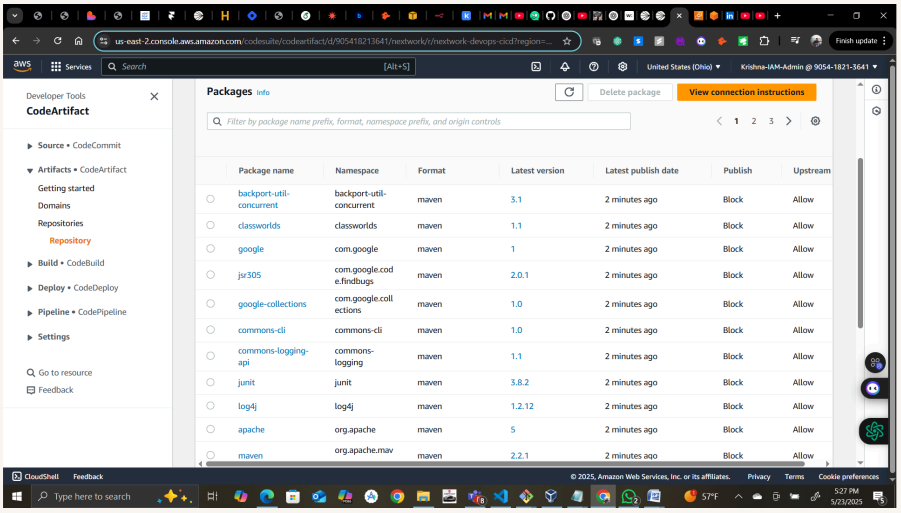
nextwork.org

Secure Packages with CodeArtifact

MO mohankrishnan802@gmail.com



Introducing Today's Project!

In this project, I will demonstrate how to set up an artifact repository using AWS CodeArtifact. I'm doing this project to learn how to store and manage software packages securely as part of a CI/CD pipeline with AWS services.

Key tools and concepts

Services I used were AWS CodeArtifact, EC2, Maven, and IAM. Key concepts I learnt include managing packages, setting up repositories, using auth tokens, and publishing custom packages in a CI/CD pipeline.

Project reflection

This project took me approximately 2 hours. The most challenging part was configuring permissions and resolving authentication errors. It was most rewarding to successfully publish and manage packages in CodeArtifact.

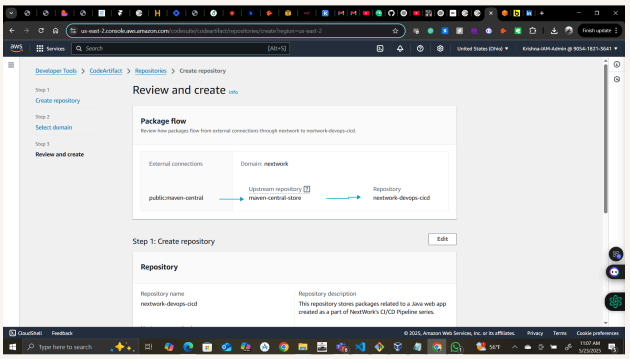
This project is part three of a series of DevOps projects where I'm building a CI/CD pipeline! I'll be working on the next project tomorrow to continue improving my skills and complete the full pipeline setup.

CodeArtifact Repository

CodeArtifact is AWS's fully managed artifact repository service. Engineering teams use artifact repositories because they securely store dependencies, enable reliable package access, and improve consistency in builds across environments.

A domain is a higher-level container in CodeArtifact that groups multiple repositories for easier management and sharing. My domain is set up to organize and manage all related repositories within this project efficiently.

A CodeArtifact repository can have an upstream repository, which means it can fetch packages from another repository if they're not found locally. My repository's upstream repository is the public Maven Central repository for Java packages.



CodeArtifact Security

Issue

To access CodeArtifact, we need an authorization token to securely authenticate requests. I ran into an error when retrieving a token because my EC2 instance lacked the correct IAM permissions to generate it.

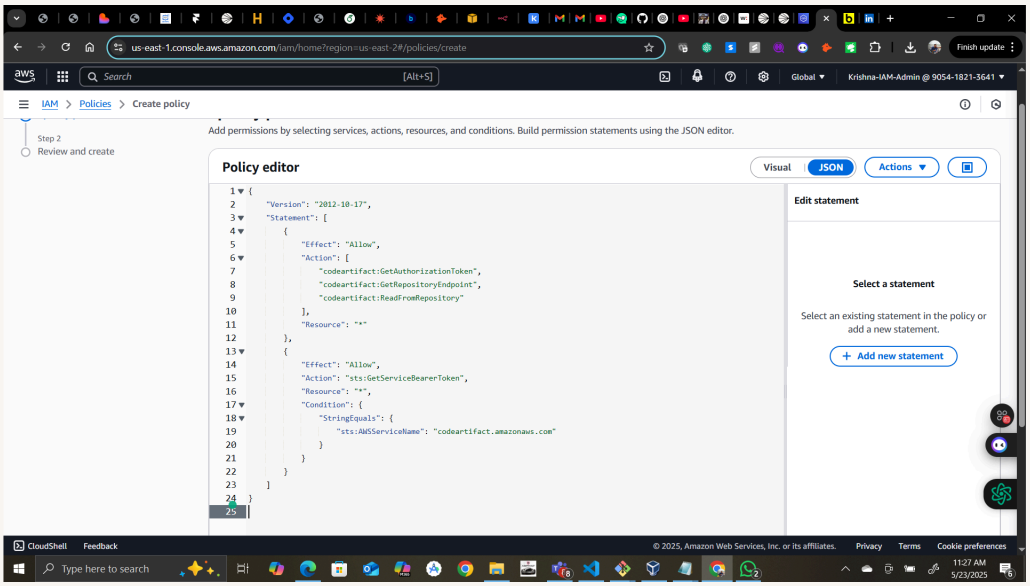
Resolution

To resolve the error with my security token, I created an IAM role with the correct CodeArtifact access policy and attached it to my EC2 instance. This resolved the error because it granted the required permissions to fetch the token securely.

It's security best practice to use IAM roles because they provide temporary credentials that reduce the risk of long-term key exposure. Roles follow the principle of least privilege, granting only the permissions needed for specific tasks.

The JSON policy attached to my role

This JSON policy allows an EC2 instance to access CodeArtifact by granting permissions to get auth tokens, repo endpoints, and read from repos. It also restricts STS token use specifically for CodeArtifact access.

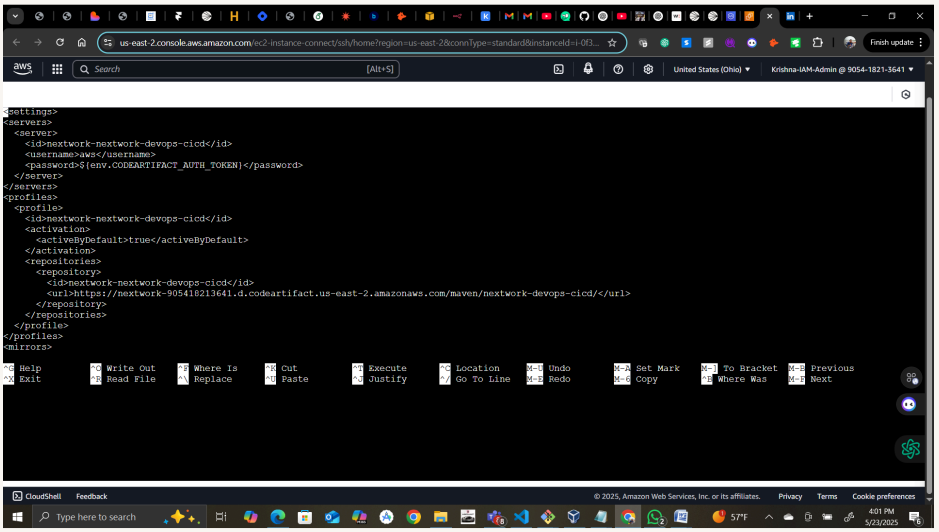


Maven and CodeArtifact

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

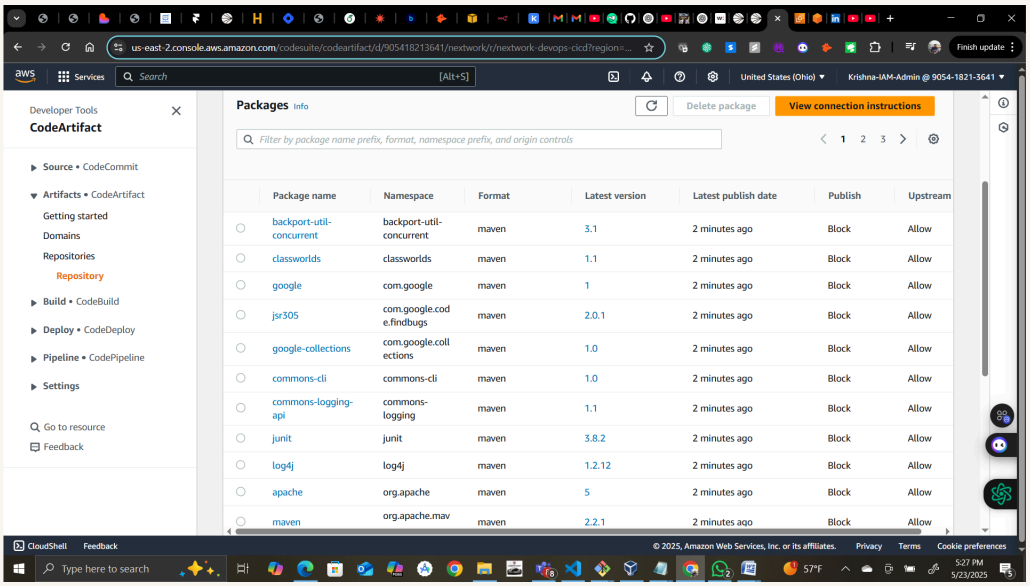
The settings.xml file configures Maven to connect securely to CodeArtifact by specifying the repository URL, authentication token, and credentials. This allows Maven to fetch and upload dependencies directly from the CodeArtifact repository.

Compiling means converting human-readable source code into machine-readable code that a computer can execute. In Java projects, this turns .java files into .class files. It ensures the code is syntactically correct and ready to run.



Verify Connection

After compiling, I checked my CodeArtifact repository. I noticed that it stored the dependencies used by my web app, confirming that Maven successfully retrieved packages from CodeArtifact and the integration was working as expected.



The place to learn & showcase your skills

Check out nextwork.org for more projects