

AWS Transform for .NET

The first agentic AI service for
modernizing .NET applications at scale

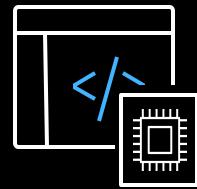
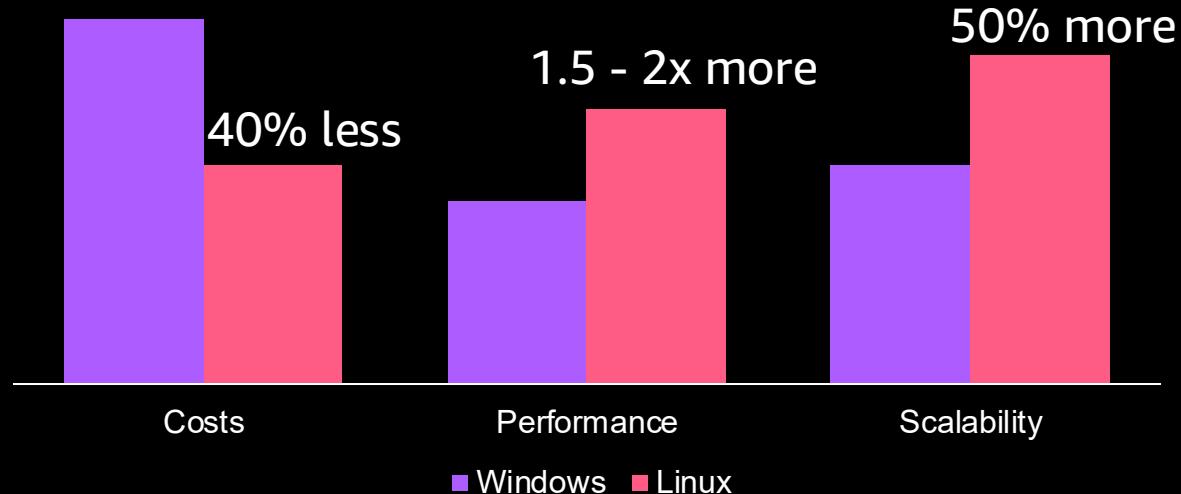


Agenda

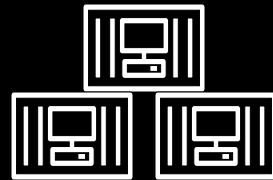
1. Customer challenges
2. How AWS Transform for .NET works
3. Customer success stories

Why modernize .NET apps from Windows to Linux?

Customers have reduced their Windows usage by over 70% to adopt Linux



Compatible with x86-64
and ARM64



Lightweight containers



Serverless architecture
with Lambda



Modernizing can cut compute costs substantially

Examples of EC2 hourly pricing for the same instance type/size

<u>Instance</u>	<u>Windows</u>	<u>Linux</u>	<u>Savings vs. Windows</u>
m7i.2xlarge	\$0.7712	\$0.4032	47.7% savings
c7i.4xlarge	\$1.450	\$0.714	50.8% savings
r7i.8xlarge	\$3.5888	\$2.1168	41.0% savings

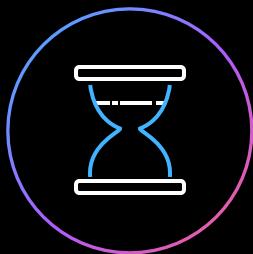
Additional savings by moving to Graviton instances

<u>Instance</u>		<u>Linux</u>	<u>vs. Intel+Windows</u>	<u>vs. Intel+Linux</u>
m7g.2xlarge	—	\$0.3264	57.7% savings	19% savings
c7g.4xlarge	—	\$0.5781	60.1% savings	19% savings
r8g.8xlarge	—	\$1.88512	47.5% savings	11% savings

*Ohio region, July 2024



Porting to cross-platform .NET is hard & slow!



Labor intensive



Error prone



Collaboration
bottlenecks

Porting .NET applications to Linux is complex, requiring code rewrites, error handling, and team coordination

INTRODUCING AWS Transform for .NET

The first agentic AI-powered
experience for modernizing
.NET applications at scale

GENERALLY AVAILABLE

Multifunctional
team collaboration

IDE experience
for developers

A unified
web experience

Foundational
LLMs

.NET
domain-expert
AI agent

Human in
the Loop

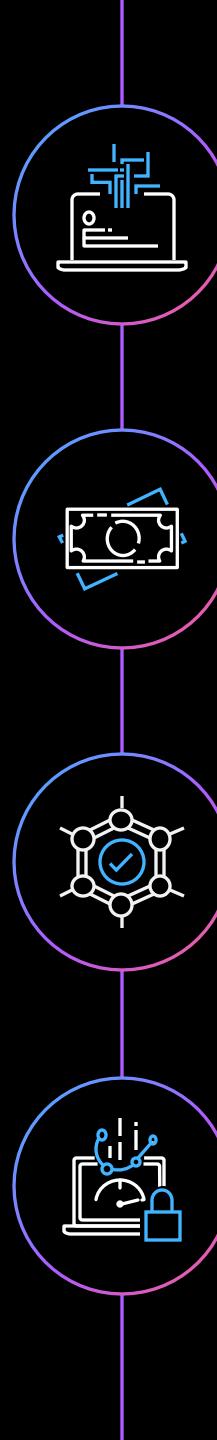


INTRODUCING

AWS Transform for .NET

The first agentic AI-powered
experience for modernizing
.NET applications at scale

GENERALLY AVAILABLE



Offload tedious .NET porting tasks, such as code analysis, dependency mapping, and refactoring, accelerating transformation of .NET applications by up to 4x

Reduce operational costs by up to 40%

Port hundreds of .NET applications in parallel with generative AI agents

Boost application security and performance



How it works



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

What's supported in AWS Transform?

PORTING .NET APPLICATIONS TO BE LINUX-READY

- Supported .NET project types:
 - Libraries
 - Console applications
 - Web API (ASP.NET) Web API
 - Business Logic Layers of SPA (Single Page Application) backends
 - Model View Controller (MVC) applications including front-end Razor views
 - Windows Communication Foundation (WCF) services
 - Unit test projects (NUnit, xUnit, and MSTest)
 - Projects with provided cross-platform versions for third-party or private NuGet packages. If a cross-platform equivalent is missing or unavailable, AWS Transform .NET will attempt a best-effort conversion.

Source versions:

- .NET Framework 3.5+
- .NET Core 3.1
- .NET 5, .NET 6, .NET 7

Target versions:

- .NET 8

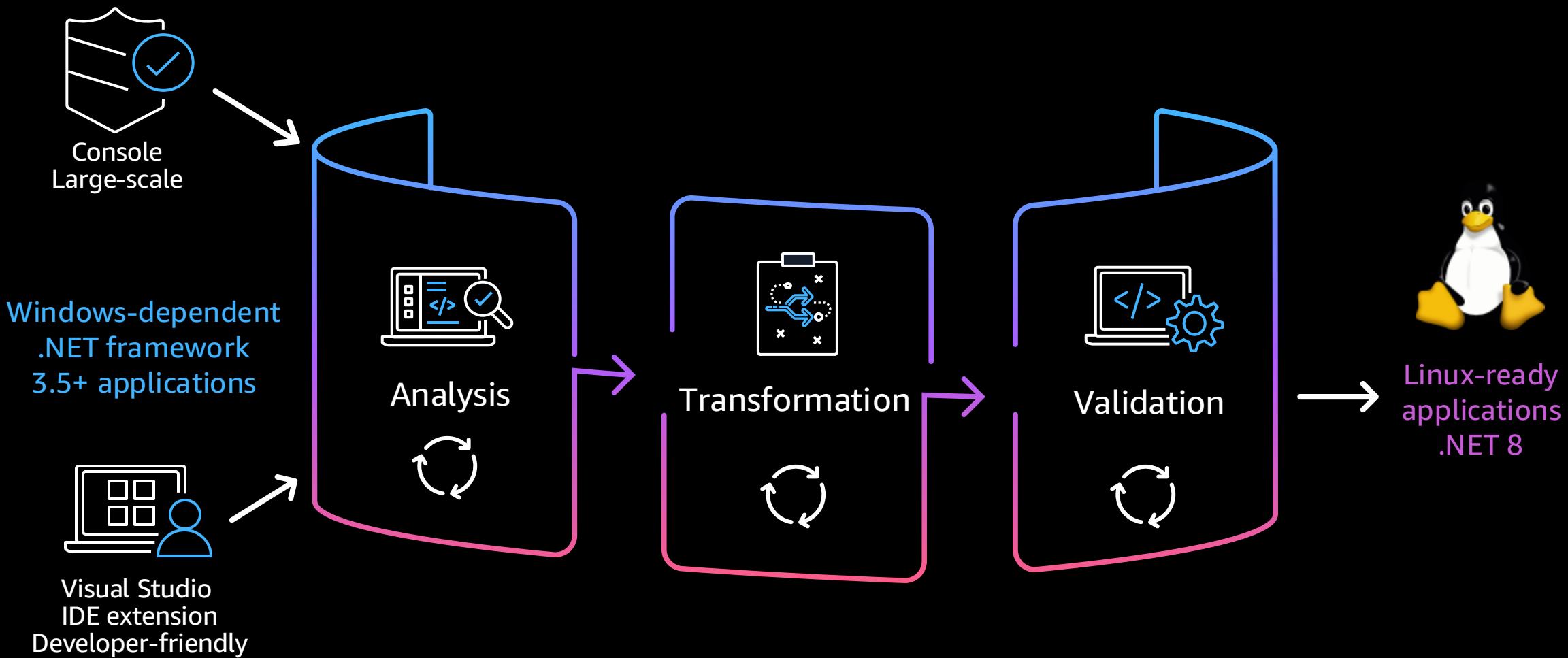
<https://docs.aws.amazon.com/pdfs/transform/latest/userguide/AWSQTransformUserGuide.pdf>



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

.NET Transformation agent

Agent for .NET application porting to Linux



Analysis

Understand code incompatibilities

- **Validates supported project types:** WCF, Web API, MVC/SPA BLLs, Class Library, and Console Apps.
- **Console-only: Automated cross-repo code dependency analysis and downloadable assessment metrics** (LoCs, projects, dependencies) to aid ProServe and partner estimates.
- **Integration with 3P tools** like CAST

Amazon Q Developer: Transform

my-first-workspace

.NET Application modernization [Edit](#)

Option 1 [region-FRA-north] [so](#) [up](#)

Job awaiting user input | Time elapsed: 1hr 1hour [Running](#) [Cancel](#) [@](#)

Review and start transformation [Reject](#) [Approve and start transformation](#)

Required In progress

The user [awscustomer] sent you a list of repositories, dependent repositories, and dependent packages that were selected for transformation. Amazon Q can transform a maximum of 100 dependencies and repositories per transformation plan. Review the list, and when you are ready, approve the list to start the transformation. If you reject the plan, please inform the user [awscustomer] to restart the job.

Job summary

Target branch destination	Target version	Exclude .NET Standard projects
qt-mybranch-10-20-2024	.NET 8.0	Yes

Number of repos: 25 Number of dependent repos: 10 Number of Private NuGet packages: 75

Total lines of code for the job: 2,209,304

Repositories selected (25)

Name	Source branch	Supported projects	Lines of code	Project detected	Projects skipped	Dependencies detected
companyApp-repo	Main	Yes	1,000	5	3 of 3	3 of 3
WebApp-framework	Main	Yes	1,000	75	3 of 3	3 of 3
CloudInfrastructure	Develop	Yes	1,000	84	3 of 3	3 of 3
DevOps-toolset						
Repo 4						
Repo.name2						

Dependent repositories added (25)

Name
companyApp-repo
WebApp-framework
CloudInfrastructure
DevOps-toolset
Repo 4

Dependent packages (25)

Name	Associated repositories	Framework version status	Core version status
AzureCore	20	Resolved	Resolved
AWS SDK Core	10	Resolved	Resolved
Autofac	5	Resolved	Resolved
Corecore.core	5	Resolved	Resolved
Repo 4	5	Resolved	Resolved

```
{ "version": "1.0", "repositories": [ { "owner": "owner1", "name": "Repo 1", "sourceBranch": "mainline", "solutionCount": 3, "projectCount": 4, "projects": [ { "name": "Project1", "version": "0.0.1", "type": "Type1", "privateNugetPackages": { "count": 5 }, "publicNugetPackages": { "count": 6 } } ] } ] }
```



Transformation

Modernize legacy .NET code to cross-platform

- Automated code transformation**
100s of applications (incl. NuGet packages [1P/top35 3P like Newtonsoft, Razor views] (only available in Console, not in IDE)
- Progress visibility and tracking with transformation logs and chat support progress check**
- Developer-led completions of remaining work (~20%) with Visual Studio IDE extension**

The screenshot displays the Amazon Q Developer interface for .NET Application modernization. It includes:

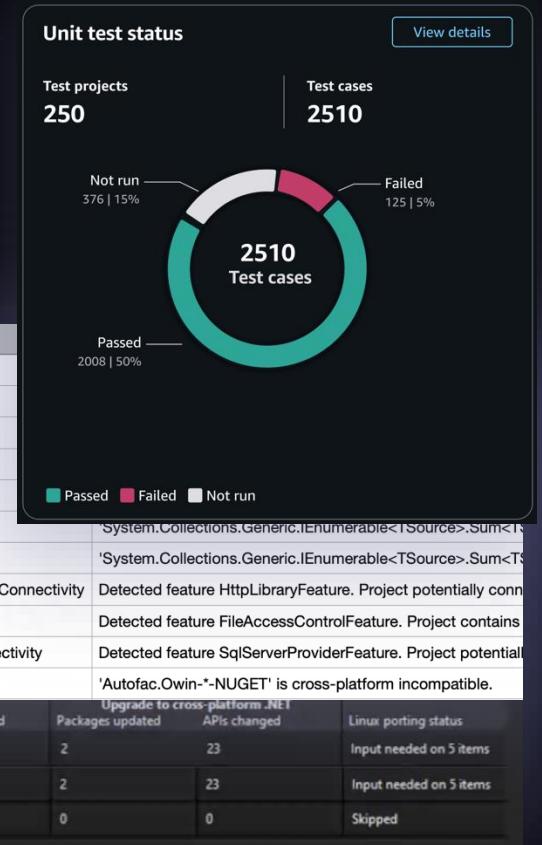
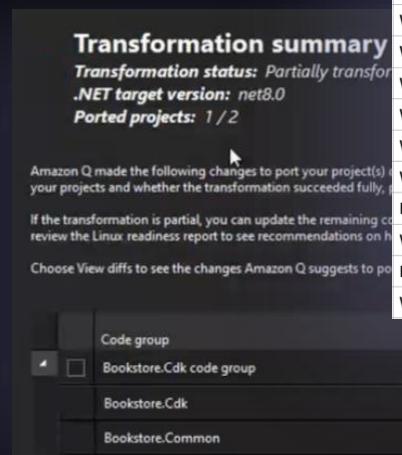
- Job Plan:** Shows the process flow: Get resources to be transformed, Discover resources for transformation, Prepare for Transformation, and Transform.
- Worklog:** A timeline of tasks and their status and duration.
- Dashboard:** Summarizes the outcome of the current job, including target branch destination, target version (.NET 8.0), and job ID.
- Transformation summary:** Shows 40 of 135 repositories transformed, 300 of 374 projects transformed, and 1,000,000 of 1,000,700 lines of code transformed.
- Repository status:** A donut chart showing repository completion status: In progress (40 units, 40%), Completed (40 units, 40%), Failed (25 units, 25%), and Not started (10 units, 10%).
- Project status:** A donut chart showing project completion status: In progress (40 units, 40%), Completed (40 units, 40%), Failed (25 units, 25%), and Not started (10 units, 10%).
- Repositories (135):** A table listing repositories by name, status, and projects.
- Code Editor:** Visual Studio code editor showing a C# file (Bookstore.Data/Bookstore.Data.csproj) with code related to Microsoft .NET SDK and Microsoft Build.
- Solution Explorer:** Shows the project structure including AddressController.cs, AuthenticationController.cs, BookController.cs, HomeController.cs, OrderController.cs, ProductController.cs, SearchController.cs, and various View files.



Validation

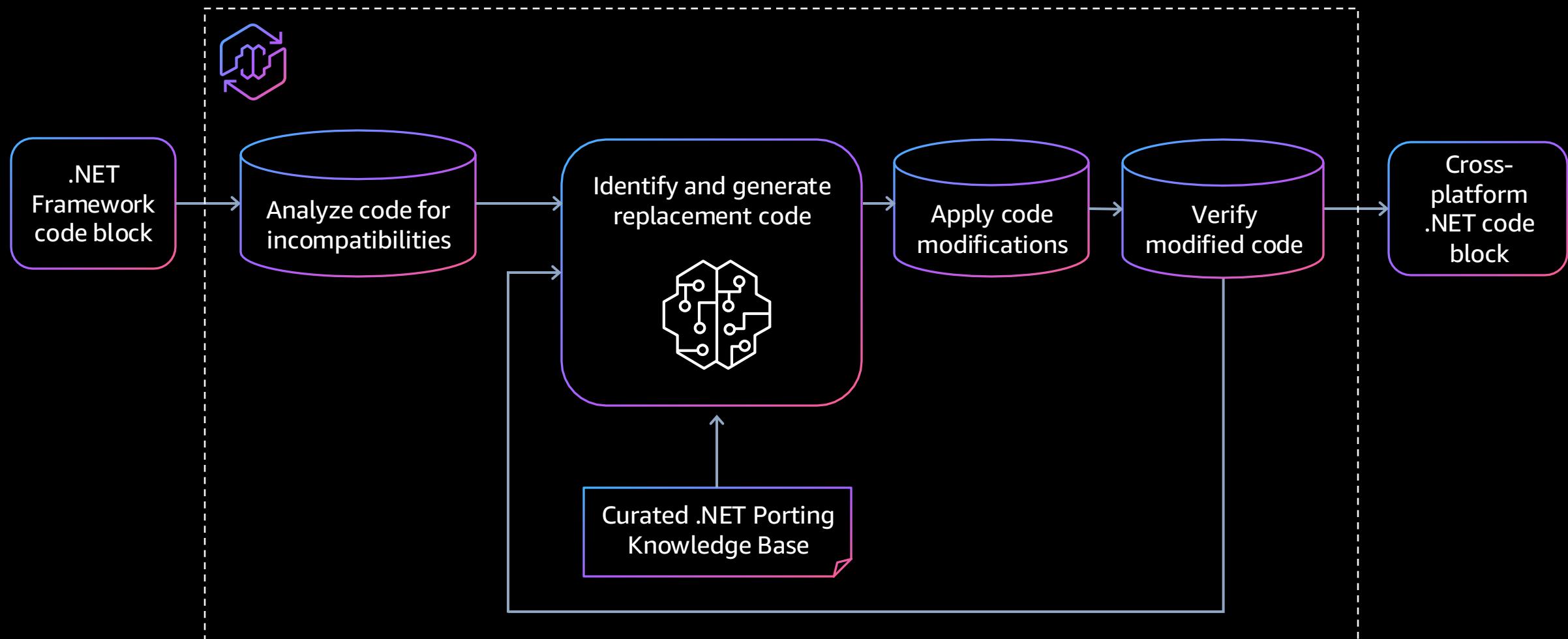
Validate transformed code with human oversight

- **Automated full build & unit test execution** to receive results of running transformed test projects
- **Transformation summary** for explainability of LLM-based code changes for validation
- IDE –only: **Linux readiness report** to identify compatibility issues requiring human attention



How .NET Transformation works?

Analyze, transform, and validate



Console Experience

The screenshot shows the AWS Transform workspace interface. At the top left is the workspace name "DotNetDemo-workspace" with an "Edit" link. Top right features a user icon, a "CA +", and a "Settings" dropdown. Below the header is a descriptive text: "This is your workspace description. Workspace is a collection of jobs, artifacts, connectors and collaborators. You can chat to Q about any of these concepts." A navigation bar at the top has tabs for "Jobs" (selected) and "Artifacts". The main area contains a large, semi-transparent 3D cube on the left and three smaller, semi-transparent 3D cubes arranged vertically on the right. In the center, there's a call-to-action section with the heading "Create your first transformation job with AWS Transform" and a paragraph explaining the process. Below the paragraph is a button labeled "Create a job in AWS Transform". The bottom right corner has a blue circular icon with a white house-like symbol.

DotNetDemo-workspace [Edit](#)

This is your workspace description. Workspace is a collection of jobs, artifacts, connectors and collaborators. You can chat to Q about any of these concepts.

[Jobs](#) [Artifacts](#)

Create your first transformation job with AWS Transform

Within a workspace, you and the collaborators you invite can create and perform transformation jobs. When creating a job, AWS Transform will work with you to define an objective for the job. Based on your objective, AWS Transform will:

1. Generate a plan
2. Start the job
3. Ask for resources or clarification when needed

[Create a job in AWS Transform](#)



Customer success



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

Thomson Reuters turbocharged .NET modernization with AWS Transform

CHALLENGES

Thomson Reuters was struggling with **expensive**, time-consuming **maintenance** of legacy .NET Framework applications that were impeding innovation and competing with their roadmap priorities.

SOLUTIONS

The company implemented AWS Transform, an agentic AI-powered tool designed for **large-scale** .NET modernization that enables parallel processing through both **web interface** and **IDE integration**.

RESULTS

- › Modernized 4x faster: **1.5M lines of code/month**
- › **30% cost savings** moving to Linux
- › 70% reduction in technical debt
- › Transformation time: **2 weeks** vs. months

"AWS Transform felt like an extension of our team—constantly learning, optimizing, and helping us move faster."

Matt Dimich

VP, Platform Engineering Enablement at Thomson Reuters

CUSTOMER PROFILE



I N D U S T R Y
Financial Services

R E G I O N
United States/Global

As a global tech and AI leader powering the legal, tax, and compliance industries, Thomson Reuters is always looking for smarter, faster ways to build. That means staying ahead of the curve—and teaming up with partners like AWS to push what's possible.



The Hartford jumpstarted .NET modernization with AWS Transform

CHALLENGES

The Hartford needed to modernize their critical Vendor Management (Skylynx) application from .NET Framework 4.8 to reduce Windows Server **licensing** costs and optimize **cloud infrastructure**.

SOLUTIONS

They implemented AWS Transform's agentic AI capabilities through both **web experience** and **Visual Studio** extension to port their application to cross-platform .NET 8.

RESULTS

- › Transformed 7,870 lines (30% of codebase) **within hours** in initial adoption
- › Slashed transformation time from months to **weeks**
- › Successfully **modernized** all 4 **core projects**
- › Achieved significant TCO reduction through eliminated Windows licensing

"AWS Transform accelerated our modernization efforts and improved the speed to market of our cloud migration initiatives. It not only helped us speed up legacy code transformation from months to weeks but also provided our developers with valuable insights and suggestions throughout the process."

Gaurav C Patrikar
Director Software Engineering, The Hartford

CUSTOMER PROFILE



The Hartford

I N D U S T R Y
Insurance/Financial Services

R E G I O N
United States

The Hartford is a leader in property and casualty insurance, group benefits, and mutual funds. For over 200 years, the company has provided people and businesses with the support and protection they need to pursue their unique ambitions, seize opportunity, and prevail through unexpected challenges.

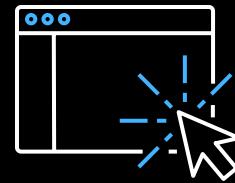


Get started today



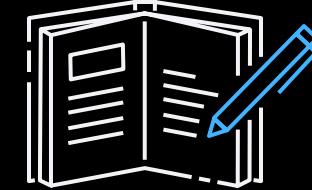
**Learn more about AWS
Transform for .NET**

[https://aws.amazon.com/
transform/net](https://aws.amazon.com/transform/net)



Hands-on demo

[https://aws.storylane.io/share/
5xy6f98m17hm](https://aws.storylane.io/share/5xy6f98m17hm)



View the user guide

[https://docs.aws.amazon.com/
transform/latest/userguide/do
tnet.html](https://docs.aws.amazon.com/transform/latest/userguide/do_tnet.html)



Thank you!

Amazon Q Developer

Coding companion for .NET in the IDE



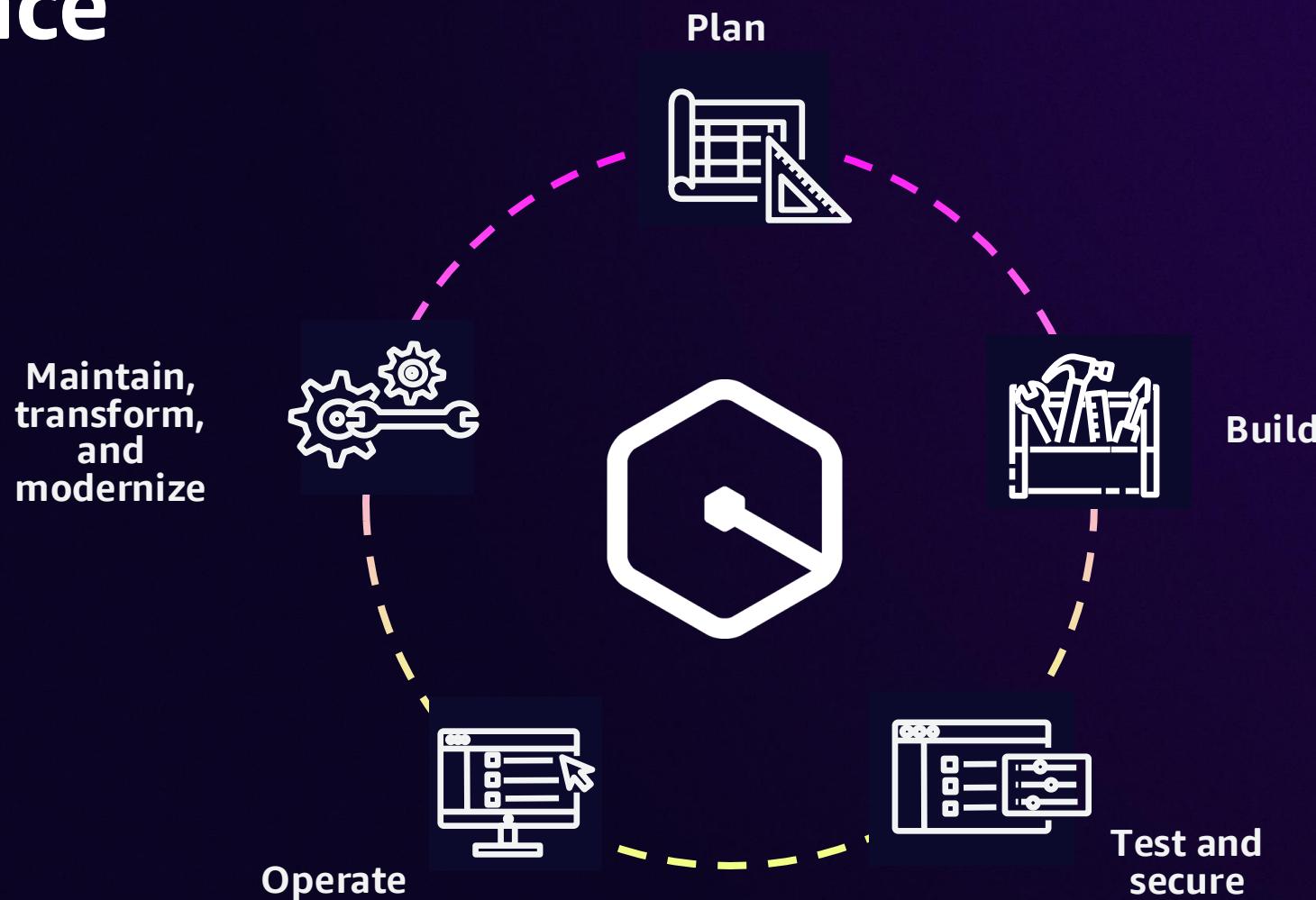
Michael Lin

Sr. Solutions Architect
Amazon Web Services

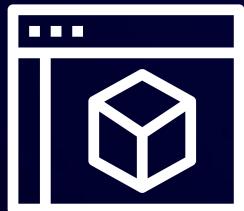


© 2025, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon Q Developer transforms the development experience



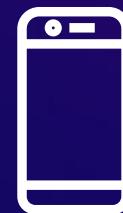
Q Developer is available where you do your work



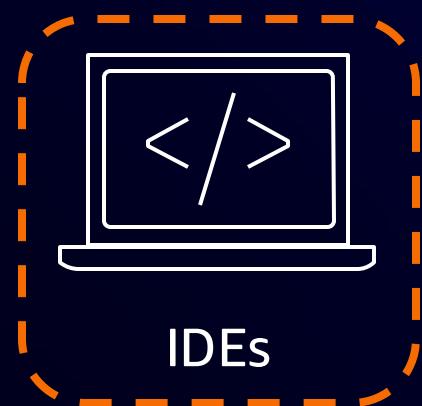
AWS Management
Console



AWS
Documentation



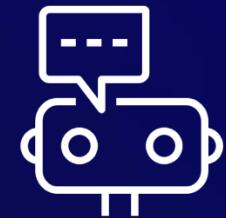
AWS Console
Mobile App



IDEs



Amazon
CodeCatalyst



Slack and Teams
(via AWS
Chatbot)

Supported IDEs for C#/.NET developers



AWS Toolkit for Visual Studio



AWS Toolkit for VS Code



AWS Toolkit for JetBrains IDEs

Install the right extension for your IDE

The screenshot shows the Visual Studio Code interface with the Extension Manager open. The search bar at the top has 'Amazon Q' typed into it. Below the search bar, there are tabs for 'Marketplace' (which is selected), 'Installed', and a gear icon. The main area displays search results for 'Amazon Q' under the 'Marketplace' tab. The results include:

- Amazon Q (by Amazon Web Services) - Version 3.28-241, 727.5K installs, 2.17 rating. Description: Amazon Q is your generative AI-powered assistant across the software development lifecycle.
- AWS Core (by Amazon Web Services) - Version 3.28-241, 775.2K installs, 3.28 rating. Description: AWS Core provides core AWS services.
- AWS Toolkit (by Amazon Web Services) - Version 3.28-241, 7.3M installs, 2.33 rating. Description: AWS Toolkit for Visual Studio Code.
- Elasticsearch Query - EDQL (by chengpohi) - Version 4.14, 33.4K installs, 4.14 rating. Description: Elasticsearch Query - EDQL.

On the left side of the screen, the sidebar shows various extensions installed, including AWS Toolkit, AWS Toolkit with Amazon Web Services, AWS Toolkit for .NET, Amazon Web Services, Avalonia Toolkit, SSIS Integration Toolkit, KingswaySoft, Tano Toolkit, Web Deploy Parameters, Multilingual App Toolkit, and Microsoft. At the bottom of the screen, there are several status bars: Error List, Immediate Window, Packager, TrevCode, Ready, and AWS: 2 of 2 Connected.

Amazon Q Developer features

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.

aws

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

Code completion/generation

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
65 public async void Put(Guid id)
66 {
67     //Update existing Order
68     var context = new DynamoDBContext(_ddbClient);
69     await context.SaveAsync(order);
70 }
71
72 // DELETE: api/ApiWithActions/{id}
73 [HttpDelete("{id}")]
74 [Authorize(Policy = "IsRegisteredUser")]
75 0 references
76 public async void Delete(Guid id)
77 {
78 }
```

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 { Suggestion 171 | Tab to accept | ↵
72     var transferUtility = new TransferUtility(_s3Client);
73
74     var stream = new MemoryStream(image);
75     transferUtility.Upload(stream, bucket, key);
76 }
```

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

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 practice

Chat x +

What are some ways I can use React.js with .NET?

There are several ways to integrate React.js with .NET, allowing you to leverage the strengths of both technologies. Here are some common approaches: [1]

1. Server-Side Rendering (SSR) with ReactJS.NET: ReactJS.NET is a popular library that allows you to render React components on the server-side within a .NET application. This approach is beneficial for improving initial page load times and SEO. You can use it with ASP.NET MVC or ASP.NET Core. [2]
2. Single Page Application (SPA) with separate frontend and backend: In this approach, you create a React.js frontend as a separate project and use .NET for your backend API. The React app communicates with the .NET backend through HTTP requests, typically using RESTful APIs or GraphQL.
3. Using React components within Razor pages: You can embed React components directly into Razor pages in ASP.NET Core applications. This allows you to use React for specific interactive parts of your application while keeping the rest of your views in Razor.
4. Blazor with React components: Blazor is a .NET framework for building interactive web UIs. You can integrate React components into a Blazor application, allowing you to use existing React libraries alongside Blazor components.
5. .NET Core with Create React App: You can set up a .NET Core backend and use Create React App for the frontend. This approach keeps the frontend and backend separate but allows you to serve the React app from your .NET Core application.
6. Using React in ASP.NET Core with JavaScriptServices: JavaScriptServices is a set of technologies for ASP.NET Core developers to build single-page applications using JavaScript frameworks like React.

When integrating React with .NET, it's important to consider how you'll handle state management, routing, and data flow between the frontend and backend. You may need to use additional libraries or tools to facilitate this integration, depending on your specific requirements.

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 React components in your views



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

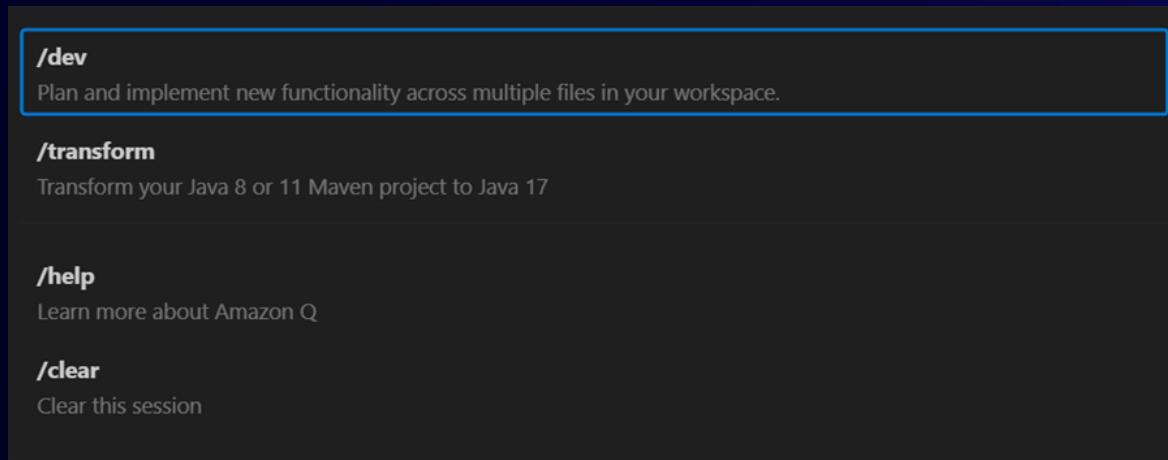
// Mark the item as "want to buy" to move it to shopping cart
wishListItem.WantToBuy = true;
```

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.

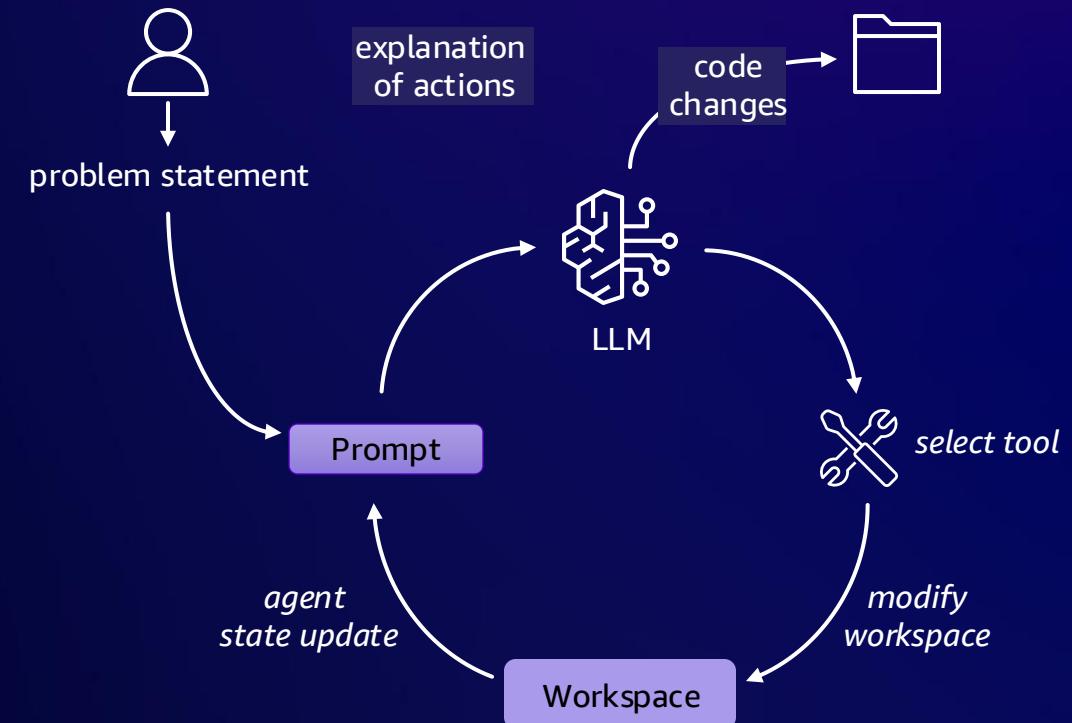


Developer agent for software development

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.



Amazon Q Developer: Transform for .NET

ACCELERATE MIGRATION FROM WINDOWS TO LINUX TO SAVE TIME AND COSTS

Amazon Q Developer Transform: for .NET can perform .NET Framework to cross-platform .NET upgrades to migrate applications from Windows to Linux faster. Avoid additional fees and gain performance and security benefits.



What's supported in Q Developer in VS?

PORTING .NET APPLICATIONS TO BE LINUX-READY

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

Source versions:

- .NET Framework 3.5+
- .NET Core 3.1
- .NET 5+

Target versions:

- .NET 8

<https://docs.aws.amazon.com/pdfs/amazonq/latest/qdeveloper-ug/amazonq-developer-ug.pdf>



© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Q Developer Agent for Code Transformation

The screenshot shows the AWS Toolkit for Visual Studio interface. On the left, the "Getting Started with the AWS Toolkit" documentation page is open, featuring the AWS logo and a green banner indicating "Amazon Q enabled and connected to IAM Identity Center (AmznCorp-IAM)". A "Try Amazon Q with examples" button is also present. Below this, there's a checkbox for allowing usage information collection and a note about AWS Privacy Policy.

In the center, a modal window titled "Port project with Q Developer" is displayed, with the sub-titel "Click to select additional projects." and a dropdown menu showing "Bookstore.Web".

On the right, the Visual Studio context menu is open, showing options like Build, Rebuild, Clean, View, Analyze and Code Cleanup, Convert, Publish, Publish to AWS Elastic Beanstalk, Port project with Amazon Q Developer (which is highlighted), Upgrade, Configure Application Insights, Run Tests, Debug Tests, Show in Test Explorer, Overview, Collapse All Descendants, Scope to This, New Solution Explorer View, Build Dependencies, Add, Manage NuGet Packages, Manage Client-Side Libraries, Manage User Secrets, Configure Startup Projects, Set as Startup Project, Debug, Initialize Interactive with Project, and Git.

The bottom left corner features the AWS logo.

Dashboard and worklog

Job details

Target branch destination
qt-mybranch-10-20-2024

Target version
.NET 8.0

Transformation summary

Repositories transformed
40 of 135

Projects transformed
300 of 374

Repository status

135 Repositories

Legend: In progress (blue), Completed (teal), Pending (light gray), Build errors (orange)

Project status

374 Projects

Legend: In progress (blue), Completed (teal), Pending (light gray), Build errors (orange)

Repositories (135)

Name	Status	Projects	Source branch	Lines
WebAppFramework	In progress	100 of 100	Main	###,###
CloudInfrastructureScripts	In progress	30 of 30	Main	###,###

Job complete | Time elapsed: 1 hour 19 minutes 57 seconds Open in new tab Copy link

Worklog

Timestamps reflect the Pacific Standard Time (PST) time zone.

Nov. 27, 2024, 11:58:14 Step 2 - Transforming project bookstore.domain to net8.0. Q will transform your project to net8.0, it...it will update packages and refactor code.

Nov. 27, 2024, 11:58:15 ...it will update packages and refactor code.

Nov. 27, 2024, 11:58:16 Step 3 - Transforming project bookstore.data to net8.0. Q will transform your project to net8.0, it...will update packages and refactor code.

Nov. 27, 2024, 11:58:17 ...will update packages and refactor code.

Nov. 27, 2024, 11:58:18 Step 4 - Transforming project bookstore.web to net8.0. Q will transform your project to net8.0, it...will update packages and refactor code.

Nov. 27, 2024, 11:58:19 ...will update packages and refactor code.

Nov. 27, 2024, 11:58:20 Step 5 - Transforming project bookstore.cdk to net8.0. Q will transform your project to net8.0, it...will update packages and refactor code.

Nov. 27, 2024, 11:58:21 ...will update packages and refactor code.

Nov. 27, 2024, 11:58:22 Step 6 - Transforming project bookstore.domain.tests to net8.0. Q will transform your project to...net8.0, it will update packages and refactor code.

Nov. 27, 2024, 11:58:23 ...net8.0, it will update packages and refactor code.

Nov. 27, 2024, 11:58:24 Step 7 - Generate transformation summary and perform Linux readiness assessment. Q will generate a...summary of the changes applied to your code and a report detailing how much effort remains to be

Nov. 27, 2024, 11:58:25 ...100% Linux compatible.

Nov. 27, 2024, 11:58:26 Q is transforming BobsBookstore.sln in BobsBookstore: completed 0 lines of code

Nov. 27, 2024, 11:58:27 Q is transforming BSTTiming.sln in BSTTiming: completed 0 lines of code

Nov. 27, 2024, 11:59:14 Q is transforming BobsBookstore.sln in BobsBookstore: completed 0 lines of code

Nov. 27, 2024, 11:59:23 Q has transformed BSTTiming.sln in BSTTiming to .NET 8.0

Nov. 27, 2024, 11:59:26 Q committed the transformed code for BSTTiming.sln

Nov. 27, 2024, 11:59:27 Results by project in qtBenchmark1/BSTTiming: success 1, partial 0, failed 0, skipped 0.

Nov. 27, 2024, 11:59:28 Q finished transformation of the .NET solutions in qtBenchmark1/BSTTiming

Nov. 27, 2024, 12:00:18 Q is transforming BobsBookstore.sln in BobsBookstore: completed 0 lines of code

Nov. 27, 2024, 12:01:22 Q is transforming BobsBookstore.sln in BobsBookstore: completed 0 lines of code

Nov. 27, 2024, 12:02:26 Q is transforming BobsBookstore.sln in BobsBookstore: completed 0 lines of code

Nov. 27, 2024, 12:03:29 Q is transforming BobsBookstore.sln in BobsBookstore: completed 5982 lines of code

Nov. 27, 2024, 12:04:33 Q has transformed BobsBookstore.sln in BobsBookstore to .NET 8.0

Nov. 27, 2024, 12:04:39 Q committed the transformed code for BobsBookstore.sln

Nov. 27, 2024, 12:04:40 Results by project in qtBenchmark1/BobsBookstore: success 5, partial 0, failed 0, skipped 0.

Nov. 27, 2024, 12:04:41 Q finished transformation of the .NET solutions in qtBenchmark1/BobsBookstore

Nov. 27, 2024, 12:04:41 Job lifecycle Q job has completed

Nov. 27, 2024, 12:04:42 Q has completed transformation of .NET solutions

Nov. 27, 2024, 12:05:39 Job lifecycle Q could not complete the job. Try again, omitting the steps that completed successfully.

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.

”

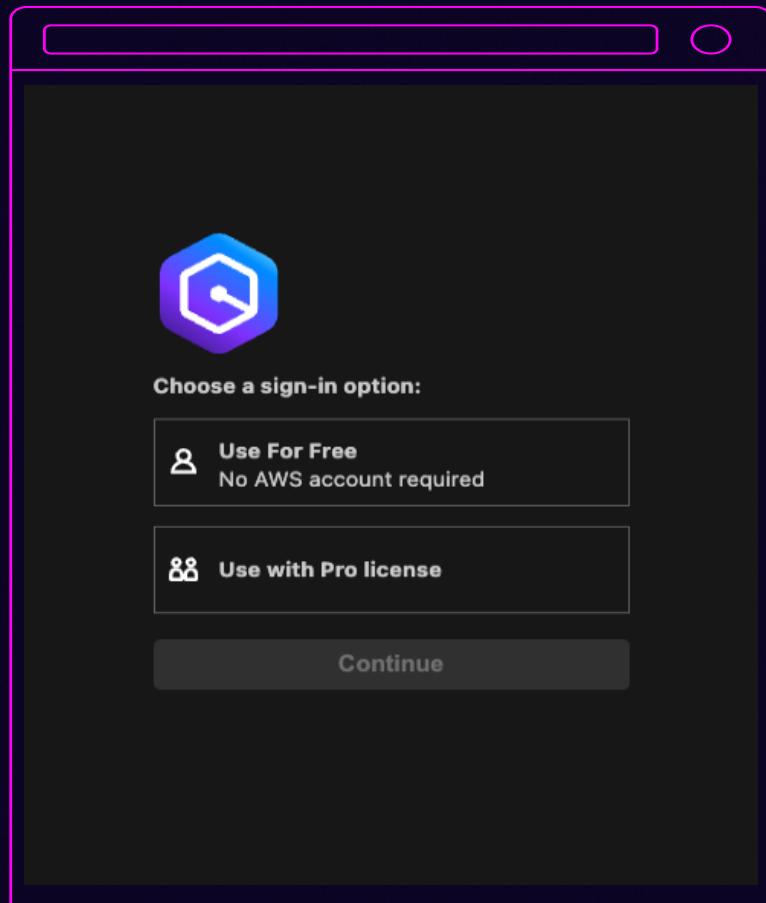
Sivaram Sivanarayanan

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



Authentication mechanisms

AWS BUILDER ID OR AWS IAM IDENTITY CENTER



A screenshot of the AWS IAM Identity Center "Applications" page. The header shows "IAM Identity Center > Applications". On the left, there's a sidebar with links for Dashboard, Users, Groups, Settings, Multi-account permissions (with "AWS accounts" and "Permission sets" sub-options), and Application assignments (with "Applications" sub-option). The main content area is titled "Applications" and describes administering users and groups for AWS managed or customer managed applications. It features an "Add application" button and tabs for "AWS managed" (which is selected) and "Customer managed". Below this, a section titled "AWS managed applications (2)" lists applications, with a search bar below it. The columns for the application list are "Application", "Service", and "Owning account ID".

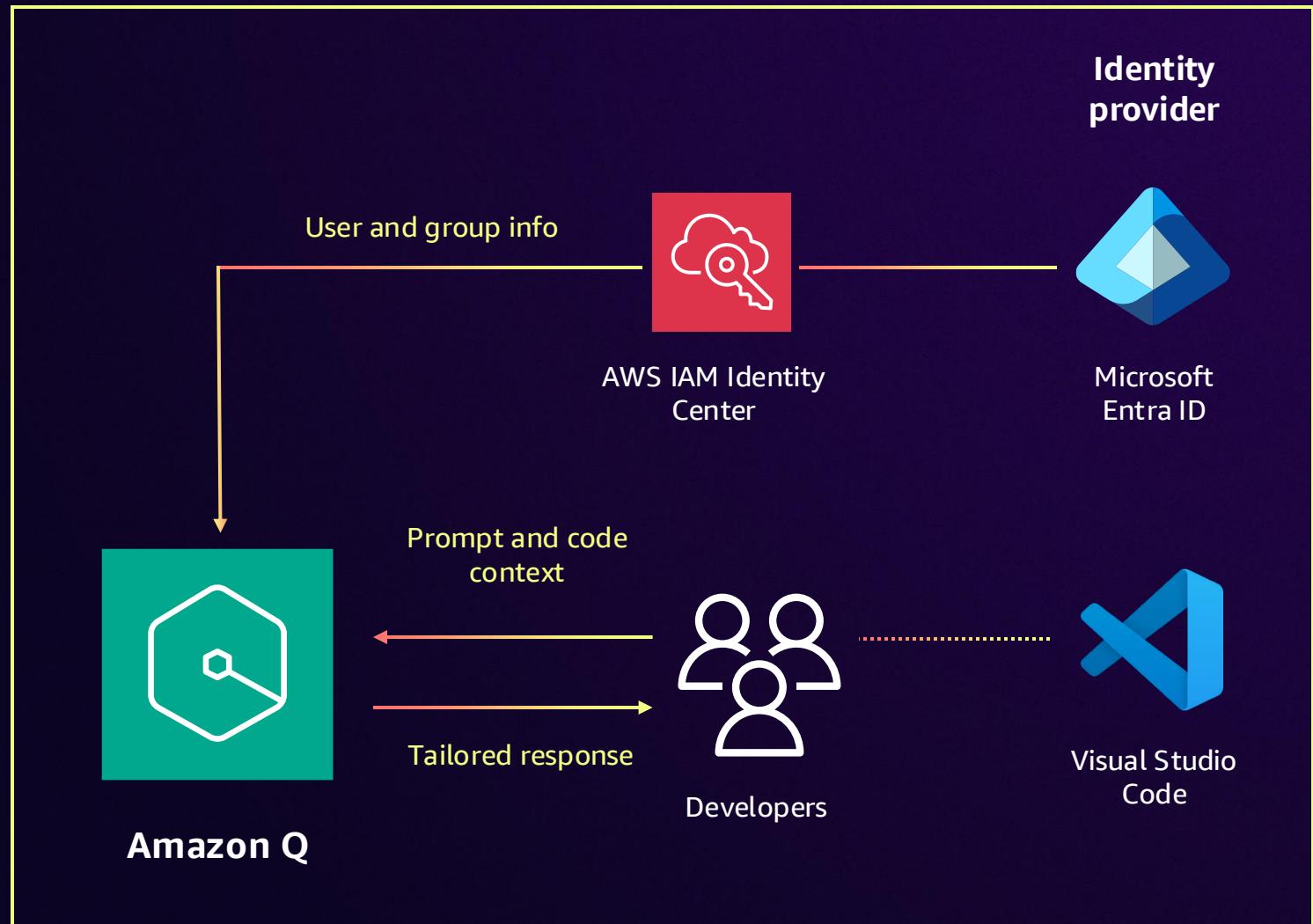
Single sign-on (SSO) integration

User management

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

Security

Developers securely access and use Amazon Q Developer



Thank you!

Michael Lin

linmicht@amazon.com



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

Customize code recommendations

The screenshot shows an IDE interface with a Java file named `FoodDeliveryFulfillment.java`. The code implements a `FoodDeliveryFulfillment` class with a `processUnassignedDeliveries` method. A cursor is positioned at line 2, after the package declaration. A yellow bar highlights the first few lines of the code, and a green bar highlights the `processUnassignedDeliveries` method. A callout box labeled "Natural language prompt" points to the highlighted code. Another callout box labeled "Generated in-line code based on your company's internal code bases" points to the `processUnassignedDeliveries` method.

```
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 }
```

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