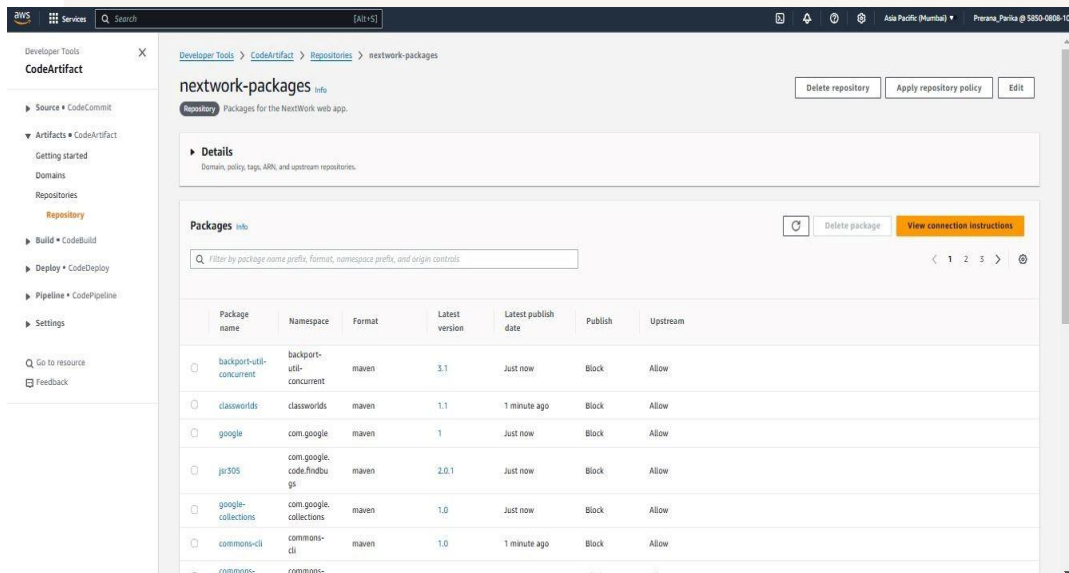


Dependencies and CodeArtifact



The screenshot displays the AWS CodeArtifact console interface. On the left, a navigation pane shows the 'Developer Tools' section with 'CodeArtifact' selected. Under 'CodeArtifact', the 'Repositories' link is highlighted. The main content area shows the 'nextwork-packages' repository details. At the top, there are buttons for 'Delete repository', 'Apply repository policy', and 'Edit'. Below this, the 'Details' section provides information about the repository. The 'Packages' section features a search bar and a table listing packages. The table has columns for Package name, Namespace, Format, Latest version, Latest publish date, Publish, and Upstream. The packages listed are:

| Package name | Namespace | Format | Latest version | Latest publish date | Publish | Upstream |
|--------------------------|--------------------------|--------|----------------|---------------------|---------|----------|
| backport-util-concurrent | backport-util-concurrent | maven | 3.1 | Just now | Block | Allow |
| classworlds | classworlds | maven | 1.1 | 1 minute ago | Block | Allow |
| google | com.google | maven | 1 | Just now | Block | Allow |
| jersey | com.google.code.findbugs | maven | 2.0.1 | Just now | Block | Allow |
| google-collections | com.google.collections | maven | 1.0 | Just now | Block | Allow |
| commons-cli | commons-cli | maven | 1.0 | 1 minute ago | Block | Allow |
| commons- | commons- | maven | 1.1 | Just now | Block | Allow |

Introducing today's project!

What is AWS CodeArtifact?

AWS CodeArtifact is a fully managed artifact repository service that allows you to securely store, share, and manage software packages used in your development projects. It supports popular package formats like Maven, npm, and Python, enabling teams to easily access and manage dependencies. It is useful because it simplifies dependency management, enhances security by controlling access to packages, and integrates seamlessly with other AWS services like AWS CodePipeline and AWS Identity and Access Management (IAM) for streamlined DevOps workflows.

How I used CodeArtifact in this project

In today's project, I used AWS CodeArtifact to manage and store the dependencies for my web app. I connected my project to CodeArtifact by configuring the `settings.xml` file in Maven to authenticate and access the repository. After compiling the web app, the dependencies were pulled from and pushed to the CodeArtifact repository, ensuring seamless management and version control of the packages used in the project. This helped in maintaining consistency across environments and streamlined the build process.

One thing I didn't expect in this project was...

while connecting to the code artifact by the given export statement it was failing later ran the compile statement as

```
export CODEARTIFACT_AUTH_TOKEN=$(aws codeartifact get-authorization-token --
domain nextwork --domain-owner 585008081047 --region ap-south-1 --query
authorizationToken --output text)
```

This project took me...

it took around 40min

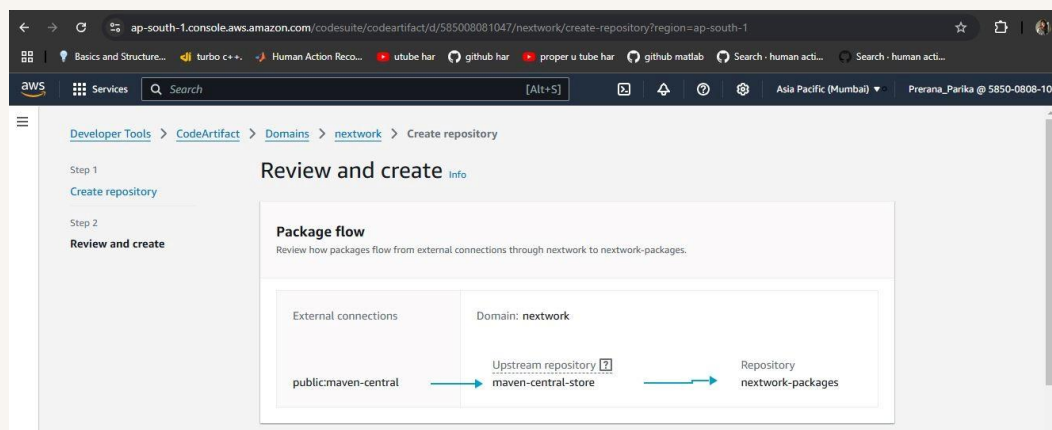
My project has three artifact repositories

The local repository is a version-controlled directory on your local computer where you store and manage files for a project. It contains all the files, folders, and the version history of the project, typically maintained by a version control system like Git. Changes made to the files in the local repository can be staged, committed, and later pushed to a remote repository for sharing or collaboration.

The upstream repository is the primary or original repository, typically hosted on a remote platform like GitHub, GitLab, or Bitbucket, from which a local repository or a

forked repository is cloned. It serves as the central source of truth for the project, and developers often pull updates from it to keep their local or forked repositories in sync. Additionally, changes made in a local or forked repository can be submitted to the upstream repository through pull requests or merge requests for review and integration.

The public repository is a version-controlled repository that is accessible to anyone on the internet, allowing others to view, clone, and contribute to the project without requiring special permissions. It is often hosted on platforms like GitHub, GitLab, or Bitbucket and is commonly used for open-source projects. A public repository enables collaboration, sharing of code, and transparency, allowing developers worldwide to contribute through pull requests, report issues, or fork the repository for their own use.



Connecting my project with CodeArtifact

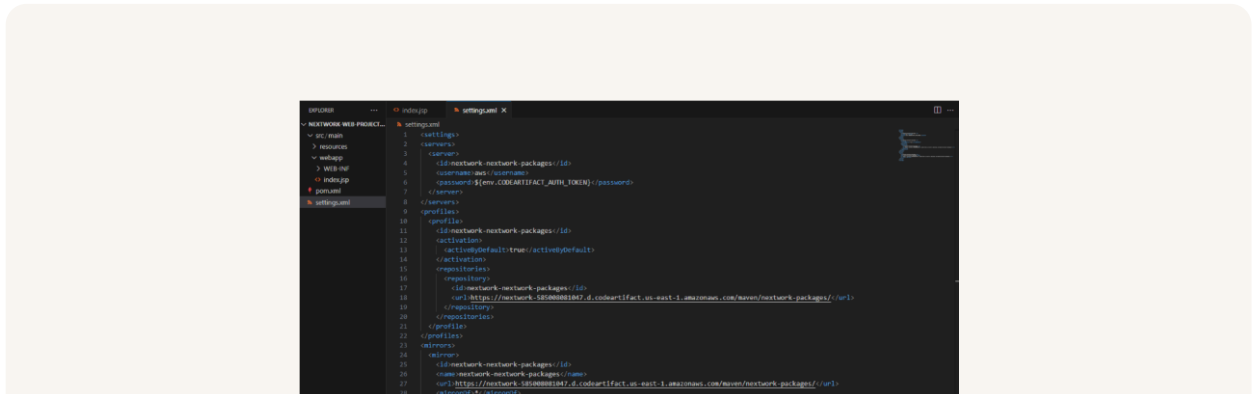
I connected my web app project (via my VSS Code (SSh connection) IDE) to CodeArtifact to securely manage and store project dependencies in a central, scalable repository. This integration allows me to pull, share, and manage libraries efficiently while ensuring consist

I created a new file, settings.xml, in my web app

settings.xml is a configuration file used by Apache Maven, a build automation tool, to define project-specific or user-specific settings for builds. It contains configurations for things like repositories, proxies, authentication details, and build profiles. This file is

typically found in the Maven installation directory or the user's home directory. It allows customization of build behavior, such as defining which remote repositories Maven should use to fetch dependencies or configuring other global or local settings for Maven projects.

The code configures Maven to authenticate with AWS CodeArtifact using an environment token. It sets the CodeArtifact repository as the default for dependencies and mirrors all repositories through it, ensuring seamless interaction with your CodeArtifact packages.



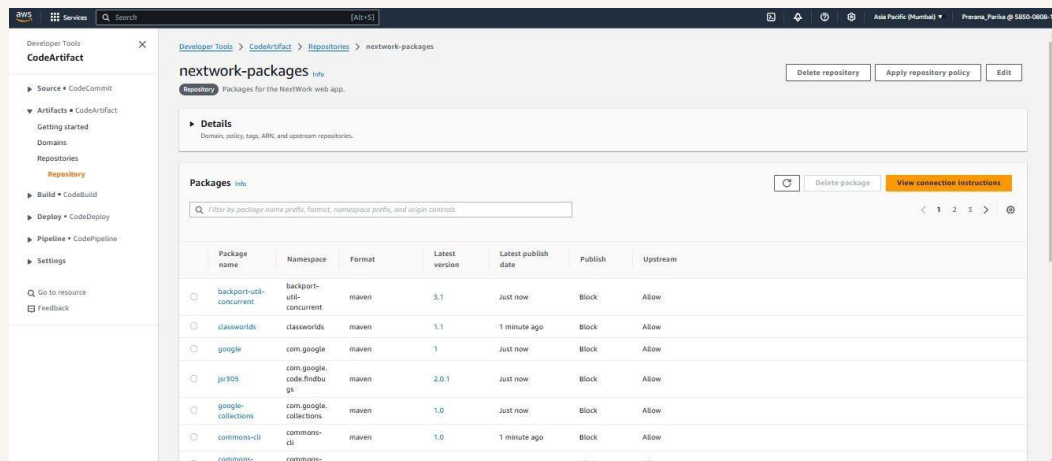
Testing the connection

To test the connection between VS Code (ec2 ssh) and CodeArtifact, I compiled my web app

Compiling means the process of converting source code written in a programming language into machine code or bytecode that can be executed by a computer. It checks for syntax errors and creates an executable or intermediary file, enabling the program to run efficiently.

Success!

After compiling, I checked the CodeArtifact repository. I saw that the compiled dependencies were successfully pushed and stored there, allowing me to verify that the necessary packages were available for use in the project, ensuring smooth builds and integration.



Create IAM policies

The importance of IAM policies

I also created an IAM policy because it defines permissions for the service to interact with CodeArtifact. It allows retrieving an authorization token, locating the repository endpoint, and reading packages from the repository, ensuring secure and controlled access to resources.

I defined my IAM policy using JSON

This policy will allow the service to perform three actions: get an authorization token (`codeartifact:GetAuthorizationToken`), locate the repository endpoint (`codeartifact:GetRepositoryEndpoint`), and read/download packages from the repository (`codeartifact:ReadFromRepository`).

