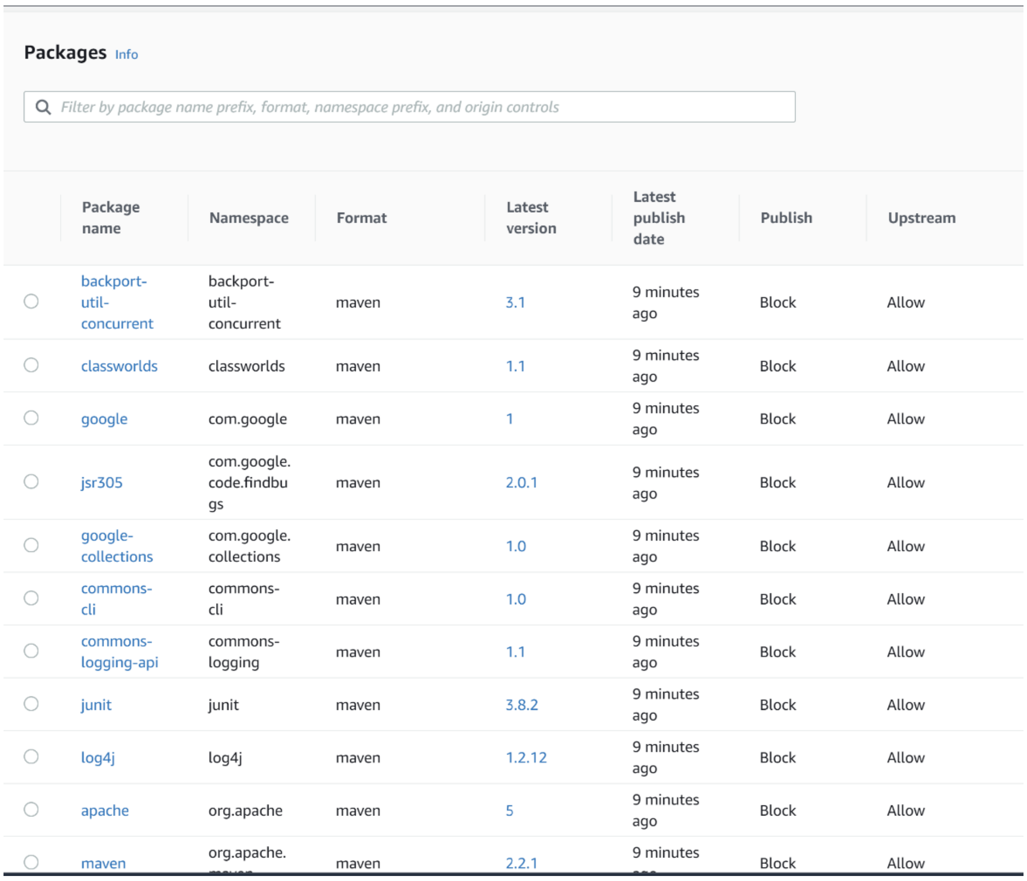
NextWork.org

Dependencies and CodeArtifact

Prajit Venkatachalam
https://www.linkedin.com/in/prajit-venkatachalam-435b2a150/



Packages Info

Filter by package name prefix, format, namespace prefix, and origin controls

	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	9 minutes ago	Block	Allow
○	google	com.google	maven	1	9 minutes ago	Block	Allow
○	jsr305	com.google.code.findbugs	maven	2.0.1	9 minutes ago	Block	Allow
○	google-collections	com.google.collections	maven	1.0	9 minutes ago	Block	Allow
○	commons-cli	commons-cli	maven	1.0	9 minutes ago	Block	Allow
○	commons-logging-api	commons-logging	maven	1.1	9 minutes ago	Block	Allow
○	junit	junit	maven	3.8.2	9 minutes ago	Block	Allow
○	log4j	log4j	maven	1.2.12	9 minutes ago	Block	Allow
○	apache	org.apache	maven	5	9 minutes ago	Block	Allow
○	maven	org.apache.	maven	2.2.1	9 minutes	Block	Allow

Introducing today's project!

What is AWS CodeArtifact?

AWS CodeArtifact is a repository service provided by AWS which enables organizations to securely store, publish, and share software packages used in the application development.

How I used CodeArtifact in this project

I used CodeArtifact in this project to store backup copies of the packages and dependencies relevant to my Java web app. This approach helps with security, risk management, and ensures continuity, even if the public packages or dependencies for my project become unavailable, as a copy will remain accessible in CodeArtifact.

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

One thing I didn't expect in this project was the issue with accessing Cloud9, as it was not available in my region. Another unexpected aspect was needing to use HTTPS Git credentials and AWS Access Key ID and Secret Access Key for Git authentication in Visual Studio Code.

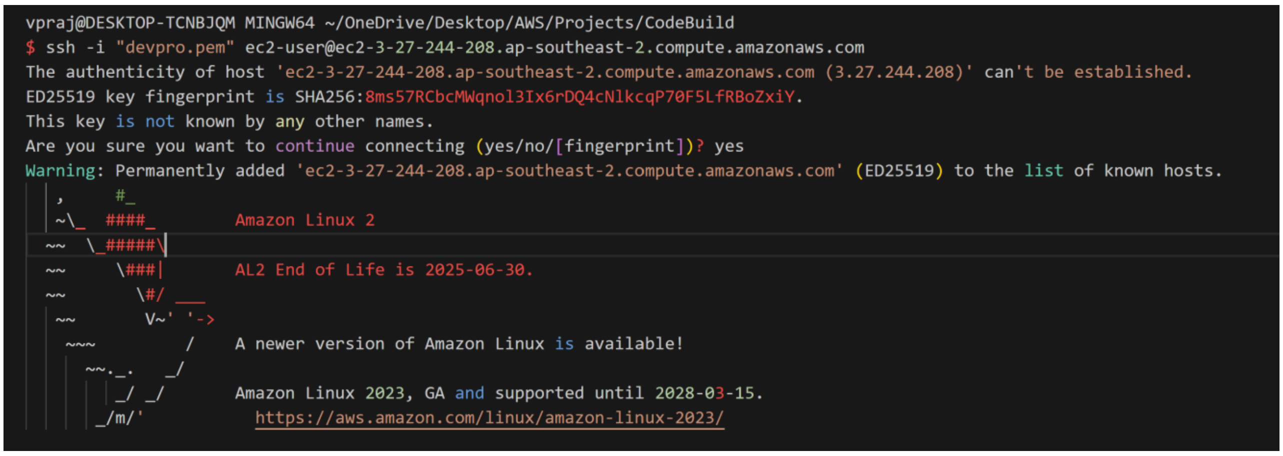
This project took me...

This project took me 2.5 hours to complete include report writing and other pre-setups.

Set up made before starting with AWS CodeArtifact

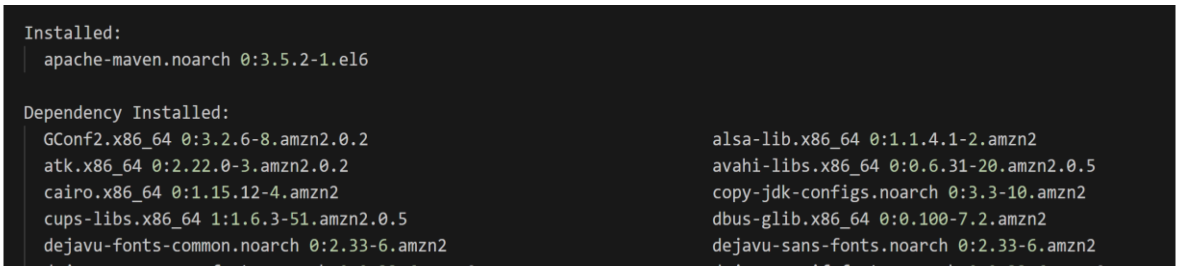
Step 1: Setting up a Web app

1. I set up my web app by launching an EC2 instance in the AWS Console, and I connected to it via SSH using Git Bash

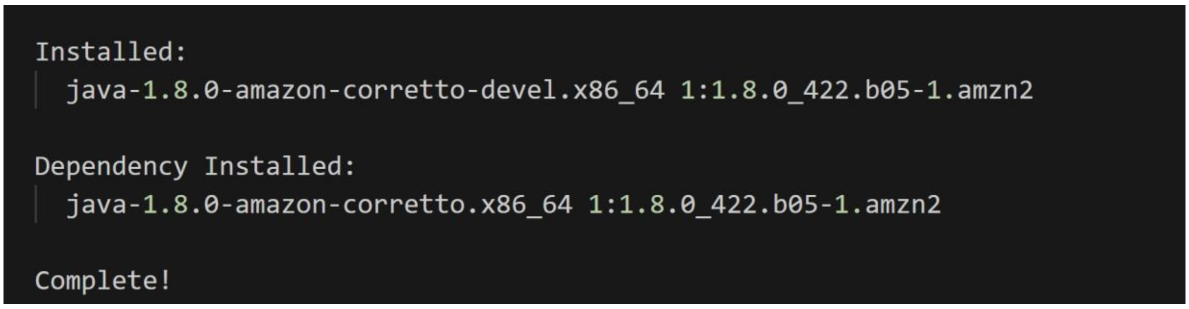


```
vpraj@DESKTOP-TCNBJQM MINGW64 ~/OneDrive/Desktop/AWS/Projects/CodeBuild
$ ssh -i "devpro.pem" ec2-user@ec2-3-27-244-208.ap-southeast-2.compute.amazonaws.com
The authenticity of host 'ec2-3-27-244-208.ap-southeast-2.compute.amazonaws.com (3.27.244.208)' can't be established.
ED25519 key fingerprint is SHA256:8ms57RCbcMWqnol3Ix6rDQ4cNlkcqP70F5LfRBoZxiY.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-3-27-244-208.ap-southeast-2.compute.amazonaws.com' (ED25519) to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2
  ~~  \_#####\
  ~~     \###|       AL2 End of Life is 2025-06-30.
  ~~       \#/ ___
   ~~       V~' '->
    ~~~         /    A newer version of Amazon Linux is available!
      ~~._.   _/
         _/ _/       Amazon Linux 2023, GA and supported until 2028-03-15.
       _/m/'           https://aws.amazon.com/linux/amazon-linux-2023/
```

2. After connecting to my EC2 instance, I set up the development environment by installing Apache Maven and Amazon Corretto 8 (Java version)



```
Installed:
  apache-maven.noarch 0:3.5.2-1.el6

Dependency Installed:
  GConf2.x86_64 0:3.2.6-8.amzn2.0.2                                  alsa-lib.x86_64 0:1.1.4.1-2.amzn2
  atk.x86_64 0:2.22.0-3.amzn2.0.2                                    avahi-libs.x86_64 0:0.6.31-20.amzn2.0.5
  cairo.x86_64 0:1.15.12-4.amzn2                                     copy-jdk-configs.noarch 0:3.3-10.amzn2
  cups-libs.x86_64 1:1.6.3-51.amzn2.0.5                              dbus-glib.x86_64 0:0.100-7.2.amzn2
  dejavu-fonts-common.noarch 0:2.33-6.amzn2                          dejavu-sans-fonts.noarch 0:2.33-6.amzn2
```

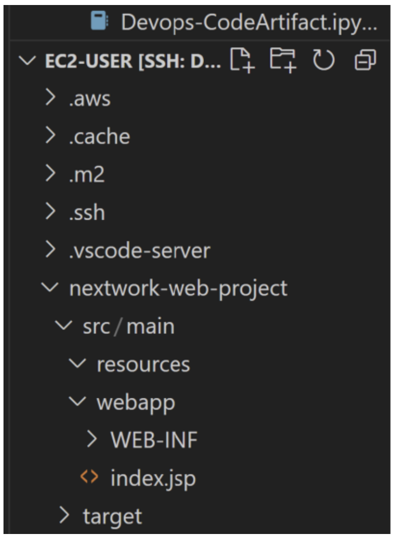


```
Installed:
  java-1.8.0-amazon-corretto-devel.x86_64 1:1.8.0_422.b05-1.amzn2

Dependency Installed:
  java-1.8.0-amazon-corretto.x86_64 1:1.8.0_422.b05-1.amzn2

Complete!
```

3. Next, I downloaded Visual Studio Code as an alternative to Cloud9. I connected to the EC2 instance via SSH and set up the Java web application by running the command:

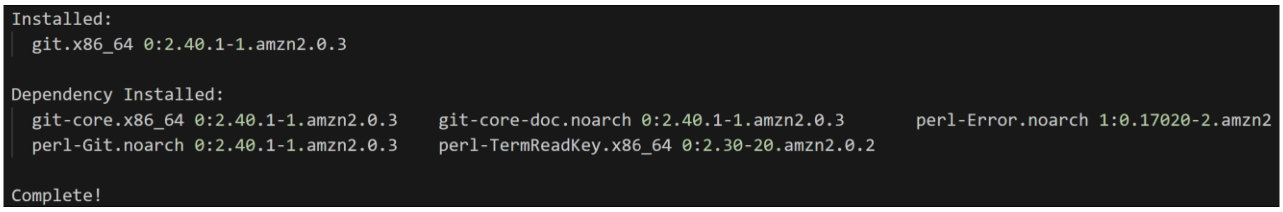
   mvn archetype:generate \
   -DgroupId=com.nextwork.app \
   -DartifactId=nextwork-web-project \
   -DarchetypeArtifactId=maven-archetype-webapp \
   -DinteractiveMode=false

4. After running the above command, a folder named 'nextwork-web-project' was created in my IDE's (VS Code) file explorer for my project.



Step 2: Creating a Git repository with AWS CodeCommit

1. I created a Git repository in AWS CodeCommit (main) and had to create a local repository in Visual Studio Code to make changes to the files and push them to the main repository. To do this, I installed Git in VSCode by running the command: **sudo yum install git**



```
Installed:
  git.x86_64 0:2.40.1-1.amzn2.0.3

Dependency Installed:
  git-core.x86_64 0:2.40.1-1.amzn2.0.3    git-core-doc.noarch 0:2.40.1-1.amzn2.0.3      perl-Error.noarch 1:0.17020-2.amzn2
  perl-Git.noarch 0:2.40.1-1.amzn2.0.3    perl-TermReadKey.x86_64 0:2.30-20.amzn2.0.2

Complete!
```

2. Next, I created a Git identity using the following commands:

   git config --global user.name "Prajit"
   git config --global user.email "v.prajit98@gmail.com"

3. To create a local repository, I created a project directory in Visual Studio Code by cloning the CodeCommit repository (main). To clone the CodeCommit repository, I used the command:
   git clone https://git-codecommit.ap-southeast-2.amazonaws.com/v1/repos/Prajit-CodeBuild-Project

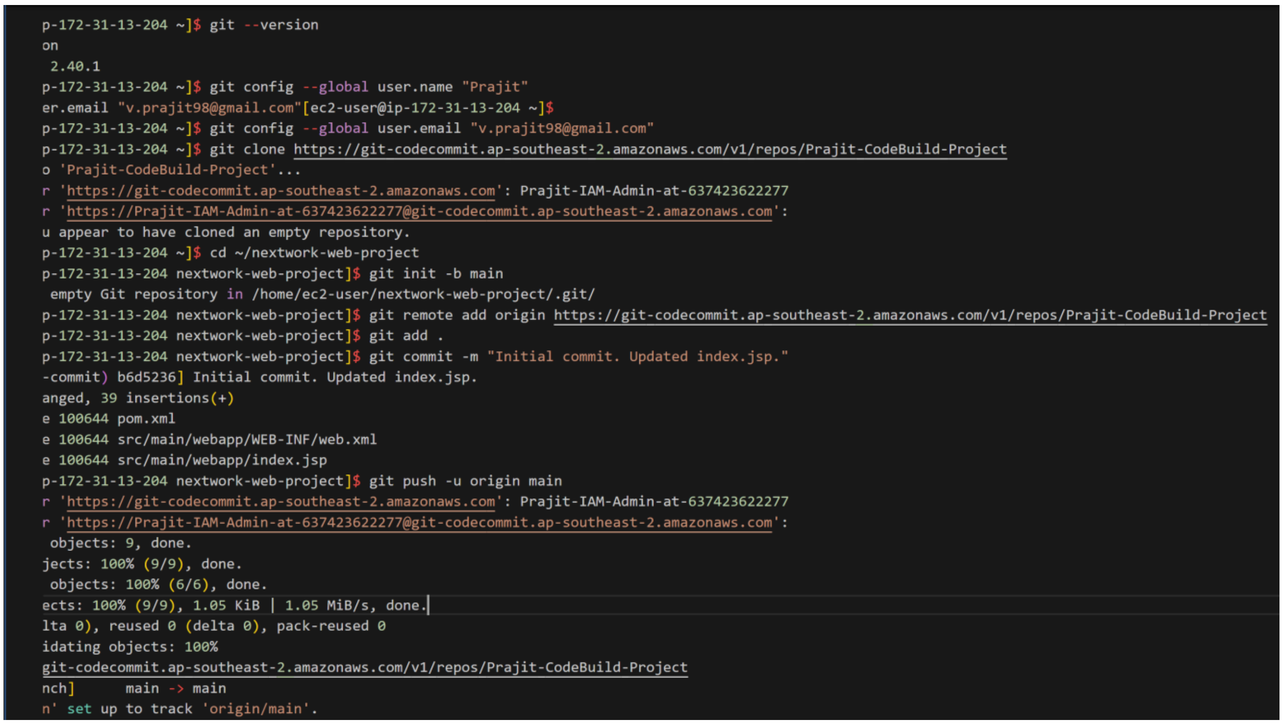
4. After running the above command, Git prompted for authentication to enter the username and password, as I was using Git over HTTPS. Therefore, I downloaded Git HTTPS credentials from IAM in the AWS Console.

5. Next, I navigated to the project directory folder to initialize my local repository. To move to the directory, I used the command:
   cd ~/nextwork-web-project, and I initialized the local repository by running the command**: git init -b main.**

6. Now, I set the remote origin to the main repository by copying the CodeCommit URL from the "Clone URL" option in the repositories section of the AWS Console. I used it in the command:
   git remote add origin https://git-codecommit.ap-southeast-2.amazonaws.com/v1/repos/Prajit-CodeBuild-Project

7. Next, I committed and pushed my code by running the following commands:
   git add .
   git commit -m "Initial commit. Updated index.jsp."
   git push -u origin main

When I use the command **git push -u origin main**, Git prompts me for authentication again, requiring me to enter the username and password that I obtained from my HTTPS Git credentials.



```
p-172-31-13-204 ~]$ git --version
on
 2.40.1
p-172-31-13-204 ~]$ git config --global user.name "Prajit"
er.email "v.prajit98@gmail.com"[ec2-user@ip-172-31-13-204 ~]$
p-172-31-13-204 ~]$ git config --global user.email "v.prajit98@gmail.com"
p-172-31-13-204 ~]$ git clone https://git-codecommit.ap-southeast-2.amazonaws.com/v1/repos/Prajit-CodeBuild-Project
o 'Prajit-CodeBuild-Project'...
r 'https://git-codecommit.ap-southeast-2.amazonaws.com': Prajit-IAM-Admin-at-637423622277
r 'https://Prajit-IAM-Admin-at-637423622277@git-codecommit.ap-southeast-2.amazonaws.com':
u appear to have cloned an empty repository.
p-172-31-13-204 ~]$ cd ~/nextwork-web-project
p-172-31-13-204 nextwork-web-project]$ git init -b main
 empty Git repository in /home/ec2-user/nextwork-web-project/.git/
p-172-31-13-204 nextwork-web-project]$ git remote add origin https://git-codecommit.ap-southeast-2.amazonaws.com/v1/repos/Prajit-CodeBuild-Project
p-172-31-13-204 nextwork-web-project]$ git add .
p-172-31-13-204 nextwork-web-project]$ git commit -m "Initial commit. Updated index.jsp."
-commit) b6d5236] Initial commit. Updated index.jsp.
anged, 39 insertions(+)
e 100644 pom.xml
e 100644 src/main/webapp/WEB-INF/web.xml
e 100644 src/main/webapp/index.jsp
p-172-31-13-204 nextwork-web-project]$ git push -u origin main
r 'https://git-codecommit.ap-southeast-2.amazonaws.com': Prajit-IAM-Admin-at-637423622277
r 'https://Prajit-IAM-Admin-at-637423622277@git-codecommit.ap-southeast-2.amazonaws.com':
 objects: 9, done.
jects: 100% (9/9), done.
 objects: 100% (6/6), done.
ects: 100% (9/9), 1.05 KiB | 1.05 MiB/s, done.
lta 0), reused 0 (delta 0), pack-reused 0
idating objects: 100%
git-codecommit.ap-southeast-2.amazonaws.com/v1/repos/Prajit-CodeBuild-Project
nch]      main -> main
n' set up to track 'origin/main'.
```

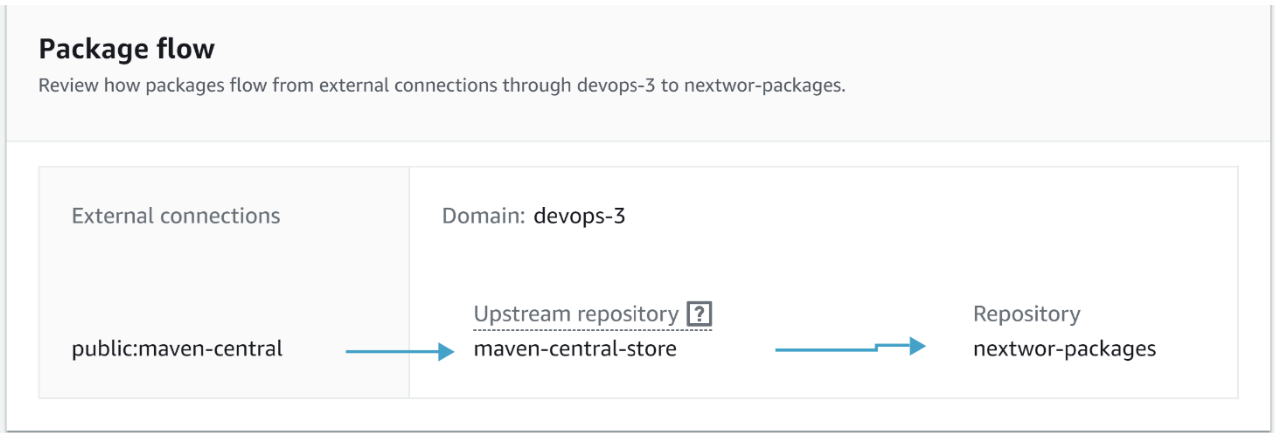
Starting with AWS CodeArtifact

My project has three artifact repositories

- There are three repositories connected to each other. The first is the local repository, where Maven primarily checks for software packages or dependencies for my web application.

- The second repository is my public upstream repository, which Maven checks next if a package is not available in the local repository.

- The third repository is the Maven Central repository, which is a public repository with the largest collection of packages for Java applications. However, Maven will only visit this repository if the first two repositories do not have the packages or dependencies it is looking for, due to high traffic.


Package flow
Review how packages flow from external connections through devops-3 to nextwor-packages.

External connections
public:maven-central

Domain: devops-3
Upstream repository
maven-central-store
Repository
nextwor-packages

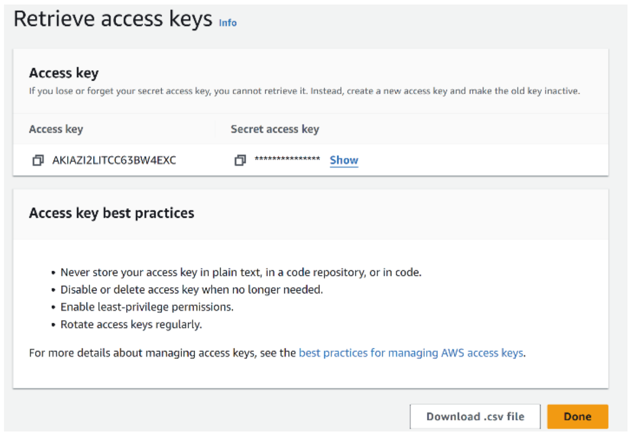
Connecting my project with CodeArtifact

I connected my web app project (via my VS Code IDE) to CodeArtifact, so that CodeArtifact knows which project it will store the packages or dependencies for.

To connect my web app project to CodeArtifact via Visual Studio Code, I had to use a few commands. The first command is:

```
<servers>
  <server>
    <id>devops-3-nextwor-packages</id>
    <username>aws</username>
    <password>${env.CODEARTIFACT_AUTH_TOKEN}</password>
  </server>
</servers>
```

This command stores the access details for the repositories that my web app project is connecting to, so I needed to configure AWS in Visual Studio Code. I ran the command **aws configure**, which authenticates the AWS Access Key ID, Secret Access Key, default region name, and default output format. In this case, I had to create an AWS access key, which provides the Access Key ID and Secret Access Key.



Retrieve access keys Info

Access key

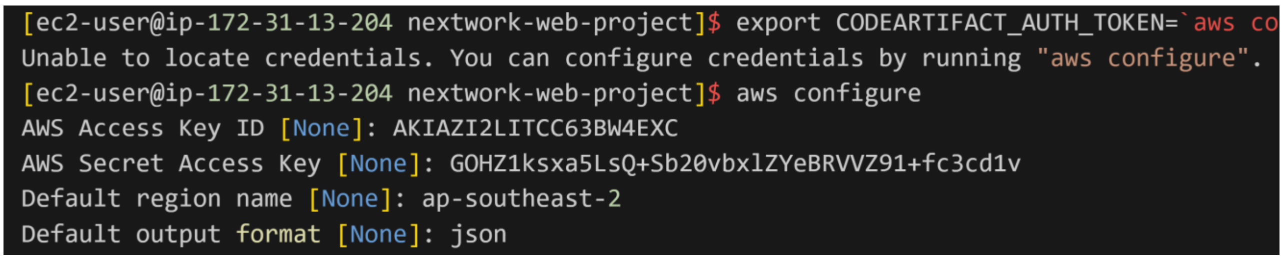
If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

Access key	Secret access key
AKIAZI2LITCC63BW4EXC	*************** Show

Access key best practices

- Never store your access key in plain text, in a code repository, or in code.
- Disable or delete access key when no longer needed.
- Enable least-privilege permissions.
- Rotate access keys regularly.

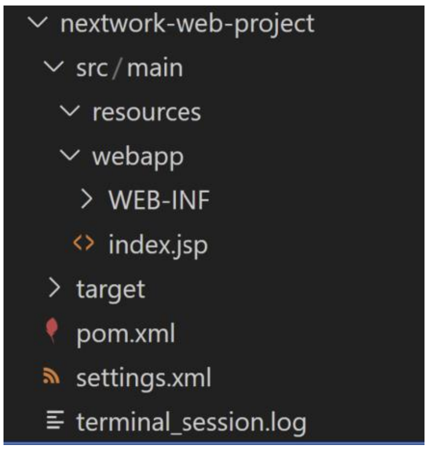
For more details about managing access keys, see the best practices for managing AWS access keys.



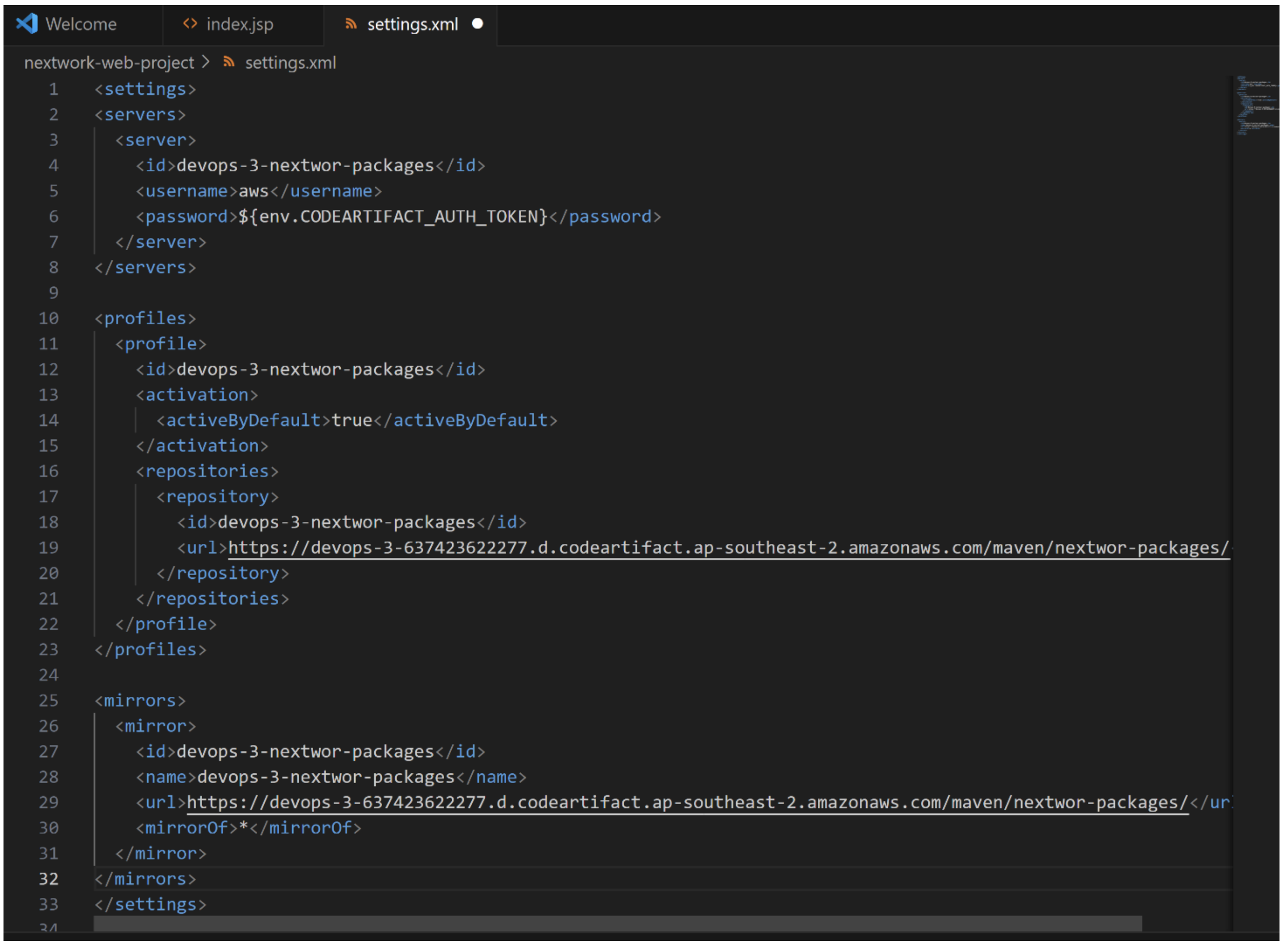
```
[ec2-user@ip-172-31-13-204 nextwork-web-project]$ export CODEARTIFACT_AUTH_TOKEN=`aws co
Unable to locate credentials. You can configure credentials by running "aws configure".
[ec2-user@ip-172-31-13-204 nextwork-web-project]$ aws configure
AWS Access Key ID [None]: AKIAZI2LITCC63BW4EXC
AWS Secret Access Key [None]: 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
Default region name [None]: ap-southeast-2
Default output format [None]: json
```

I created a new file, settings.xml, in my web app. The settings.xml file provides Maven with instructions on where to find the dependencies it needs to fetch and how Maven will gain access to the repositories storing these dependencies.

Creating the settings.xml file involves writing three snippets of code in it, as shown in the picture below. To create the settings.xml file, I used the command **cd ~/nextwork-web-project** to navigate to that directory, and then ran **echo $'<settings>\n</settings>' > settings.xml** to create the file. Once the file created, it will start appearing in the “nextwork-web-project” folder in the vscode explorer.



```
nextwork-web-project
  src / main
    resources
    webapp
      WEB-INF
      index.jsp
  target
  pom.xml
  settings.xml
  terminal_session.log
```



```xml
<settings>
<servers>
  <server>
    <id>devops-3-nextwor-packages</id>
    <username>aws</username>
    <password>${env.CODEARTIFACT_AUTH_TOKEN}</password>
  </server>
</servers>

<profiles>
  <profile>
    <id>devops-3-nextwor-packages</id>
    <activation>
      <activeByDefault>true</activeByDefault>
    </activation>
    <repositories>
      <repository>
        <id>devops-3-nextwor-packages</id>
        <url>https://devops-3-637423622277.d.codeartifact.ap-southeast-2.amazonaws.com/maven/nextwor-packages/
      </repository>
    </repositories>
  </profile>
</profiles>

<mirrors>
  <mirror>
    <id>devops-3-nextwor-packages</id>
    <name>devops-3-nextwor-packages</name>
    <url>https://devops-3-637423622277.d.codeartifact.ap-southeast-2.amazonaws.com/maven/nextwor-packages/</ur
    <mirrorOf>*</mirrorOf>
  </mirror>
</mirrors>
</settings>
```

These are the three commands written inside the settings.xml file.

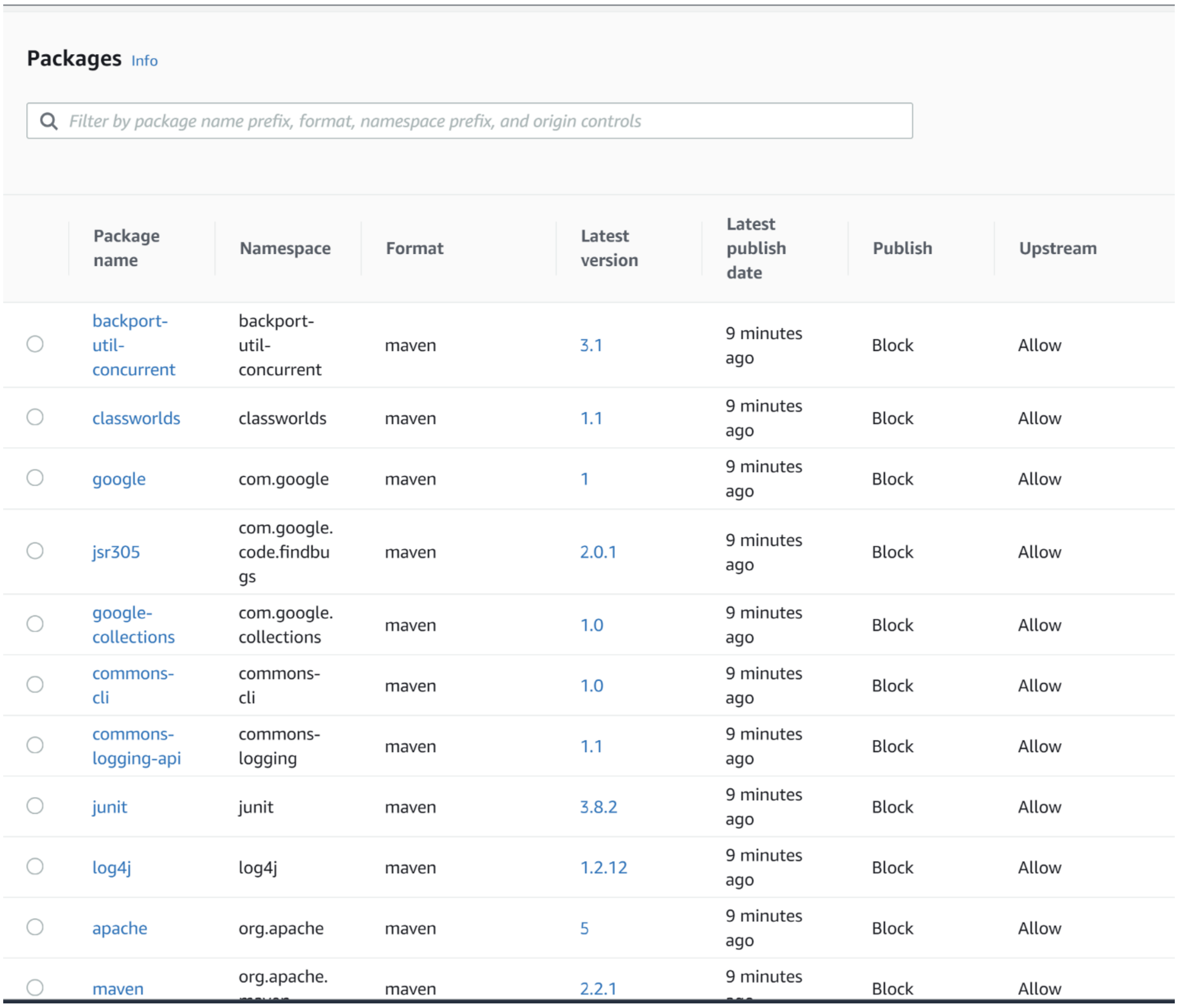
The code I pasted into the settings.xml file was provided by CodeArtifact, so I did not have to write it from scratch. The snippets of code store authentication tokens for CodeArtifact and define when Maven will visit each repository, as well as where Maven can find backup local repositories.

Testing the connection

To test the connection between Visual Studio Code and CodeArtifact, I compiled my web app. Compiling means translating my web application's code into machine code that servers can understand and run.

After compiling, I checked my local repository and saw that it now contained numerous packages. This means Maven has successfully retrieved packages from the upstream repository and the Maven Central repository and has installed them locally.

For compiling, I used a command: **mvn -s settings.xml compile**



Packages Info

Filter by package name prefix, format, namespace prefix, and origin controls

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	9 minutes ago	Block	Allow
google	com.google	maven	1	9 minutes ago	Block	Allow
jsr305	com.google.code.findbugs	maven	2.0.1	9 minutes ago	Block	Allow
google-collections	com.google.collections	maven	1.0	9 minutes ago	Block	Allow
commons-cli	commons-cli	maven	1.0	9 minutes ago	Block	Allow
commons-logging-api	commons-logging	maven	1.1	9 minutes ago	Block	Allow
junit	junit	maven	3.8.2	9 minutes ago	Block	Allow
log4j	log4j	maven	1.2.12	9 minutes ago	Block	Allow
apache	org.apache	maven	5	9 minutes ago	Block	Allow
maven	org.apache.	maven	2.2.1	9 minutes	Block	Allow

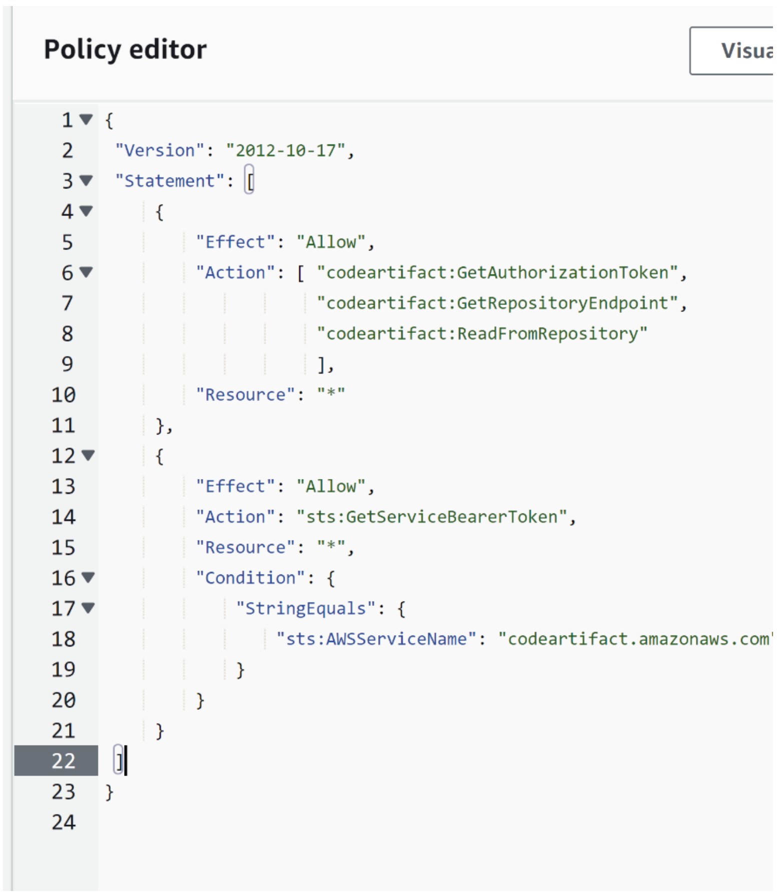
Create IAM policies

The importance of IAM policies

I also created an IAM policy because other services in my CI/CD pipeline, such as CodeBuild and CodePipeline, will need access to the packages and dependencies stored in CodeArtifact. By default, these services do not have access, so they need to be granted permissions through an IAM policy

I defined my IAM policy using JSON

I defined my IAM policy using JSON. This policy enables the policy holder to obtain authorization tokens (i.e., access to CodeArtifact) and fetch the packages stored in CodeArtifact's repositories.



```
{
 "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
            }
        }
    }
 ]
}
```

Everyone should be in ajob they love.

Check out nextwork.org formore projects