pipeline {

agent any

triggers {

githubPush()

}

environment {

SONARQUBE\_ENV = 'SonarQube'

BRANCH\_NAME = 'main'

}

tools {

jdk 'jdk17'

maven 'Maven'

}

stages {

stage('Clean Workspace') {

steps {

echo '🧹 Cleaning Jenkins workspace...'

deleteDir() // Built-in Jenkins step to clean the workspace

}

}

stage('Checkout Code') {

steps {

git branch: "${BRANCH\_NAME}", url: 'https://github.com/KaminiNegi/Hotel-Management-App.git'

}

}

stage('Build') {

steps {

echo '🔨 Building project...'

sh 'java -version'

sh 'mvn clean install -DskipITs'

}

}

stage('Run Unit Tests') {

steps {

echo '🧪 Running unit tests...'

sh 'mvn test'

}

post {

always {

echo '✅ Unit tests completed'

junit '\*\*/target/surefire-reports/\*.xml'

recordCoverage(

tools: [[parser: 'JACOCO']],

id: 'jacoco',

name: 'JaCoCo Coverage',

sourceCodeRetention: 'EVERY\_BUILD'

)

}

}

}

stage('Trigger Chrome Permission') {

steps {

echo '⚙️ Triggering Chrome to grant macOS permissions...'

sh 'open -a "Google Chrome"'

}

}

stage('Run Selenium Tests') {

steps {

echo '🌐 Running Selenium UI tests (headless)...'

// Adjust this command based on your test suite

sh 'mvn test -Dtest=\*\*/\*TestHotelApp.java -Dheadless=true'

}

post {

always {

echo '🧪 Selenium tests completed'

junit '\*\*/target/surefire-reports/\*.xml'

}

}

}

stage('SonarQube Analysis') {

steps {

withSonarQubeEnv("${SONARQUBE\_ENV}") {

sh '''

mvn clean verify sonar:sonar \

-DskipITs=true \

-Dsonar.qualitygate.wait=false \

-Dsonar.junit.reportPaths=target/surefire-reports \

-Dsonar.coverage.jacoco.xmlReportPaths=target/jacoco-report/jacoco.xml \

-Dsonar.sources=src/main/java \

-Dsonar.tests=src/test/java \

-Dsonar.java.binaries=target/classes \

-Dsonar.inclusions=\*\*/auth/AuthenticationService.java,\*\*/auth/AuthenticationController.java,\*\*/booking/RoomService.java,\*\*/booking/RoomController.java,\*\*/booking/RoomRepository.java,\*\*/config/JwtAuthenticationFilter.java,\*\*/config/JwtService.java,\*\*/config/LogoutService.java

'''

}

}

}

stage("Quality Gate") {

steps {

timeout(time: 10, unit: 'MINUTES') {

waitForQualityGate abortPipeline: true

}

}

}

stage('Package') {

steps {

echo '📦 Packaging application...'

sh 'mvn package -DskipTests'

}

}

}

post {

success {

echo '✅ Build successful!'

}

failure {

echo '❌ Build failed.'

}

}

}