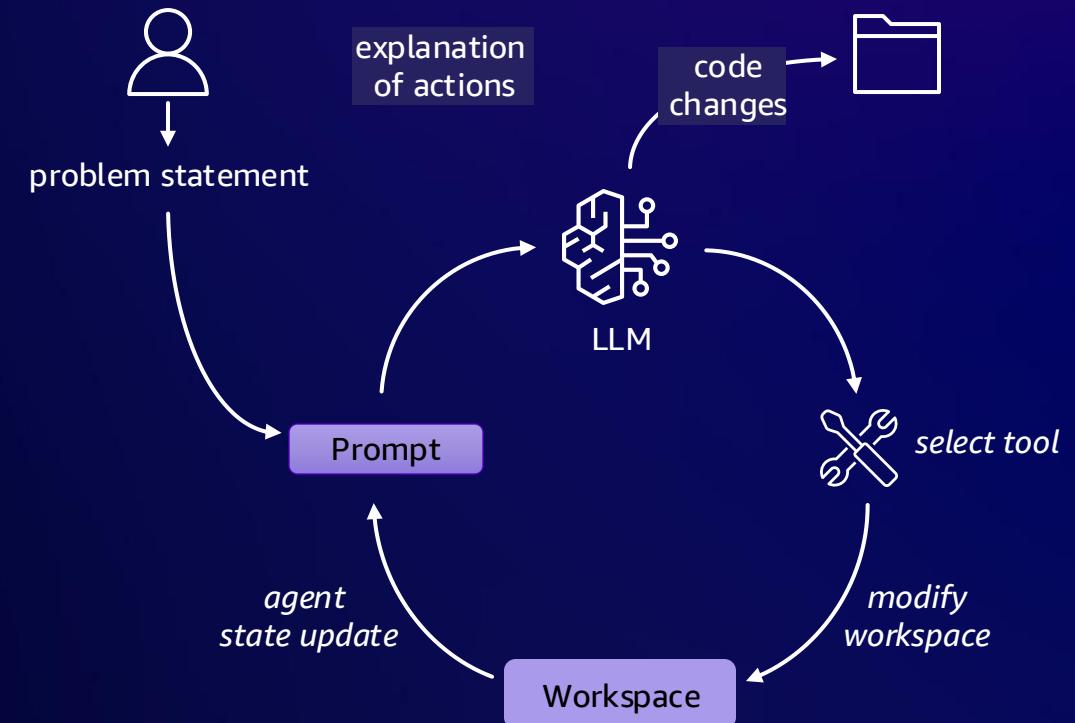


Developer agent for software development

PLAN AND IMPLEMENT NEW FUNCTIONALITY ACROSS MULTIPLE FILES IN YOUR WORKSPACE



Currently available in the Amazon Q extension for VS Code and in Amazon CodeCatalyst. Support is planned for other IDEs.





AMAZON Q: CHAT

Chat x Q - Dev x +

Welcome to feature development.

I can generate code to implement new functionality across your workspace. We'll start by discussing an implementation plan, and then we can review and regenerate code based on your feedback.

To get started, describe the task you are trying to accomplish.

Create a class to save a log of the tokens generated to dynamodb

Ok, let me create a plan. This may take a few minutes.

1. Create a new class for logging token generation to DynamoDB

1. Create a new file: `TokenLogger.cs`

2. Import the required namespaces:

- `Amazon.DynamoDBv2`

- `Amazon.DynamoDBv2.DataModel`

3. Define a new class `TokenLogger` that implements `IDisposable`

4. Define a constructor that initializes the DynamoDB client and table name

5. Define a method `LogToken` with the following input parameters:

- `string token`

- `DateTime expirationTime`

6. Inside `LogToken`, create a new instance of a DynamoDB model class

How can this plan be improved?

C# JwtBuilder.cs

C# Function.cs X

C# Function.cs > Function

```
1  using Amazon.Lambda.Core;
2  using Amazon.Lambda.APIGatewayEvents;
3  using Amazon.SimpleNotificationService;
4  using Amazon.SimpleNotificationService.Model;
5  using Octokit;
6  using System.Security.Cryptography;
7  using System.Text.Json;
8  using System.Linq;
9
10 // Assembly attribute to enable the Lambda function's JSON input to be converted into a .NET
11 [assembly: LambdaSerializer(typeof(Amazon.Lambda.Serialization.SystemTextJson.DefaultLambda
12
13 namespace orgbranchprotection;
14
15 1 reference
16 public class Function
17 {
18     3 references
19     private static IAmazonSimpleNotificationService? client;
20     3 references
21     private static JwtBuilder? jwtBuilder;
22     4 references
23     private static AccessToken? accessToken;
24     2 references
25     private readonly long Installation = 11111111;
26     2 references
27     private readonly string SnsTopicArn = "";
28
29     //test comment
30
31     0 references
32     public Function()
33     {
34         if (jwtBuilder != null) return;
35     }
36 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS CODE REFERENCE LOG

NuGet

Determining projects to restore...

All projects are up-to-date for restore.

Describe the task
Q tells you its plan to accomplish the task

1. Create a new class for logging token generation to DynamoDB

1. Create a new file: `TokenLogger.cs`
2. Import the required namespaces:
 - `Amazon.DynamoDBv2`
 - `Amazon.DynamoDBv2.DataModel`
3. Define a new class `TokenLogger` that implements `IDisposable`
4. Define a constructor that initializes the DynamoDB client and table name
5. Define a method `LogToken` with the following input parameters:
 - `string token`
 - `DateTime expirationTime`
6. Inside `LogToken`, create a new instance of a DynamoDB model class representing a log entry
7. Save the log entry to the DynamoDB table using the client

2. Update the existing `JwtBuilder` class

1. `./JwtBuilder.cs`
2. Import the `TokenLogger` class
3. Instantiate the `TokenLogger` in the constructor
4. In the `GetToken` method, call the `LogToken` method of `TokenLogger` with the generated token and its expiration time

3. Add tests for the `TokenLogger` class

1. Create a new file: `TokenLoggerTests.cs`
2. Import the required namespaces for unit testing
3. Define a test class `TokenLoggerTests`

4. Define a test method `TestLogToken` with the following steps:

- Arrange:
 1. Set up mock DynamoDB client and table
 2. Create an instance of `TokenLogger`
 - Act:
 - Call `LogToken` with a sample token and expiration time
 - Assert:
 - Verify that the log entry was saved to the DynamoDB table
4. Update the project file to include the new files
 1. `./orgbranchprotection.csproj`
 2. Add the new `TokenLogger.cs` and `TokenLoggerTests.cs` files to the project
 5. Update the deployment configuration file for DynamoDB access
 1. `./aws-lambda-tools-defaults.json`
 2. Add the required DynamoDB permissions and environment variables for the DynamoDB table name
 6. Update the GitHub workflow or deployment scripts to run the new tests
 1. Modify the existing GitHub Actions workflow or deployment scripts
 2. Add a step to run the new `TokenLoggerTests` unit tests

By following these steps, a new class `TokenLogger` will be introduced to log token generation events to a DynamoDB table. The existing `JwtBuilder` class will be updated to use the `TokenLogger`. Additionally, unit tests will be added to verify the functionality of the `TokenLogger` class. The project file and deployment configuration will be updated to include the new files and DynamoDB access respectively. Finally, the GitHub Actions workflow or deployment scripts will be modified to run the new tests.

Note: To handle current demand, Q will use a smaller, higher-speed model for code generation.

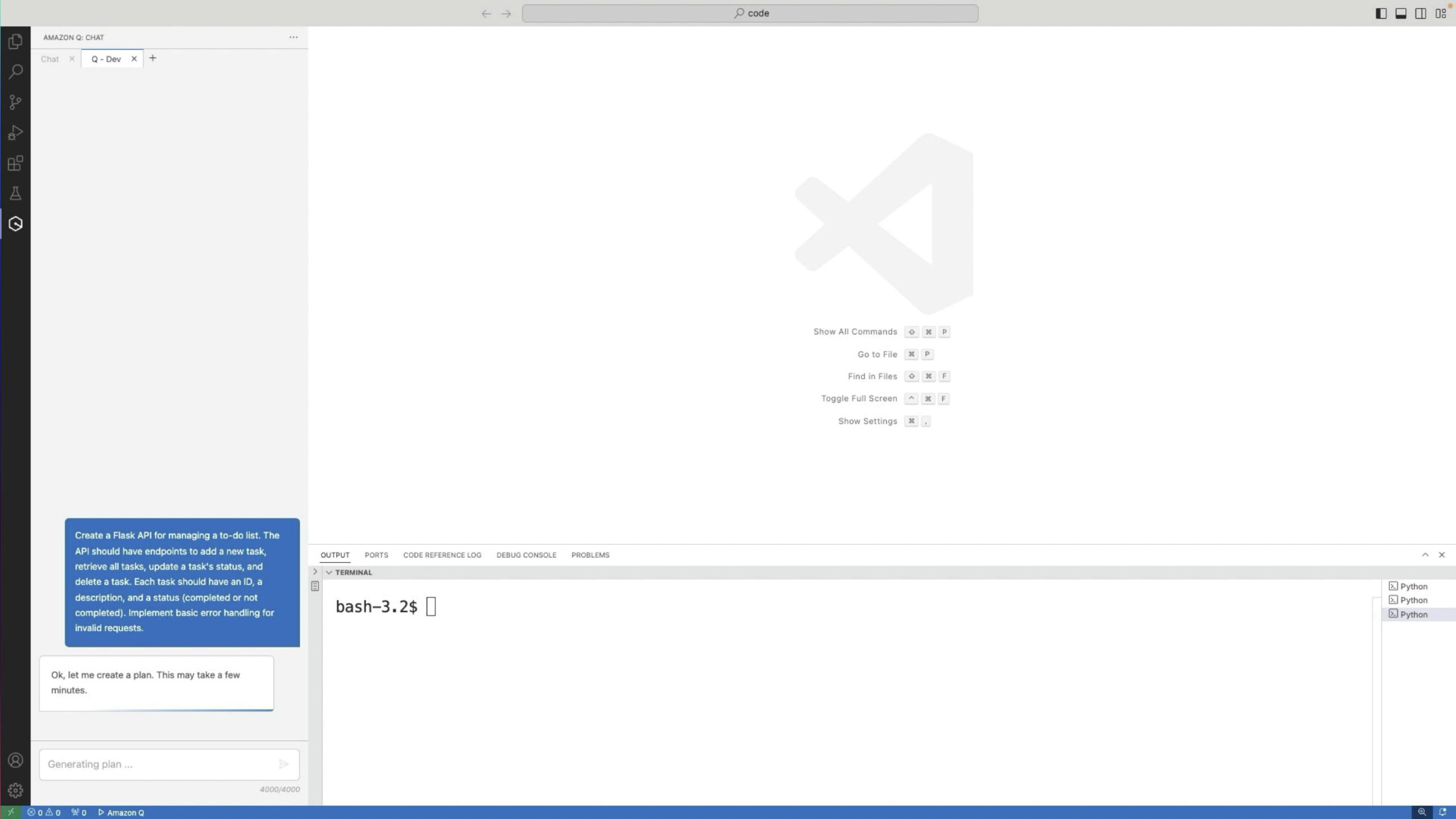


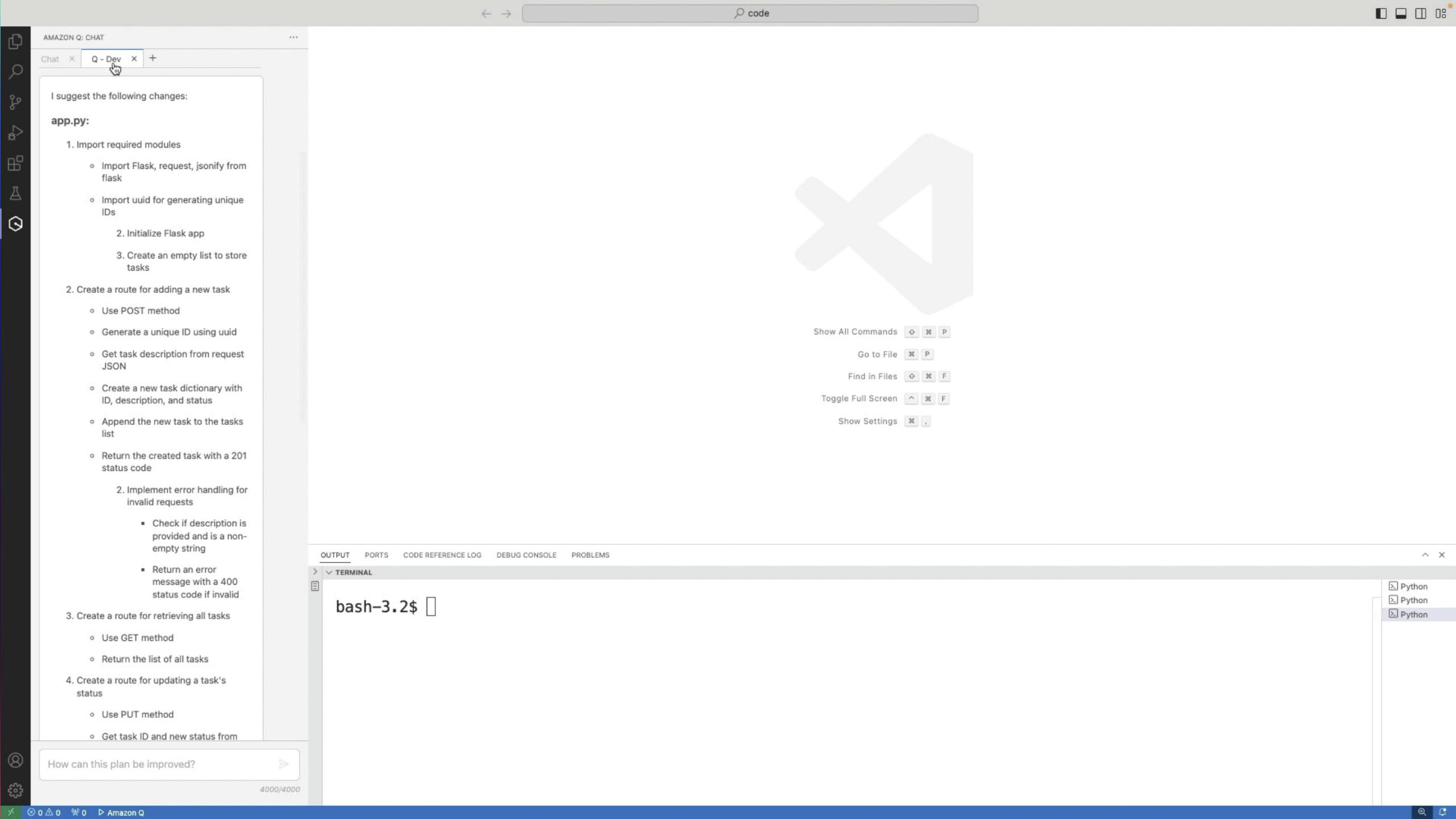
Sample prompt



Create a Flask API for managing a to-do list. The API should have endpoints to add a new task, retrieve all tasks, update a task status, and delete a task.

Ok, let me create a plan. This may take a few minutes.





AMAZON Q: CHAT

Chat X Q - Dev X +

I suggest the following changes:

app.py:

1. Import required modules
 - o Import Flask, request, jsonify from flask
 - o Import uuid for generating unique IDs
 2. Initialize Flask app
 3. Create an empty list to store tasks
2. Create a route for adding a new task
 - o Use POST method
 - o Generate a unique ID using uuid
 - o Get task description from request JSON
 - o Create a new task dictionary with ID, description, and status
 - o Append the new task to the tasks list
 - o Return the created task with a 201 status code
 2. Implement error handling for invalid requests
 - Check if description is provided and is a non-empty string
 - Return an error message with a 400 status code if invalid
3. Create a route for retrieving all tasks
 - o Use GET method
 - o Return the list of all tasks
 4. Create a route for updating a task's status
 - o Use PUT method
 - o Get task ID and new status from

How can this plan be improved? ➤

4000/4000

Show All Commands ⌘ ⌥ P

Go to File ⌘ P

Find in Files ⌘ ⌥ F

Toggle Full Screen ⌘ ⌥ F

Show Settings ⌘ ,

OUTPUT PORTS CODE REFERENCE LOG DEBUG CONSOLE PROBLEMS

> V TERMINAL

bash-3.2\$

Python
Python
Python



The image shows a screenshot of a code editor interface, likely Visual Studio Code, displaying a Python file named `app.py`. The code implements a simple REST API for managing tasks using the Flask framework.

```
1 from flask import Flask, request, jsonify
2 import uuid
3
4 app = Flask(__name__)
5
6 tasks = []
7
8 @app.route('/tasks', methods=['POST'])
9 def add_task():
10     data = request.json
11     if not data or 'description' not in data or not isinstance(data['description'], str) or data
12         ['description'].strip() == '':
13             return jsonify({"error": "Invalid task description"}), 400
14
15     task = {
16         "id": str(uuid.uuid4()),
17         "description": data['description'],
18         "status": "not completed"
19     }
20     tasks.append(task)
21     return jsonify(task), 201
```

Below the code editor, there is a terminal window titled "TERMINAL" showing a bash prompt:

```
bash-3.2$
```

On the right side of the interface, there is a sidebar with several "Python" entries listed under "RECENT FILES".

← → Home Workspaces API Network

Search Postman

Invite ⚙️ Bell Upgrade

My Workspace New Import GET Get Tasks POST Add Task +

HTTP Amazon Q LiL / Add Task

POST http://127.0.0.1:5000/tasks

Params Authorization Headers (8) Body **Body** Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

```
1 {
2   "description": "check in code to repo",
3   "status": "not started"
4 }
```

Status: 201 CREATED Time: 5 ms Size: 292 B Save as example

Collections Environments History ChatGPT ChatGPT Action NASA APIs NDC OpenAI API

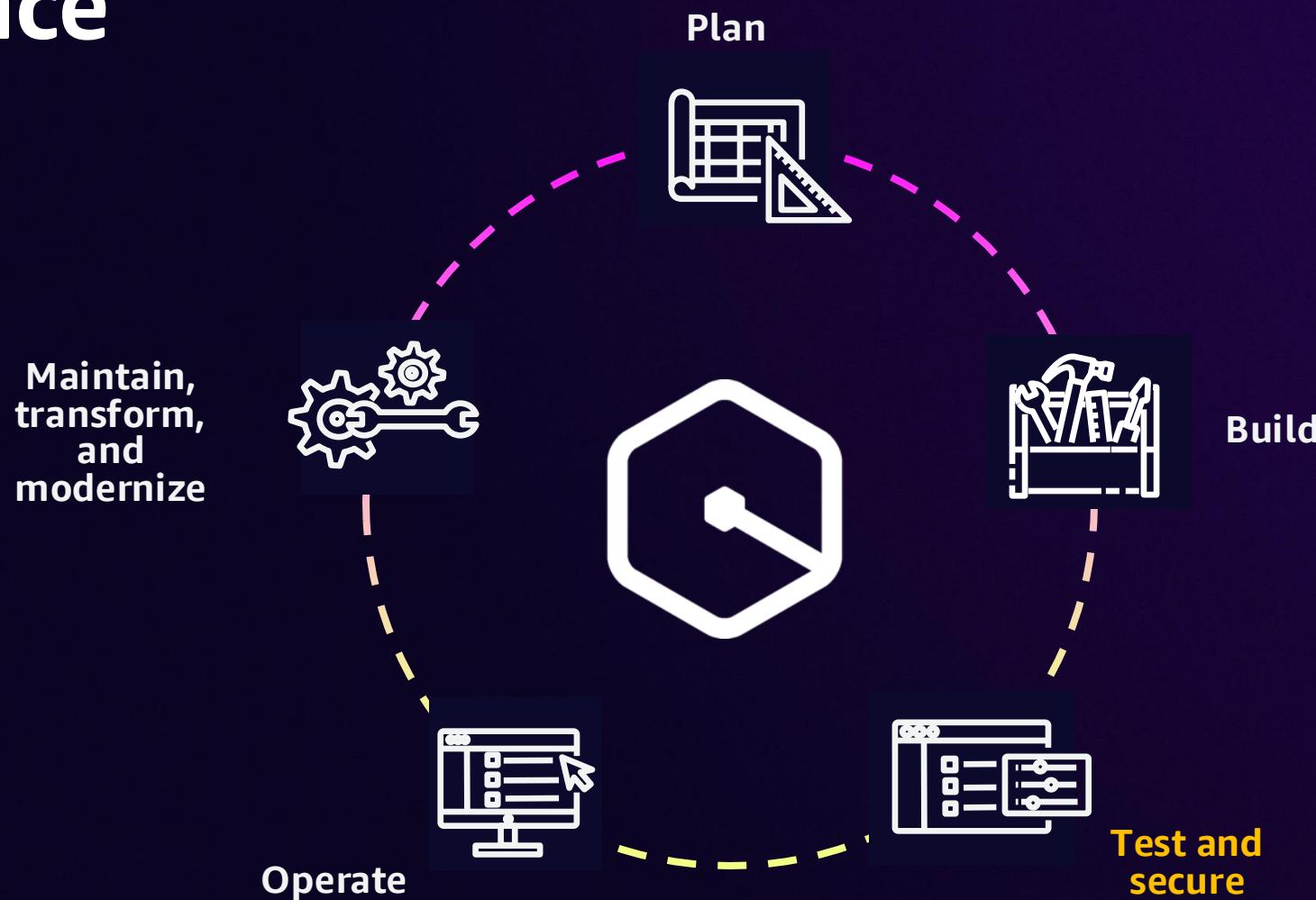
Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "description": "check in code to repo",
3   "id": "d84309ef-8d00-4a1d-940a-8e987ac93b0f",
4   "status": "not completed"
5 }
```

Online Find and replace Console Postbot Runner Start Proxy Cookies Vault Trash

Amazon Q Developer transforms the development experience



Unit test generation (/test)

Amazon Q Developer agent for unit tests

☰ /test

Included in your Q Developer Agent subscription

I can generate unit tests for your active file.

After you select the functions or methods I should focus on, I will:

1. Generate unit tests
2. Place them into relevant test file

To learn more, check out our [user guide](#).



← →

demo-feat-2048-game

□ □ □ 0%

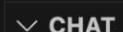


AMAZON Q

...

> CODE ISSUES

...



CHAT

Chat X Q - Test X +

To learn more, check out our User Guide.

Sure. This may take a few minutes. I'll share updates here as I work on this.

Generating unit tests for the following methods in app.py

new_game

make_move

get_game_state



Generating unit tests... 71%

Cancel

Amazon Q Developer uses generative AI. You may need to verify responses. See the [AWS Responsible AI Policy](#). Amazon Q Developer

app.py game.py

src > app >

app.py

```
46     async def new_game() -> dict:
47         """Start a new game and return the initial state."""
48         game = Game2048()
49         game_id = str(len(games))
50         games[game_id] = game
51         return {"game_id": game_id, "state": game.get_state()}
52
53     @app.post("/game/move")
54     async def make_move(move_request: MoveRequest) -> dict:
55         """Make a move in the specified direction."""
56         game_id = move_request.game_id
57         if game_id not in games:
58             raise HTTPException(status_code=404, detail="Game not found")
59
60         game = games[game_id]
61         try:
62             moved = game.move(move_request.direction)
63             return {"moved": moved, "state": game.get_state()}
64         except ValueError as e:
65             raise HTTPException(status_code=400, detail=str(e))
66
67     @app.get("/game/{game_id}")
68     async def get_game_state(game_id: str) -> dict:
69         """Get the current state of a game."""
70         if game_id not in games:
71             raise HTTPException(status_code=404, detail="Game not found")
72         return games[game_id].get_state()
```





← →

demo-feat-2048-game

□ □ □ 0%

app.py

test_app.py (Generated by Amazon Q) ×

game.py

↑ ↓ ⌛ ¶ ☰ ...

tmp > aws-toolkit-vscode > q-testgen > resultArtifacts > tests > test_app.py

```
40+
41+def test_get_game_state_empty_game_id(test_client):
42+    """
43+    Test get_game_state with an empty game ID.
44+
45+    response = test_client.get("/game/")
46+    assert response.status_code == 404
47+    assert "detail" in response.json()
48+
49+
50+def test_get_game_state_existing_game():
51+    """
52+    Test get_game_state when the game_id exists in games.
53+    It should return the game state.  {}
54+
55+    game_id = "test_game_id"
56+    mock_game = MagicMock(spec=Game2048)
57+    mock_game.get_state.return_value = {
58+        "board": [[0, 2, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0]],
59+        "score": 0,
60+        "game_over": False
61+    }
62+
63+    games[game_id] = mock_game
64+
65+    with TestClient(app) as client:
66+        response = client.get(f"/game/{game_id}")
67+
68+    assert response.status_code == 200
69+    assert response.json() == mock_game.get_state.return_value
70+
71+    # Clean up the test game
72+    del games[game_id]
```

Automated code reviews (/review)

Amazon Q Developer agent for code reviews



/review

Included in your Q Developer subscription

I can review your workspace for vulnerabilities and issues.

After you begin a review, I will:

1. Review all relevant code in your workspace or your current file
2. Provide a list of issues for your review

You can then investigate, fix, or ignore issues.

To learn more, check out our [User Guide](#).

This screenshot shows a developer workspace interface with a focus on AI-powered code review.

The top navigation bar includes standard icons for file operations, search, and help, followed by a search bar containing "demo". On the far right are additional UI elements like a refresh icon and a timestamp.

The left sidebar features a "CODE ISSUES" section and a "CHAT" section. The "CHAT" section has tabs for "Chat" and "Q - Review", with the latter being active.

A prominent blue callout box highlights the "/review" feature, stating it's included in the Q Developer subscription. It explains that the AI will review the workspace for vulnerabilities and issues, and provides a list of what it will do:

1. Review all relevant code in your workspace or your current file
2. Provide a list of issues for your review

It also mentions that users can investigate, fix, or ignore issues, and points to the User Guide for more information.

Below this, a question is posed: "Would you like to review your active file or the workspace you have open?", with two buttons: "Review workspace" and "Review active file".

The main content area displays a Python FastAPI application code snippet:

```
src > app > app.py > ...
28
29     app = FastAPI(title="2048 Game API", debug=stage == "development")
30
31     app.add_middleware(
32         CORSMiddleware,
33         allow_origins=origins,
34         allow_credentials=True,
35         allow_methods=["*"],
36         allow_headers=["*"],
37     )
38     # Store active games
39     games = {}
40
41     class MoveRequest(BaseModel):
42         direction: str
43         game_id: str
44
45     @app.post("/game/new")
46     async def new_game() -> dict:
47         """Start a new game and return the initial state."""
48         game = Game2048()
49         game_id = str(len(games))
50         games[game_id] = game
51         return {"game_id": game_id, "state": game.get_state()}
52
53     @app.post("/game/move")
54     async def make_a_move(move_request: MoveRequest):
55         """Make a move (parameter) move_request: MoveRequest"""
56         game_id = move_request.game_id
57         if game_id not in games:
58             raise HTTPException(status_code=404, detail="Game not found")
59
60         game = games[game_id]
61         try:
62             moved = game.move(move_request.direction)
63             return {"moved": moved, "state": game.get_state()}
64         except ValueError as e:
65             raise HTTPException(status_code=400, detail=str(e))
66
67     @app.get("/game/{game_id}")
68     async def get_game(game_id: str):
69         game = games.get(game_id)
70         if not game:
71             raise HTTPException(status_code=404, detail="Game not found")
72
73         return {"game_id": game_id, "state": game.get_state()}
```

At the bottom, a message says "Waiting on your inputs..." with a right-pointing arrow. A note at the very bottom states: "Amazon Q Developer uses generative AI. You may need to verify responses. See the [AWS Responsible AI Policy](#). Amazon Q Developer processes data across all US Regions. See [AWS Responsible AI Policy](#) for more information."

The screenshot shows the AWS Lambda Code Editor interface with the following components:

- Left Sidebar:** Includes icons for file operations (New, Open, Save, Delete), search, and code navigation.
- Top Bar:** Shows the project name "AMAZON Q" and a search bar with the query "demo".
- Code Issues Panel:** A tree view of code issues:
 - Critical:** 3 issues
 - CWE-798,259 - Hardcoded credentials cert-key.pem [Ln 1, Col 1]
 - CWE-798 - Hardcoded credentials cert-key.pem [Ln 1, Col 1]
 - CWE-77,78 - Package Vulnerability poetry.lock [Ln 95, Col 1]
 - High:** 2 issues
 - CWE-1333 - Package Vulnerability poetry.lock [Ln 8, Col 1]
 - CWE-770 - Package Vulnerability poetry.lock [Ln 615, Col 1]
 - Medium:** 0 issues
 - Low:** 0 issues
 - Info:** 1 issue
- Code Editor:** The main workspace displays the "app.py" file content:

```
src > app > app.py > new_game
27
28
29     app = FastAPI(title="2048 Game API", debug=stage == "development")
30
31     app.add_middleware(
32         CORSMiddleware,
33         allow_origins=origins,
34         allow_credentials=True,
35         allow_methods=["*"],
36         allow_headers=["*"],
37     )
38     # Store active games
39     games = {}
40
41     class MoveRequest(BaseModel):
42         direction: str
43         game_id: str
44
45     @app.post("/game/new")
46     async def new_game() -> dict:
47         """Start a new game and return the initial state."""
48         game = Game2048()
49         game_id = str(len(games))
50         games[game_id] = game
51         return {"game_id": game_id, "state": game.get_state()}
52
53     @app.post("/game/move")
54     async def make_move(move_request: MoveRequest) -> dict:
55         """Make a move in the specified direction."""
56         game_id = move_request.game_id
57         if game_id not in games:
58             raise HTTPException(status_code=404, detail="Game not found")
59
60         game = games[game_id]
61         try:
62             moved = game.move(move_request.direction)
63             return {"moved": moved, "state": game.get_state()}
64         except ValueError as e:
65             raise HTTPException(status_code=400, detail=str(e))
```
- Chat Panel:** A sidebar with tabs for "Chat" and "Q - Review". The "Q - Review" tab is active, showing a summary of issues:
 - Critical: 3 issues
 - High: 2 issues
 - Medium: 0 issues
 - Low: 0 issues
 - Info: 1 issue
- Bottom Panel:** Includes a message input field ("Ask a question or enter '/' for quick actions"), a "Help" icon, and a note about AI usage: "Amazon Q Developer uses generative AI. You may need to verify responses. See the [AWS Responsible AI Policy](#). Amazon Q Developer processes data across all US Regions. See [AWS Responsible AI Policy](#) for more information."

The screenshot shows a code editor interface with several tabs at the top: `app.py`, `test_app.py`, and `game.py`. The `app.py` tab is active, displaying Python code for a FastAPI application. A sidebar on the left contains icons for file operations like Open, Save, Find, and Refresh.

In the center, a modal window titled "Code Issue Details" is open, showing a warning for "CWE-1333 - Package Vulnerability" (High severity). The modal contains the following text:

The `py` library through 1.11.0 for Python allows remote attackers to conduct a ReDoS (Regular expression Denial of Service) attack via a Subversion repository with crafted info data, because the `InfoSvn` command argument is mishandled. Note: This has been disputed by multiple third parties as not being reproduceable and they argue this is not a valid vulnerability.

Relevant link: <https://github.com/advisories/GHSA-w596-4wvx-j9j6>

The modal also displays two tables:

Common Weakness	Detector library
Enumeration (CWE) CWE-1333	-

File path
poetry.lock [Ln 8]

At the bottom of the modal are four buttons: "Generate Fix" (highlighted in blue), "Explain", "Ignore", and "Ignore All".

```
27
28 app = FastAPI(title="2048 Game API", debug=stage == "development")
29
30 app.add_middleware(
31     CORSMiddleware,
32     allow_origins=origins,
33     allow_credentials=True,
34     allow_methods=["*"],
35     allow_headers=["*"],
36 )
37
38 # Store active games
39 games = {}
40
41 class MoveRequest(BaseModel):
42     direction: str
43     game_id: str
44
45 @app.post("/game/new")
46 async def new_game() -> dict:
47     """Start a new game and return the initial state."""
48     game = Game2048()
49     game_id = str(len(games))
50     games[game_id] = game
51     return {"game_id": game_id, "state": game.get_state()}
52
53 @app.post("/game/move")
54 async def make_move(move_request: MoveRequest) -> dict:
55     """Make a move in the specified direction."""
56     game_id = move_request.game_id
57     if game_id not in games:
58         raise HTTPException(status_code=404, detail="Game not found")
59
60     game = games[game_id]
61     try:
62         moved = game.move(move_request.direction)
63         return {"moved": moved, "state": game.get_state()}
64     except ValueError as e:
65         raise HTTPException(status_code=400, detail=str(e))
```

demo

Code Issue Details

```
app.py 5 poetry.lock: Original ↔ poetry.lock test_app.py M ↑ ↓ ¶ ⌂ ...
```

1 # This file is automatically @generated by Poetry 1.8.3 and should not be change
2
3 [[package]]
4 name = "annotated-types"
5 version = "0.7.0"
6 description = "Reusable constraint types to use with typing.Annotated"
7 optional = false
8 - python-versions = ">=3.8"
8+ python-versions = ">=3.9"
9 files = [
10 {file = "annotated_types-0.7.0-py3-none-any.whl", hash = "sha256:1f02e8b43a8c09a53a08bc8cfcc"},
11 {file = "annotated_types-0.7.0.tar.gz", hash = "sha256:aff07c09a53a08bc8cfcc"}]
12]
13
14 [[package]]
15 name = "anyio"
16 version = "4.4.0"
17 description = "High level compatibility layer for multiple asynchronous event loops"
18 optional = false
19 python-versions = ">=3.8"
20 files = [
21 {file = "anyio-4.4.0-py3-none-any.whl", hash = "sha256:c1b2d8f46a8a812513012"},
22 {file = "anyio-4.4.0.tar.gz", hash = "sha256:5aadc6a1bbb7cdb0bede386cac5e294"}]
23]
24
25 [package.dependencies]
26 idna = ">=2.8"
27 sniffio = ">=1.1"
28
29 [package.extras]
30 doc = ["Sphinx (>=7)", "packaging", "sphinx-autodoc-typehints (>=1.2.0)", "sphinxcontrib-apidoc (>=0.10.6)"]
31 test = ["anyio[trio]", "coverage[toml] (>=7)", "exceptiongroup (>=1.2.0)", "hyperset (>=0.10.0)"]
32 trio = ["trio (>=0.23)"]
33
34 [[package]]
35 name = "attrs"
36 version = "24.2.0"
37 description = "Classes Without Boilerplate"
38 optional = false
39 python-versions = ">=3.7"
40 files = [

CWE-1333 - Package Vulnerability High

The py library through 1.11.0 for Python allows remote attackers to conduct a ReDoS (Regular expression Denial of Service) attack via a Subversion repository with crafted info data, because the InfoSvnCommand argument is mishandled. Note: This has been disputed by multiple third parties as not being reproduceable and they argue this is not a valid vulnerability.

Relevant link: <https://github.com/advisories/GHSA-w596-4wvx-j9j6>

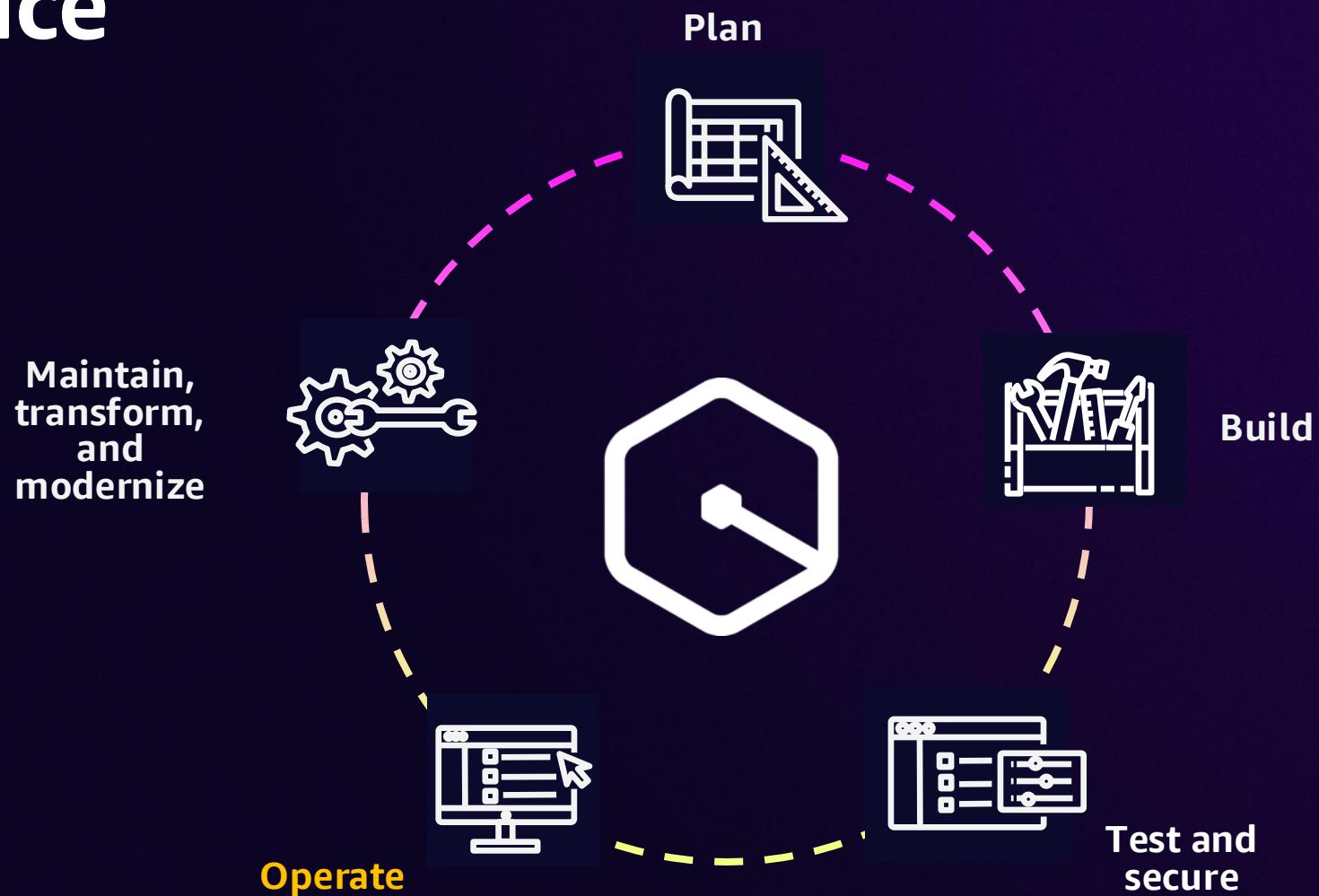
Common Weakness Enumeration (CWE)	Detector library
CWE-1333	-

File path
[poetry.lock \[Ln 8\]](#)

Suggested code fix preview

```
version = "0.7.0"  
description = "Reusable constraint types to use with typing.Annotated"  
optional = false  
python-versions = ">=3.9"  
files = [  
    {file = "annotated_types-0.7.0-py3-none-any.whl", hash = "sha256:1f02e8b43a8c09a53a08bc8cfcc"},  
    {file = "annotated_types-0.7.0.tar.gz", hash = "sha256:aff07c09a53a08bc8cfcc"}]
```

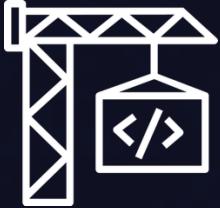
Amazon Q Developer transforms the development experience



Cloud Operations use cases for Amazon Q Developer

Build

(in the AWS Management Console)



Get expert guidance for building and deploying on AWS

- Ask for expert advice
- Receive actionable recommendations (e.g., Amazon SageMaker)
- Identify and resolve common errors in the console
- Generate Infrastructure as Code (IaC)

Operate



Improve operational efficiency

- Discover AWS resources
- Investigate and remediate operational issues
- Assisted incident investigation
- Third-party integrations (Wiz, Datadog)

Optimize



Understand and optimize cost

- Investigate your AWS bill
- Identify trends and optimization opportunities
- Get EC2 Instance sizing advice

Generate Infrastructure as Code (IaC)

The screenshot shows the AWS RDS Databases page. A blue callout box at the top left indicates that a database named "console-code-demo-db" is "Creating". Below it, there are seven databases listed:

DB identifier	Status	Role	Engine
console-code-demo-db	Creating	Regional c...	Aurora Po...
console-code-demo-db-instance-1	Creating	Reader ins...	Aurora Po...
console-demo-db	Available	Regional c...	Aurora Po...
demo-database	Available	Regional c...	Aurora Po...

The screenshot shows the "Console-to-Code" feature in progress. It displays a list of recorded actions:

- CDK
 - Generate CDK Java
 - Generate CDK Python
 - Generate CDK TypeScript
- CloudFormation
 - Generate CFN JSON
 - Generate CFN YAML
- RDS / Launch Dbinst...
 - createDBinstance Write
 - createDBCluster Write
 - authorizeSecurityGroup Write
 - createSecurityGroup Write
 - createDBSubnetGroup Write

Accelerates creation of Infrastructure as Code (IaC) - CDK, Terraform, CloudFormation

Console to code (Preview) sets foundation for IaC best practice patterns

Generate Infrastructure as Code (IaC)

Successfully created database [console-code-demo-db](#)

You can use settings from console-code-demo-db to simplify configuration of suggested database add-ons while we finish creating your DB for you.

[View connection details](#)

RDS > Databases

Databases (7)

Group resources Modify Actions ▾ Restore from S3 Create database

Filter by databases

DB identifier	Status	Role	Engine
console-code-demo-db	Available	Regional c...	Aurora Po...
console-code-demo-db-instance-1	Creating	Reader ins...	Aurora Po...
console-demo-db	Available	Regional c...	Aurora Po...
demo-database	Available	Regional c...	Aurora Po...

CDK Python

Amazon Q Developer uses generative AI to create code. Use the code as a starting point for your development. See the AWS Responsible AI Policy.

```
```python
from aws_cdk import (
 Stack,
 aws_rds as rds,
 aws_ec2 as ec2,
 CfnOutput
)
from constructs import Construct

class RdsStack(Stack):
 def __init__(self, scope: Construct, construct_id: str, **kwargs) -> None:
 super().__init__(scope, construct_id, **kwargs)

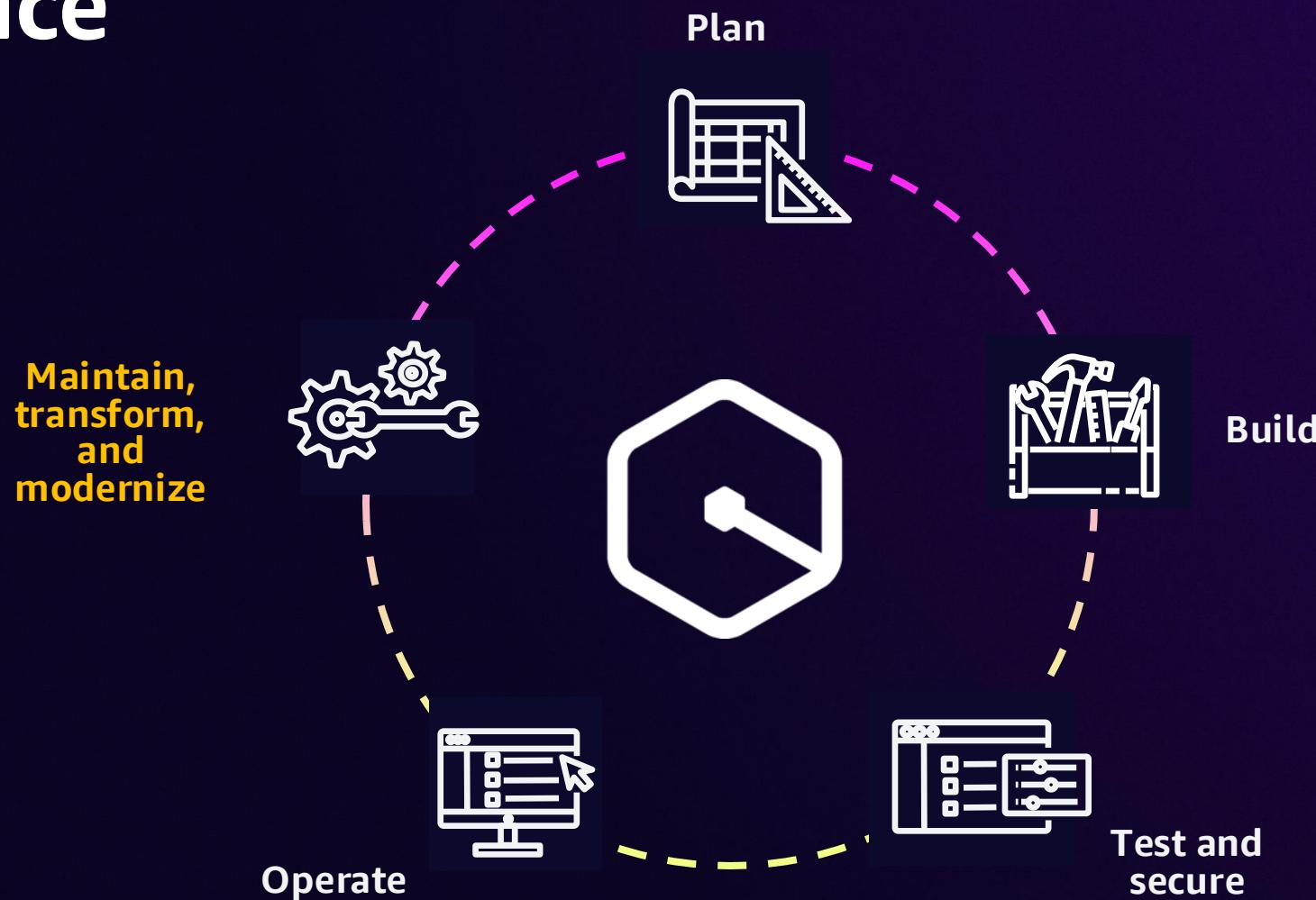
 # Create DB Subnet Group
 subnet_ids = ["subnet-0caf3c76ef61bd6a",
 "subnet-02fdf4d3e6b890c22", "subnet-05be166ae0aed2834", "subnet-01d2af67ffdd05d0c"]
 db_subnet_group = rds.SubnetGroup(
 self, "DefaultVpcSubnetGroup",
 description="Created from the RDS Management Console",
 subnet_group_name="default-vpc-0052c11fff911fa89",
 vpc_id="vpc-0052c11fff911fa89",
 vpc_subnets=ec2.SubnetSelection(subnet_ids=sub
```

```

Accelerates creation of Infrastructure as Code (IaC) - CDK, Terraform, CloudFormation

Console to code ([Preview](#)) sets foundation for IaC best practice patterns

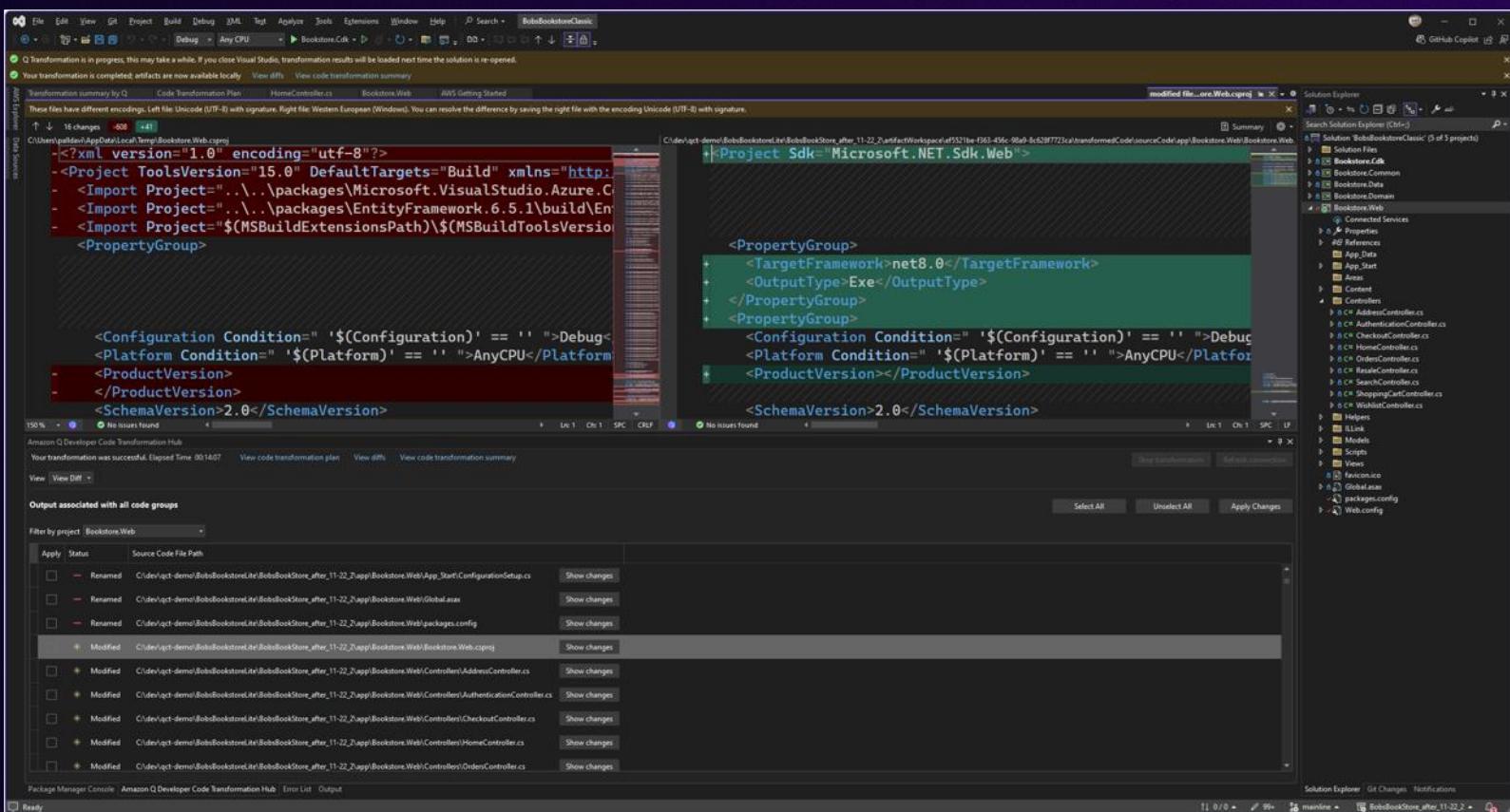
Amazon Q Developer transforms the development experience



Amazon Q Developer transformation capabilities

available in IDE or on CLI

Generative AI agents to accelerate transformation of .NET and Java applications



The screenshot shows a Visual Studio interface with several windows open:

- Solution Explorer:** Shows a solution named "BobBookstoreClassic" containing five projects: Bookstore.Cdk, Bookstore.Data, Bookstore.Domain, Bookstore.Web, and Bookstore.Web (another instance).
- Code Transformation Plan:** A central window displaying XML transformation code. It includes sections for Project ToolsVersion, Configuration, Platform Condition, ProductVersion, and SchemaVersion.
- Output associated with all code groups:** A list of file changes, mostly renamed files, under the "Source Code File Path" tab.
- Package Manager Console:** A small window at the bottom left showing "Ready".
- Amazon Q Developer Code Transformation Hub:** A bottom bar with tabs for "Transformation summary by file", "Code Transformation Plan", "HomeController.cs", "Bookstore.Web", and "AWS Getting Started".



Amazon Q Developer transformation capabilities

WORKLOADS



**.NET Framework
applications
on Windows**

Repository connection
Repository selection
Transformation
Review



**Linux-ready
cross-platform
.NET applications**



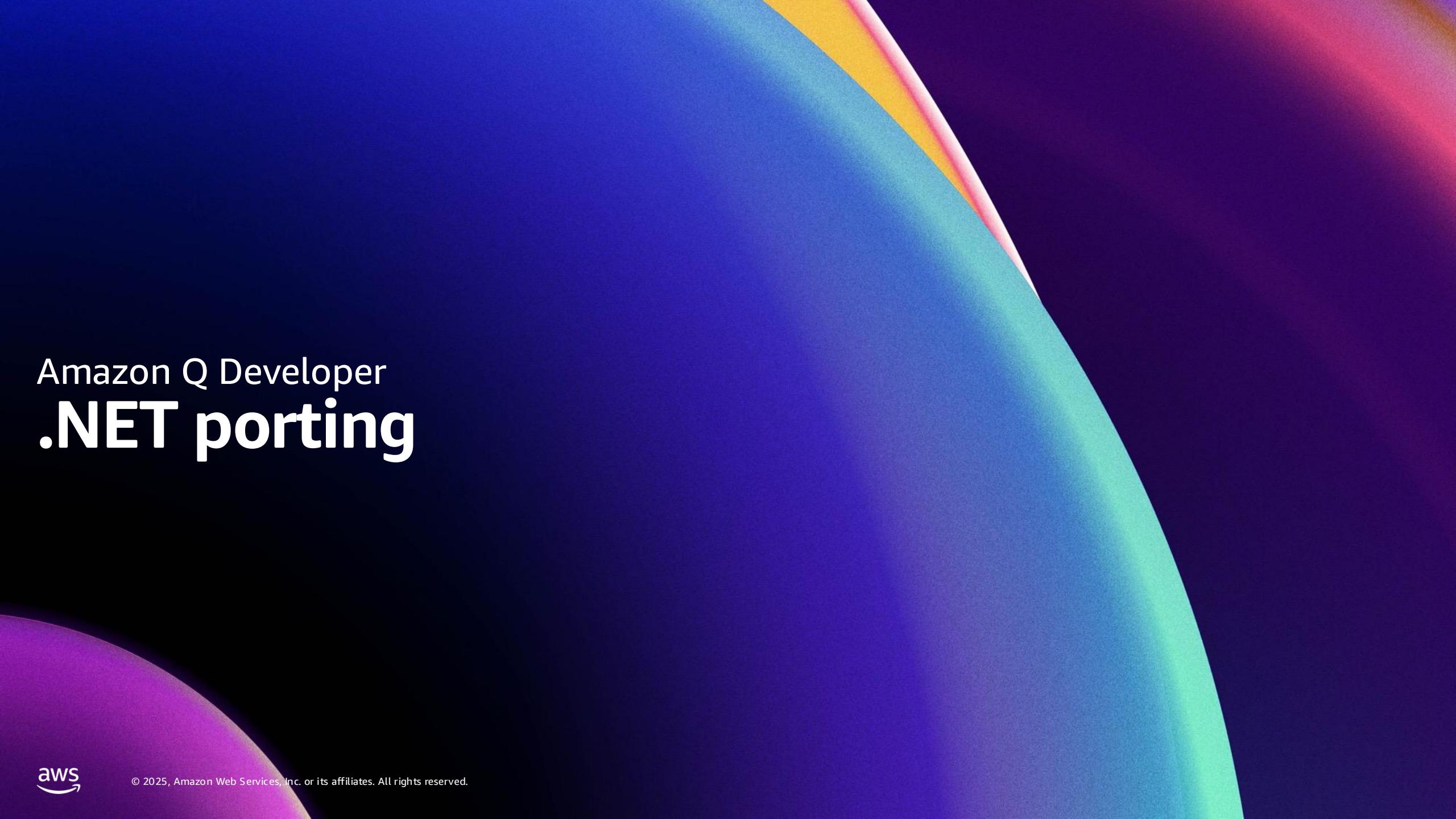
**Java applications
in v8, v11, v17, v21**

Build/test v8/v11/v17/v21
Apply knowledge base
Build/test v17/v21
Error fixing using genAI
Continuous maintenance



**Java applications
in v17/v21**

Amazon Q Developer **.NET porting**

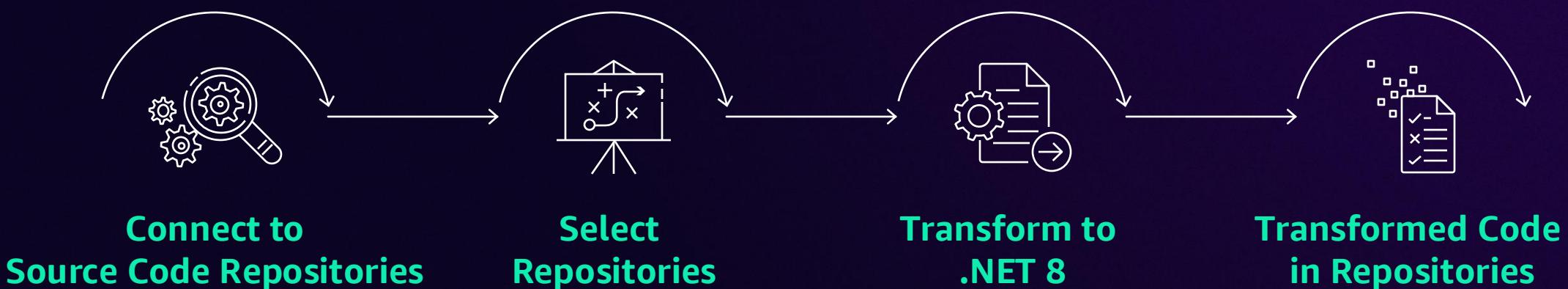
The background features a large, abstract, curved shape that resembles a rainbow or a stylized planet. It has a gradient of colors from dark purple to bright yellow and red, with a soft, glowing edge. This shape is positioned in the upper right quadrant of the slide, creating a dynamic and modern feel.

Q Developer .NET porting

THE FIRST GENERATIVE AI-POWERED ASSISTANT FOR LARGE-SCALE PORTING OF .NET APPLICATIONS

.NET Framework
on Windows

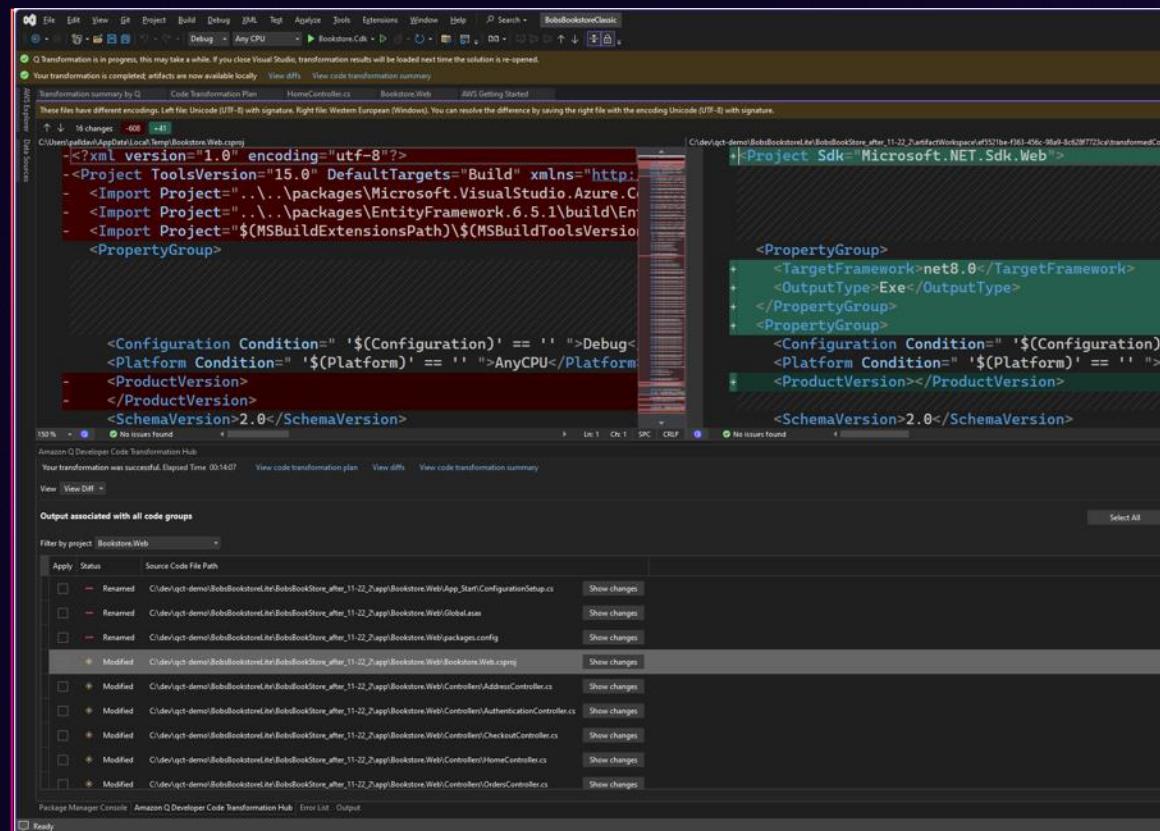
Cross-platform
.NET on Linux



Two experiences for .NET porting

THE SAME GENERATIVE AI AGENT CAPABILITIES

.NET porting in Visual Studio IDE (for developers)

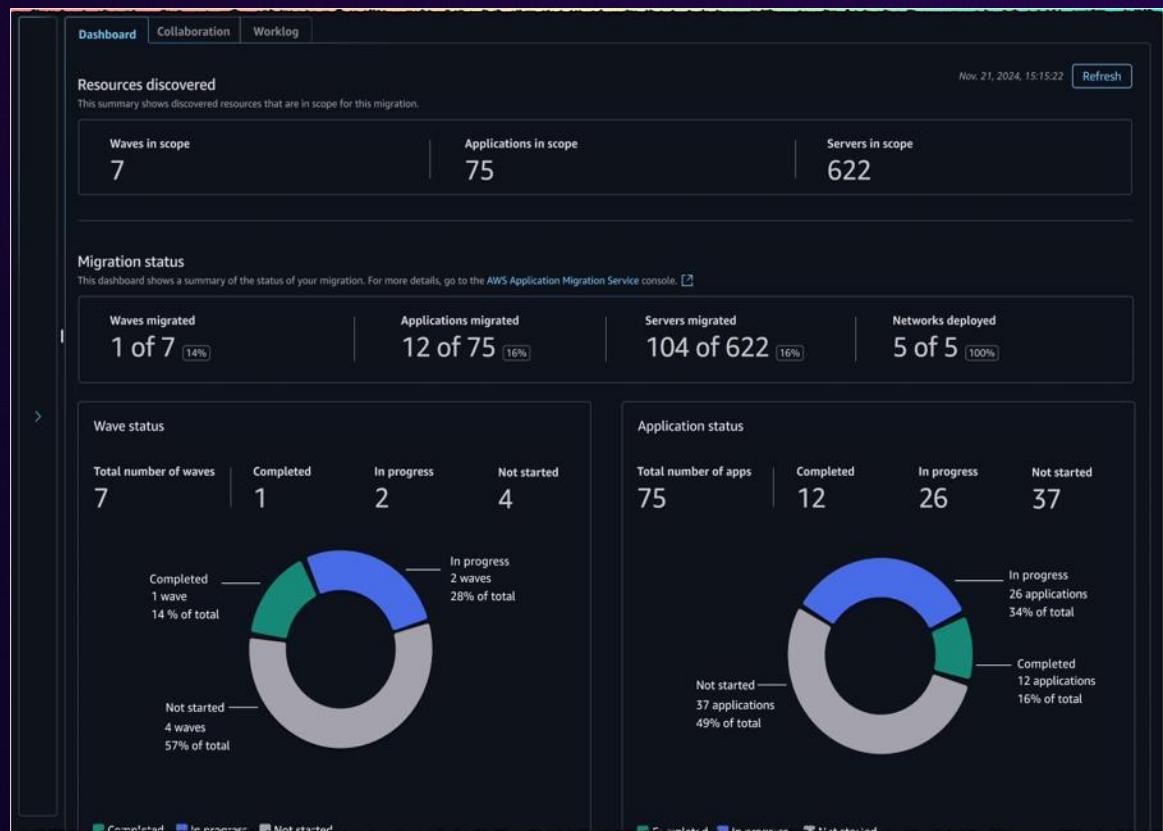


Amazon Q Developer



© 2025, Amazon Web Services, Inc. or its affiliates. All rights reserved.

.NET porting at large-scale using a web experience (for IT and modernization teams)



AWS Transform

Upgrade .NET apps faster with Amazon Q



Tata Consultancy Services

“

Amazon Q Developer transform is transforming the way we approach .NET modernization for clients. By leveraging its advanced AI-driven automation, we've already seen early success in helping customers upgrade their legacy .NET applications. This has enabled us to deliver modernization projects much faster, with projected reduction in operating cost of up to 15%. For our clients, this means faster time-to-market for their modernized applications, and higher value as they shift development capacity towards innovation by saving on maintenance efforts. We're excited to leverage the transformation capabilities of Amazon Q Developer to help our clients achieve greater scalability while adopting modern development practices for building secure and high-performance applications.

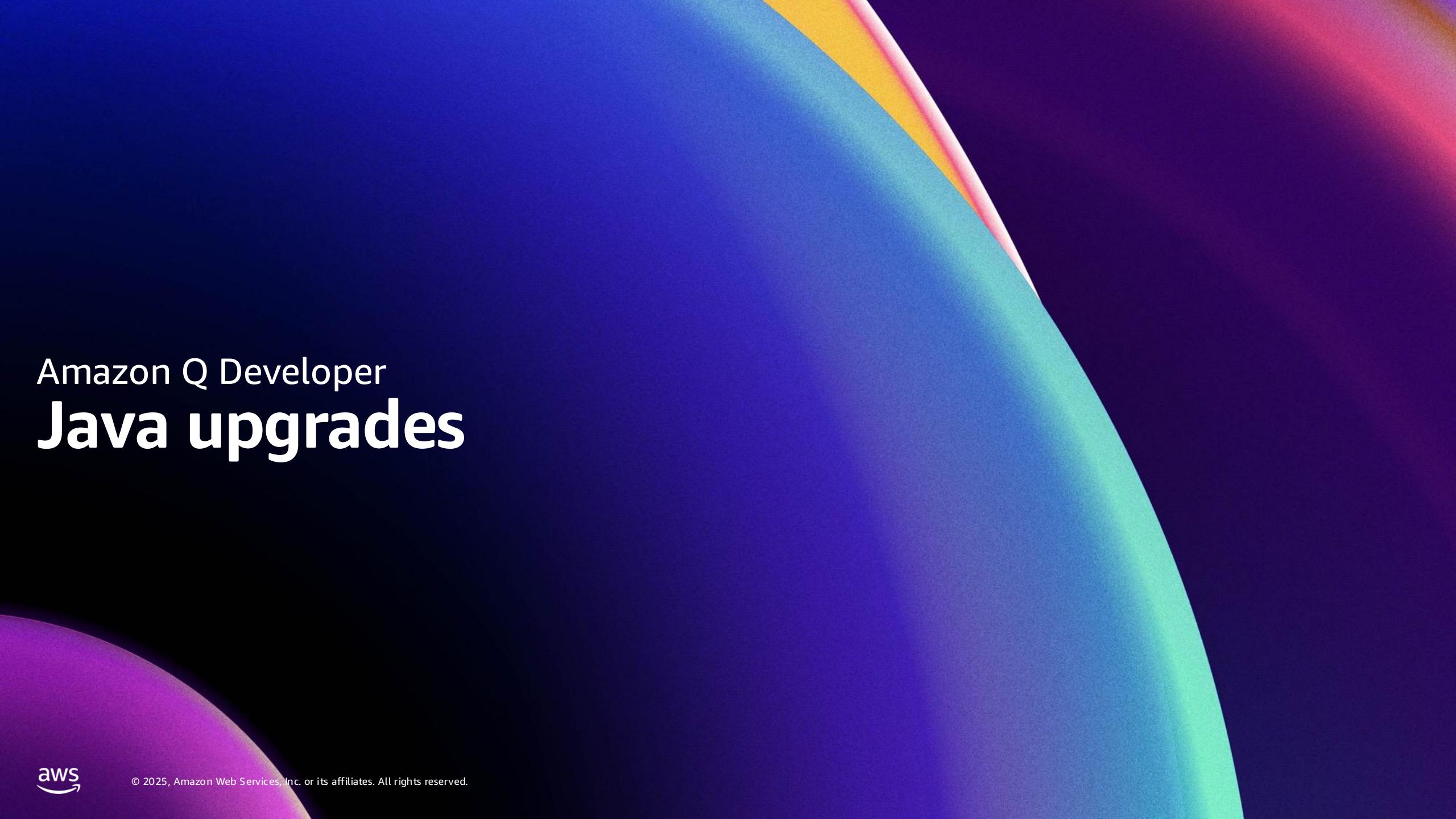
”

Sivaram Sivanarayanan

Global Head of AWS Products & Offerings, AI.Cloud, TCS



Amazon Q Developer **Java upgrades**

The background features a large, abstract, curved shape that resembles a rainbow or a stylized planet. It has a gradient of colors from dark purple to bright yellow and red, with a soft, glowing edge. This shape is positioned in the upper right quadrant of the slide, creating a dynamic and modern feel.

Q Developer Java upgrades



V8, v11, v17, v21



V17, v21



Build & test
Java 8,11,17,21



Apply Knowledge Base



Build & Test
Java 17,21



Fix Errors with GenAI



Continuous Maintenance

Amazon's internal use of Amazon Q



CEO Andy Jassy reported in a recent [LinkedIn post](#) that the internal developer tooling team used Q Developer's new code transformation capability—specifically integrating it with internal development systems—to automate the process of upgrading over 30,000 applications from Java 8 or 11 to Java 17, saving over 4,500 developer years compared to manual upgrades.

Source: <https://news.a2z.com/contents/38885081>

LinkedIn post: https://www.linkedin.com/posts/andy-jassy-8b1615_one-of-the-most-tedious-but-critical-tasks-activity-7232374162185461760-AdSz/?utm_source=share&utm_medium=member_desktop

Getting Started



© 2025, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Deployment decisions

Free tier



Code faster with code suggestions in the IDE and CLI



Use where you work: Your IDE, CLI, the AWS Management Console, etc.



Limited monthly access of advanced features

Pro tier



Manage users and policies with enterprise access controls



Customize Amazon Q to your code base



High limits of advanced features

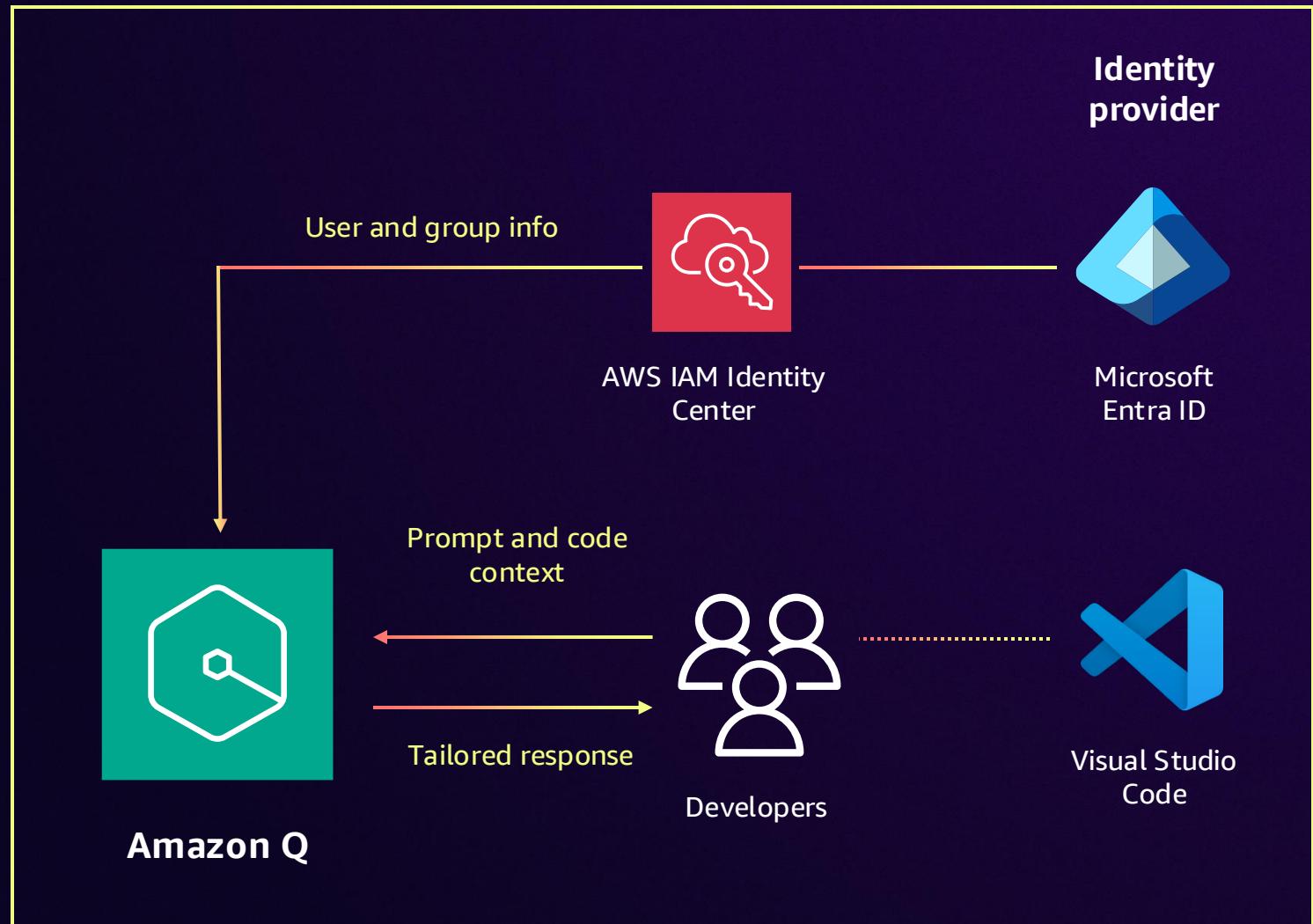
Single sign-on (SSO) integration

User management

Seamlessly match users' corporate credentials and provide access to Amazon Q

Security

Developers securely access and use Amazon Q Developer



Accelerate Evaluation with Workshop Exercises

aws workshop studio

Amazon Q Developer Workshop - Building the Q-Words App

- Introduction
- ▶ Setup and Configuration
- ▶ Code Comprehension and Summarization
- ▶ Code Debugging, Testing, and Optimization
- ▶ Natural Language to New Feature
- ▶ Application Modernization and Transformation
- ▶ Generative AI for Accelerating Software Development
- Resource Cleanup
- Wrapping Up: Next Steps and

Amazon Q Developer Workshop - Building the Q-Words ...

Amazon Q Developer Workshop - Building the Q-Words App



The workshop page displays the title "Amazon Q Developer Workshop - Building the Q-Words App". Below the title is a large image featuring the Amazon Q Developer logo, which consists of the text "Amazon Q Developer" and "Your generative AI-powered assistant" next to a stylized hexagonal icon.

aws workshop studio

Easily port .NET Frameworks apps to Linux with Amazon Q Developer .NET transformation capabilities

- Access Lab Environment
- ▶ Setup
- ▶ Lab 1 - IDE
- ▶ Lab 2 - Web Experience
- Cleanup
- Summary

AWS Documentation Homepage ↗

Content preferences

Language

Easily port .NET Frameworks apps to Linux with Amazon ...

Easily port .NET Frameworks apps to Linux with Amazon Q Developer .NET transformation capabilities

In this workshop, gain hands-on experience porting a .NET Framework application to cross-platform .NET. Use Amazon Q Developer .NET transformation capabilities, a modernization tool powered by generative AI that significantly reduces the time to migrate applications from Windows Server to Linux. Migrating to Linux helps you avoid additional licensing fees and gain performance and security benefits.

<https://catalog.workshops.aws/qwords/en-US>
<https://catalog.workshops.aws/qdevtransformdotnet/en-US>



Developers can be more productive

Accelerate workflow

Reduce repetitive work

Focus on higher value activities



© 2025, Amazon Web Services, Inc. or its affiliates. All rights reserved.

THE WEB THE API THE ML THE AI THE DB THE FS

Thank you!

Michael Lin

linmicht@amazon.com



© 2025, Amazon Web Services, Inc. or its affiliates. All rights reserved.