**Project Documentary: Setting Up a DevOps Pipeline on AWS**

**Objective**

To set up a **fully automated CI/CD pipeline** using Jenkins, SonarQube, Nexus, and Tomcat on an AWS EC2 instance. This pipeline will:

1. **Retrieve source code** from GitHub.
2. **Build the project** using Maven.
3. **Perform static code analysis** with SonarQube.
4. **Store artifacts** in Nexus Repository.
5. **Deploy the application** to an Apache Tomcat server.

**1. AWS Infrastructure Setup**

**1.1 Launch EC2 Instance**

* **AMI**: Choose Amazon Linux 2 / Ubuntu / CentOS 9
* **Instance Type**: t2.medium (recommended for Jenkins & SonarQube)
* **Storage**: Minimum 30GB
* **Security Groups**:
  + Allow required ports:
    - SSH (22)
    - Jenkins (8080)
    - SonarQube (9000)
    - Nexus (8081)
    - Tomcat (8080)
    - Any additional services as needed

**1.2 Install Required Dependencies**

sudo yum update -y

sudo yum install -y git wget unzip curl nano java-11-openjdk-devel

**2. Install and Configure Jenkins**

**2.1 Install Jenkins**

wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo

rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key

yum install -y jenkins

systemctl enable --now jenkins

**2.2 Retrieve Admin Password and Setup Jenkins**

cat /var/lib/jenkins/secrets/initialAdminPassword

* Access Jenkins: http://<AWS\_PUBLIC\_IP>:8080
* Install recommended plugins.
* Create an admin user and configure global settings.

**2.3 Install Required Plugins**

* Git Plugin
* Pipeline Plugin
* Maven Integration
* SonarQube Scanner
* Nexus Artifact Uploader
* SSH Pipeline Steps (for deployment to Tomcat)

**3. Install and Configure SonarQube**

**3.1 Install PostgreSQL for SonarQube**

sudo yum install -y postgresql-server postgresql-contrib

sudo postgresql-setup initdb

sudo systemctl enable --now postgresql

**3.2 Create SonarQube Database and User**

sudo -i -u postgres psql

CREATE DATABASE sonarqube;

CREATE USER sonar WITH ENCRYPTED PASSWORD 'sonar@123';

GRANT ALL PRIVILEGES ON DATABASE sonarqube TO sonar;

\q

**3.3 Install SonarQube**

wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-9.9.1.zip

unzip sonarqube-9.9.1.zip -d /opt/

sudo useradd sonar

sudo chown -R sonar:sonar /opt/sonarqube

**3.4 Start SonarQube**

su - sonar

cd /opt/sonarqube/bin/linux-x86-64/

./sonar.sh start

* Access SonarQube at http://<AWS\_PUBLIC\_IP>:9000
* Login using admin / admin and configure settings.

**4. Install and Configure Nexus Repository Manager**

**4.1 Install Nexus**

wget https://download.sonatype.com/nexus/3/latest-unix.tar.gz

tar -xvzf latest-unix.tar.gz -C /opt/

**4.2 Start Nexus Service**

cd /opt/nexus/bin

./nexus start

* Access Nexus: http://<AWS\_PUBLIC\_IP>:8081
* Default Login: admin / admin123

**4.3 Create a Maven Repository in Nexus**

1. Go to **Repositories > Create repository**
2. Select **hosted** for a private repository.
3. Use format **Maven 2** and configure settings.

**5. Install and Configure Apache Tomcat**

**5.1 Install Tomcat**

wget https://downloads.apache.org/tomcat/tomcat-9/v9.0.75/bin/apache-tomcat-9.0.75.tar.gz

tar -xvzf apache-tomcat-9.0.75.tar.gz -C /opt/

**5.2 Configure Tomcat Users for Deployment**

Edit the tomcat-users.xml file:

xml

<role rolename="manager-gui"/>

<role rolename="manager-script"/>

<user username="admin" password="admin123" roles="manager-gui,manager-script"/>

**5.3 Start Tomcat**

cd /opt/apache-tomcat-9.0.75/bin

./startup.sh

* Access Tomcat Manager: http://<AWS\_PUBLIC\_IP>:8080

**6. Creating a Jenkins CI/CD Pipeline**

**6.1 Create a New Pipeline Job in Jenkins**

1. **Go to Jenkins Dashboard → New Item → Pipeline.**
2. **In the pipeline script, add the following:**

pipeline {

agent any

environment {

GIT\_URL = 'https://github.com/suffixscope/maven-web-app.git'

// SonarQube

SONAR\_URL = 'http://43.204.238.132:9000'

// Nexus

NEXUS\_URL = 'http://43.204.30.233:8081'

NEXUS\_REPO = 'maven-releases'

GROUP\_ID = 'com.example'

ARTIFACT\_ID = 'maven-web-app'

VERSION = '1.0'

PACKAGING = 'war'

// Tomcat

TOMCAT\_URL = 'http://3.108.59.248:8080'

WAR\_NAME = 'maven-web-app.war'

}

tools {

maven 'Maven-3.8.7' // Using Jenkins Maven tool

}

stages {

stage('Checkout Code') {

steps {

git branch: 'master', url: "${GIT\_URL}"

}

}

stage('Build with Maven') {

steps {

sh 'mvn clean package'

}

}

stage('SonarQube Analysis') {

steps {

withSonarQubeEnv('SonarQube') {

withCredentials([string(credentialsId: 'SONAR\_TOKEN', variable: 'SONAR\_AUTH\_TOKEN')]) {

sh 'mvn sonar:sonar -Dsonar.host.url=$SONAR\_URL -Dsonar.login=$SONAR\_AUTH\_TOKEN'

}

}

}

}

stage('Quality Gate') {

steps {

timeout(time: 5, unit: 'MINUTES') {

script {

def qg = waitForQualityGate()

if (qg.status != 'OK') {

error "Pipeline failed due to SonarQube Quality Gate failure!"

}

}

}

}

}

stage('Upload to Nexus') {

steps {

withCredentials([usernamePassword(credentialsId: 'nexus-admin', usernameVariable: 'NEXUS\_USER', passwordVariable: 'NEXUS\_PASS')]) {

sh """

mvn deploy:deploy-file \

-DgroupId=${GROUP\_ID} \

-DartifactId=${ARTIFACT\_ID} \

-Dversion=${VERSION} \

-Dpackaging=${PACKAGING} \

-Dfile=target/${ARTIFACT\_ID}.war \

-DrepositoryId=nexus \

-Durl=${NEXUS\_URL}/repository/${NEXUS\_REPO}/ \

-DgeneratePom=true \

-Dauth.basic=\$NEXUS\_USER:\$NEXUS\_PASS

"""

}

}

}

stage('Deploy to Tomcat') {

steps {

withCredentials([usernamePassword(credentialsId: 'TOMCAT\_CREDENTIALS', usernameVariable: 'TOMCAT\_USER', passwordVariable: 'TOMCAT\_PASS')]) {

sh """

curl -v -u $TOMCAT\_USER:$TOMCAT\_PASS \

--upload-file target/${WAR\_NAME} \

"${TOMCAT\_URL}/manager/text/deploy?path=/maven-web-app&update=true"

"""

}

}

}

}

}

**7. Final Testing**

* **Trigger the Jenkins pipeline**
* Verify the application is deployed successfully in Tomcat:  
  http://<AWS\_PUBLIC\_IP>:8080/my-app/
* Monitor logs for any issues using:

tail -f /var/log/jenkins/jenkins.log

tail -f /opt/apache-tomcat-9.0.75/logs/catalina.out

**8. Monitoring & Maintenance**

* **Log Monitoring:**
  + Use tail -f to monitor Jenkins, SonarQube, Nexus, and Tomcat logs.
* **Regular Updates:**
  + Keep all packages updated using yum update or apt update.
* **Automated Backups:**
  + Set up backups for Jenkins, Nexus, and PostgreSQL databases.