**LOGIN TO AWS ACCOUNT**

**CODE COMMIT**

**1.Create codecommit repo** - vprofile-maven-repo > Public upstream: maven-central-store > This account > Domain: visualpath. We see 2 repositories one for maven-central-store used to store dependencies. Click on maven-central-store > view connection instruction > mvn > It will show you steps 1.Export CodeArtifact authorization token 2.Changes in settings.xml

**2.Create iam user with codecommit policy** – vprofile-arti-admin > AWSCodeArtifactAdminAccess > Copy Access & Secret Access key. On Gitbash aws configure.

Run command export CODEARTIFACT\_AUTH\_TOKEN=`aws codeartifact get-authentication-token --domain visualpath –domain-owner <account\_id> --query authorizationToken –-output text`. Run echo $CODEARTIFACT\_AUTH\_TOKEN This will generate token which is used to interact CodeArtifact with CodeBuild

Changes in settings.xml see CHANGEHERE.Settings.xml used to configure Maven settings, particularly for repository management and authentication.Total 3 changes.

Changes in pom.xml see CHANGEHERE.pom.xml is Maven POM (Project Object Model) configuration file which contains project's metadata, dependencies, build settings, and repository information for a Java web application.It contains artifactId(artifactName), packaging(war), version, properties, dependencies, build, finalName, plugin, repositories.Added 2 lines of timestamp in properties.Total 2 changes.

Login sonarcloud with github credentials. Generate token: My account > Security > Token name: vprofile-sonar-cloud > copy token. Click plus + symbol > Analyze new project > Create a project manually > Select Organization > Project key: vprofile-repo > set up

We need : 1.Token 2.Project key/name: vprofile-repo 3.URL sonarcloud: <https://sonarcloud.io> 4.Organization name:xyz

AWS > System manager > Parameter store > Create parameter

1.Name: Organization > Value: < organization\_name\_sonarCloud> xyz

2. Name: HOST > Value: <URL\_sonarcloud> https://sonarcloud.io

3. Name: Project > Value: <Project name> vprofile-repo

4.Name: sonartoken > Type: SecureString Value: <Paste token here>

5.Name: codeartifact-token > Type: SecureString Value: <paste token by gitbash echo $CODEARTIFACT\_AUTH\_TOKEN>

AWS > codebuild > Create build project > vprofile-build > Repository: select > Branch: select > Operating system: ubuntu > Runtime: standard > image: 3.0 > Role name: give role name > Insert build command > Paste buildspec.yml with changes below

Buildspec.yml contains : version, env: <environment variables from parameter store>, phases >> **install:** runtime-versions(java: openjdk8), commands(copy settings.xml to /root/.m2/settings.xml) ; **pre-build :** update, install jq checkstyle, wget download maven, extract maven, softlink to maven directory, wget download sonarscanner, unzip sonarscanner, export path for sonar export PATH=$PATH:/sonar-scanner/bin/; **build :** commands: - mvn test, – mvn checkstyle:checkstyle, mvn sonar:sonar with all parameter store values, - sleep 5, - command to create result.json, - cat result.json, - with if condition check quality gate pass or fail

Changes in buildspec.yml : CHANGEHERE

> Logs > Cloudwatch > Group name: vprofile-nvir-buildlogs > stream name: sonarbuildjob > create build project

Build project > vprofile-build > Edit > Environment > copy rolename > go to IAM roles > paste role find it > attach policies > Create policy > service system manager, Actions: DescribeParameters, DescribeDocumentParameters, GetParameters, GetParametersHistory, GetParameter, GetParametersByPath > Policy name: vprofile-sonarparameteraccess > attach policy to role. > start build

In sonarcloud we see result of bugs & everything . Set quality gates on it

Create project with build\_buildspec.yml file content which create artifact Changes in build\_buildspec.yml CODEARTIFACT\_AUTH\_TOKEN: sonartoken. Added extra step artifacts in build\_buildspec.yml

AWS > SNS > Create topic > vprofile-pipeline-notification > Create topic > Create subscription > email > emailid > goto email confirm subscription >

AWS > Codebuild > pipelines > Create pipeline > vprofile-ci-pipeline > Codecommit > Repo > branch > cloudwatch > codebuild > select project 2nd > Skip deploy stage > create pipeline > stop execution > edit > Add stage > Test > Add action group > Sonar-code-analysis > Codenbuild > 1st project > SourceArtifact > Done

Create s3 bucket with pipeline-artifact folder in it

Add stage > Deploy > Deploy-to-s3 > Amazon s3 > BuildArtifact > bucket name > Directory name > check Extract artifact > Done > Save

Git pull repository in local make changes in README file commit & push ci-aws we see pipeline

3.Generate ssh keys locally

4.Exchange keys with IAM user

5.Put source code from github repo to cc repository and push

CODE ARTIFACT

Create an IAM user with code artifact access

Install AWS CLI, configure

Export auth token

Update settings.xml file in source code top level directory with below details

Update pom.xml file with repo details

SONAR CLOUD

Create sonar cloud account

Generate token

Create SSM parameters with sonar details

Create Build project

Update codebuild role to access SSMparameterstore

CREATE NOTIFICATIONS FOR SNS OR SLACK

BUILD PROJECT

Update pom.xml with artifact version with timestamp

Create variables in SSM => parametersore

Create build project

Update codebuild role to access SSMparameterstore

CREATE PIPELINE

Codecommit

Testcode

Build

Deploy to s3 bucket

TEST PIPELINE

12 - Continuous Delivery on AWS Cloud [Java Application]

Choose region with Code Artifact service

Create iam with policy codecommit with All codecommit actions & resource Add arn region & repo name.