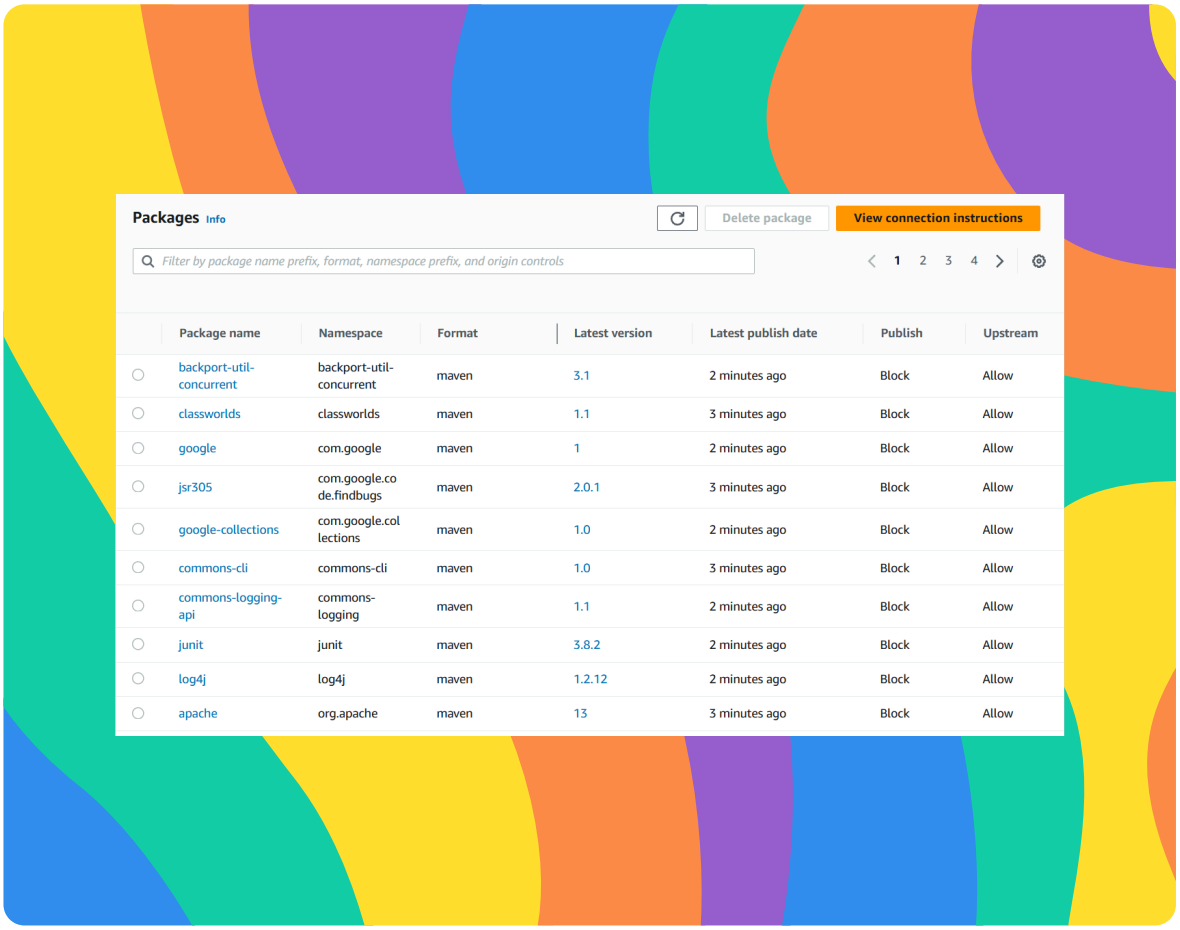
nextwork.org

Secure Packages with CodeArtifact

Mohammed Furqanuddin



Introducing Today's Project!

In this project, I will demonstrate how to create and use a CodeArtifact repository in AWS. I’m doing this project to understand how application packages are stored securely and used in a CI/CD workflow.

Key tools and concepts

Services I used were Amazon EC2, AWS CodeArtifact, IAM, and Apache Maven. Key concepts I learnt include managing cloud servers, securing access with IAM roles, storing and retrieving dependencies using CodeArtifact, and understanding how artifacts are used in real CI/CD pipelines.

Project reflection

This project took me approximately 2–3 hours to complete. The most challenging part was configuring permissions and integrating Maven with CodeArtifact correctly. It was most rewarding to finally see my dependencies appear in CodeArtifact and confirm that the build was working end to end.

placeholder

This project is part three of a series of DevOps projects where I’m building a CI/CD pipeline. I’ll be working on the next project soon, continuing step by step to strengthen my hands-on DevOps and cloud skills.

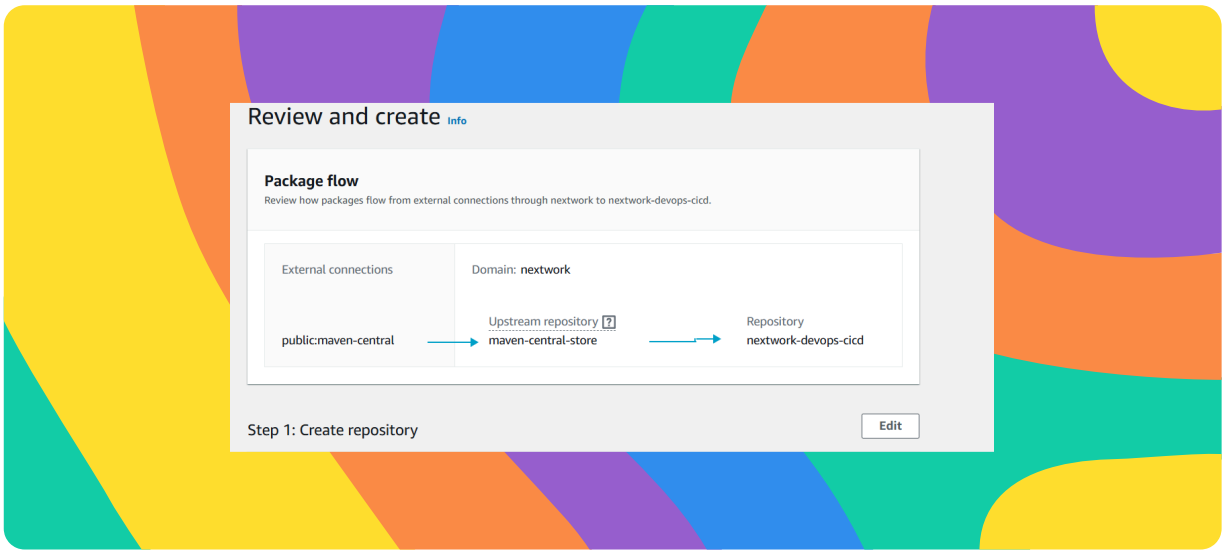
Mohammed Furqanud...
NextWork Student

CodeArtifact Repository

CodeArtifact is an AWS service that acts as a managed artifact repository for storing and sharing software packages and dependencies. Engineering teams use artifact repositories because they provide a secure, centralized place to manage dependencies, control versions, and ensure consistent and reliable builds in a CI/CD pipeline.

A domain in CodeArtifact helps manage multiple repositories by grouping them under a single namespace and access policy. My domain acts as the central place where my repositories are created and managed.

An upstream repository in CodeArtifact allows my repository to pull dependencies from an external source when they are not already available. My repository's upstream repository is a public package repository used to retrieve required packages.



Mohammed Furqanud...
NextWork Student

CodeArtifact Security

Issue

To access CodeArtifact, we need an authorization token to authenticate requests and ensure only permitted resources can download or publish packages. I ran into an error when retrieving a token because my EC2 instance did not yet have the required IAM permissions to access CodeArtifact.

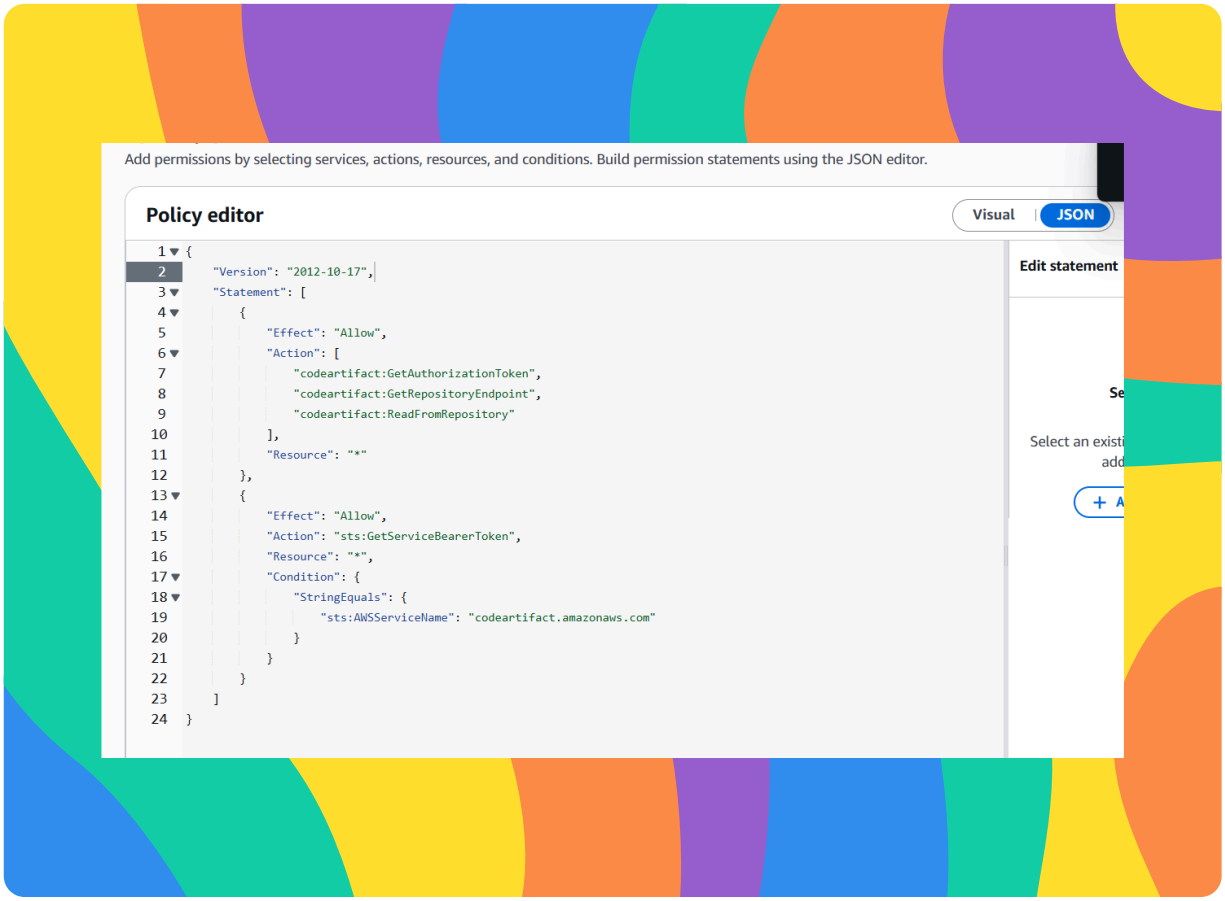
Resolution

To resolve the error with my security token, I created an IAM role with the required CodeArtifact permissions and attached it to my EC2 instance. This resolved the error because the EC2 instance was then able to automatically obtain temporary credentials and authenticate with CodeArtifact.

It’s considered a security best practice to use IAM roles because they allow AWS services to securely access other AWS resources without hardcoding credentials. IAM roles provide temporary, automatically rotated permissions, which reduces the risk of credential leakage and follows the principle of least privilege.

The JSON policy attached to my role

The JSON policy I set up grants permissions to retrieve an authorization token, access the CodeArtifact repository endpoint, and read packages from the repository. These permissions are necessary so my EC2 instance can authenticate with CodeArtifact and download the dependencies required by my Maven project.



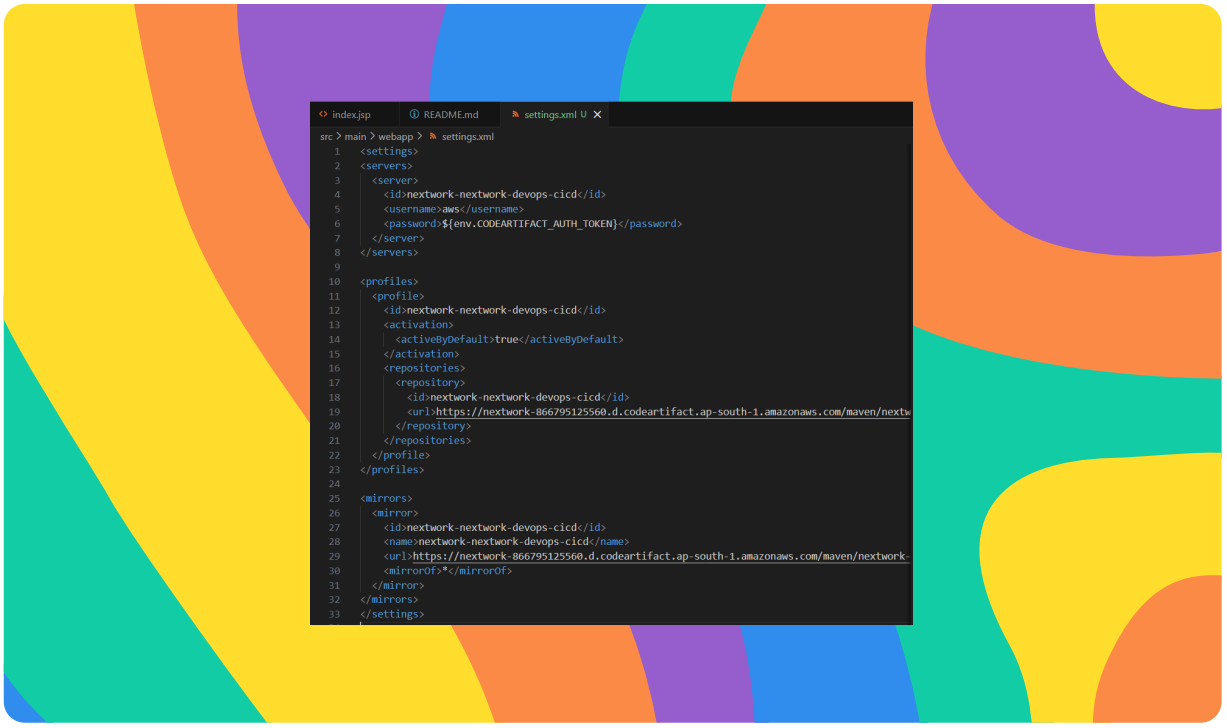
Mohammed Furqanud...
NextWork Student

Maven and CodeArtifact

To test the connection between Maven and CodeArtifact, I compiled my web app using settings.xml

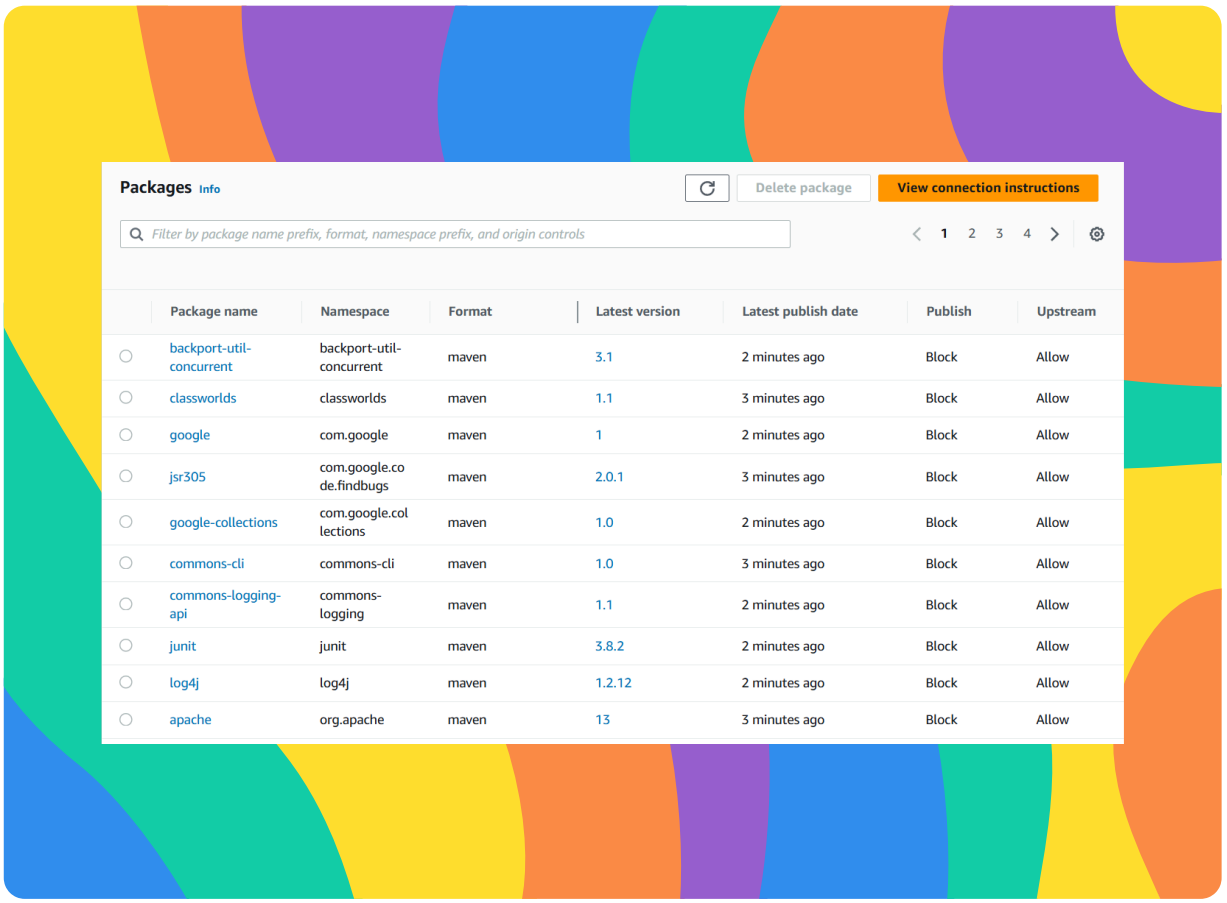
The settings.xml file configures Maven to authenticate with CodeArtifact by specifying the repository endpoint and the authorization token. This allows Maven to securely download dependencies from CodeArtifact during the build process.

Compiling means taking the code that I write as a developer and turning it into something the computer can actually run.



Verify Connection

After compiling my web app, I checked the nextwork-devops-cicd repository in AWS CodeArtifact. I noticed that multiple Maven dependencies, such as junit, log4j, and other Apache and Google libraries, had appeared in the Packages section. This confirmed that Maven successfully pulled the dependencies through CodeArtifact and that the repository was correctly storing and managing my project's artifacts



Package name	Namespace	Format	Latest version	Latest publish date	Publish	Upstream
backport-util-concurrent	backport-util-concurrent	maven	3.1	2 minutes ago	Block	Allow
classworlds	classworlds	maven	1.1	3 minutes ago	Block	Allow
google	com.google	maven	1	2 minutes ago	Block	Allow
jsr305	com.google.code.findbugs	maven	2.0.1	3 minutes ago	Block	Allow
google-collections	com.google.collections	maven	1.0	2 minutes ago	Block	Allow
commons-cli	commons-cli	maven	1.0	3 minutes ago	Block	Allow
commons-logging-api	commons-logging	maven	1.1	2 minutes ago	Block	Allow
junit	junit	maven	3.8.2	2 minutes ago	Block	Allow
log4j	log4j	maven	1.2.12	2 minutes ago	Block	Allow
apache	org.apache	maven	13	3 minutes ago	Block	Allow

Mohammed Furqanud...
NextWork Student
nextwork.org

Uploading My Own Packages

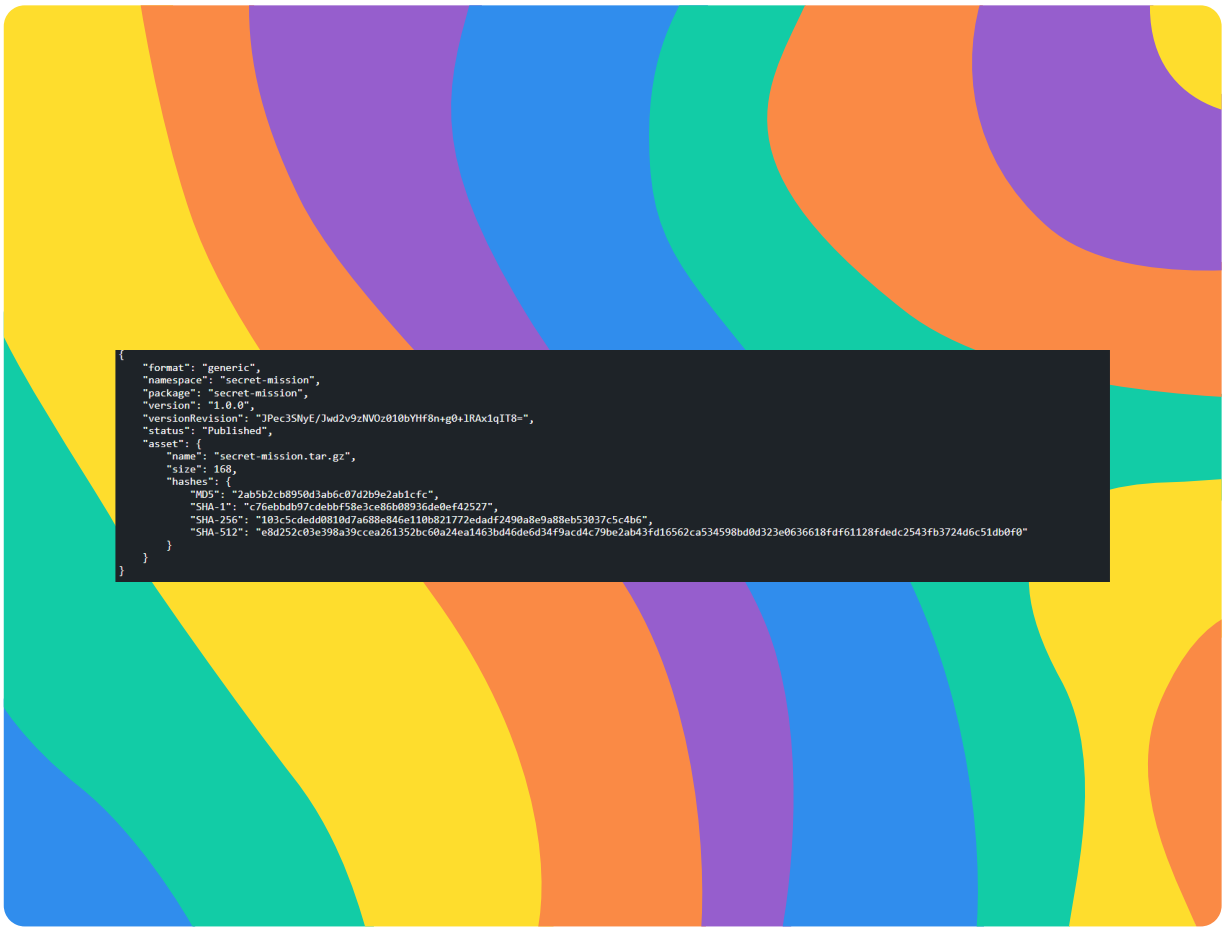
In a project extension, I also decided to create and publish my own custom package to CodeArtifact. This is useful in situations where teams need to share private, internal libraries securely without exposing them to public repositories

To create my own package, I created a simple custom artifact and packaged it so it could be published to my CodeArtifact repository. I also generated a security hash for the package to verify its integrity and ensure that the file was not modified or corrupted during upload or download.

After publishing my package, I checked CodeArtifact and saw my custom package secret-mission listed with version 1.0.0. This confirmed that the package was successfully published and stored in my repository.

I downloaded the package that I uploaded to CodeArtifact to complete the full package lifecycle publish and consume. In real-world teams, one team publishes a package while other teams download and use it. This step proves that my package can be successfully retrieved and used just like a real enterprise dependency. After downloading, I extracted the package and verified its contents to confirm everything worked correctly

Mohammed Furqanud...
NextWork Student



The place to learn & showcase your skills

Check out nextwork.org for more projects