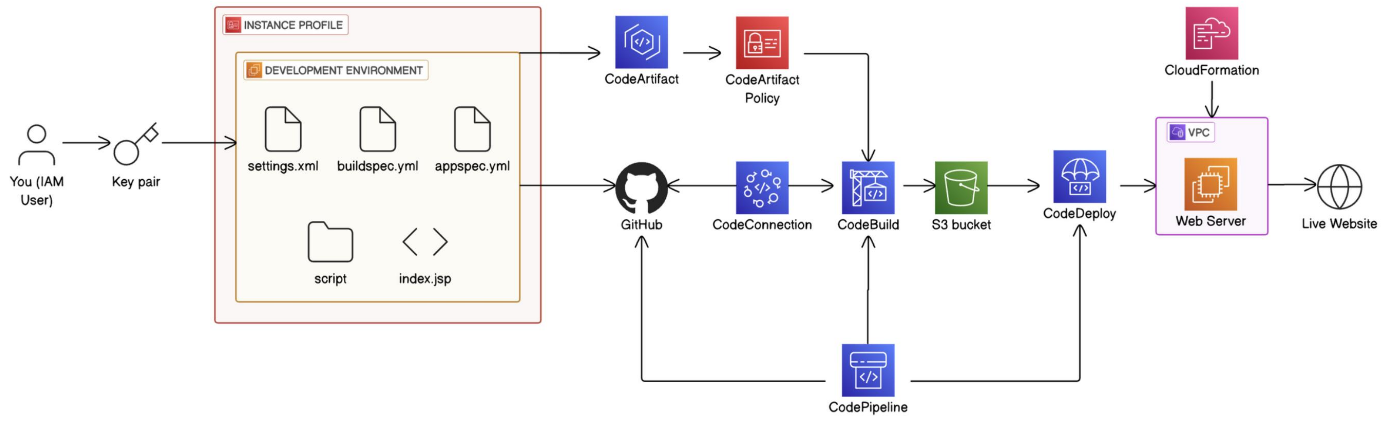
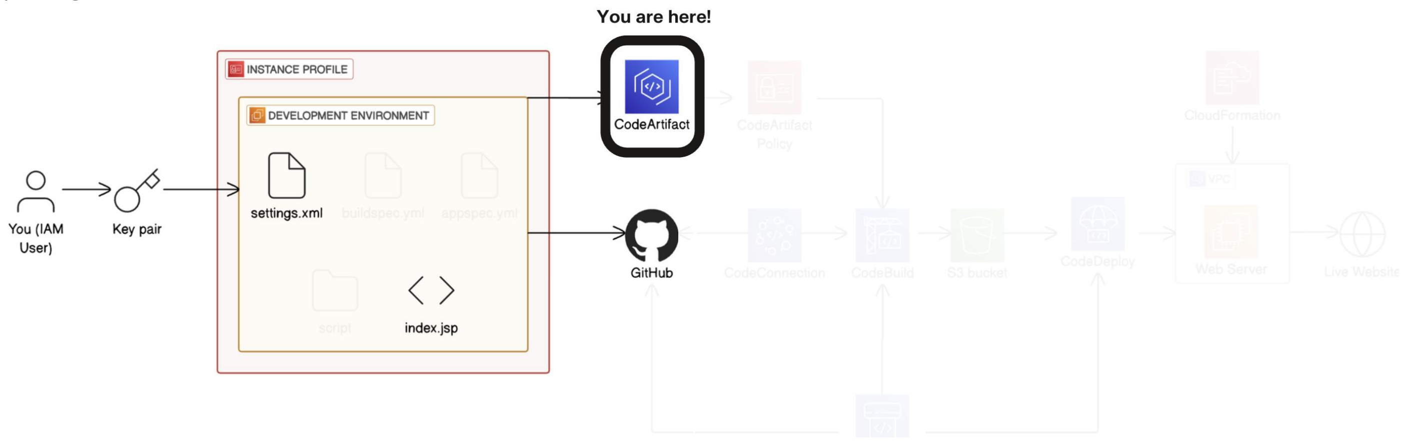
Secure Packages With Code-artifact



Introducing Today's Project!

In this project, I will demonstrate how to set up AWS CodeArtifact, configure IAM roles, and connect it to a web app. I'm doing this to learn how to manage packages in AWS by setting up, securing, and verifying CodeArtifact, then uploading packages!



Key tools and concepts

Services I used were AWS CodeArtifact for package management and AWS CLI for interaction. Key concepts I learnt include dependency resolution, publishing custom packages, versioning, and managing artifact life-cycles using Maven in Java projects.

Project reflection

This project took me approximately a few hours to complete. The most challenging part was configuring the AWS CLI and setting up CodeArtifact. It was most rewarding to publish my custom package and see it successfully integrated into the build process

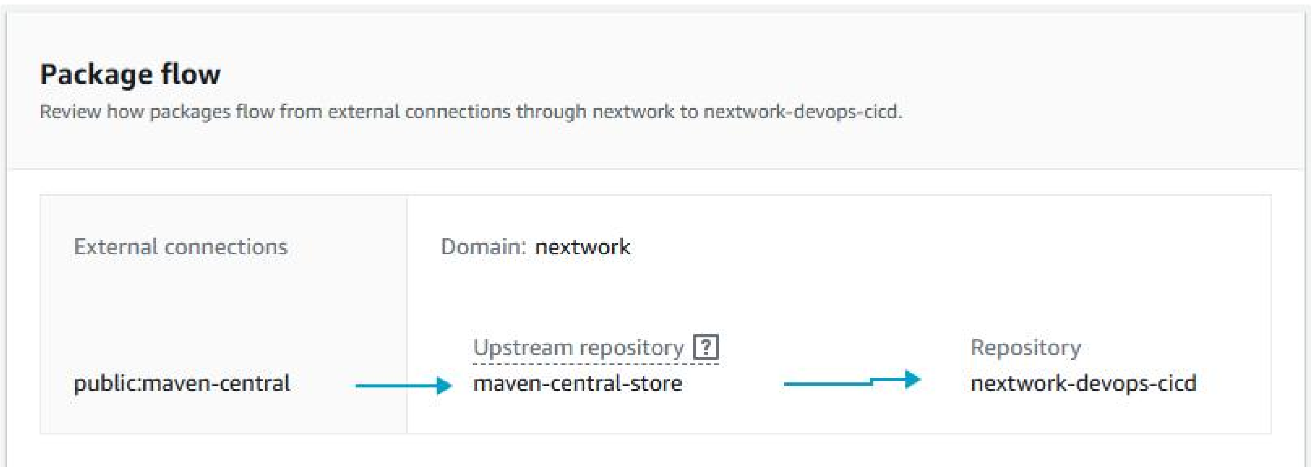
This project is part three of a series of DevOps projects where I'm building a CI/CD pipeline! I'll be working on the next project soon, focusing on automating deployment and integrating testing to further enhance the pipeline.

CodeArtifact Repository

CodeArtifact is an AWS service for securely storing and managing software packages. Engineering teams use artifact repositories because they ensure dependency consistency, enhance security, and simplify package management across projects and teams.

A domain is a central hub in AWS CodeArtifact that lets you manage multiple repositories under one umbrella, enabling shared dependencies and access control. My domain is a scalable solution that simplifies package management across teams and project

A CodeArtifact repository can have an upstream repository, which means it inherits packages while storing its own. My repository's upstream repository is not defined, but in AWS, it could be another CodeArtifact repo or an external registry like PyPI



CodeArtifact Security

Issue

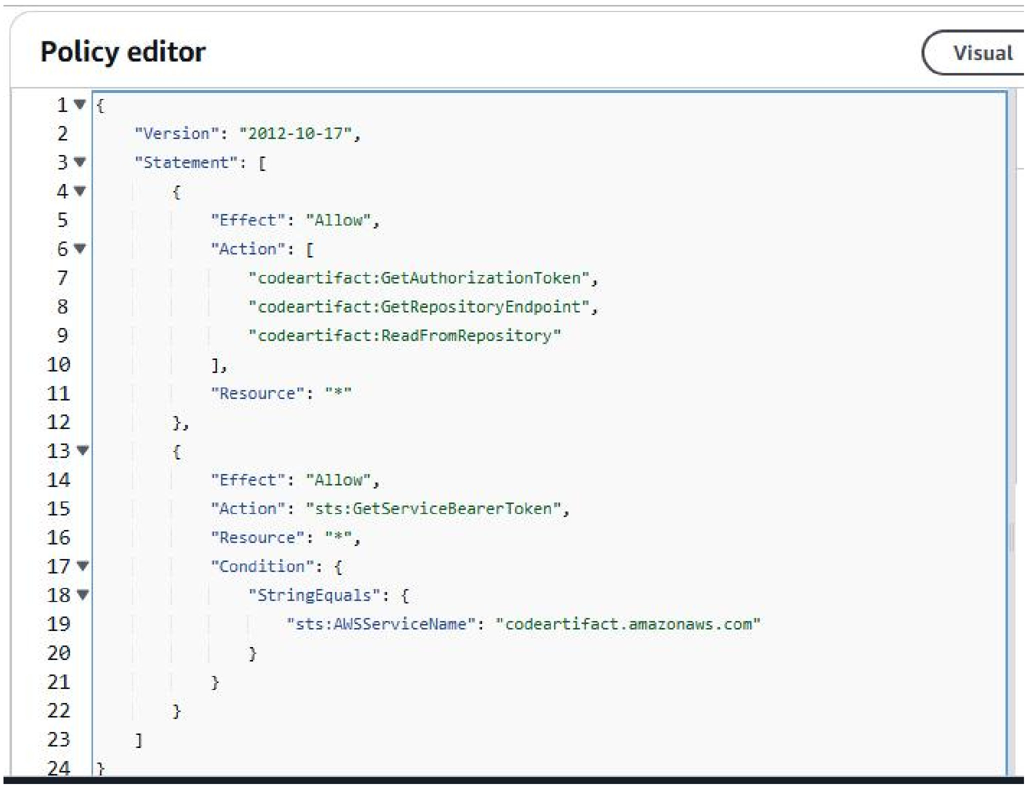
To access CodeArtifact, we need an auth token to securely fetch/publish packages. My error occurred due to missing IAM permissions, expired AWS creds, wrong region, or network issues. Fix: Check policies, renew creds & verify repo details.

Resolution

Fixed the error by correcting IAM policy typos, attaching it to an EC2 role, and verifying AWS region. This ensured proper CodeArtifact token permissions with auto-rotated credentials for secure access.
Using IAM roles is a security best practice because they provide temporary credentials instead of permanent keys, enforce least-privilege access, auto-rotate tokens, and enable secure service-to-service access without credential management overhead.

The JSON policy attached to my role

This IAM policy grants permissions to get CodeArtifact auth tokens, read packages, and access repo endpoints. Required for secure package management in npm/pip/Maven workflows.

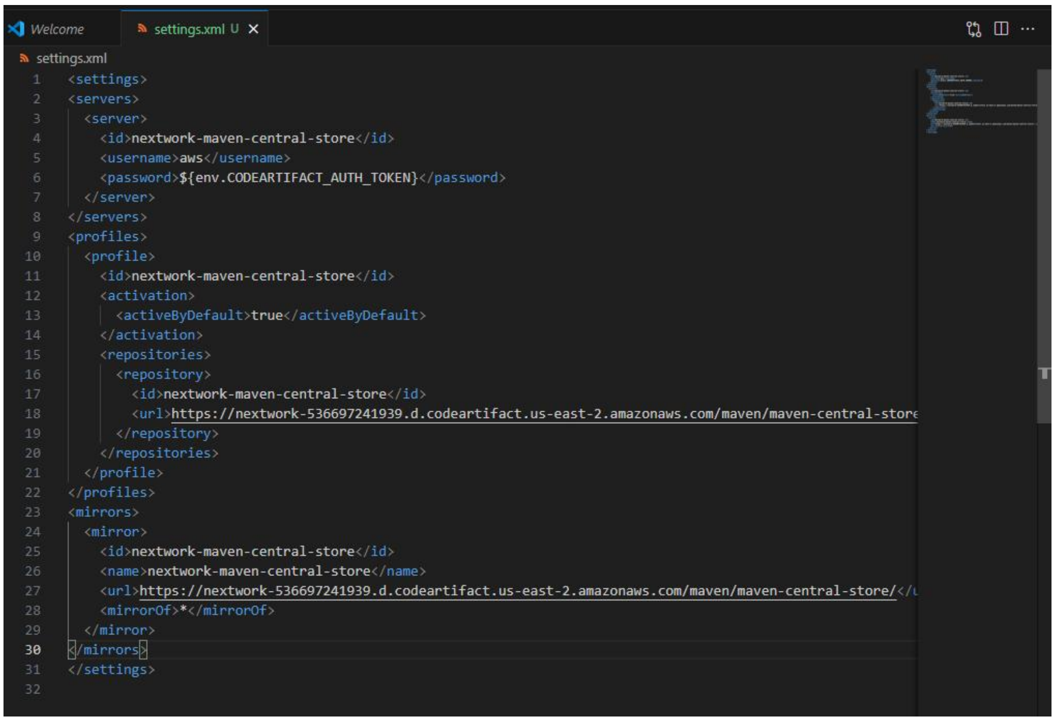


```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codeartifact:GetAuthorizationToken",
                "codeartifact:GetRepositoryEndpoint",
                "codeartifact:ReadFromRepository"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "sts:GetServiceBearerToken",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "sts:AWSServiceName": "codeartifact.amazonaws.com"
                }
            }
        }
    ]
}
```

Maven and CodeArtifact

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

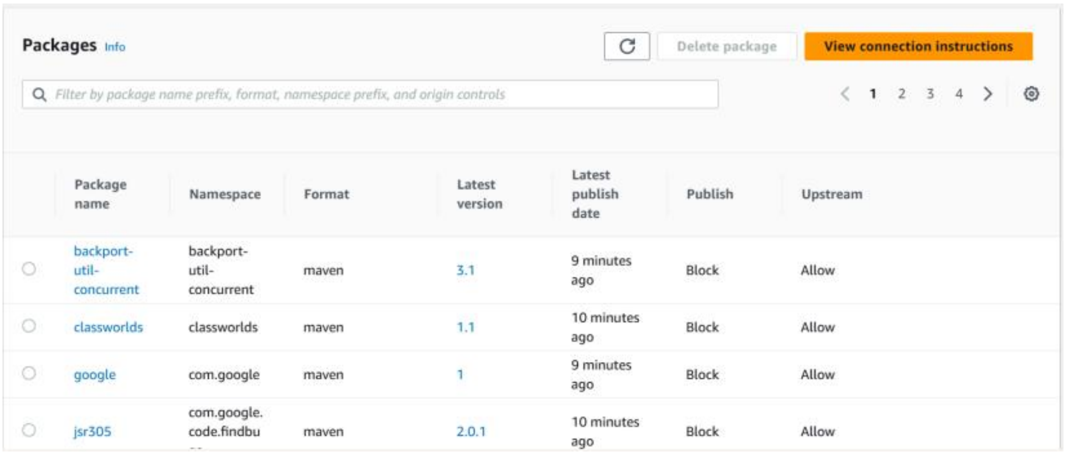
The settings.XML file tells Maven to use your CodeArtifact repo by adding its URL and AWS auth token. This lets Maven securely fetch/publish packages through your private repository instead of public ones.
Compiling means converting source code into byte-code for the JVM. Maven checks CodeArtifact for dependencies and caches them for faster builds. If not found, it pulls from an upstream repo like Maven Central, ensuring efficient and reliable builds.



```
<settings>
<servers>
  <server>
    <id>nextwork-maven-central-store</id>
    <username>aws</username>
    <password>${env.CODEARTIFACT_AUTH_TOKEN}</password>
  </server>
</servers>
<profiles>
  <profile>
    <id>nextwork-maven-central-store</id>
    <activation>
      <activeByDefault>true</activeByDefault>
    </activation>
    <repositories>
      <repository>
        <id>nextwork-maven-central-store</id>
        <url>https://nextwork-536697241939.d.codeartifact.us-east-2.amazonaws.com/maven/maven-central-store
      </repository>
    </repositories>
  </profile>
</profiles>
<mirrors>
  <mirror>
    <id>nextwork-maven-central-store</id>
    <name>nextwork-maven-central-store</name>
    <url>https://nextwork-536697241939.d.codeartifact.us-east-2.amazonaws.com/maven/maven-central-store/</u
    <mirrorOf>*</mirrorOf>
  </mirror>
</mirrors>
</settings>
```

Verify Connection

After compiling, I checked the CodeArtifact repository for the required dependencies. I noticed that Maven first looked there, and if the dependencies weren't found, it fetched them from Maven Central and cached them for faster builds in the future.


Packages Info
Delete package
View connection instructions
Filter by package name prefix, format, namespace prefix, and origin controls
1 2 3 4

	Package name	Namespace	Format	Latest version	Latest publish date	Publish	Upstream
○	backport-util-concurrent	backport-util-concurrent	maven	3.1	9 minutes ago	Block	Allow
○	classworlds	classworlds	maven	1.1	10 minutes ago	Block	Allow
○	google	com.google	maven	1	9 minutes ago	Block	Allow
○	jsr305	com.google.code.findbu	maven	2.0.1	10 minutes ago	Block	Allow

In this project, you've learned how to:

- Set up and configure AWS CodeArtifact as a private Maven repository for managing dependencies.
- Use IAM roles and policies to let your EC2 instance access CodeArtifact.
- Verify your web app's connection to CodeArtifact and ensure Maven can download dependencies from it.
- Create and add your own packages to your CodeArtifact repository!

In the next project, you'll learn how to set up **CodeBuild** to automatically build your web app. No more running commands to get CodeArtifact to compile your project!