**Jenkins CICI Pipe Line Code for Dictionary App**Step 1) First crate ec2 instance with Ubuntu os and select t2 medium with 16 gb ram and login through moboxterm to that ect and install Jenkins first using below command  
  
#!/bin/bash

sudo apt-get update

sudo apt-get install openjdk-21-jdk -y

#now install jenkins

curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee /usr/share/keyrings/jenkins-keyring.asc > /dev/null

echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] https://pkg.jenkins.io/debian-stable binary/ | sudo tee /etc/apt/sources.list.d/jenkins.list > /dev/null

sudo apt-get update

sudo apt-get install jenkins -y

sudo systemctl enable jenkins

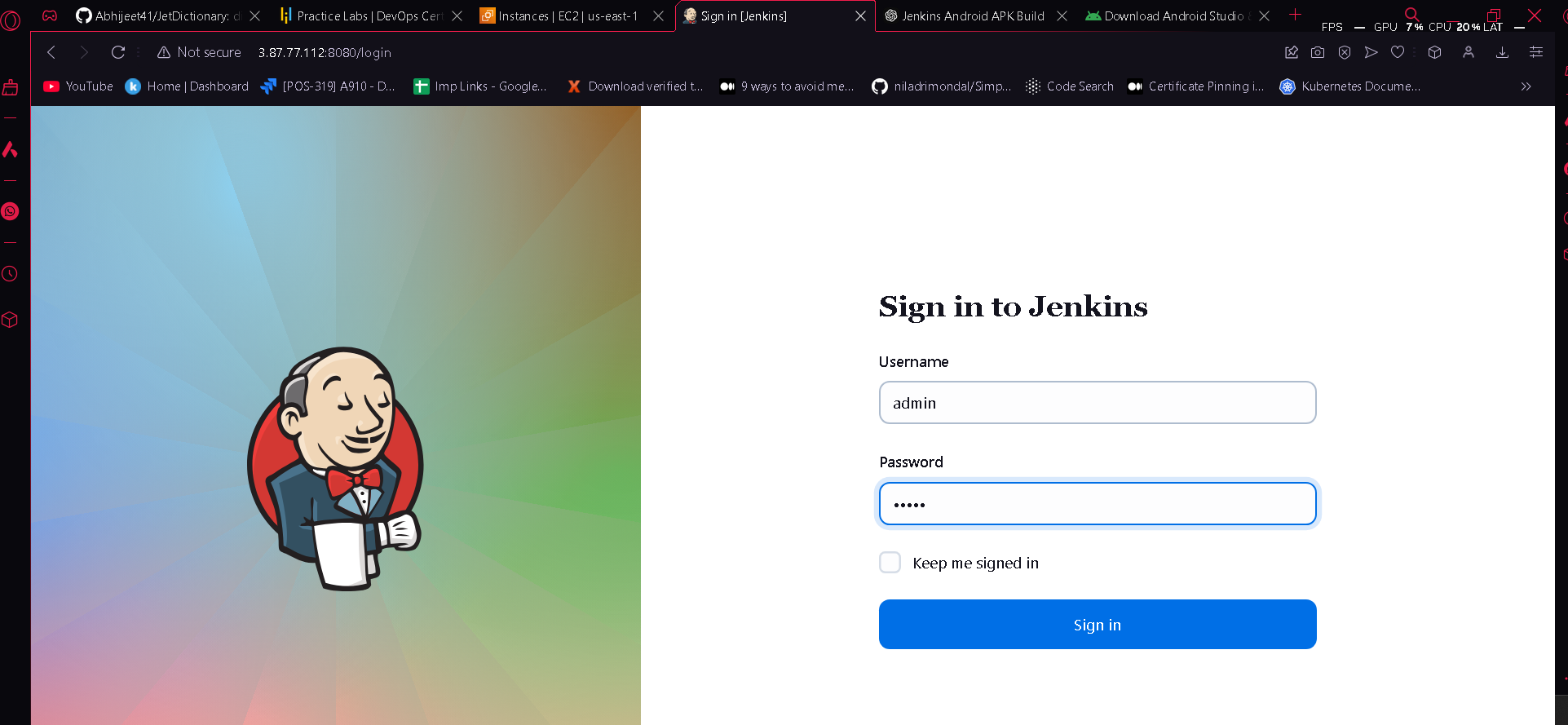
sudo systemctl start jenkins

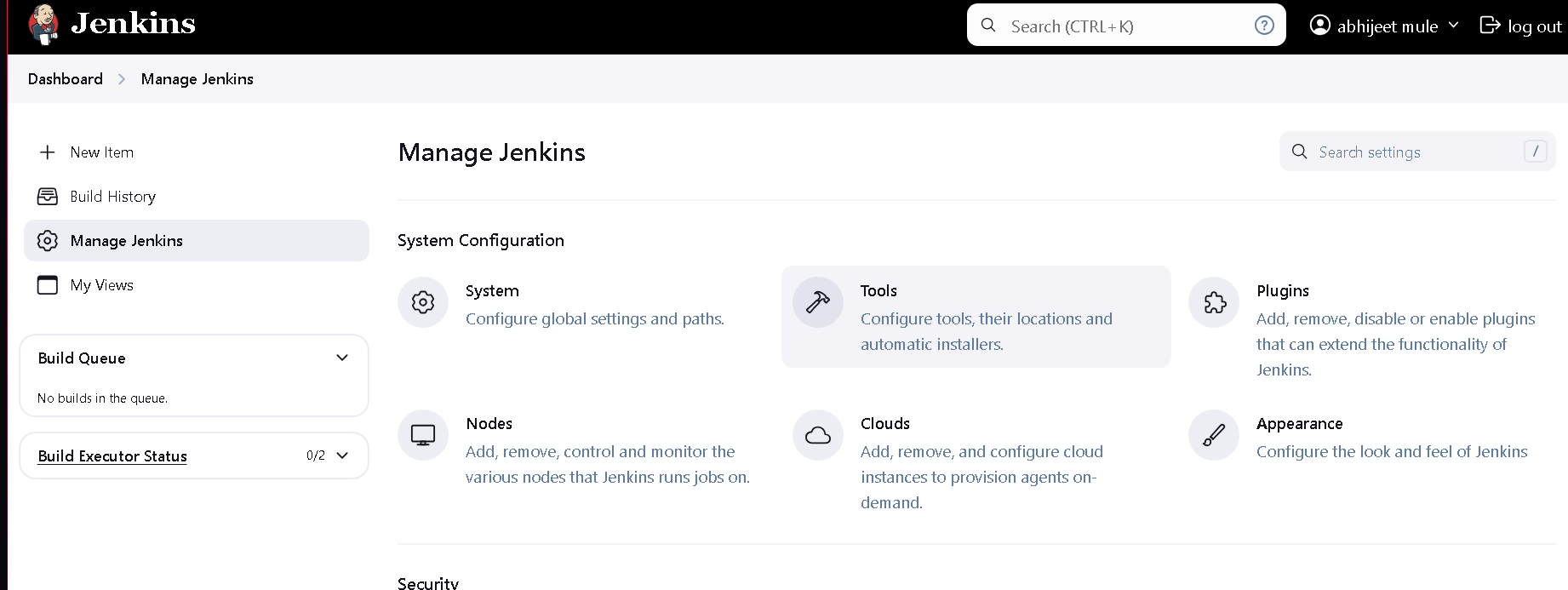
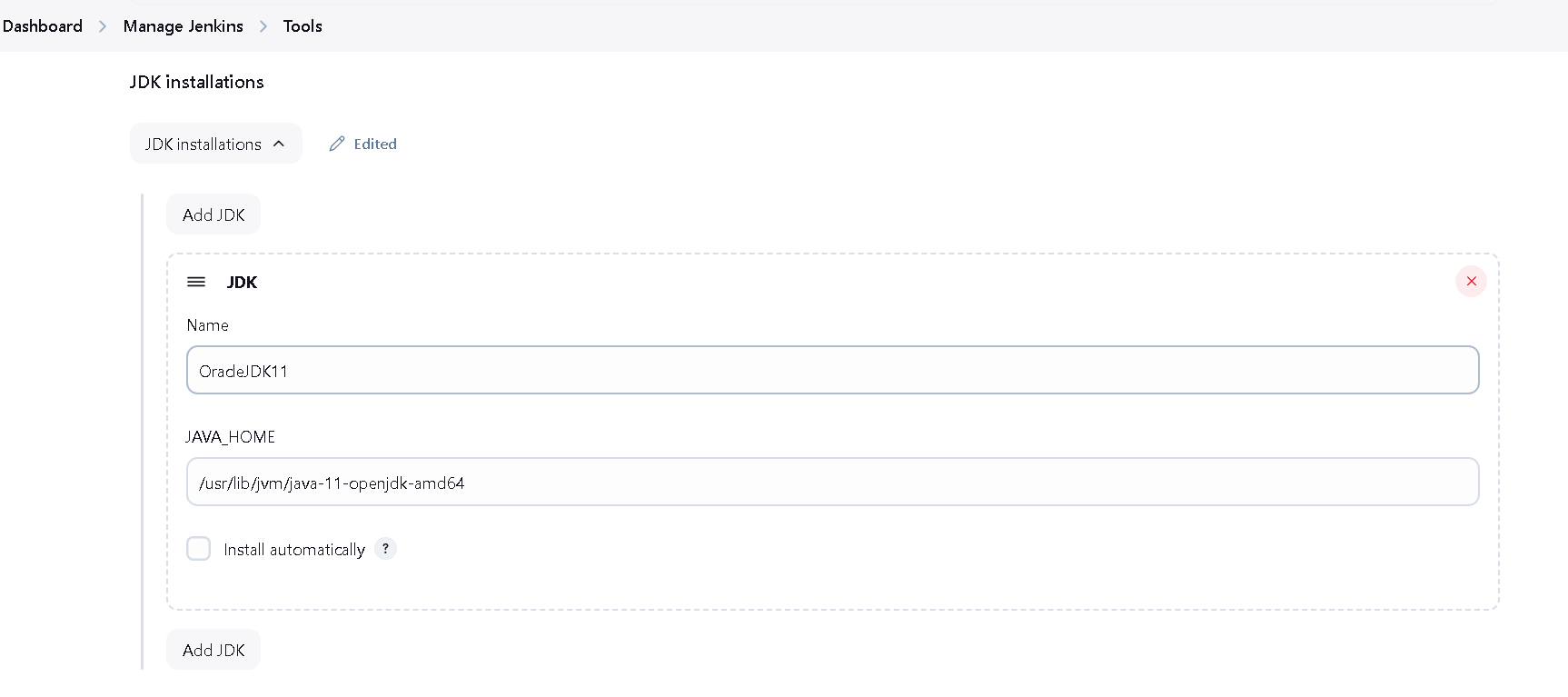
sudo systemctl status jenkins

#read jeknins password

sudo cat /var/lib/jenkins/secrets/initialAdminPassword

copy public ip of ec2 and paste on browser with port 8080



Now got to manage Jenkins and then select tools option and add JDK path  
  
  


Now go to **Manage Jenkins** and Select **System** option in **Global properties** check **Environment variables** and add path that we are gonna install in Ubuntu ec2 instance  
/home/ubuntu/android-sdk  


Now use below command to install Command Line tools in Ubuntu Ec2

mkdir -p ~/android-sdk/cmdline-tools

cd ~/android-sdk/cmdline-tools

wget https://dl.google.com/android/repository/commandlinetools-linux-xxxx\_latest.zip # replace 'xxxx' with latest version number

unzip commandlinetools-linux-\*.zip

mv cmdline-tools latest  
  
**Configure ANDROID\_HOME and update PATH**: Add the following lines to ~/.bashrc to set the environment variables:  
  
export ANDROID\_HOME=~/android-sdk

export PATH=$ANDROID\_HOME/cmdline-tools/latest/bin:$ANDROID\_HOME/platform-tools:$PATH  
  
Apply the changes:  
source ~/.bashrc

We used below pipeline script for Dictionary app

pipeline {

agent any

tools {

jdk 'OracleJDK11'

}

stages {

stage('Check ANDROID\_HOME') {

steps {

sh 'echo "ANDROID\_HOME is set to: $ANDROID\_HOME"'

sh 'ls $ANDROID\_HOME'

}

}

stage('Set Permissions') {

steps {

sh 'chmod +x ./gradlew'

}

}

stage('Set up SDK path') {

steps {

sh 'echo "sdk.dir=$ANDROID\_HOME" > local.properties'

}

}

stage('Clean Gradle') {

steps {

sh './gradlew clean --no-daemon'

}

}

stage('Build') {

steps {

sh './gradlew assembleRelease'

}

}

}

post {

success {

archiveArtifacts artifacts: 'app/build/outputs/apk/release/\*.apk', allowEmptyArchive: true

echo 'Build completed successfully!'

}

failure {

echo 'Build failed. Please check the logs for errors.'

}

}

After running pipeline now getting below error that indicates permission denied  
  
echo ANDROID\_HOME is set to: /home/ubuntu/android-sdk ANDROID\_HOME is set to: /home/ubuntu/android-sdk [Pipeline] sh + ls /home/ubuntu/android-sdk ls: cannot access '/home/ubuntu/android-sdk': Permission denied  
  
The "Permission denied" error indicates that the Jenkins process doesn’t have permission to access the Android SDK directory on the EC2 instance. This is common if Jenkins is running under a different user (e.g., jenkins), while the SDK directory is owned by ubuntu. Here’s how to resolve it:

**Log in to Your EC2 Instance**:  
  
ssh -i /path/to/your-key.pem ubuntu@your-ec2-instance-ip  
  
**Check Ownership and Permissions**: Run the following command to check the current permissions on the SDK directory:

ls -ld /home/ubuntu/android-sdk

You’ll likely see something like this:  
o/p drwxr-xr-x 3 ubuntu ubuntu 4096 Oct 31 10:00 /home/ubuntu/android-sdk

**Step 2: Grant Jenkins Access to the SDK Directory**

There are a few ways to give Jenkins access:

1. **Option 1: Change Ownership** (Recommended)
   * Change the owner of the SDK directory to the jenkins user

sudo chown -R jenkins:jenkins /home/ubuntu/android-sdk

**Option 2: Add Jenkins to the Ubuntu User Group**

* Alternatively, add jenkins to the ubuntu group to give it access:

sudo usermod -aG ubuntu Jenkins

### Step 3: Restart Jenkins

After modifying permissions, restart Jenkins to apply any group changes:

sudo systemctl restart Jenkins  
  
  
history of commands that used  
  
ubuntu@ip-172-31-29-146:~$ history

1 sudo systemctl status jenkins

2 java --version

3 clear

4 curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee /usr/share/keyrings/jenkins-keyring.asc > /dev/null

5 echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] https://pkg.jenkins.io/debian-stable binary/ | sudo tee /etc/apt/sources.list.d/jenkins.li st > /dev/null

6 sudo apt-get update

7 sudo apt-get install jenkins -y

8 sudo systemctl enable jenkins

9 sudo systemctl start jenkins

10 sudo systemctl status jenkins

11 sudo cat /var/lib/jenkins/secrets/initialAdminPassword

12 sudo apt update

13 sudo apt install openjdk-11-jdk -y

14 sudo update-alternatives --config java

15 clear

16 mkdir -p ~/android-sdk/cmdline-tools

17 cd ~/android-sdk/cmdline-tools

18 wget https://dl.google.com/android/repository/commandlinetools-linux-11076708\_latest.zip

19 unzip commandlinetools-linux-\*.zip

20 sudp apt install unzip

21 sudo apt install unzip

22 unzip commandlinetools-linux-\*.zip

23 mv cmdline-tools latest

24 export ANDROID\_HOME=~/android-sdk

25 export PATH=$ANDROID\_HOME/cmdline-tools/latest/bin:$ANDROID\_HOME/platform-tools:$PATH

26 source ~/.bashrc

27 sdkmanager "platform-tools" "build-tools;33.0.0" "platforms;android-33"

28 sudo update-alternatives --config java

29 sdkmanager "platform-tools" "build-tools;33.0.0" "platforms;android-33"

30 sudo apt install gradle -y

31 ls ~/android-sdk

32 cd ..

33 cd ~/android-sdk

34 cd ..

35 ls

36 echo "ANDROID\_HOME is set to: $ANDROID\_HOME"

37 ls -ld /home/ubuntu/android-sdk

38 sudo chown -R jenkins:jenkins /home/ubuntu/android-sdk

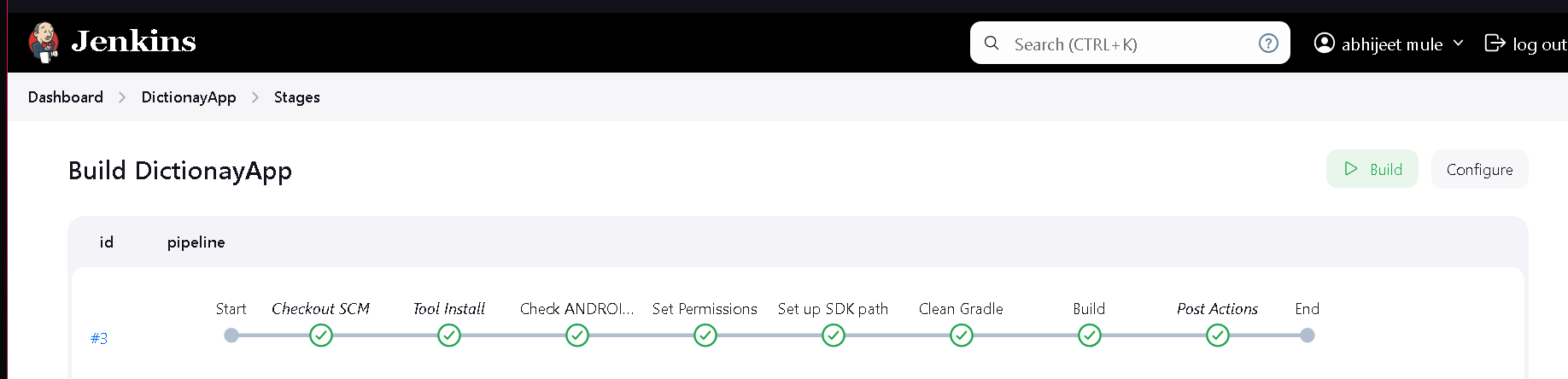
39 sudo usermod -aG ubuntu jenkins

40 sudo chmod -R 755 /home/ubuntu/android-sdk

41 sudo systemctl restart jenkins

42 history

ubuntu@ip-172-31-29-146:~$  
  
final output



For vagrant setup do same only for path instead of **ubuntu** set **vagrant**

The error indicates that certain Android SDK components are missing or their licenses have not been accepted. Follow these steps to resolve the issue:

### ****1. Install Missing SDK Components****

You need to install the missing components and accept the licenses.

#### ****Install the Required SDK Components****

Run the following commands to install the missing packages:

$ sdkmanager "platforms;android-32" "build-tools;30.0.3"

#### ****Accept the Licenses****

Accept the licenses for the installed components

$ yes | sdkmanager –licenses

### ****2. Verify SDK Components****

To verify that the required components are installed, use:

$ sdkmanager –list

The error indicates that certain Android SDK components are missing or their licenses have not been accepted. Follow these steps to resolve the issue:

### ****1. Install Missing SDK Components****

You need to install the missing components and accept the licenses.

#### ****Install the Required SDK Components****

Run the following commands to install the missing packages:

bash

Copy code

sdkmanager "platforms;android-32" "build-tools;30.0.3"

#### ****Accept the Licenses****

Accept the licenses for the installed components:

bash

Copy code

yes | sdkmanager --licenses

This will ensure all required licenses are accepted.

### ****2. Verify SDK Components****

To verify that the required components are installed, use:

bash

Copy code

sdkmanager --list

Ensure you see entries for:

* platforms;android-32
* build-tools;30.0.3

### ****3. Grant Jenkins Access****

Ensure the Jenkins user has access to the SDK directory. Run the following commands:

$ sudo chown -R jenkins:jenkins /home/vagrant/Android/Sdk

$ sudo chmod -R 755 /home/vagrant/Android/Sdk

**Now Add SonarQube Setup in Dictionary app**

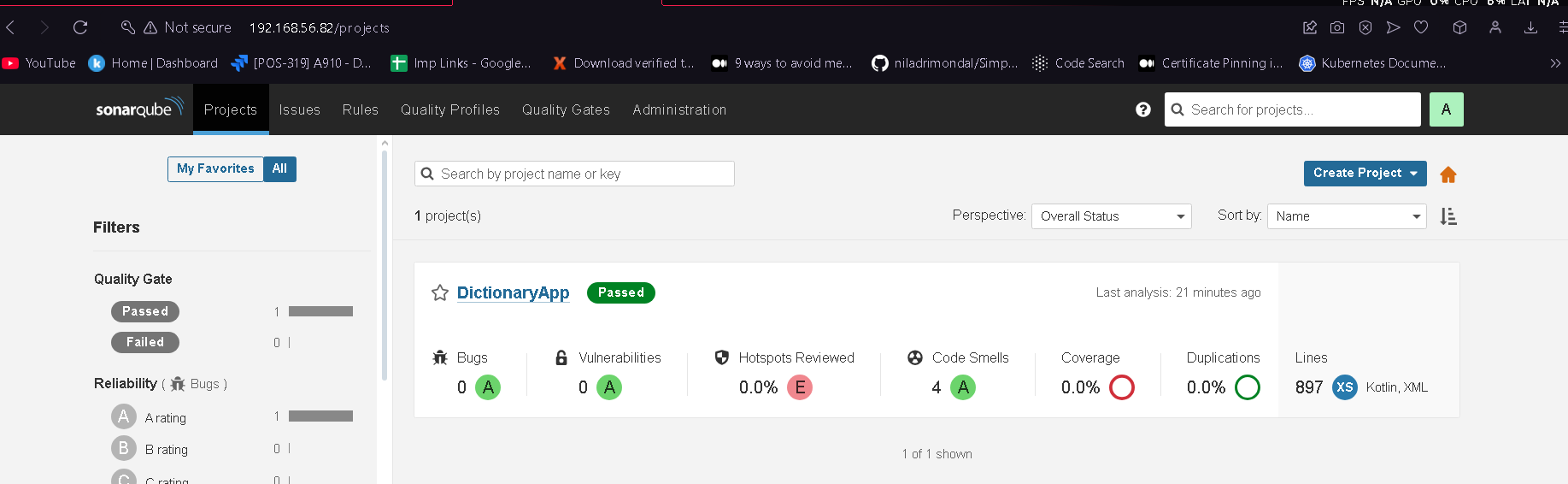
Step1) Add Sonar.sh script in vagrant vm or ec2 Instance

Refer below github link

<https://github.com/Abhijeet41/vprofile-project-aws/blob/atom/userdata/sonar-setup.sh>

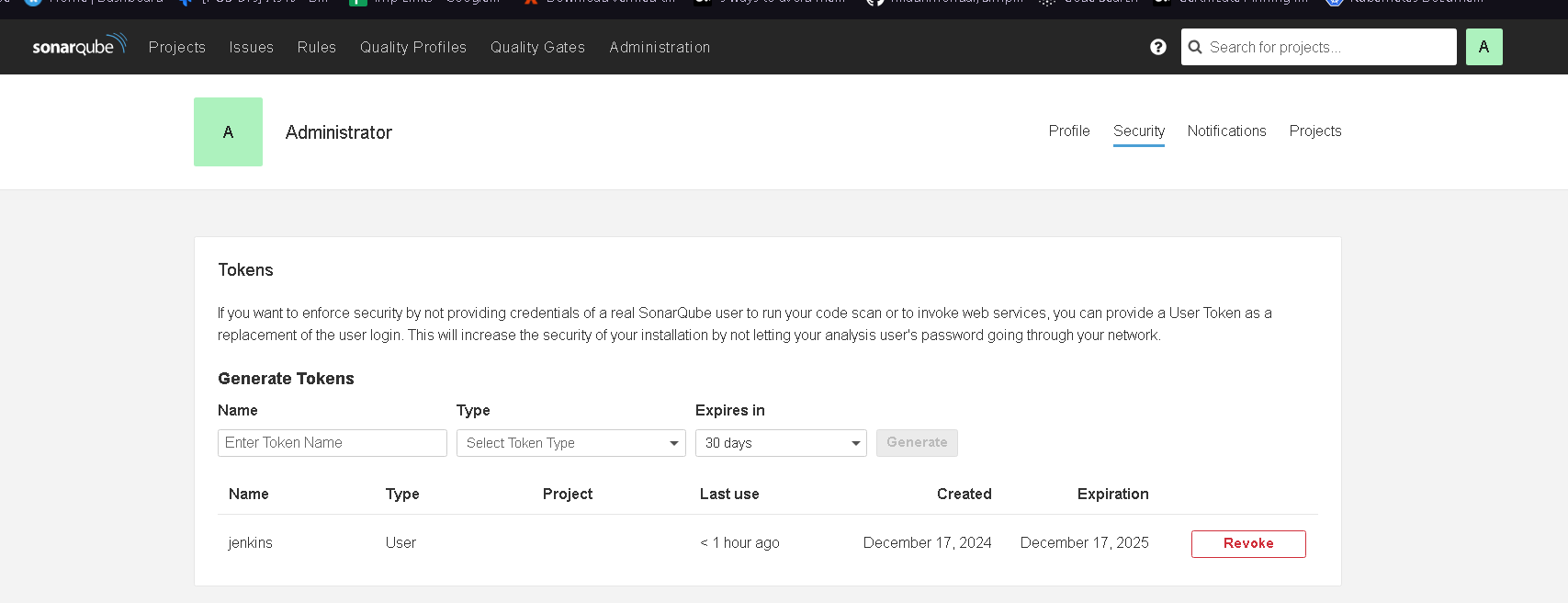
you will get this page after paste url with port 80 in browser

<http://192.168.56.82:80>

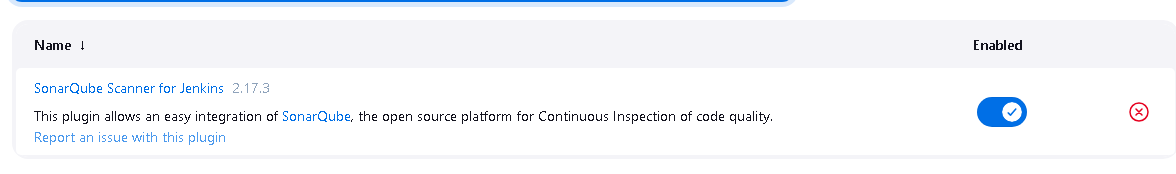


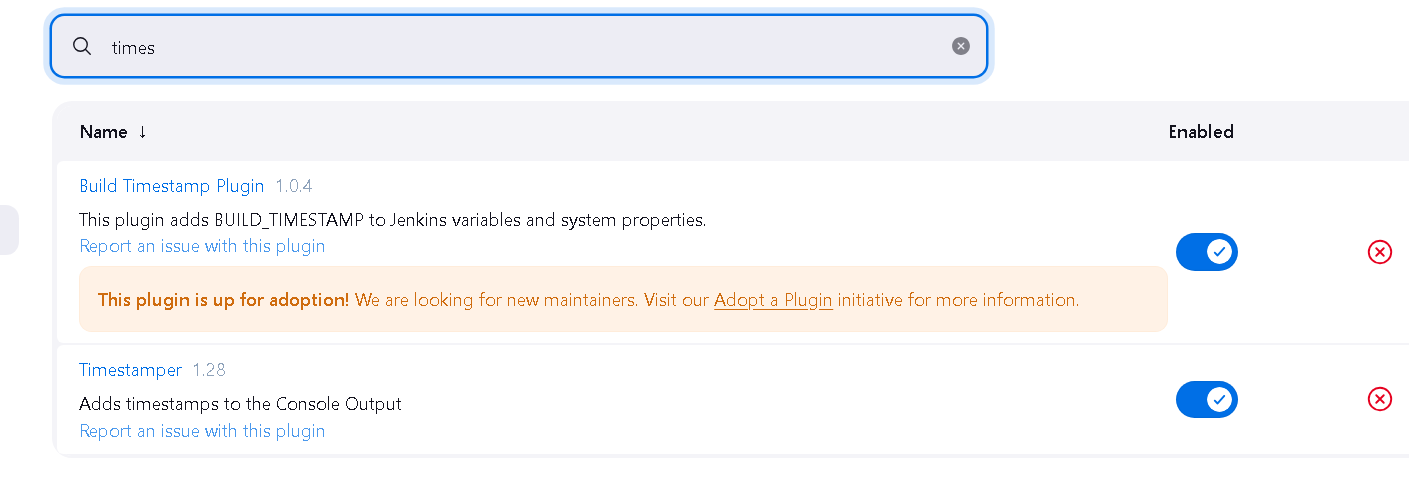
Now click on Administrator and then MyAccount and then select Security tab

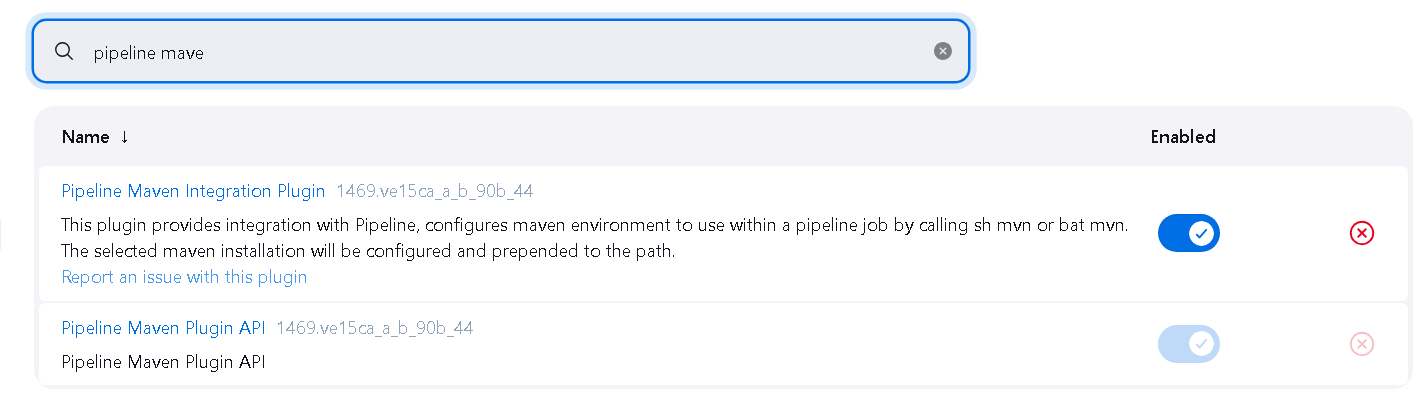
And here you should generate the token I set token name Jenkins

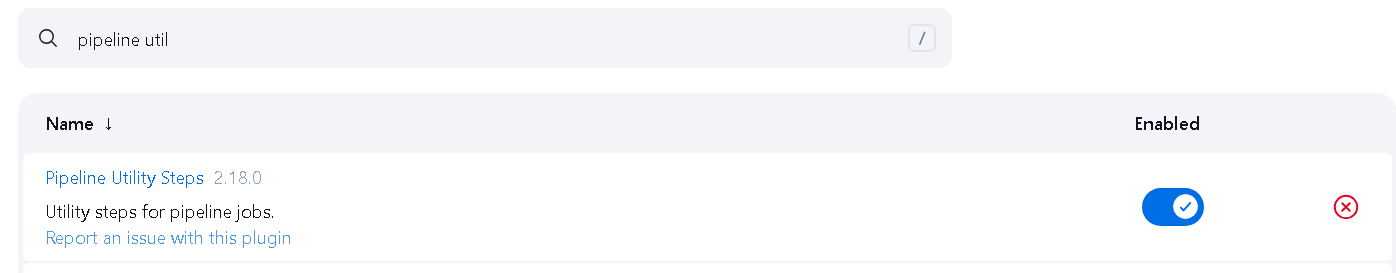


now in Jenkins add below plugins

****

****

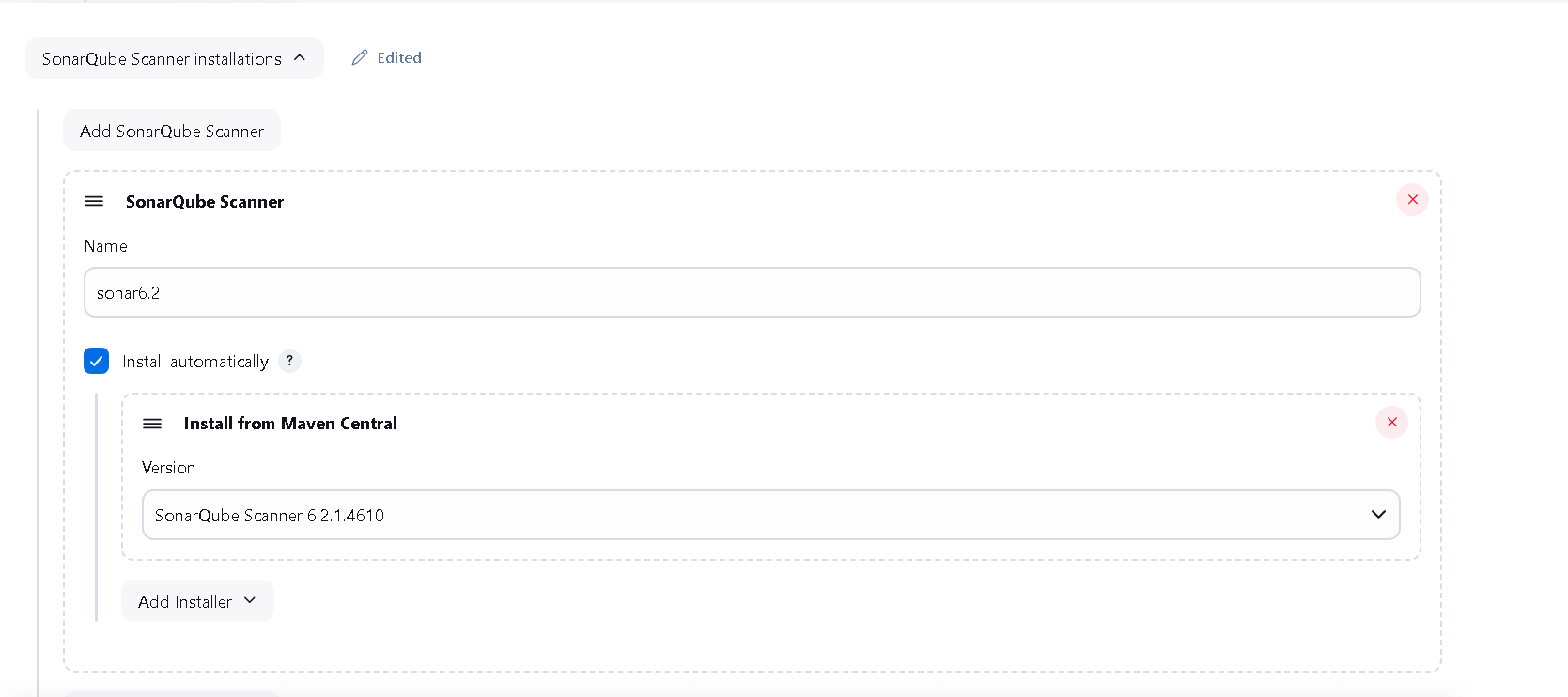
****



Now Add SonarServer Details in Jenkins

Go to Dashboard > Manage Jenkins -> Tools

Search SonarQube Scanner and click on add sonarqube scanner and set name sonar6.2 and hit apply and save

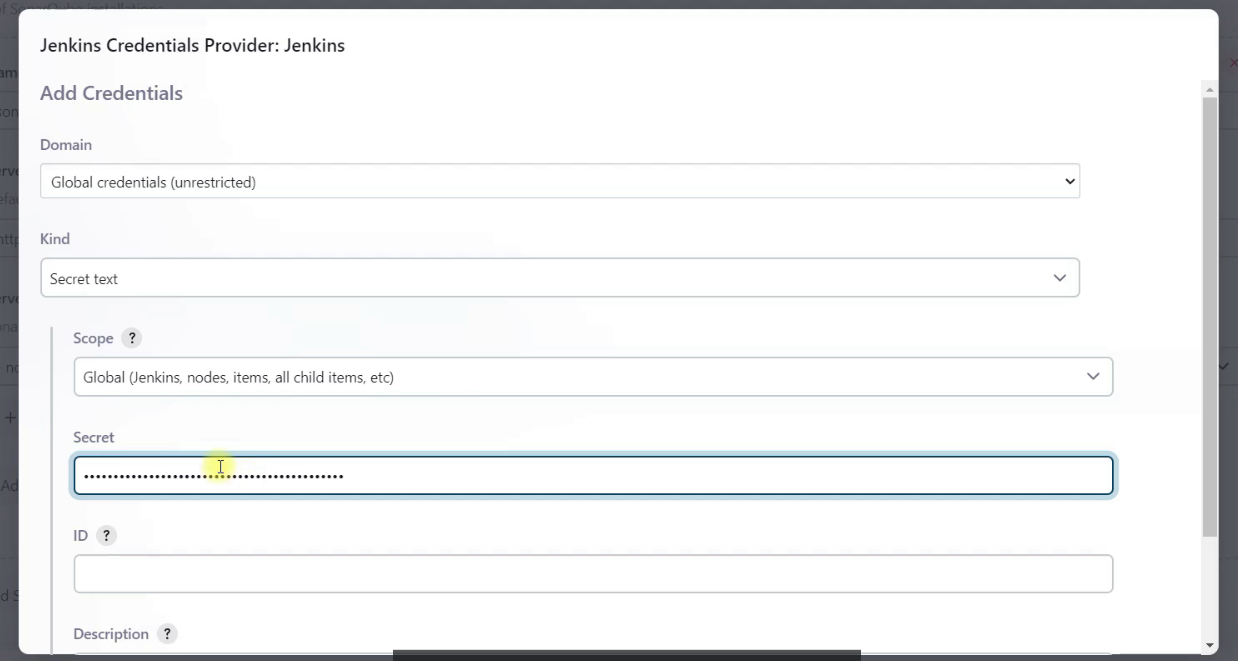


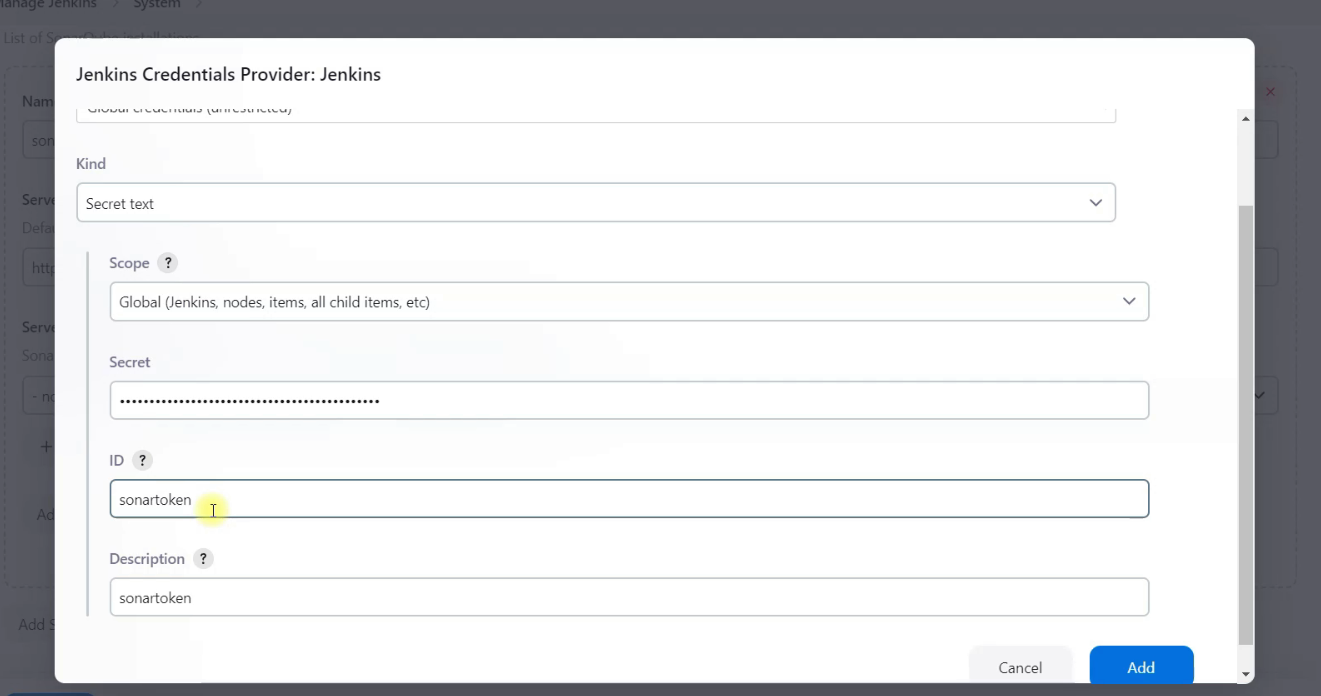
Now for upload result on sonarqube server go to system and set sonar details

System scroll down until see SonarQube servers and check mark Enviroment variables and click on add sonarqube I set my vagrant private ip if you are using ec2 then set ec2 private ip



Now here add token that we generate in sonarqube page





Now Integrate webhook so sonar qube will be send the result using webhook to Jenkins

1) Click on your project name mine is Dictionary App

Now finally in Jenkins file add this below stage

stage('SonarQube Analysis') {  
 environment {  
 SONAR\_TOKEN = credentials('sonartoken') // Reference the credential ID created in Jenkins  
 SONARQUBE\_URL = 'http://192.168.56.82:9000'  
 }  
 steps {  
 withSonarQubeEnv('sonarserver') { // Replace 'sonarserver' with the name of your SonarQube server in Jenkins  
 sh '''  
 sonar-scanner \  
 -Dsonar.projectKey=DictionaryApp \  
 -Dsonar.sources=. \  
 -Dsonar.host.url=$SONARQUBE\_URL \  
 -Dsonar.login=$SONAR\_TOKEN'''  
 }  
 }  
 }

Now after adding SonarQube final Jenkinsfile will be

pipeline {  
 agent any  
  
 tools {  
 jdk 'OracleJDK11'  
 }  
  
 stages {  
 stage('Check ANDROID\_HOME') {  
 steps {  
 sh 'echo "ANDROID\_HOME is set to: $ANDROID\_HOME"'  
 sh 'ls $ANDROID\_HOME'  
 }  
 }  
  
 stage('Set Permissions') {  
 steps {  
 sh 'chmod +x ./gradlew'  
 }  
 }  
  
 stage('Set up SDK path') {  
 steps {  
 sh 'echo "sdk.dir=$ANDROID\_HOME" > local.properties'  
 }  
 }  
  
 stage('Clean Gradle') {  
 steps {  
 sh './gradlew clean --no-daemon'  
 }  
 }  
  
 stage('Build') {  
 steps {  
 sh './gradlew assembleRelease'  
 }  
 }  
 stage('SonarQube Analysis') {  
 environment {  
 SONAR\_TOKEN = credentials('sonartoken') // Reference the credential ID created in Jenkins  
 SONARQUBE\_URL = 'http://192.168.56.82:9000'  
 }  
 steps {  
 withSonarQubeEnv('sonarserver') { // Replace 'sonarserver' with the name of your SonarQube server in Jenkins  
 sh '''  
 sonar-scanner \  
 -Dsonar.projectKey=DictionaryApp \  
 -Dsonar.sources=. \  
 -Dsonar.host.url=$SONARQUBE\_URL \  
 -Dsonar.login=$SONAR\_TOKEN'''  
 }  
 }  
 }  
 }  
  
 post {  
 success {  
 archiveArtifacts artifacts: 'app/build/outputs/apk/release/\*.apk', allowEmptyArchive: true  
 echo 'Build completed successfully!'  
 }  
 failure {  
 echo 'Build failed. Please check the logs for errors.'  
 }  
 }  
}