**Jenkins-03**

**Title: Task 1  
Create one Jenkins job using the below code and create three stages:**

* **stage1: Git clone to download the source code.**
* **stage2: SonarQube Integration to check the quality of code.**
* **stage3: Slack Integration to send the alerts to slack.**

**URL:**[**https://github.com/betawins/VProfile-1.git**](https://github.com/betawins/VProfile-1.git)

**objective:** To create a Jenkins pipeline job that:

1. Downloads source code from a GitHub repository
2. Analyses code quality using SonarQube
3. Sends build status notifications to Slack

This ensures continuous integration, code quality enforcement, and real-time alerts.

**prerequisites**:

Jenkins server installed with instance type m7 flexi large and running Jenkins

require another instance to run SonarQube with instance type m7flex large

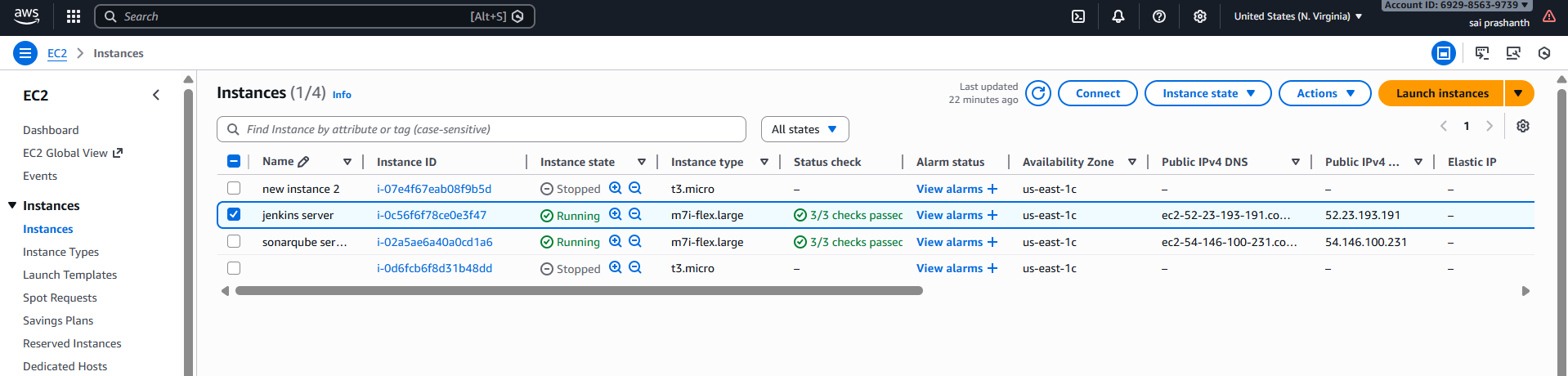
sonar scanner plugin download

creating token in SonarQube

configuration changes in Jenkins GUI level

**step by step process:**

first install two instances one for Jenkins and another one for SonarQube and install Jenkins in Jenkins instance..

****

sudo yum install -y java-17-amazon-corretto

now go to your browser and search for Jenkins download

click on redhat option---copy the link and paste it here

use the command--- sudo wget -O /etc/yum.repos.d/jenkins.repo \

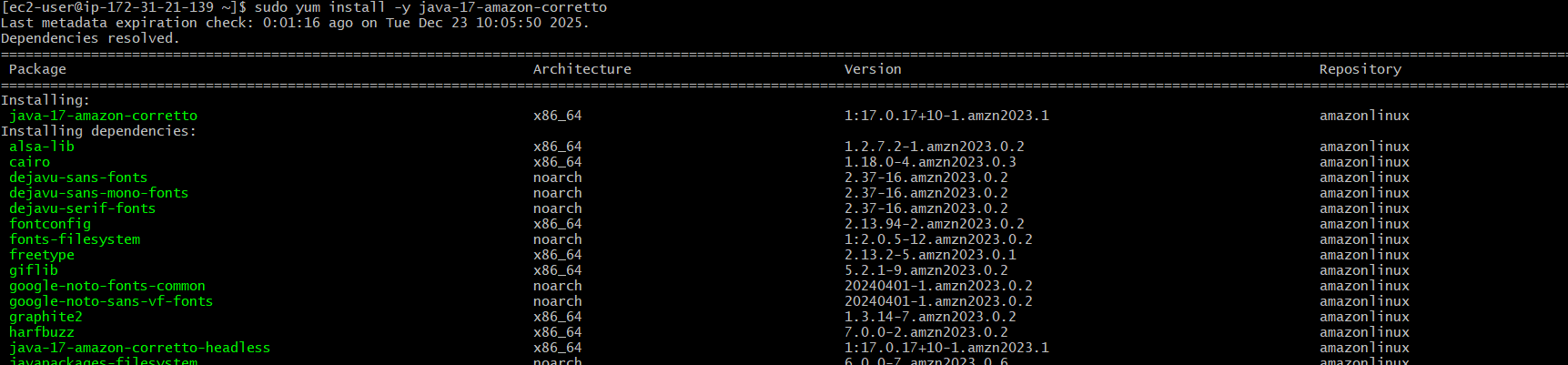
https://pkg.jenkins.io/redhat-stable/jenkins.repo

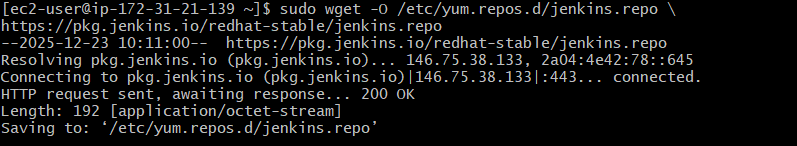
download the rpm key also which is available

use the command

sudo rpm --import <https://pkg.jenkins.io/redhat-stable/jenkins.io.key>

now install Jenkins---use the command---yum install Jenkins

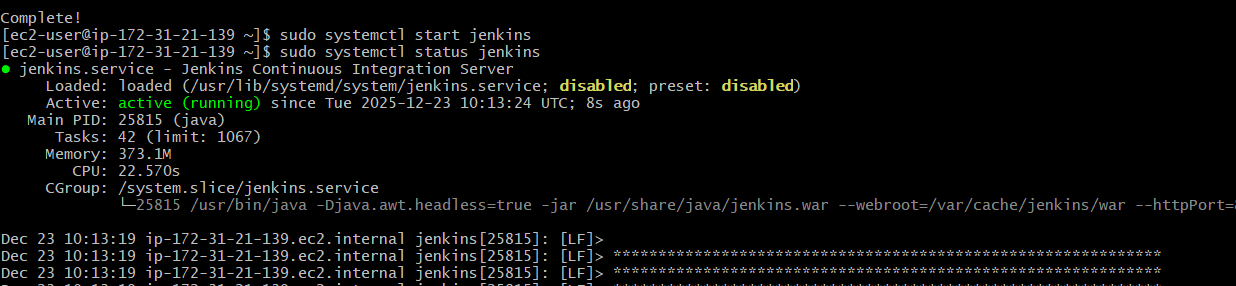




after getting installed check the status of Jenkins

systemctl start Jenkins

systemctl status Jenkins



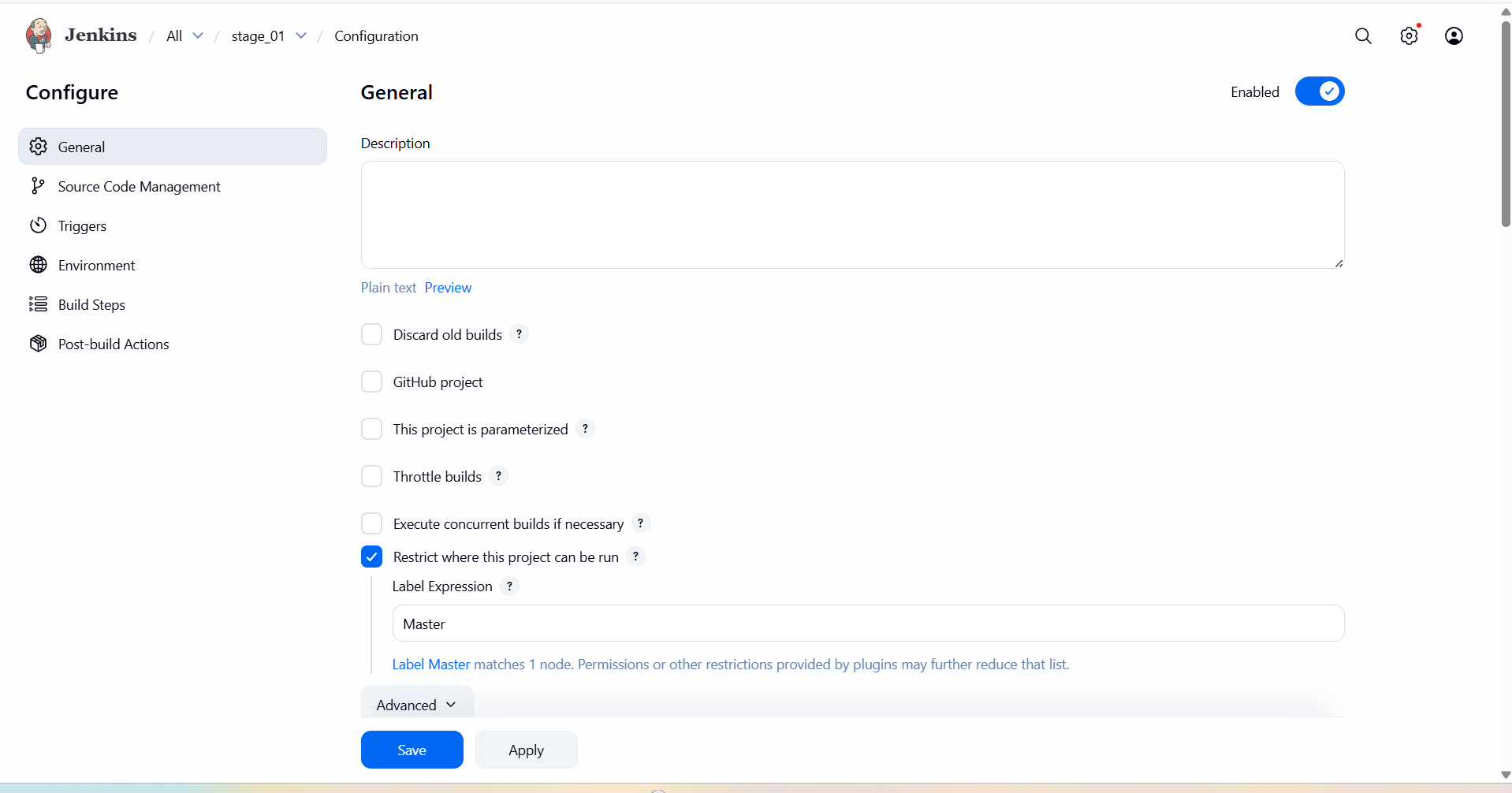
now open in browser and check with public key with 8080 port number and we need to clone the repository from GitHub from URL <https://github.com/betawins/VProfile-1.git>

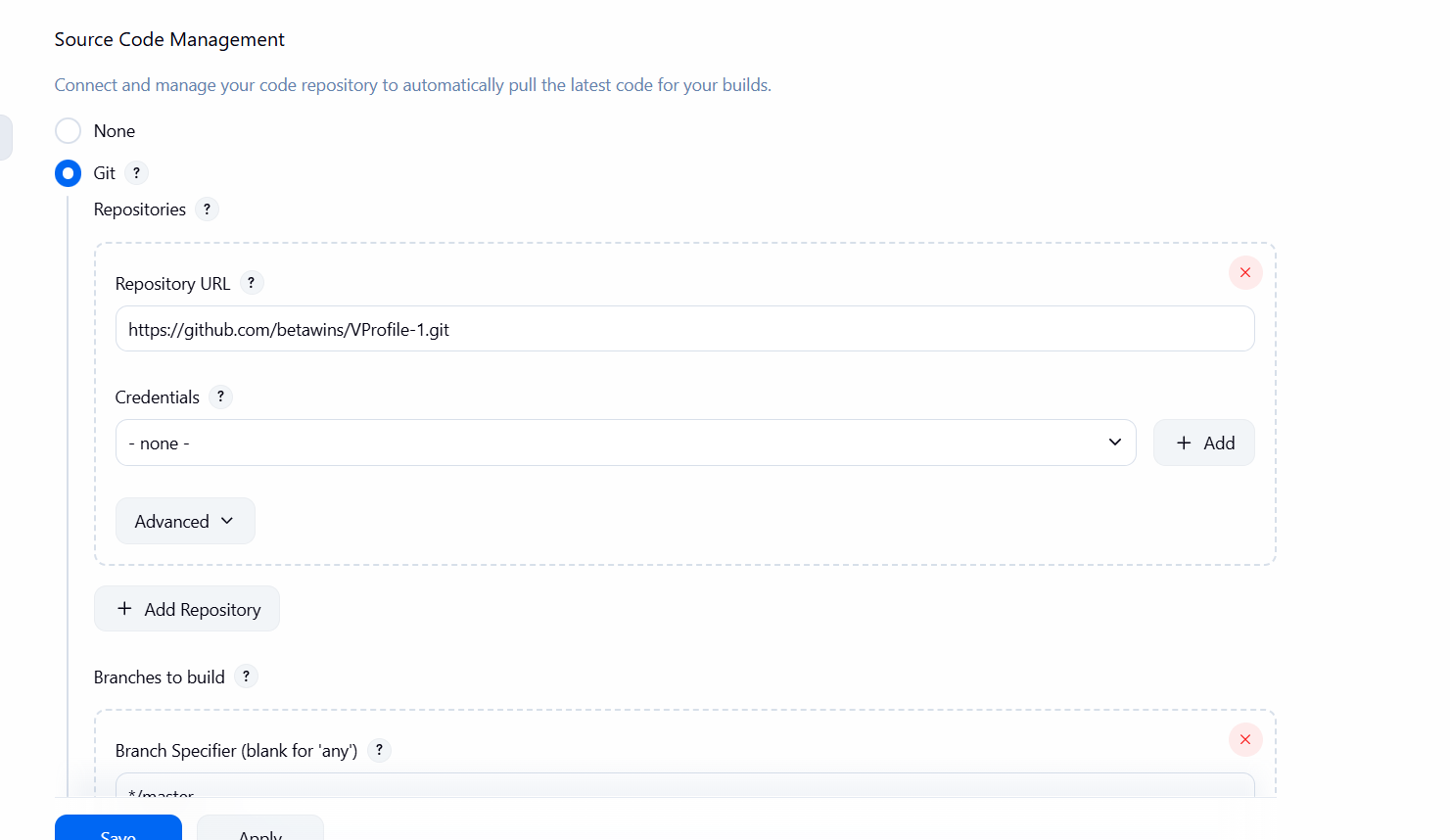
create one job…label name--master

in source code management---select git---paste the URL

and click on save changes….

now we need to integrate SonarQube



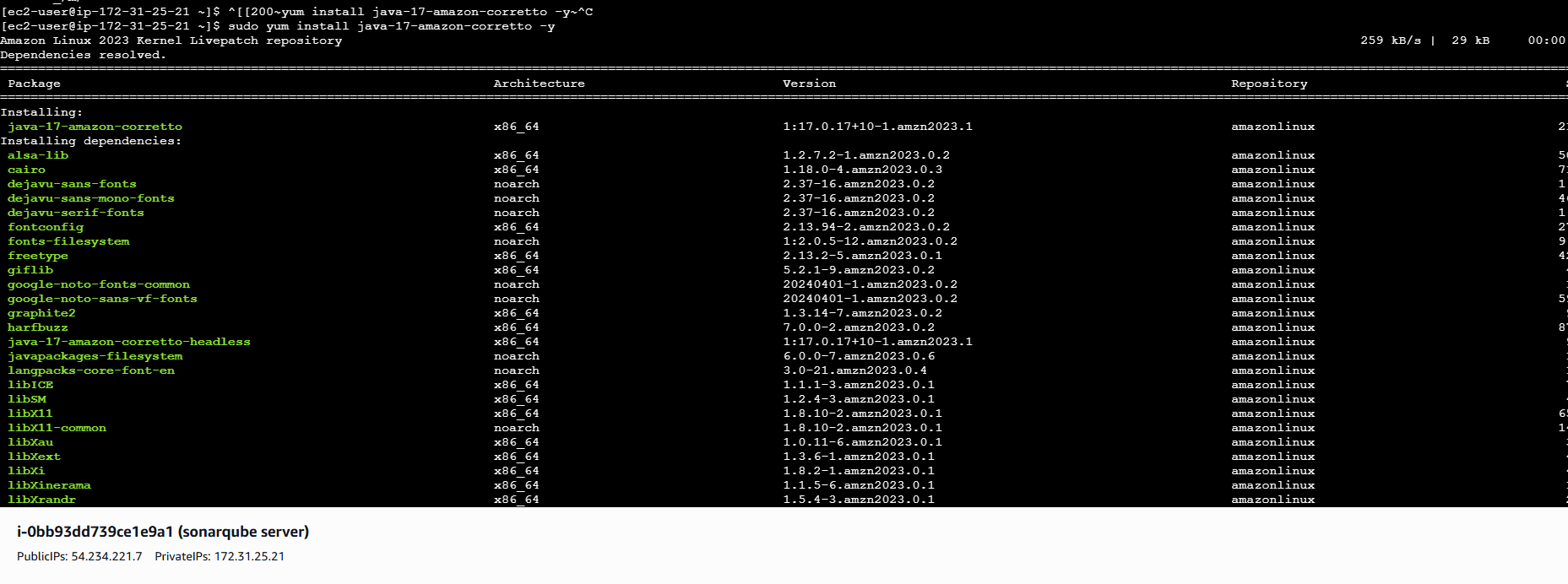




now after cloning the repository….

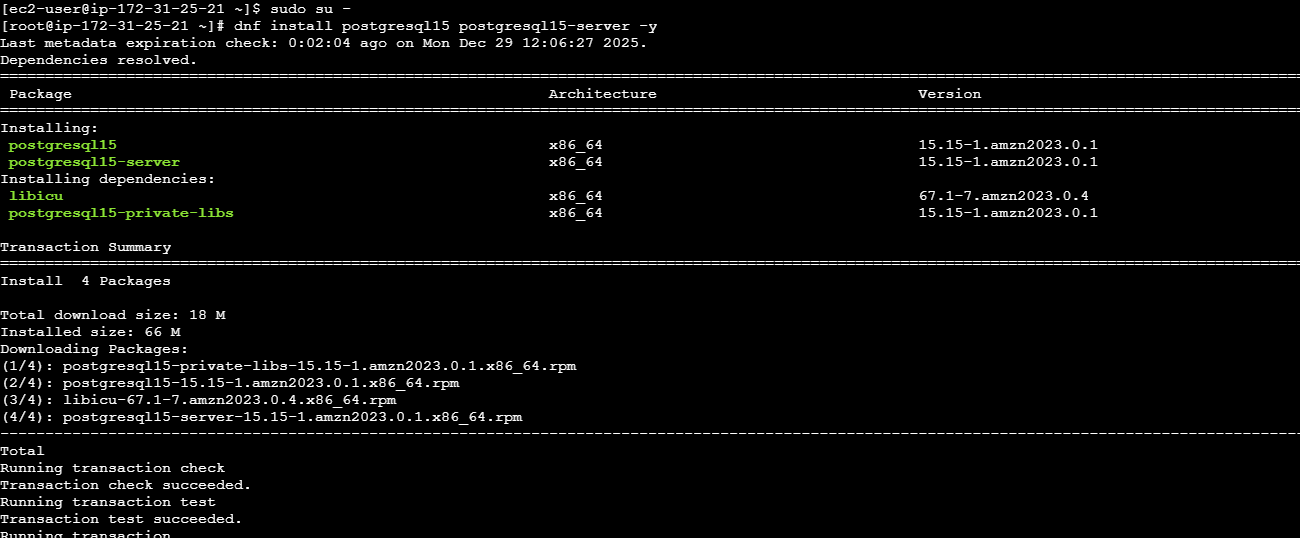
connect with another instance now for installation of SonarQube

install java first in SonarQube server



after getting installed java---first we need to install one database for dependency of SonarQube

here installation of postgres----use command-- dnf install postgresql15 postgresql15-server -y



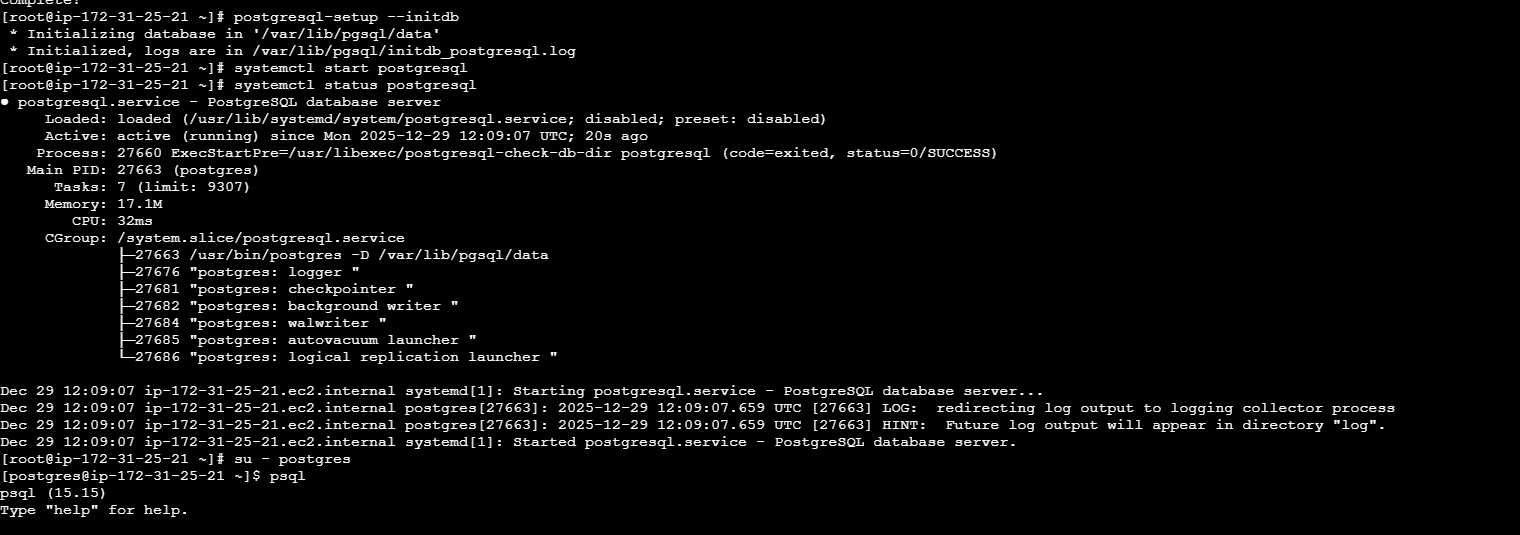
to initialize postgres now…use command--- postgresql-setup –initdb

now enable and start postgres—use command—

systemctl start postgresql

systemctl enable postgresql

systemctl status postgresql



now you created Postgres database for SonarQube.

switch to postgres user---su -postgres

we need to execute the commands…

CREATE DATABASE sonarqube;

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

GRANT ALL PRIVILEGES ON DATABASE sonarqube TO sonar;

ALTER DATABASE sonarqube OWNER TO sonar;

to verify the commands…

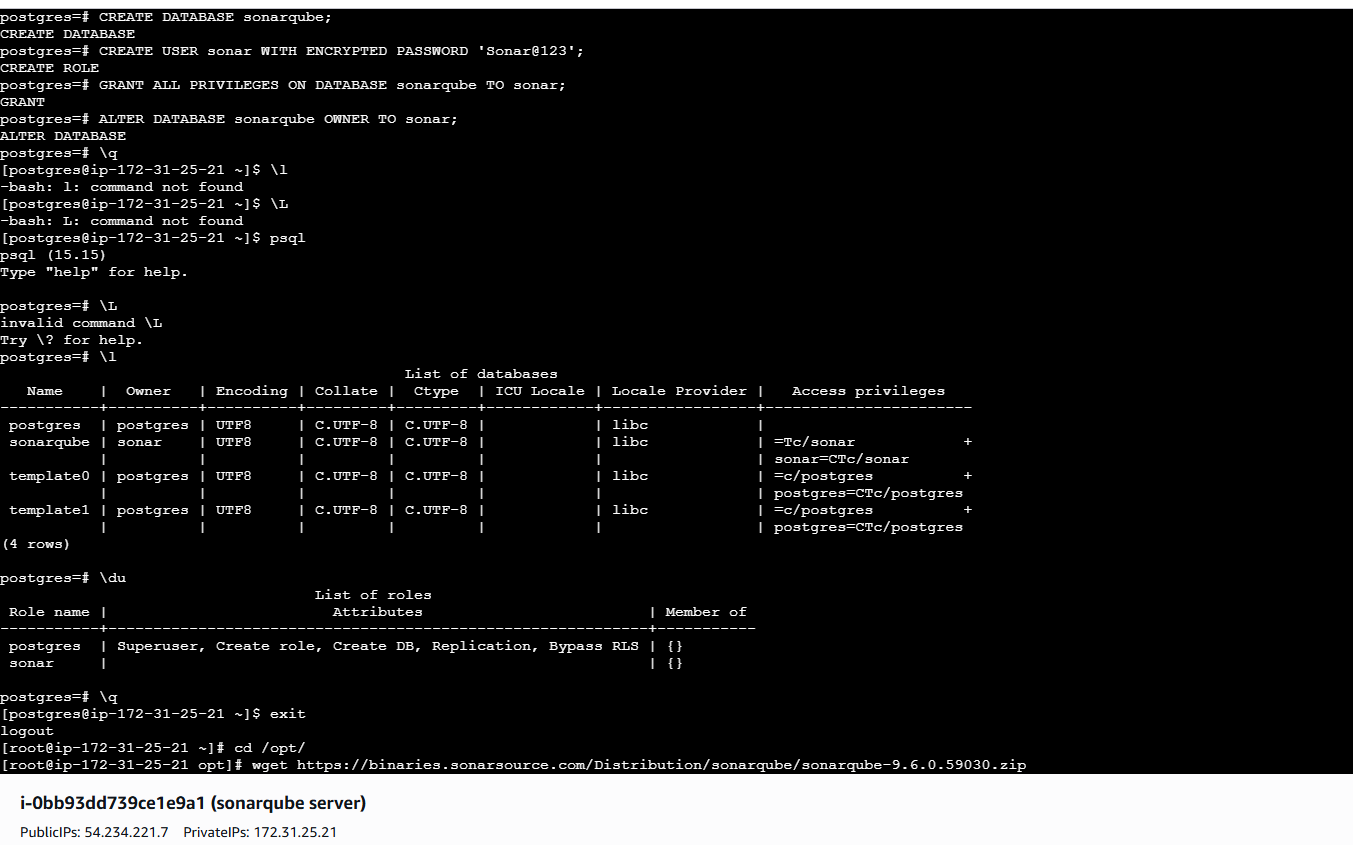
\l

\du

**Exit**

\q

exit



once it has been done…

we need to install now SonarQube.

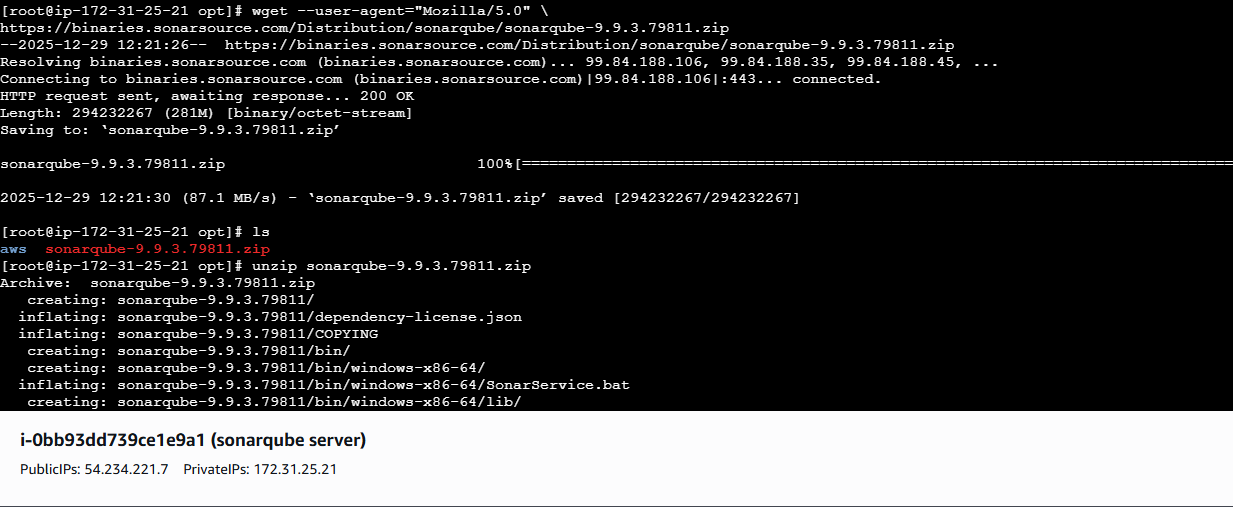
for sonarqube installation---use the command—

cd /opt

wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-9.6.0.59030.zip

unzip sonarqube-9.6.0.59030.zip

mv sonarqube-9.6.0.59030 sonar for renaming



now we need to create one user for executing

execute the commands…and give permissions

useradd sonar

chown -R sonar:sonar /opt/sonar

chmod -R 775 /opt/sonar

now we need to make someconfigirations…

use command vi sonar.properties

add the username and password

add the website as 0.0.0.0

sonar.jdbc.username=sonar

sonar.jdbc.password=Sonar@123

sonar.jdbc.url=jdbc:postgresql://localhost/sonarqube

sonar.web.port=9000

and save it

now we need to start the SonarQube

execute the commands…

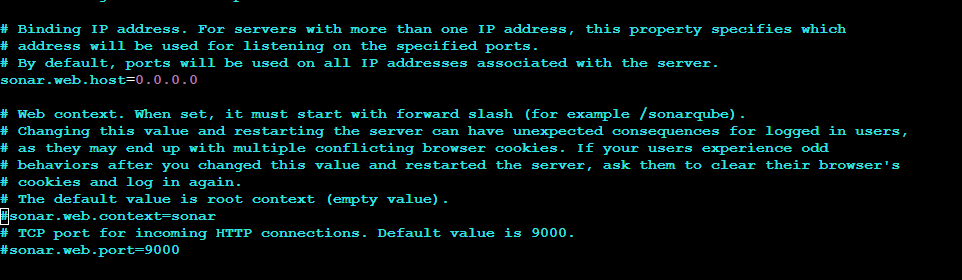
su - sonar

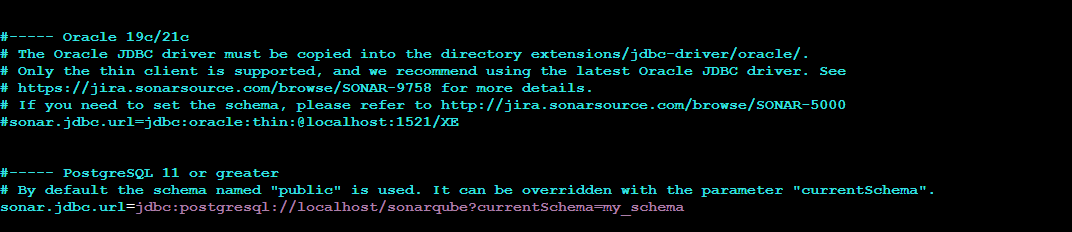
cd /opt/sonar/bin/linux-x86-64

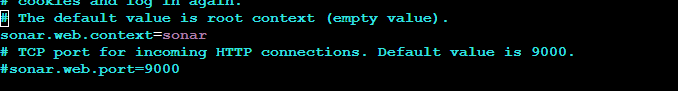
./sonar.sh start

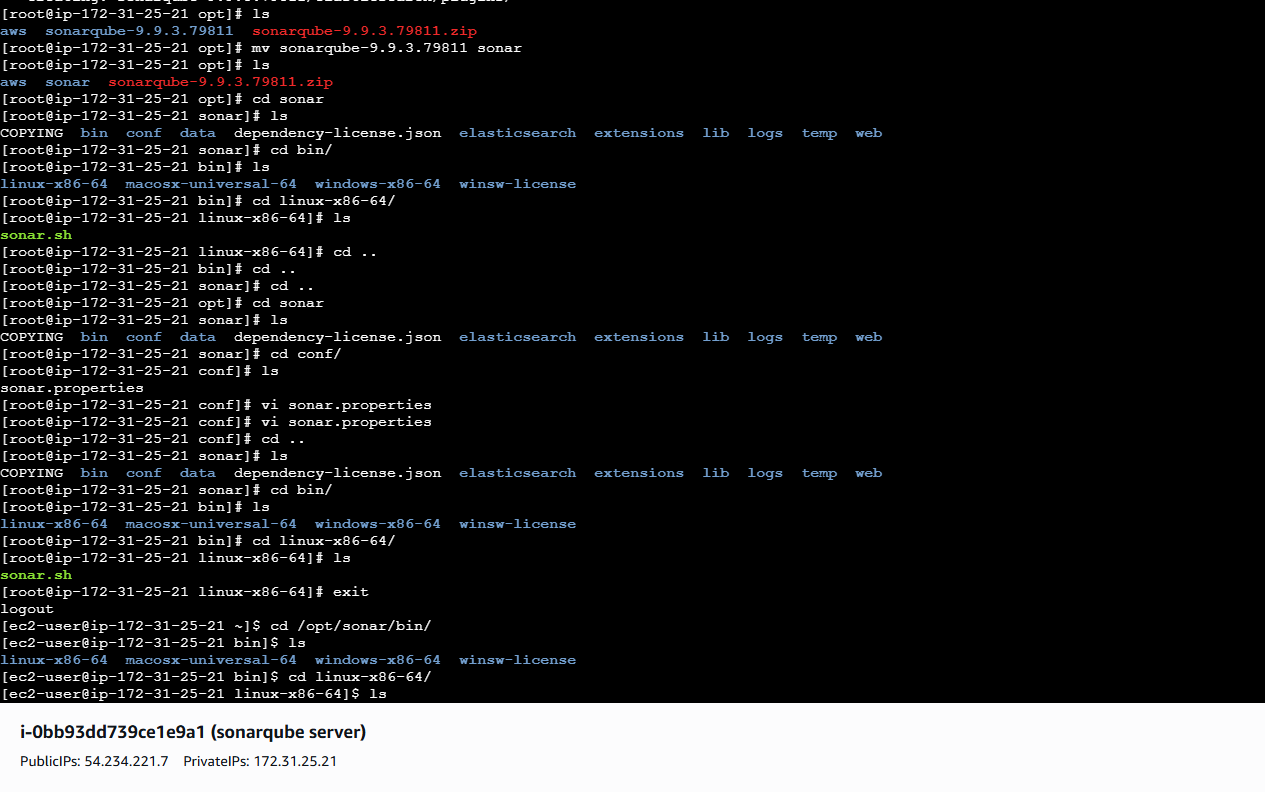
./sonar.sh status

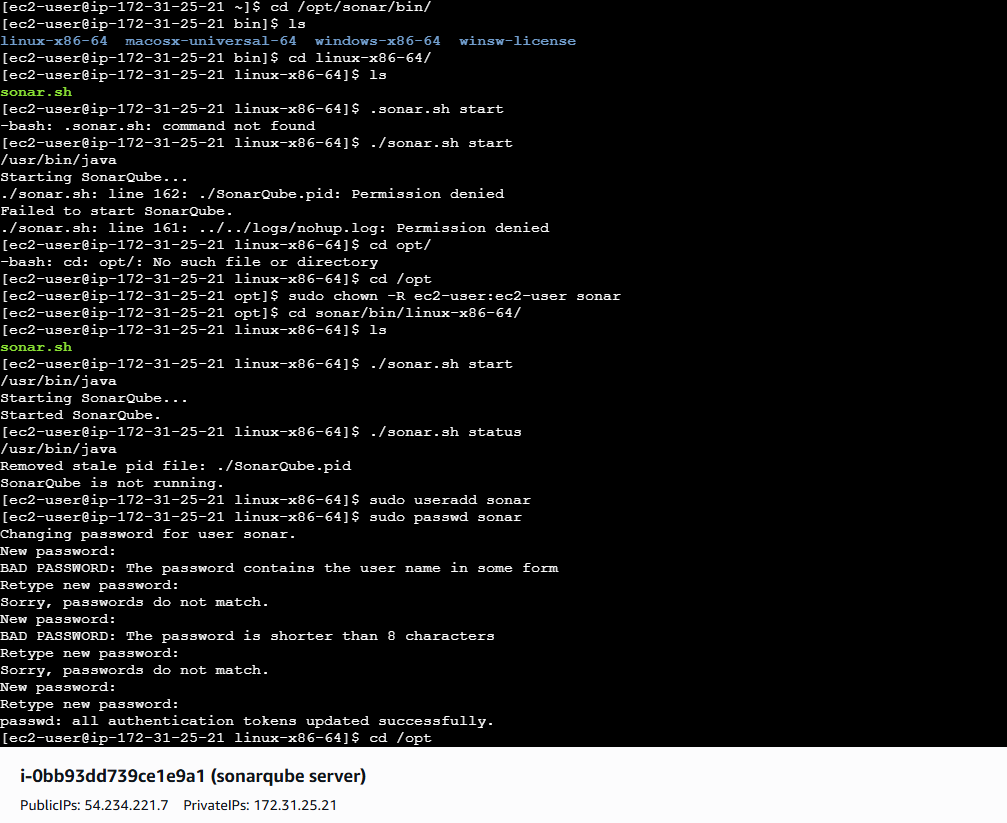


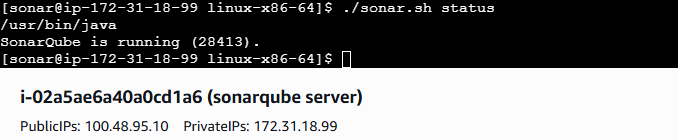






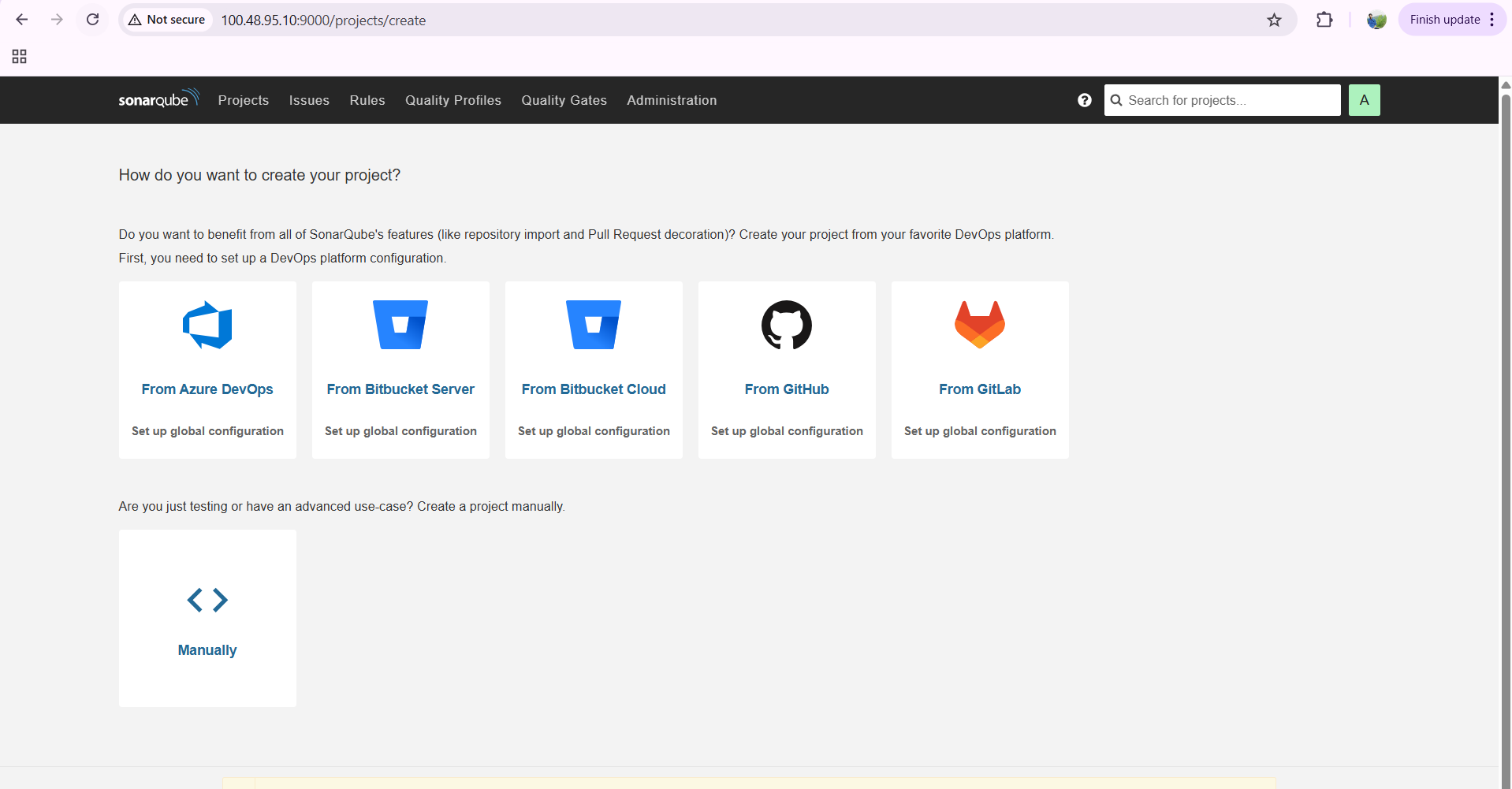






now enter with SonarQube public Ip and with port number 9000

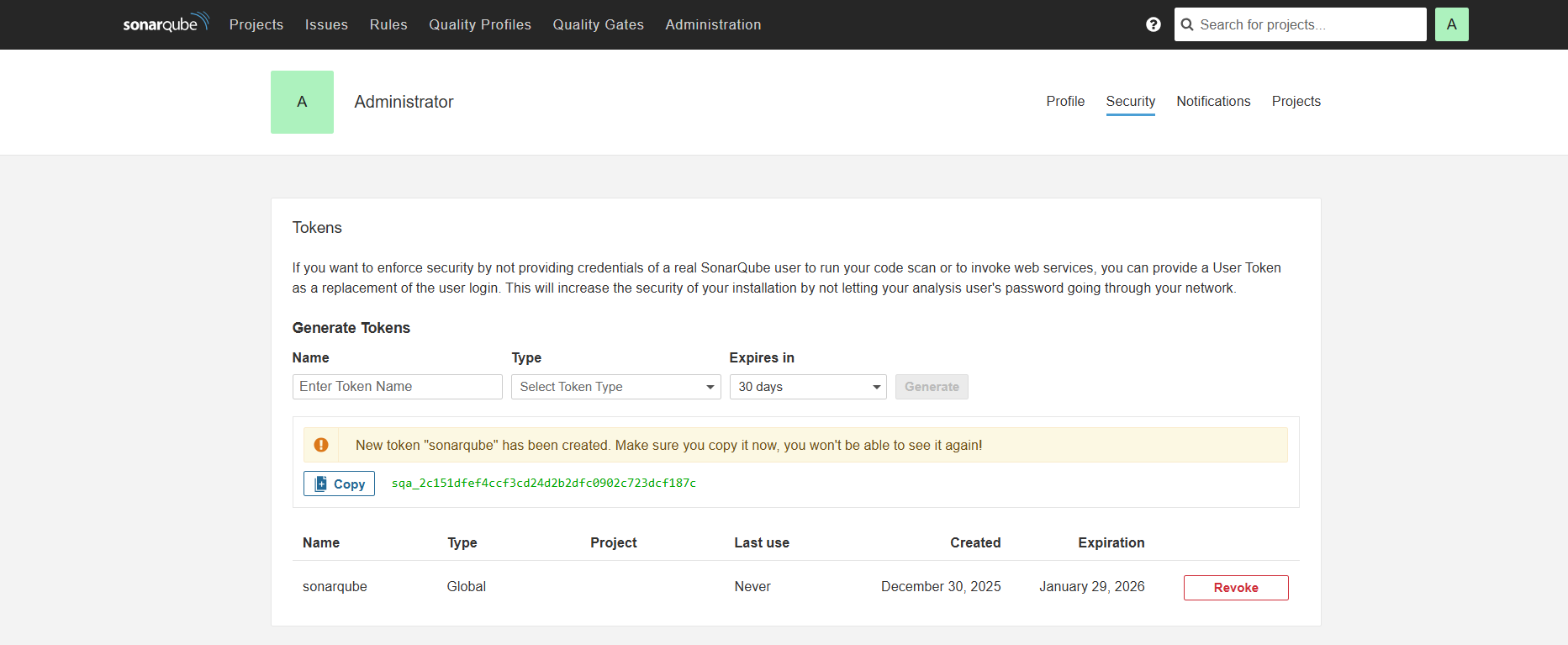
you can see SonarQube has been running in browser now



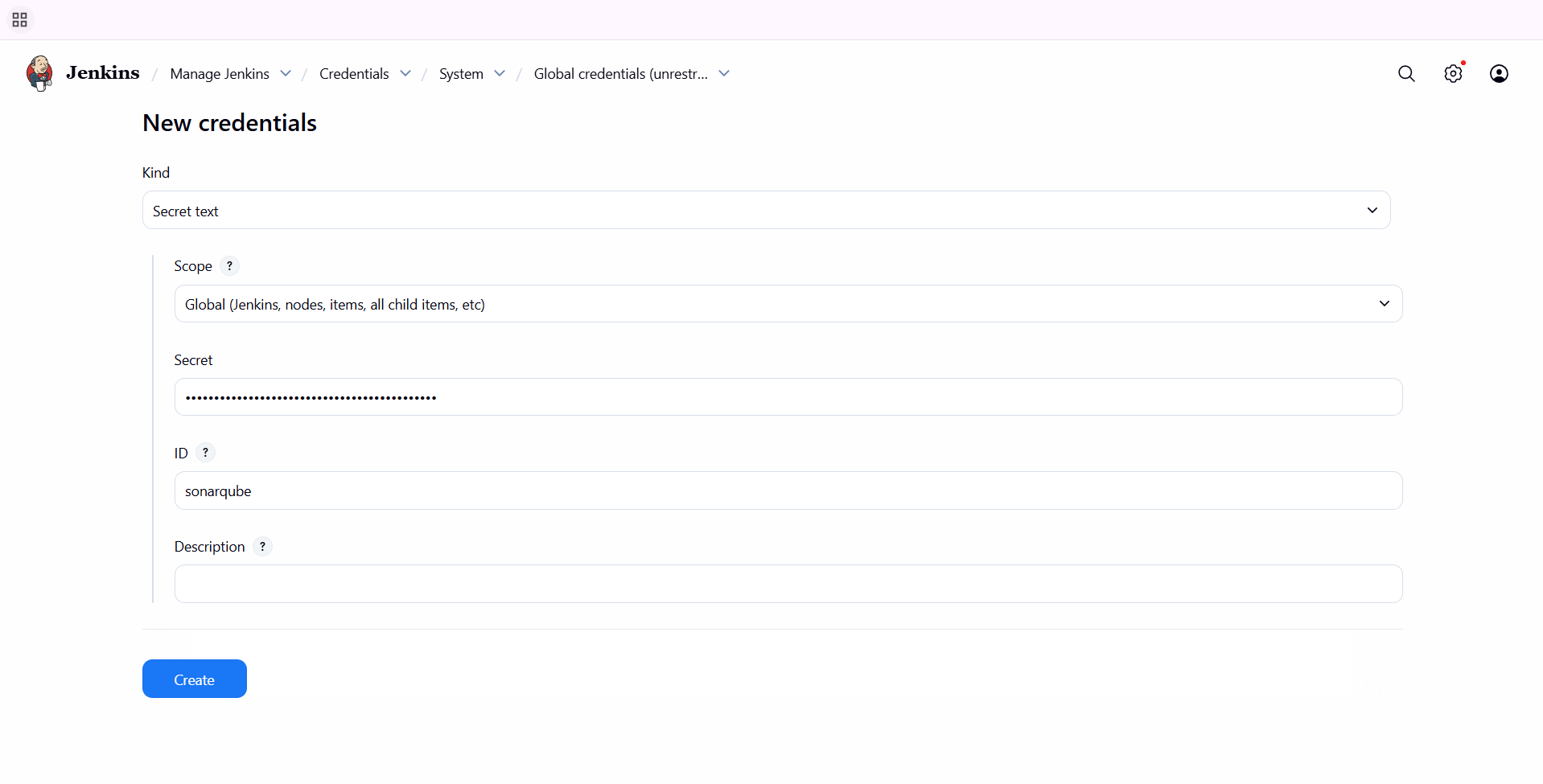
now we need to create one token to add the credentials in Jenkins

go to administrator---security---create one token—enter token name—type as global and create

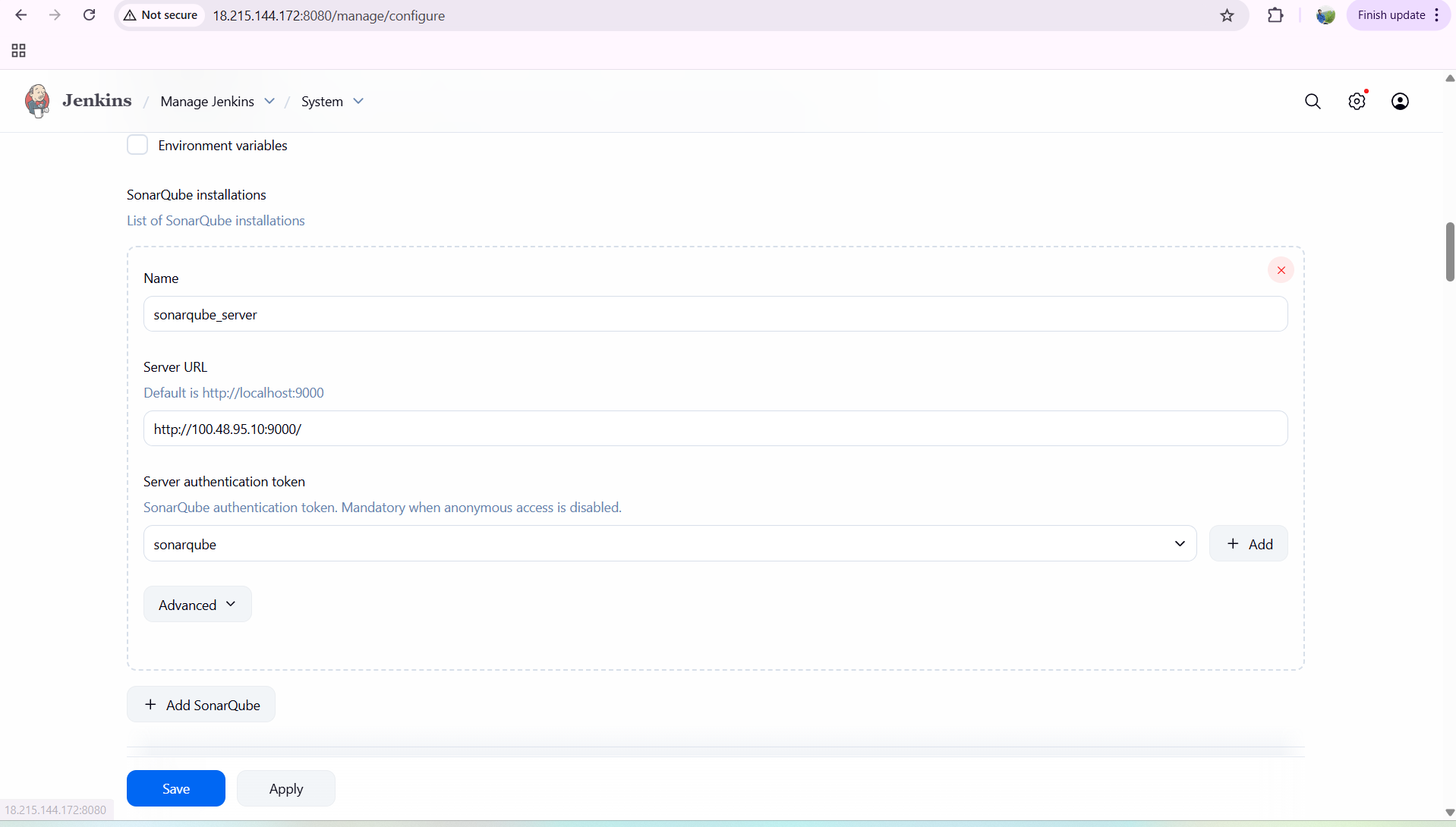
you can see one token has been created



now come to jenkins dashboard---credentials—system---type as secret text—and copy that token key and paste it in secret tab…and click on create

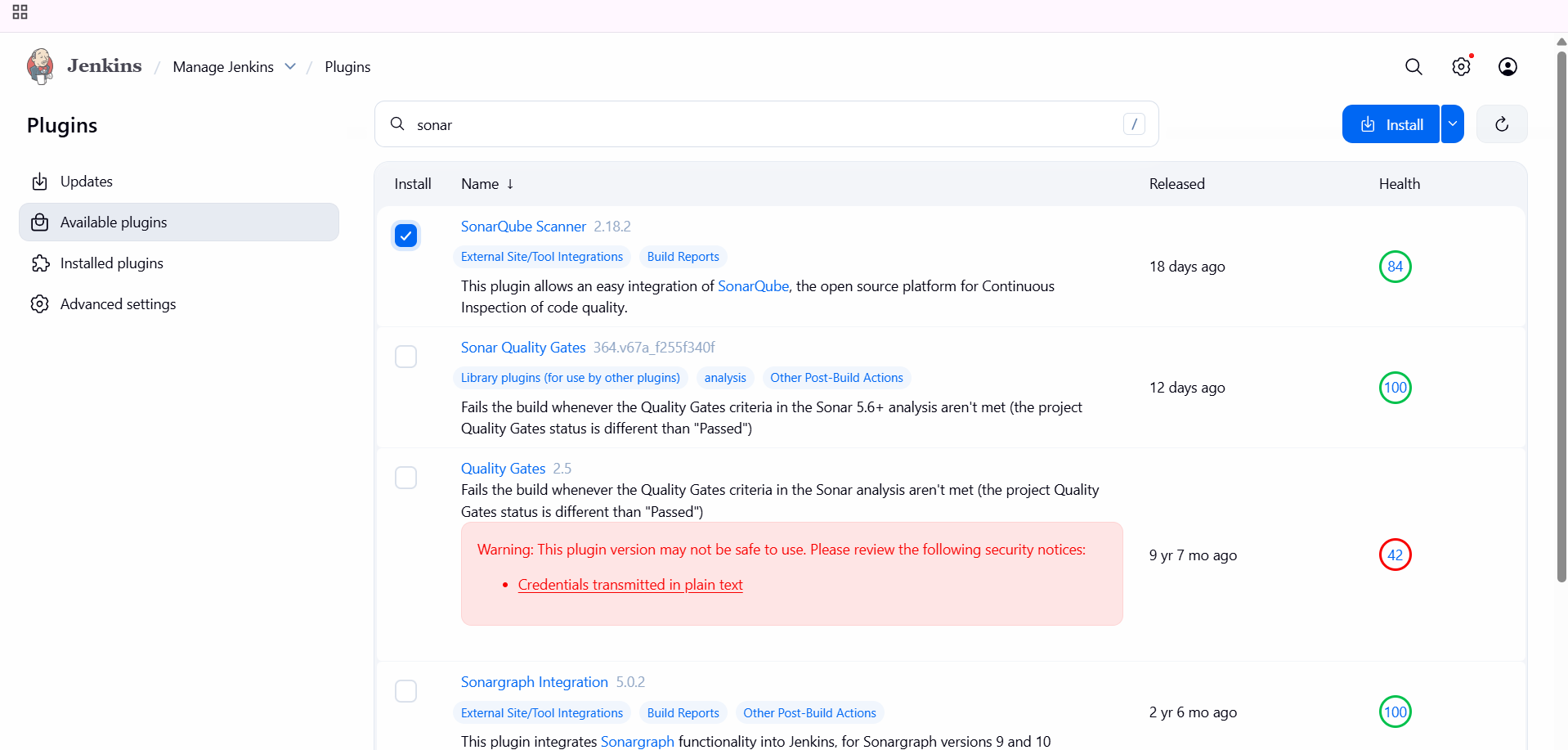


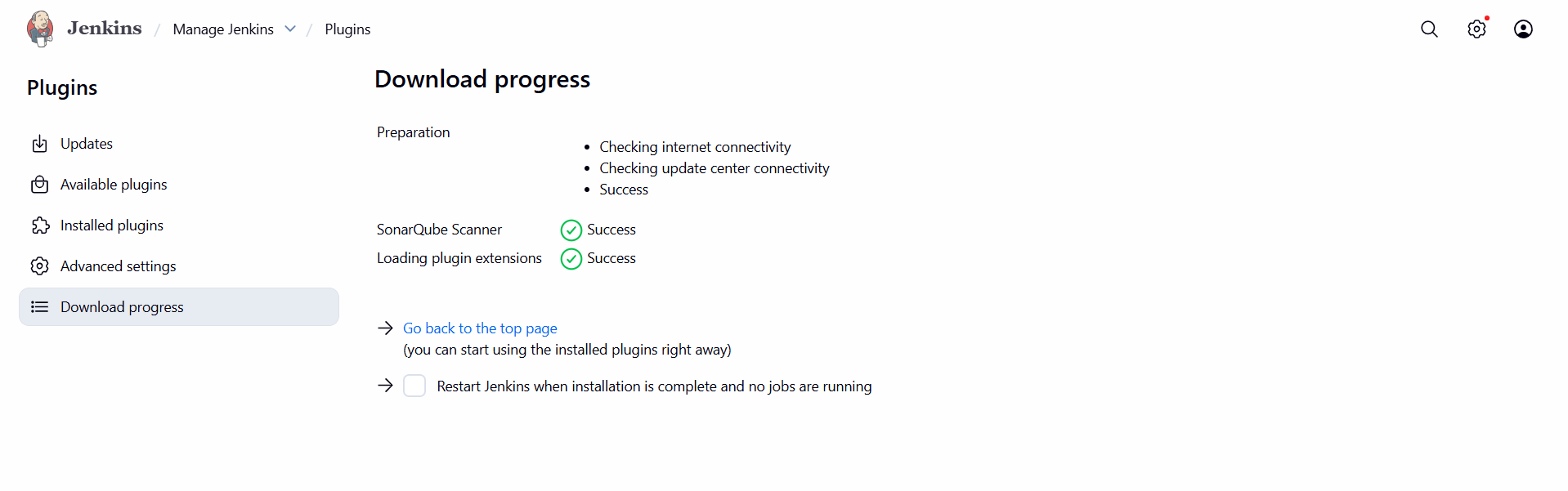
now come to manage jenkins---system—sonarqube installations—enter name—add the url of your sonarqube—and add the token which you have created and save it..

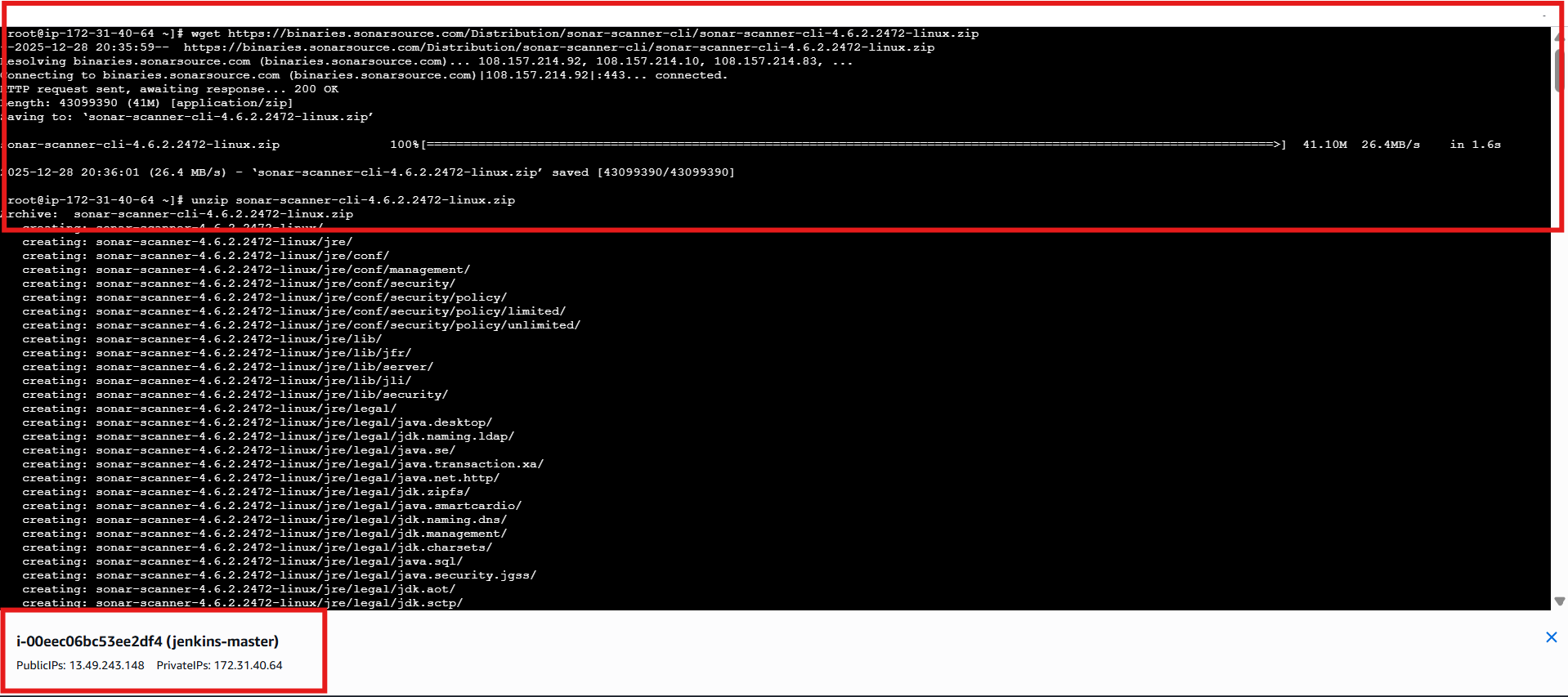


to build a communication between Jenkins and SonarQube we need to install sonar scanner

now we need to install sonar scanner in both GUI and in Jenkins CLI level.



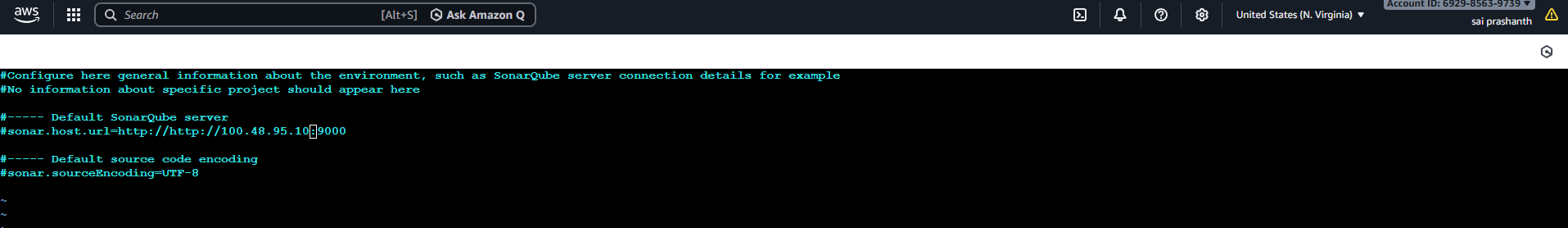




then after instalklation---to make configurationchanges

vi sonar-scanner.properties

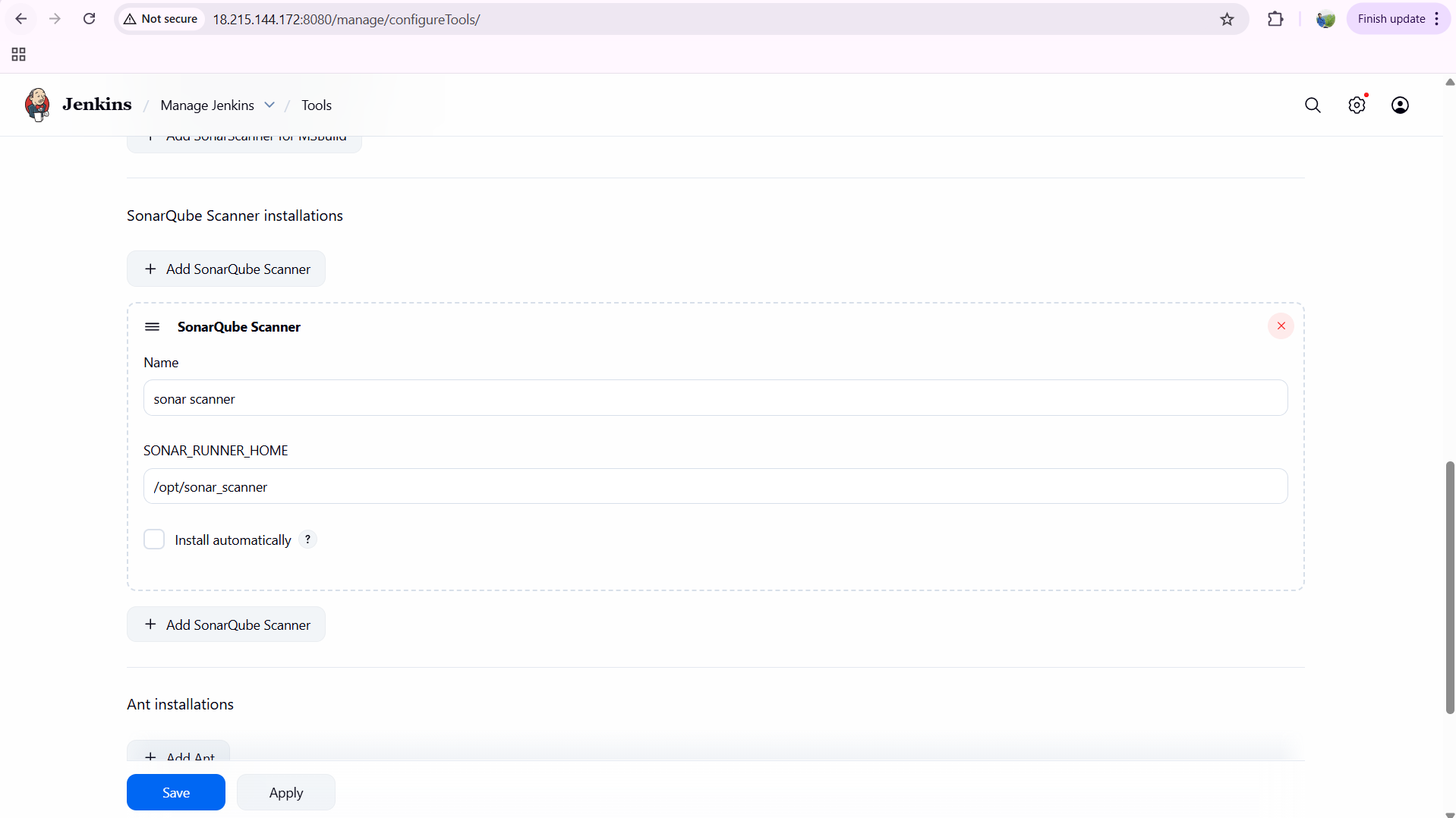
after making changes—save now



now to get the path of sonar scanner---cd /opt/Sonar scanner—cd bin/--ls—pwd

now copy the path

now to Jenkins dashboard---manage Jenkins---tools—SonarQube scanner—name—enter the path and save it



now go to configuration---build step---click on execute SonarQube scanner—and enter the commands as below

sonar.projectKey=Sabear

sonar.projectName=Sabear

sonar.projectVersion=1.0

sonar.sources=/var/lib/jenkins/workspace/$JOB\_NAME/src/

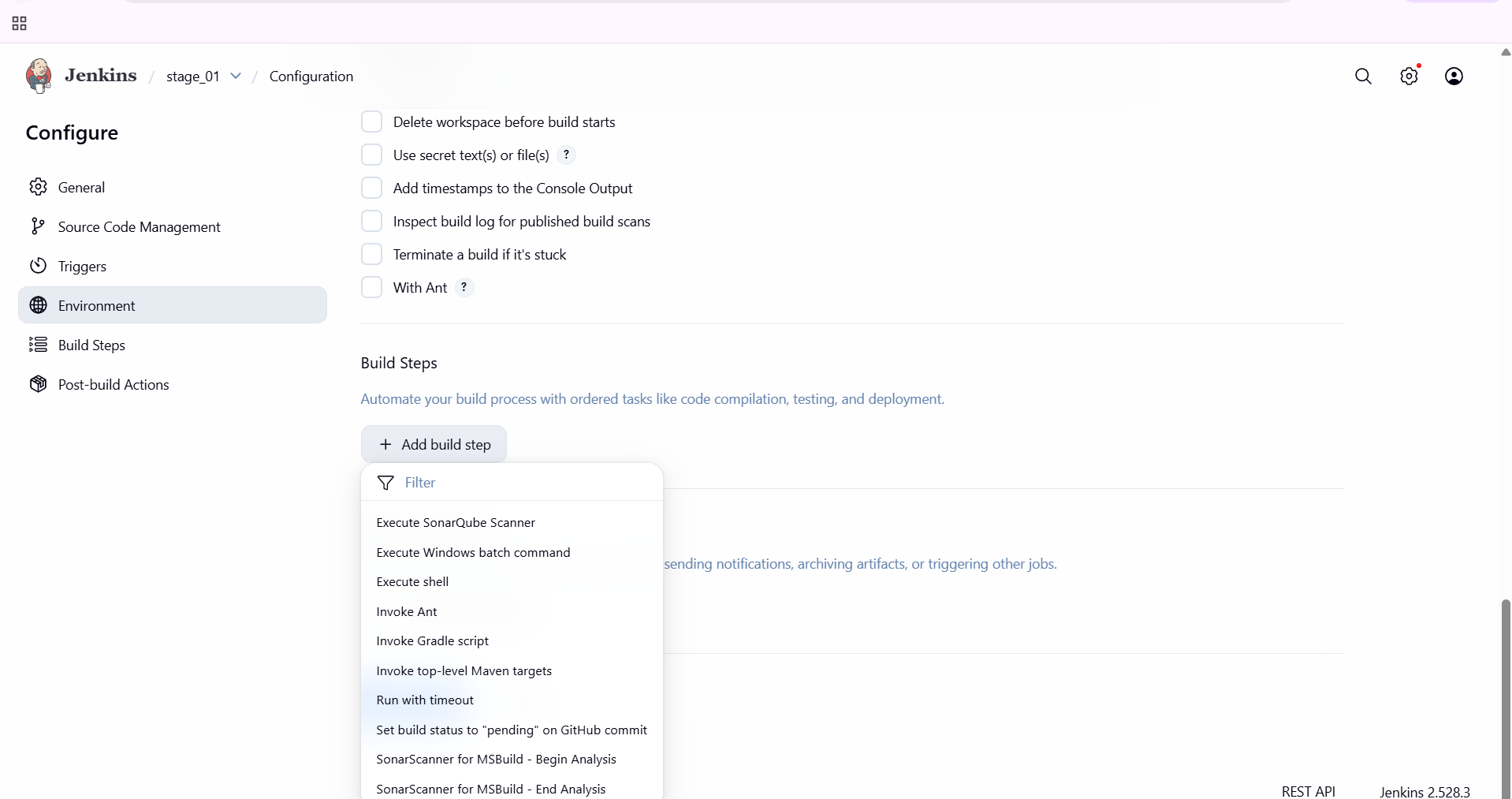
sonar.binaries=target/classes/com/visualpathit/account/controller/

sonar.junit.reportsPath=target/surefire-reports

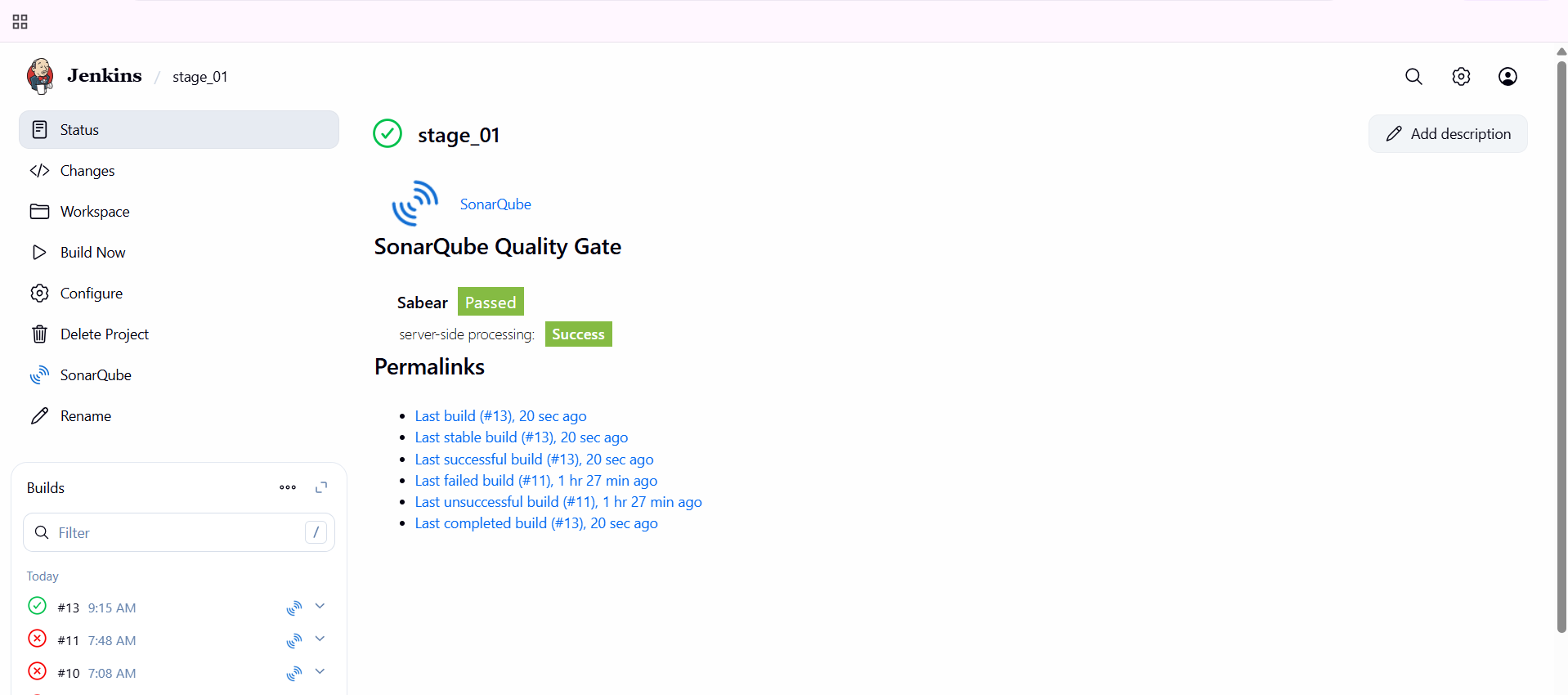
sonar.jacoco.reportPath=target/jacoco.exec

sonar.java.binaries=src/main/java/com/visualpathit/account/

and save it



now click on build now….it was passed and success



errors faced:

getting not success…as we need to select the instance type of m7 flex large for both instances..

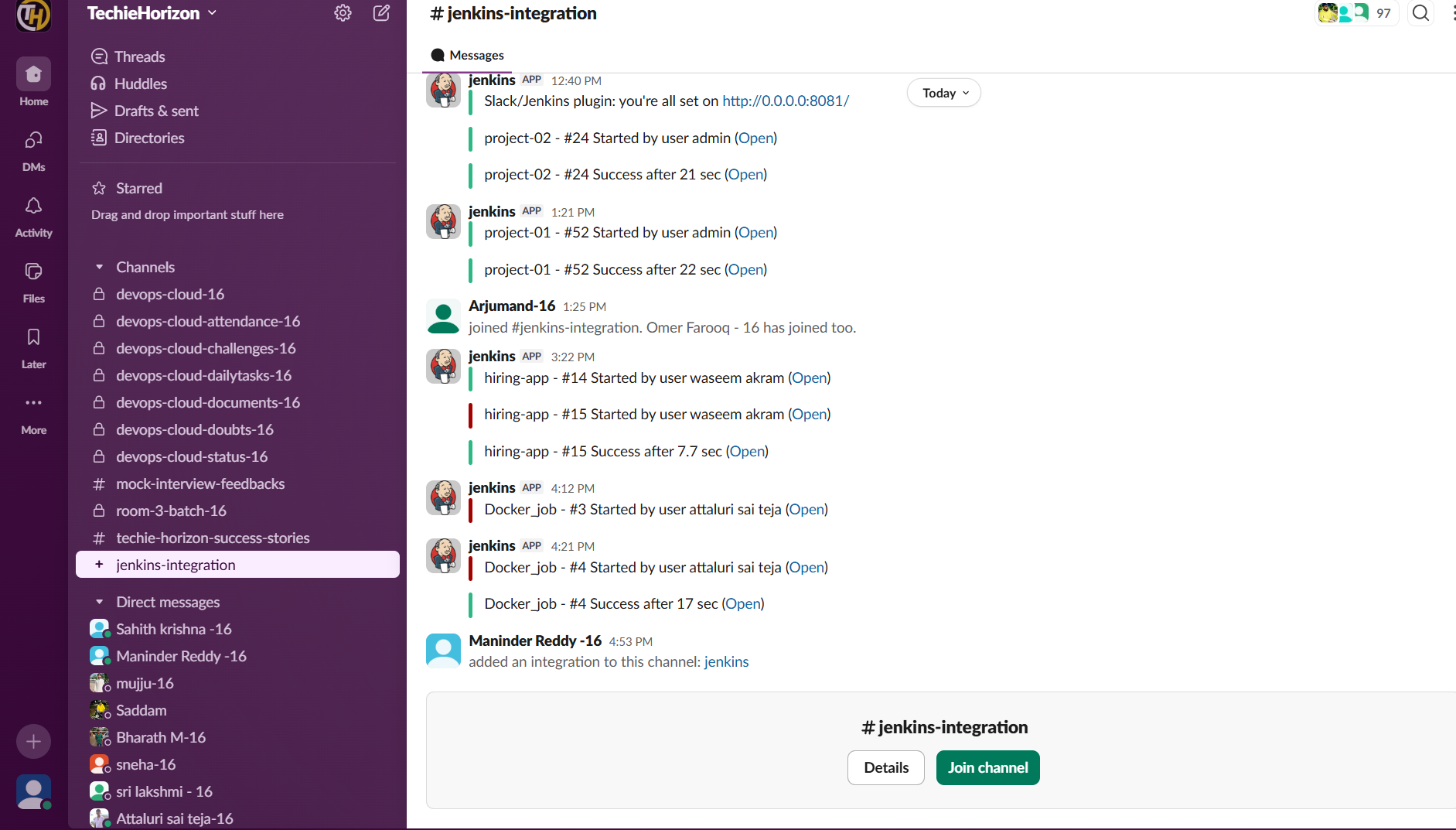
and check the path correctly

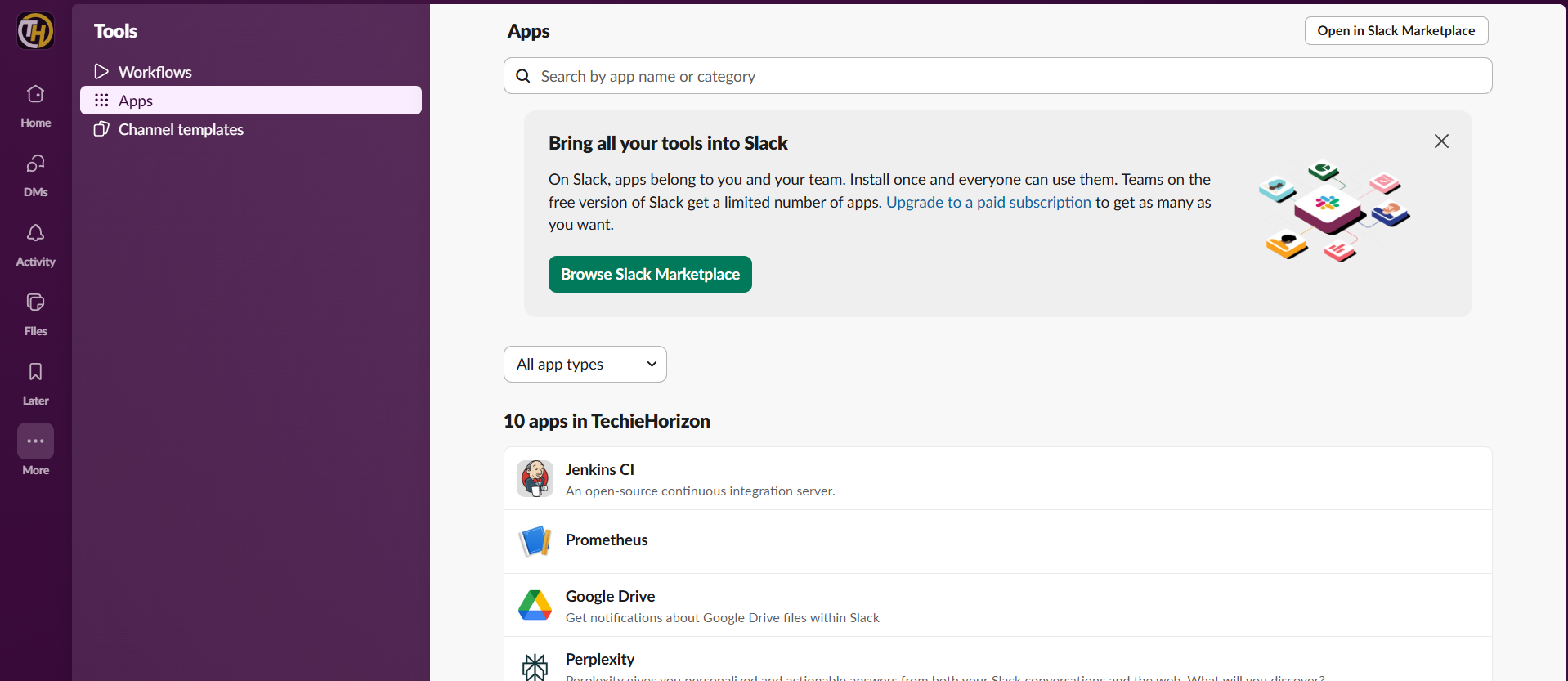
check the content what you have added in vi sonar-scanner

* **stage3**: Slack Integration to send the alerts to slack.

step by step process:

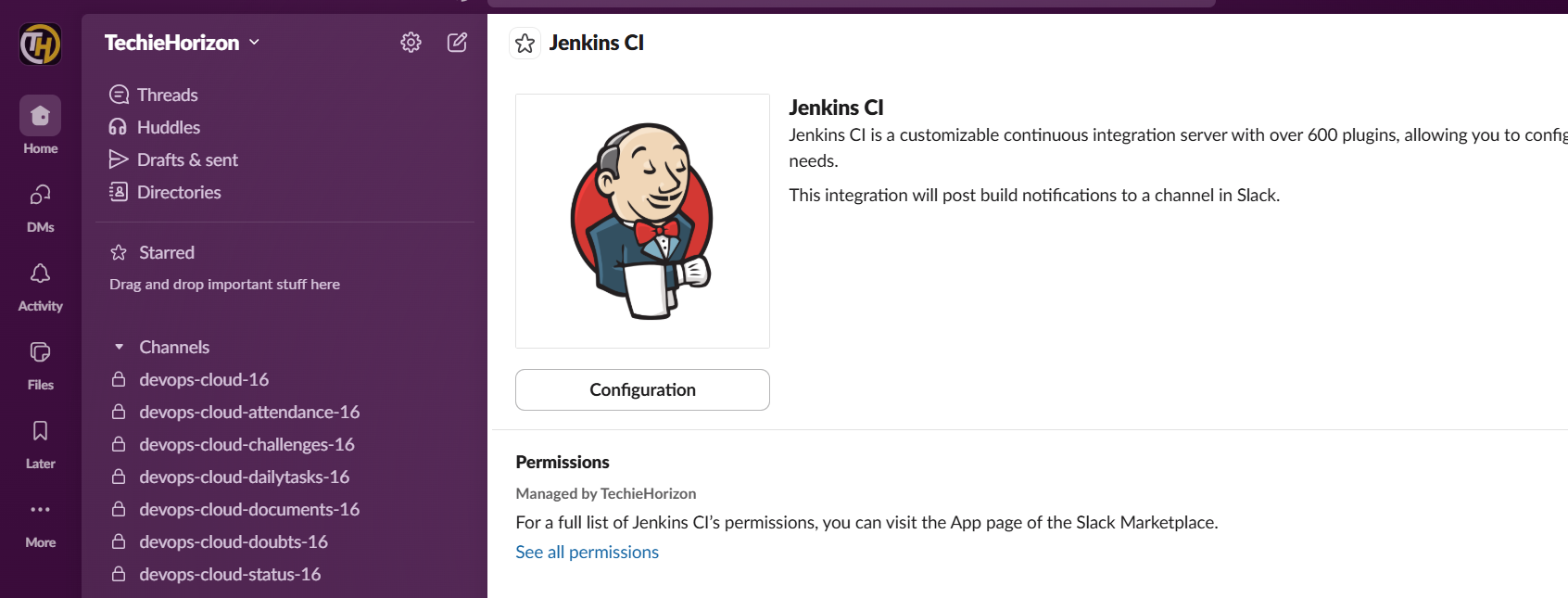
first go to your existing slack---apps---click on Jenkins CI

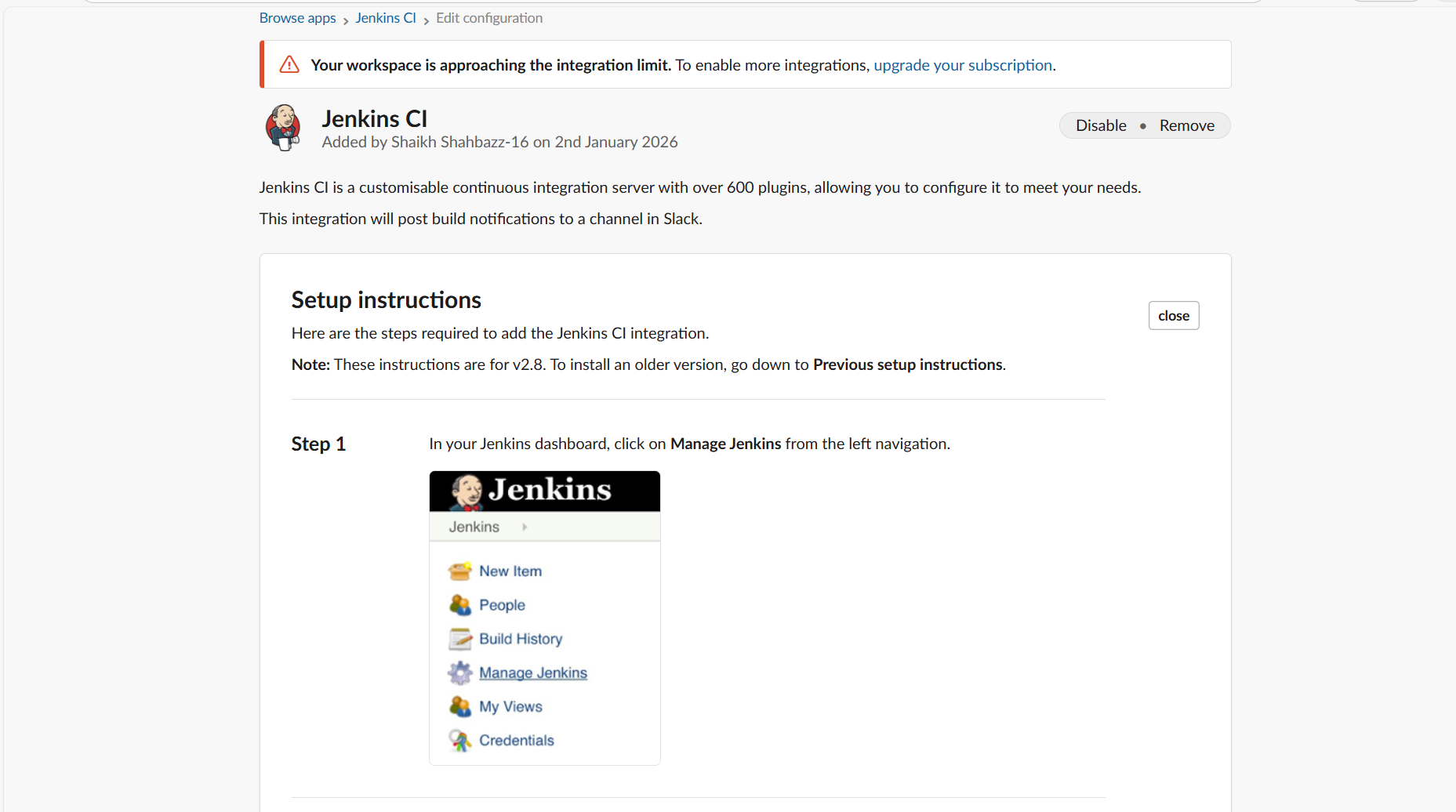


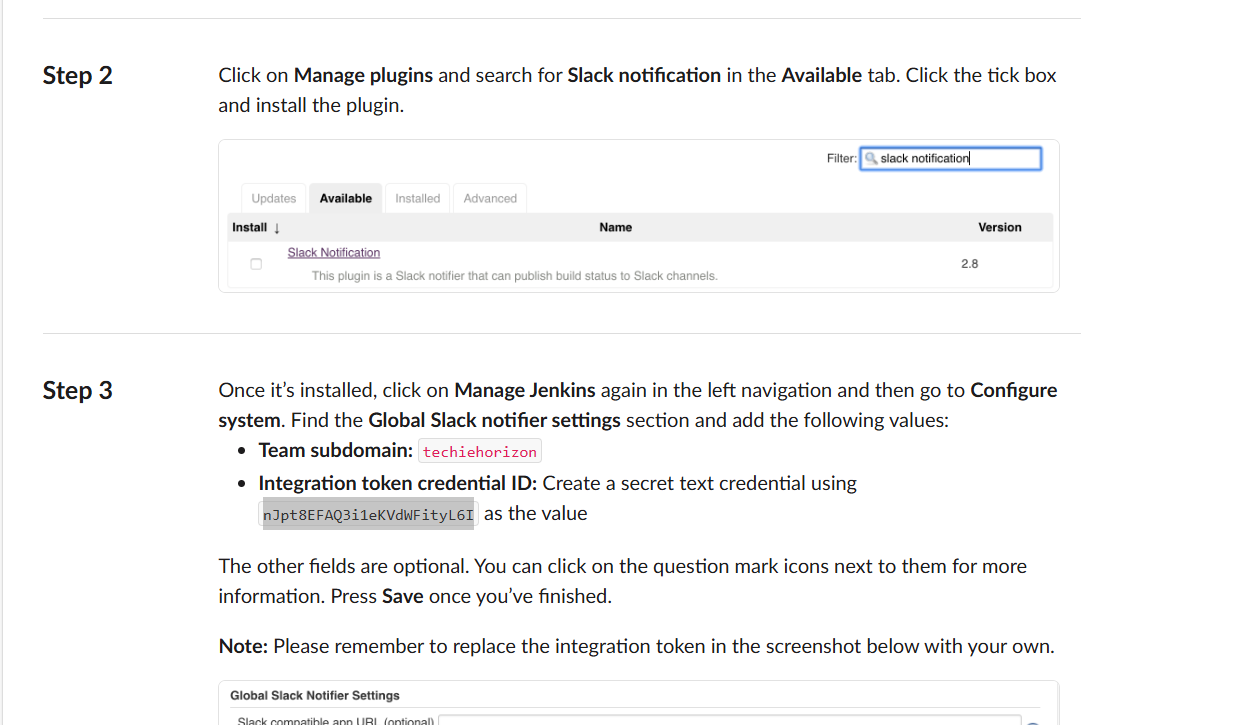


then go to configurations---edit configuration---enter your email and enter the code which you have been received

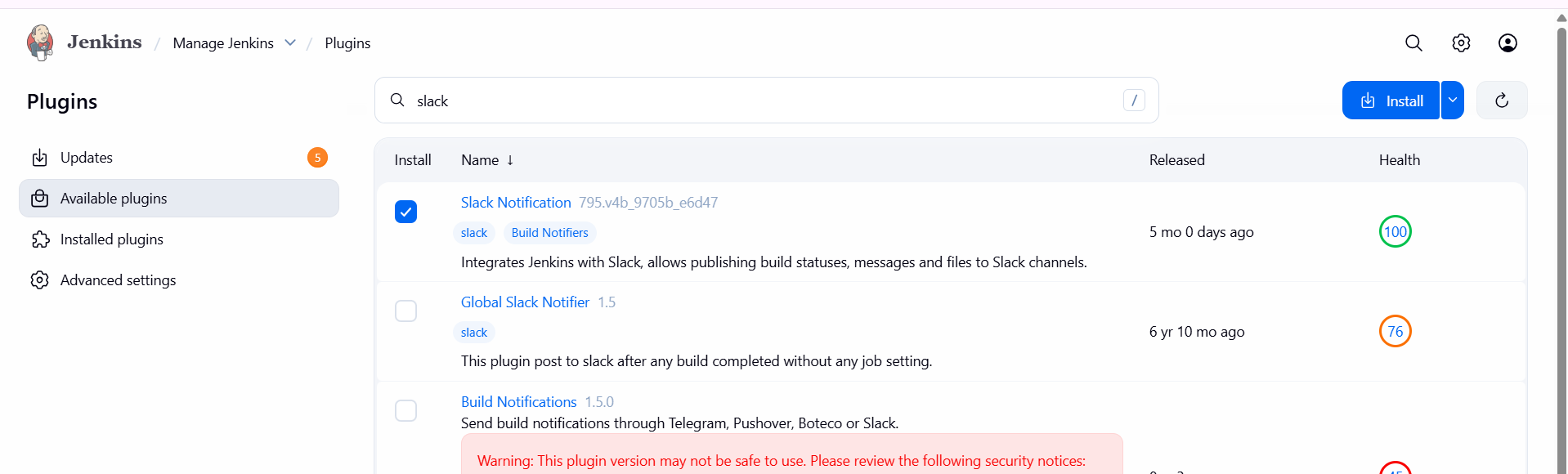
you can see different configuration steps.



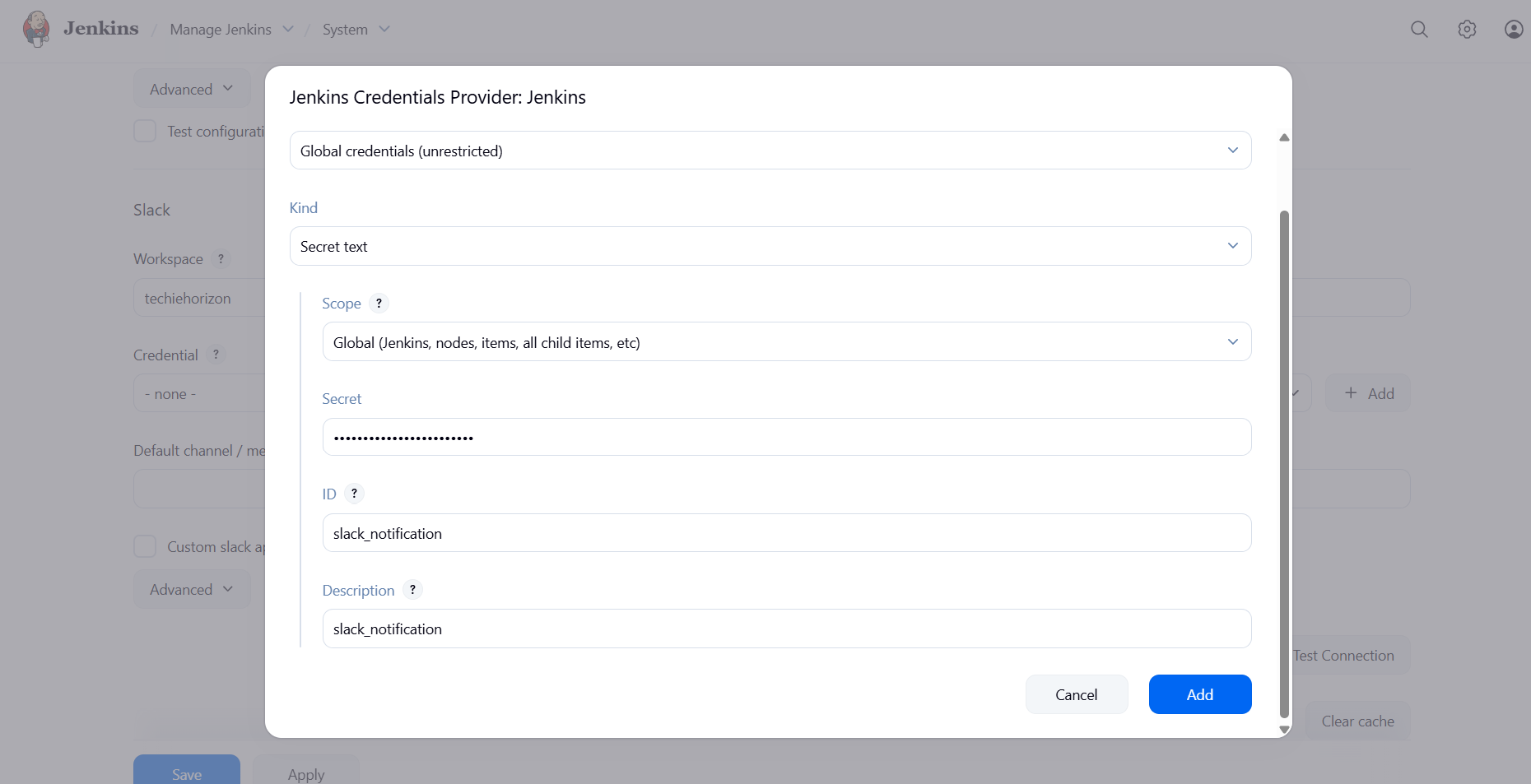




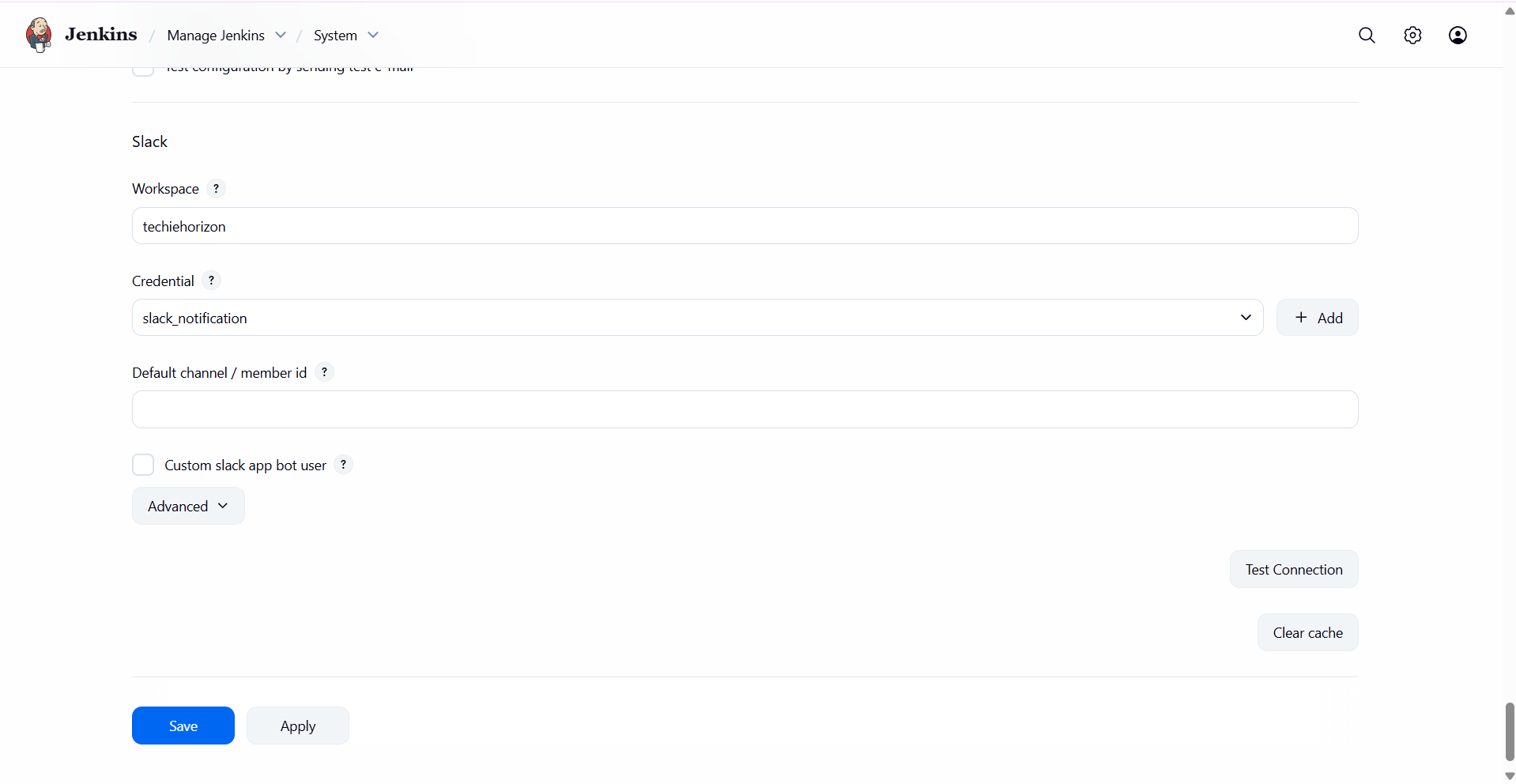
now go to Jenkins----manage Jenkins---install slack notification plugin



now go to system---click on slack---workspace name---techie horizon as mentioned in slack configuration and secret key—we need to copy that and paste it in credentials…and give one id name and add the credentials…



now click on test connection---save now

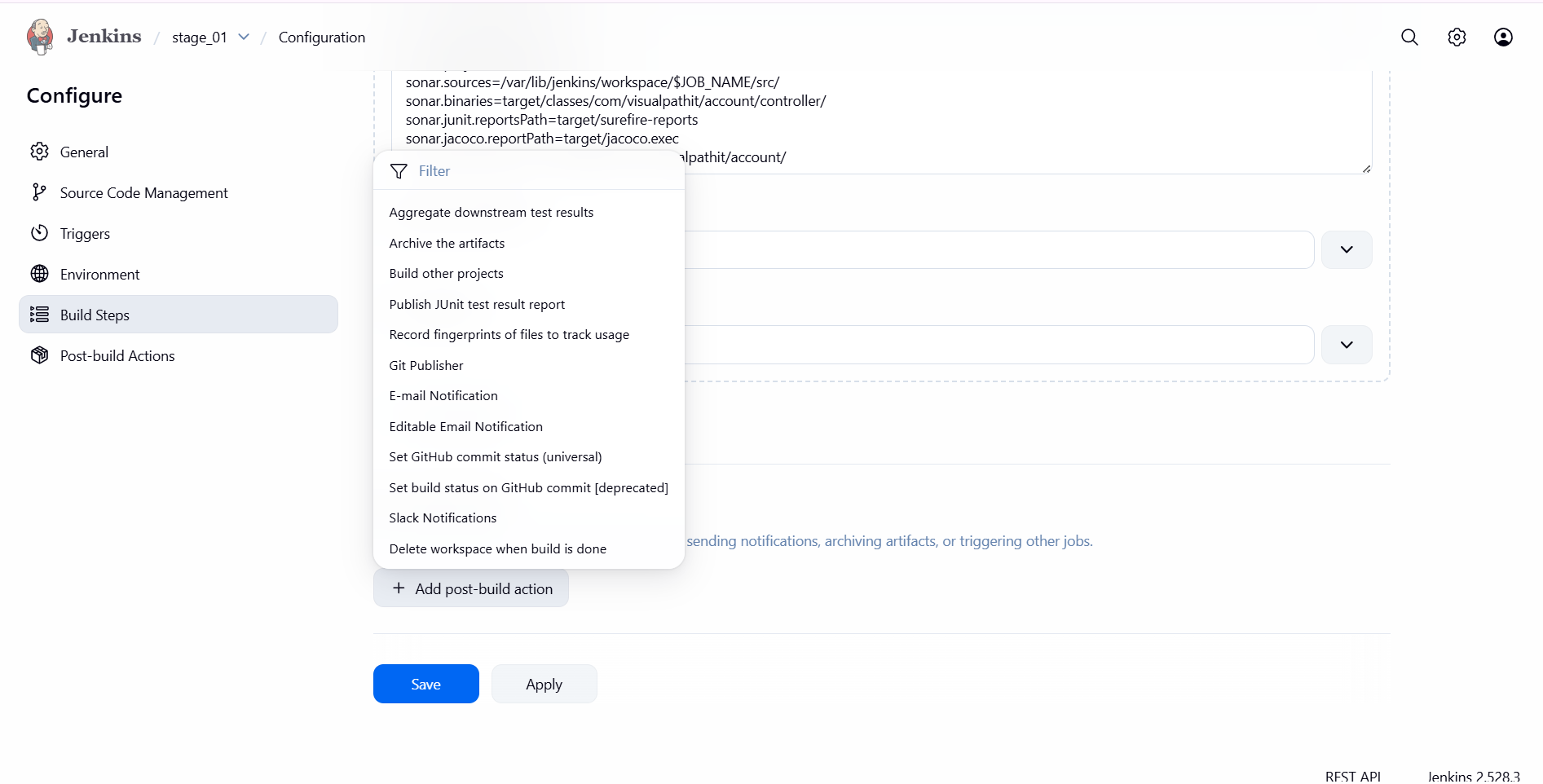


now go to your job---configure---post build actions---click on slack notification—and allow the permissions

notify build start

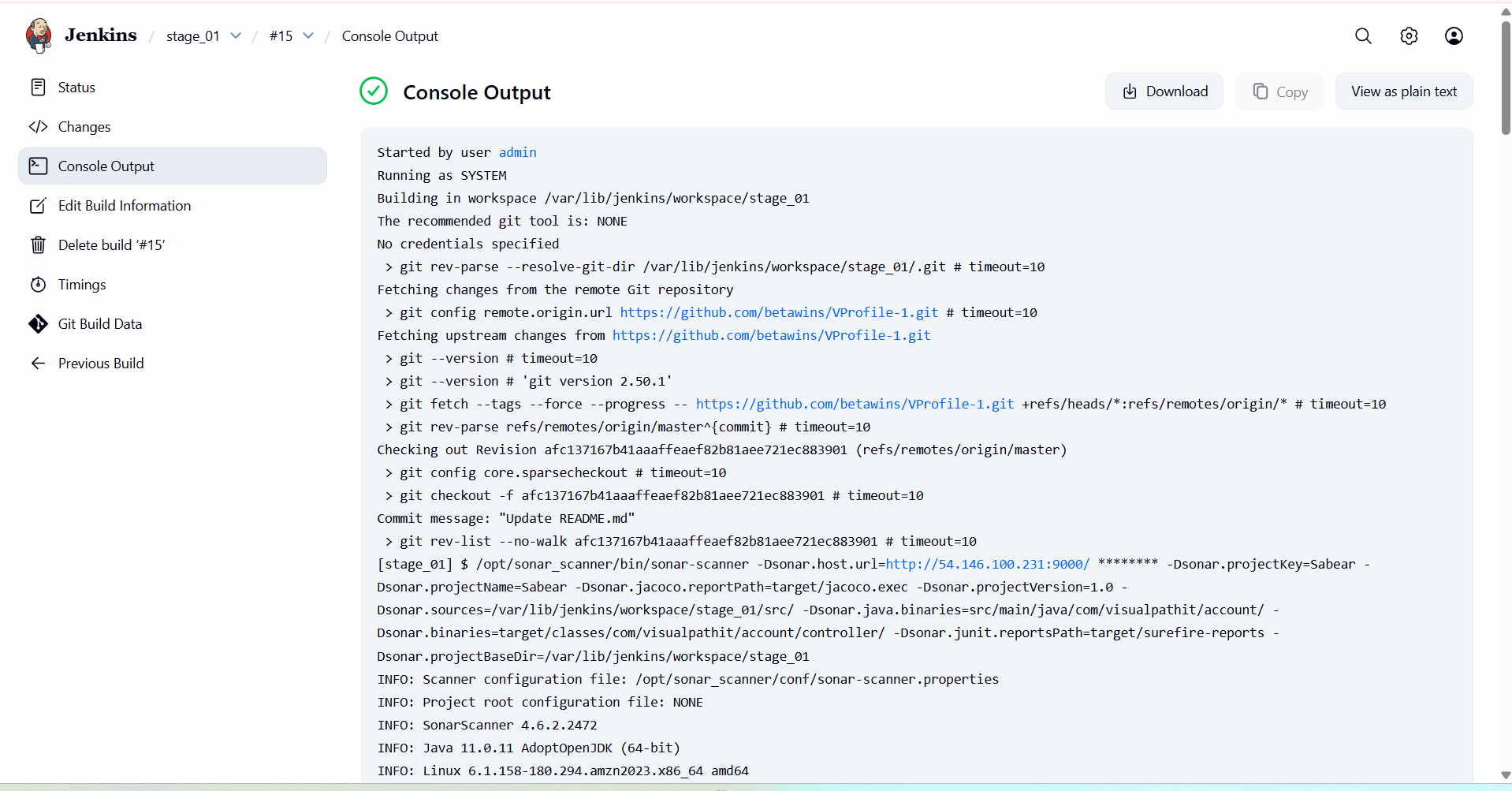
notify success

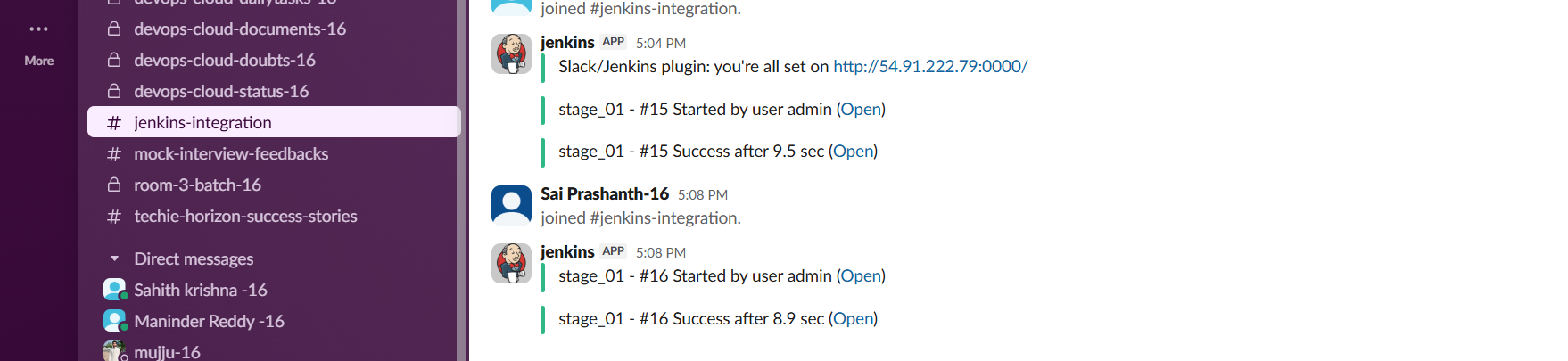
notify every failure and click on save





now you can verify the console output and check In your slack…





**Task 2**  
Create one Jenkins job using the below code and create three stages:

* **stage1**: Git clone to download the source code.
* **stage2**: SonarQube Integration to check the quality of code.
* **stage3**: Slack Integration to send the alerts to slack.

**URL**: <https://github.com/betawins/hiring-app.git>

**objective**: To create a Jenkins Pipeline job that:

1. Clones source code from a GitHub repository
2. Performs static code analysis using SonarQube to check code quality
3. Sends build status notifications to Slack

Repository used: <https://github.com/betawins/hiring-app.git>

**prerequisites**:

Jenkins server up and running

SonarQube server running and accessible

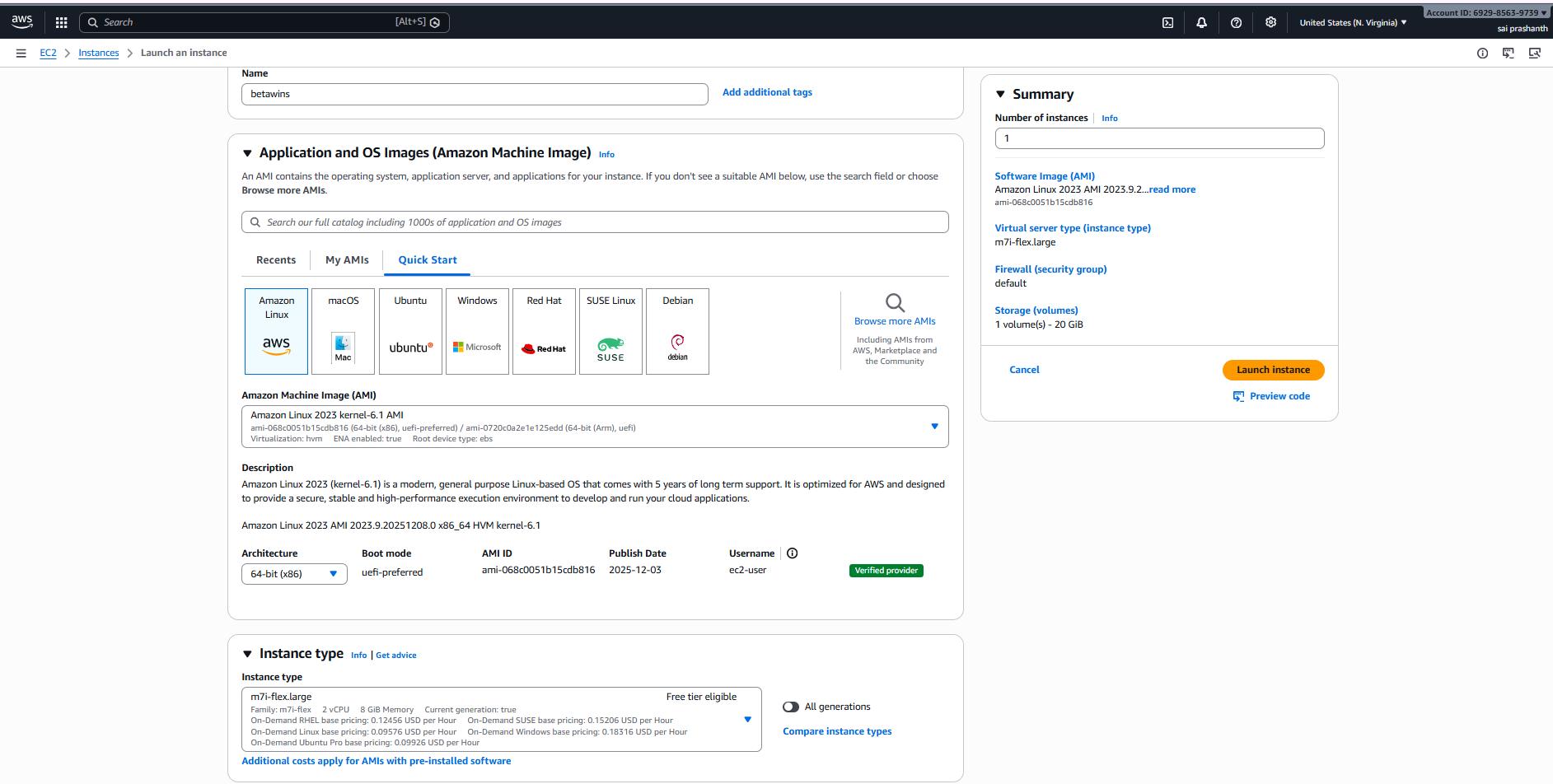
Java 17 configured in Jenkins

Sonar Scanner installed on Jenkins server

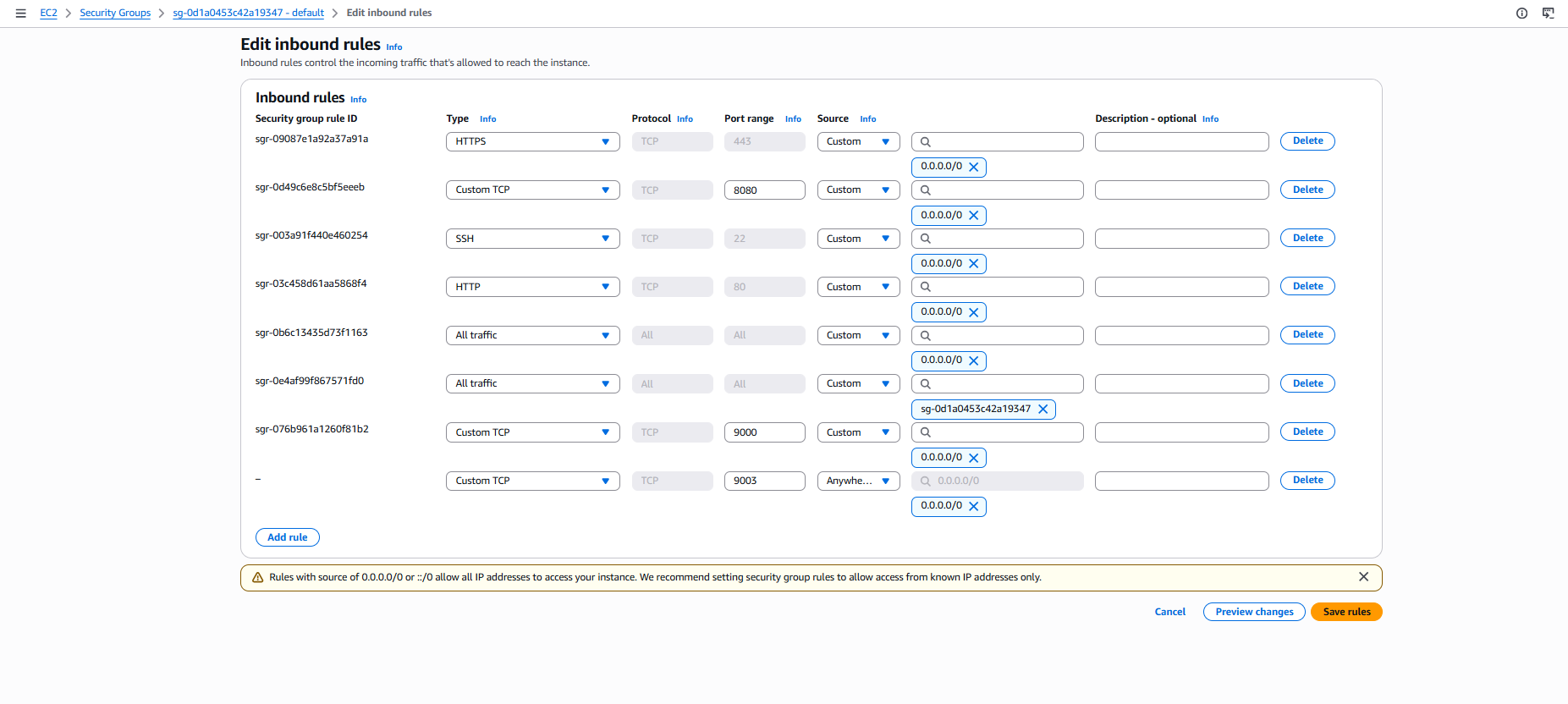
SonarQube token added in Jenkins credentials

**step by step process:**

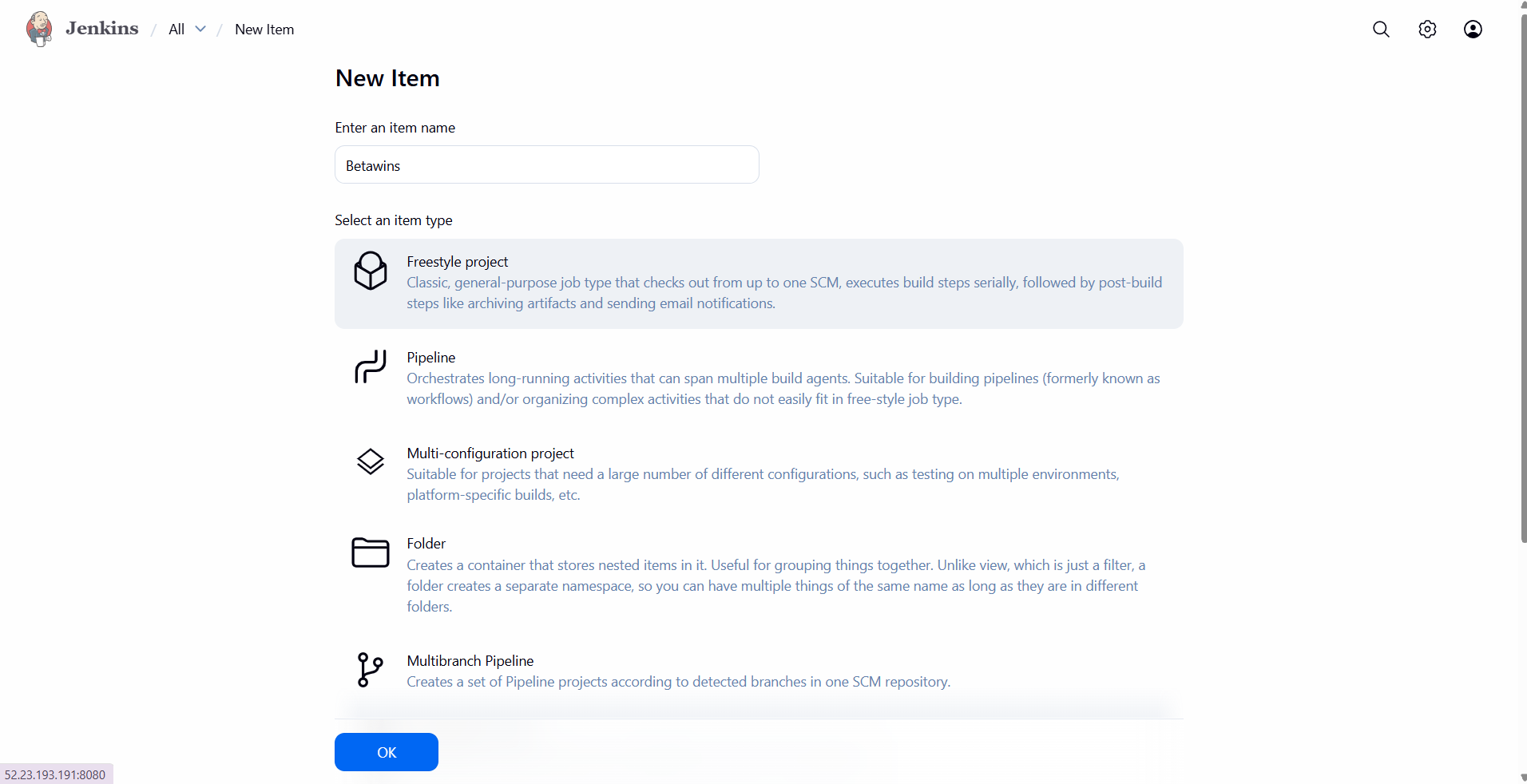
**install one ec2 with name betawins---instance type m7flex.large—launch instance**



here we are using docker container…so we need to give the inbound rule 9003 in security groups…



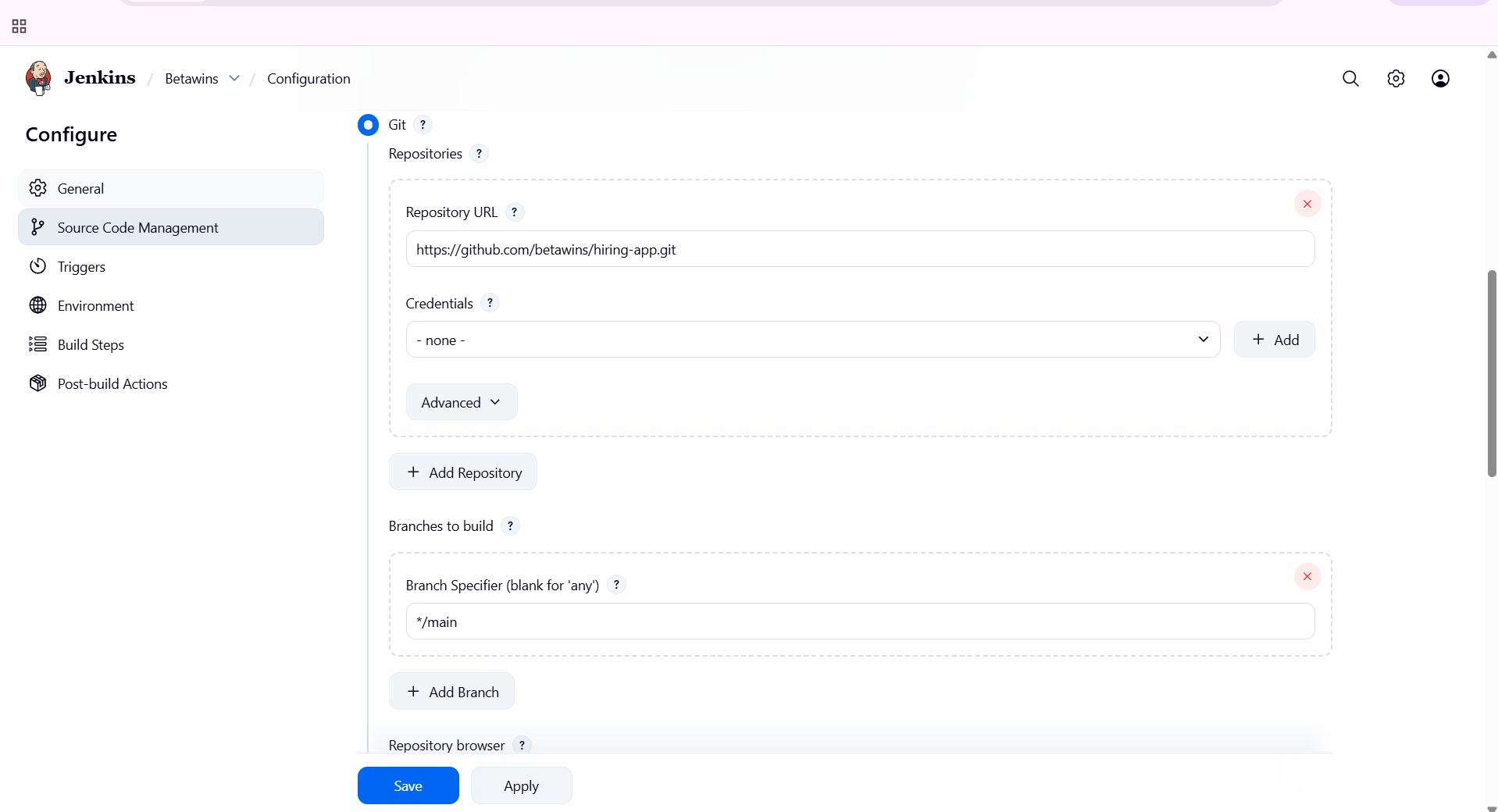
now come to Jenkins dashboard---mew item---betawins name---freestyle project—click on ok



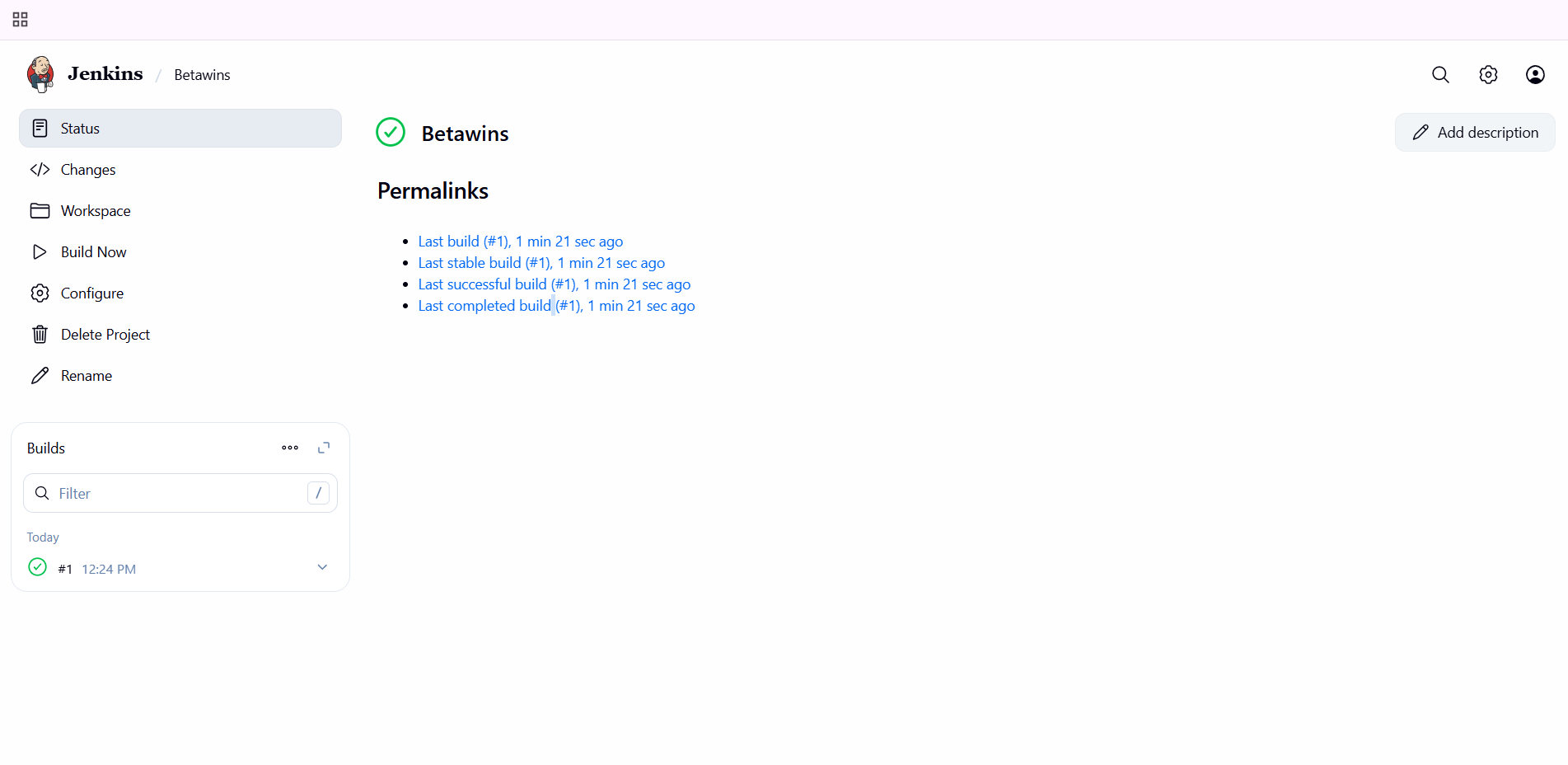
now go to configuration---source code management---select git—add repository URL <https://github.com/betawins/hiring-app.git>

edit the branch name as main…

click on save



now the job has been configured…you can see in console output

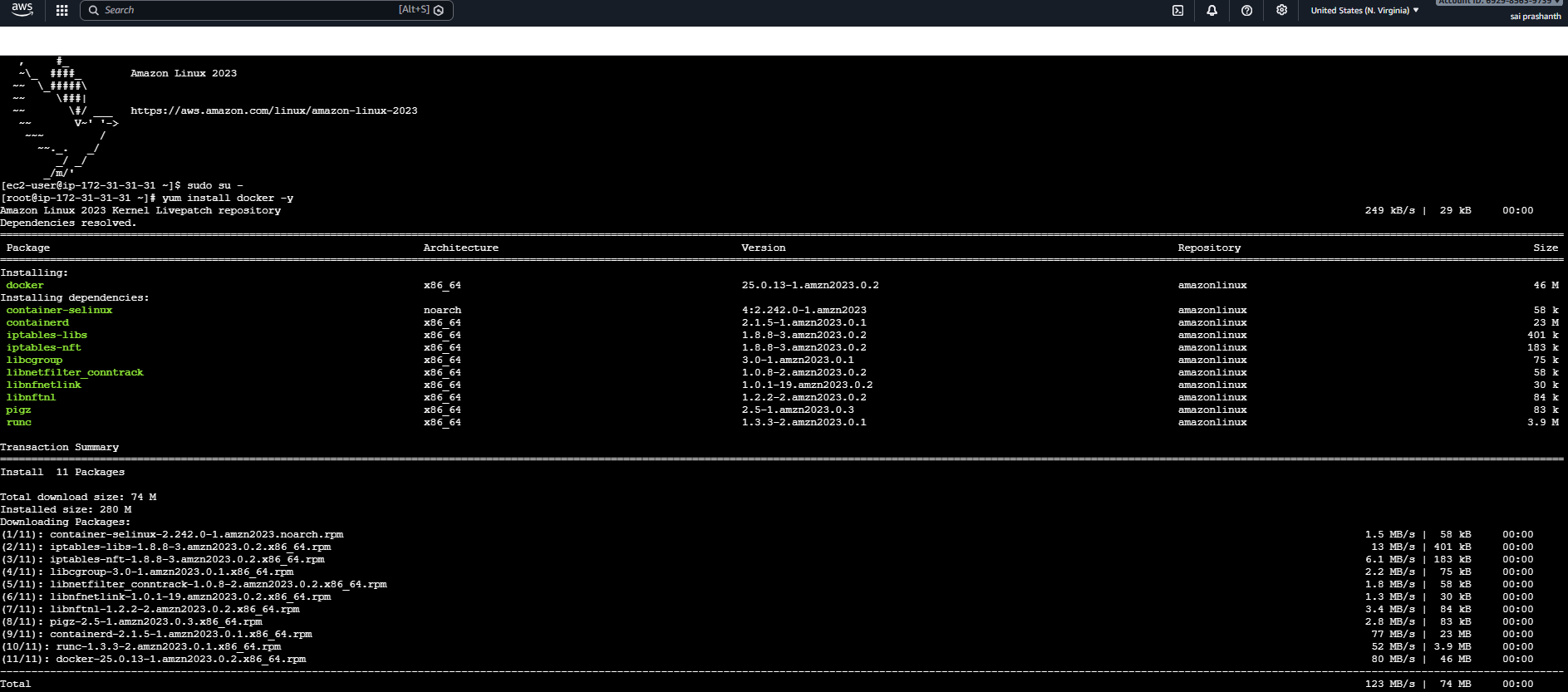


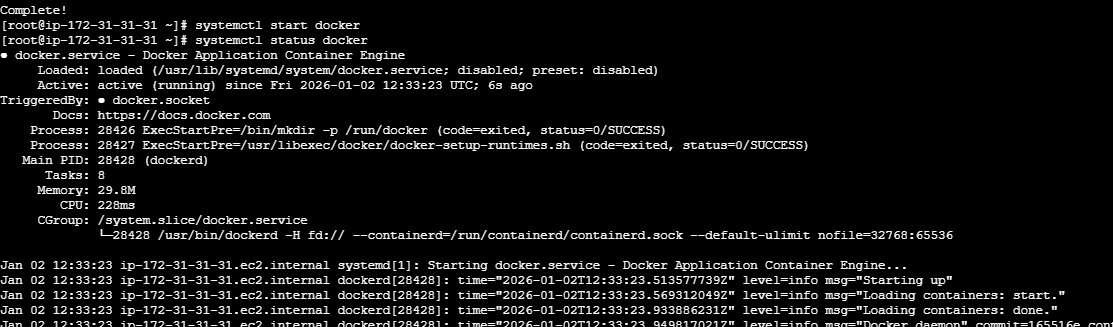
now connect the instance we need to install docker…

yum install docker -y

systemctl start docker

systemctl status docker



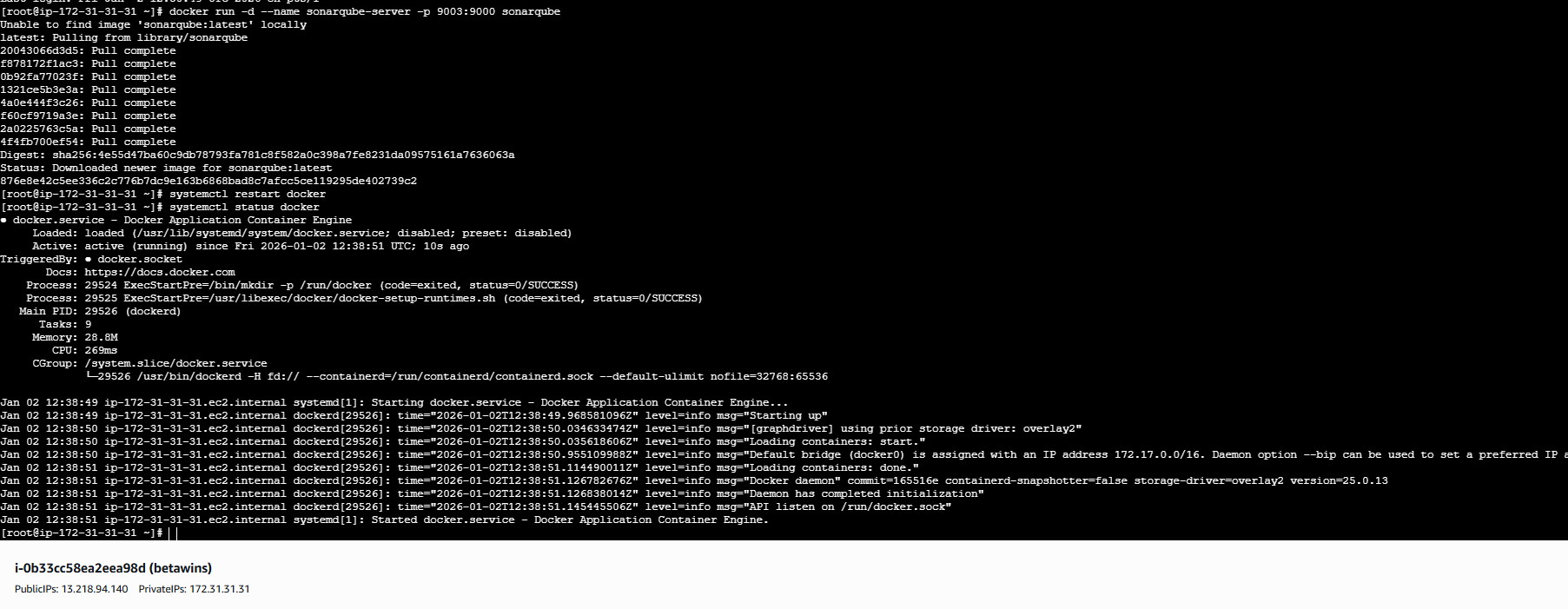


now to run the image from container we need to use the command…

docker run -d --name sonarqube-server -p 9003:9000 sonarqube

systemctl restart docker

systemctl status docker



to check the list of containers running—use the command—docker ps

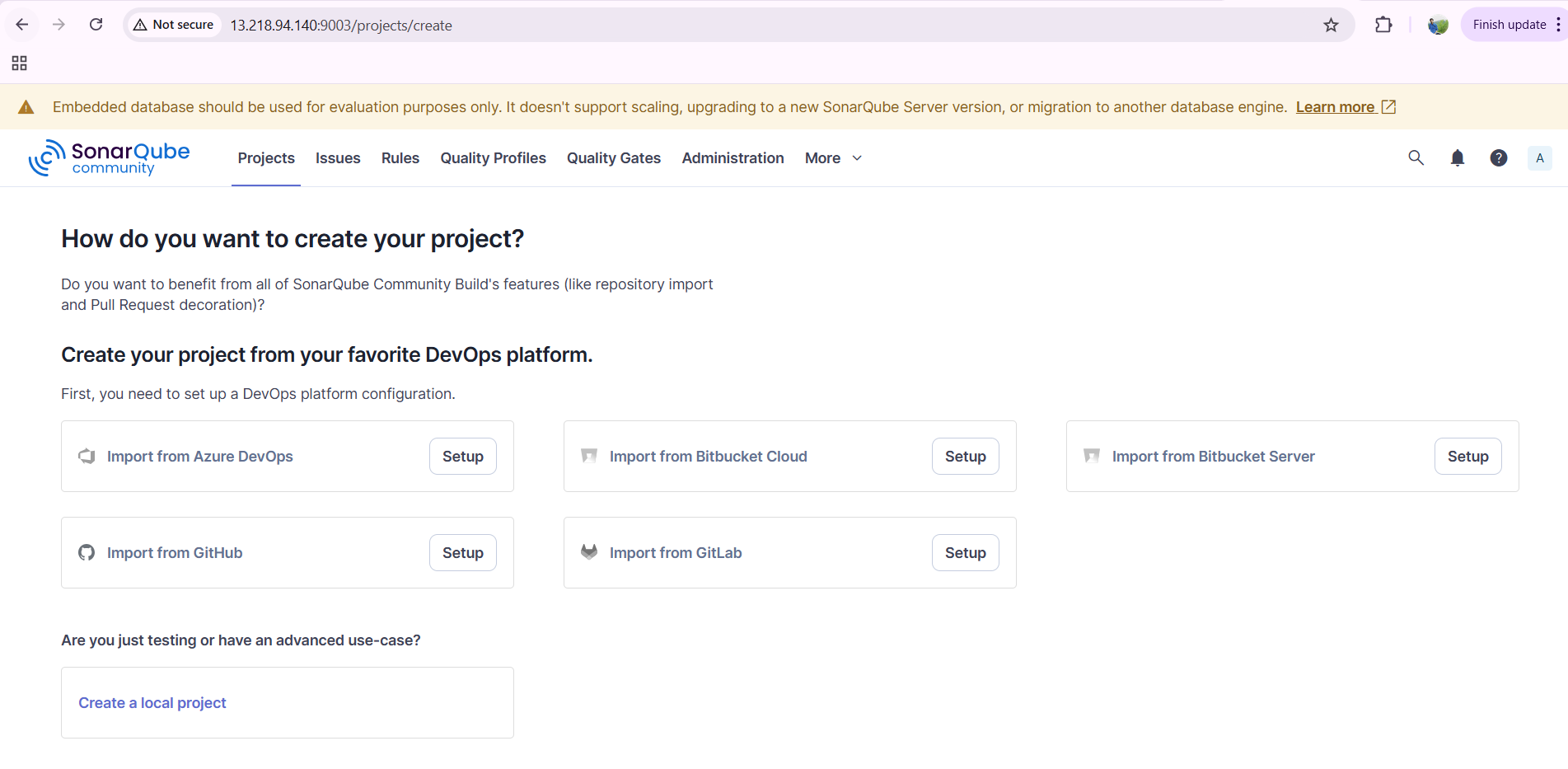
now we need to pull---docker pull sonarqube

now again check the status…systemctl status docker

now we need to start sonarqube…use the command—docker start sonarqube-server

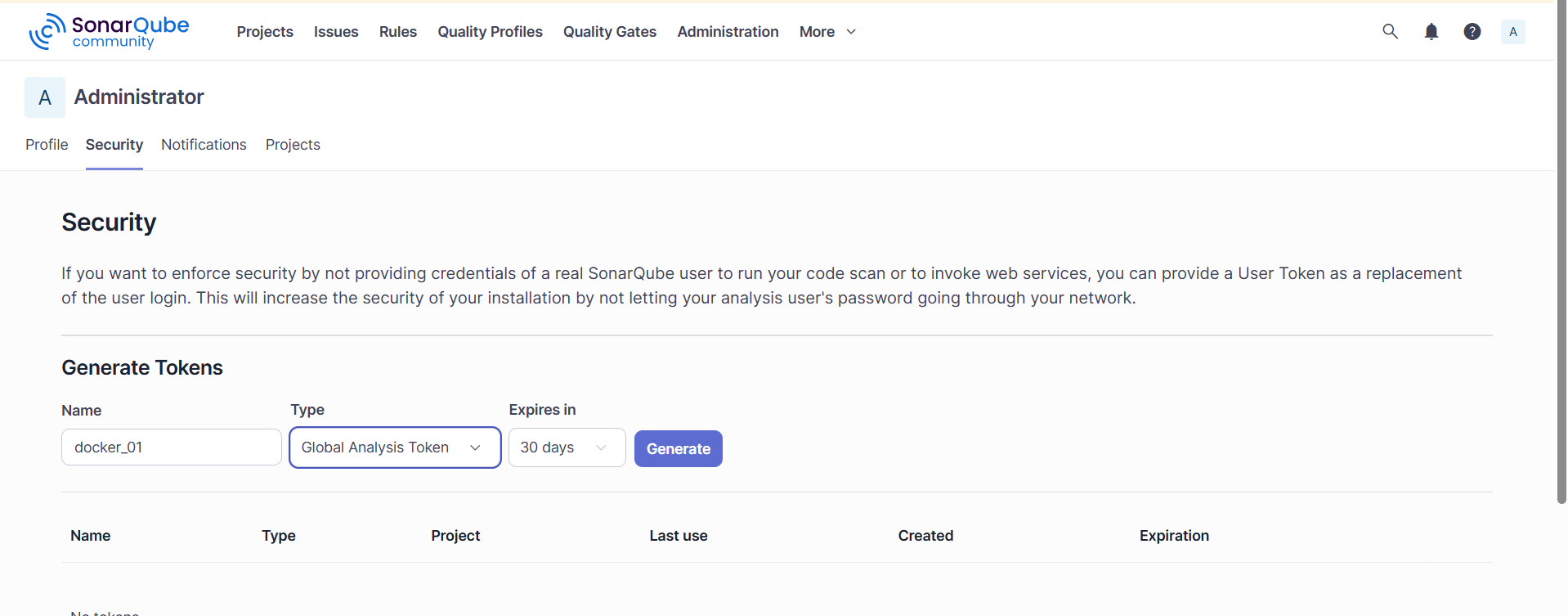


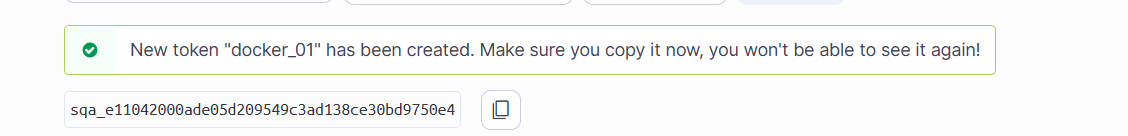
now check in browser---public ip with poer 9003 you can see the sonarqube page has been opened…



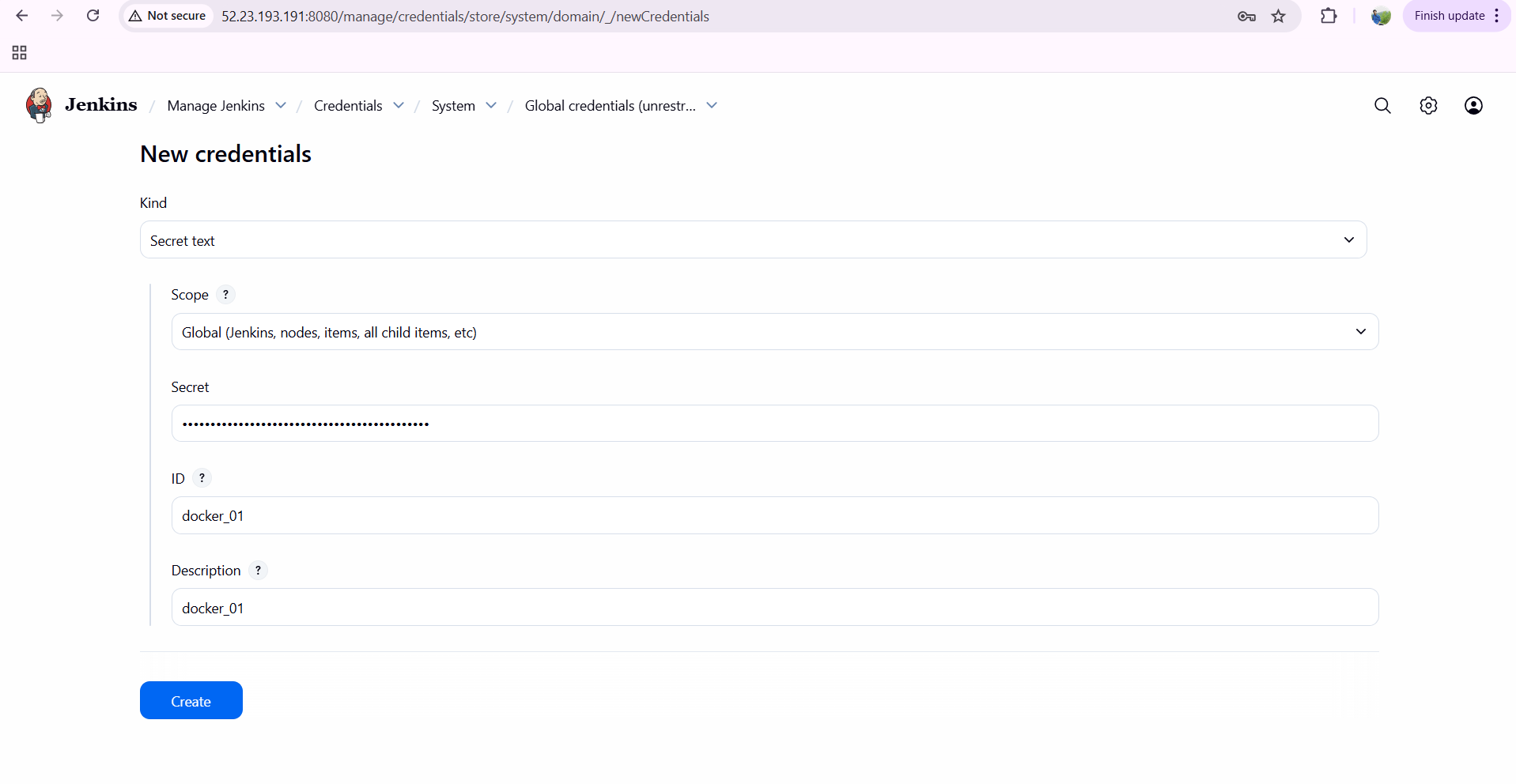
create one token—administrator—security—name—and generate

token has been generated now

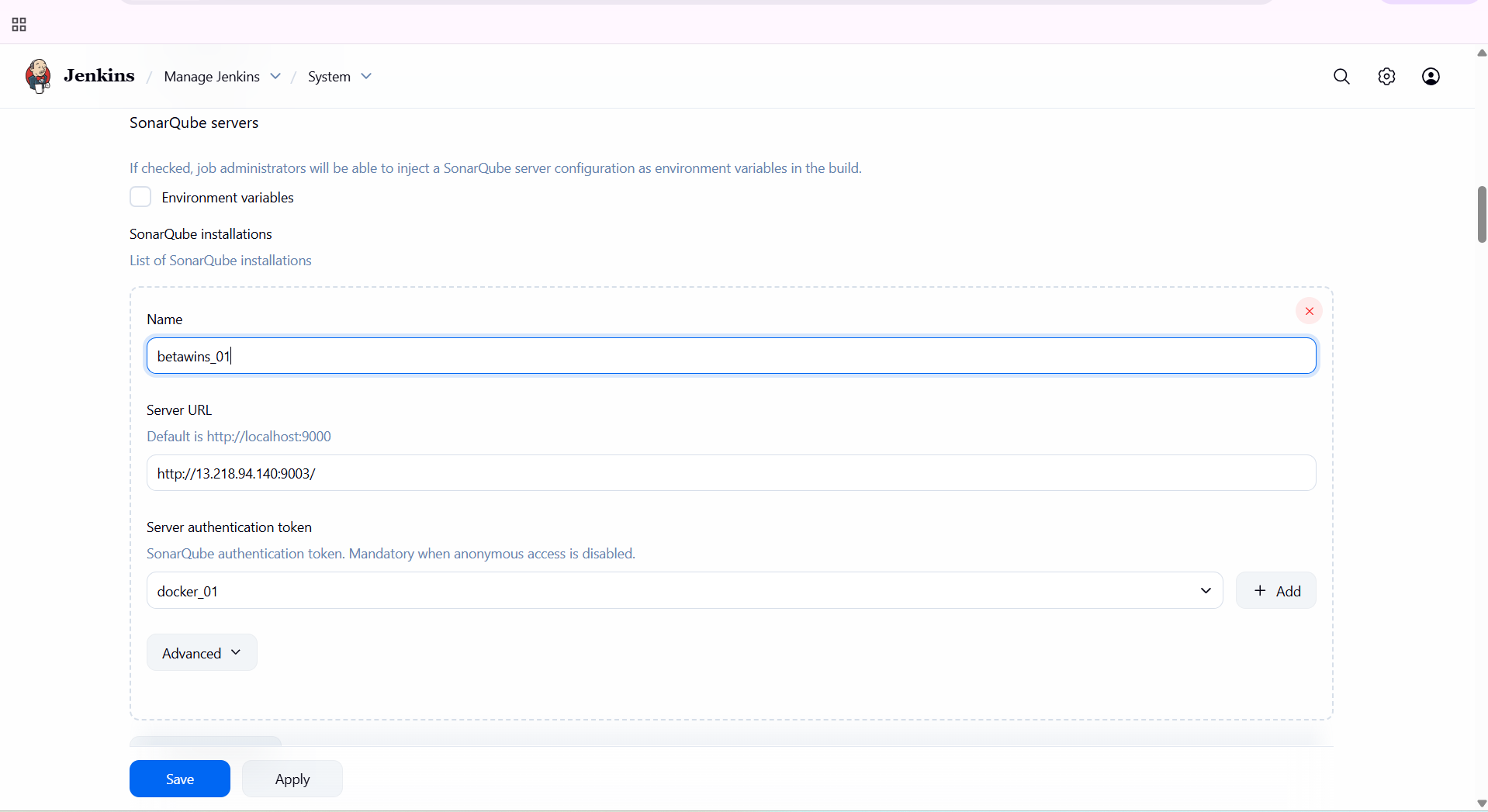




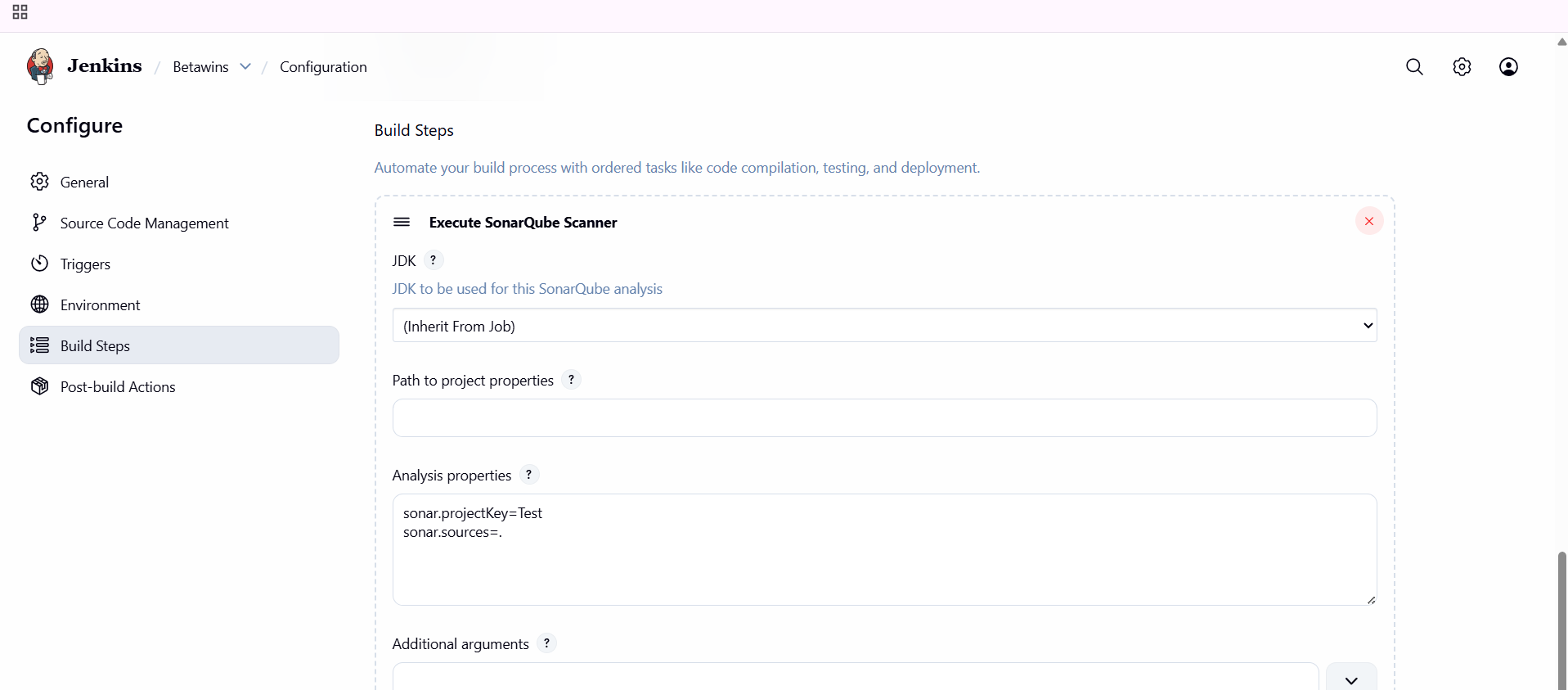
now come to Jenkins---manage Jenkins—credentials—secret text—enter the token in secret tab—add open id and description name and save it.



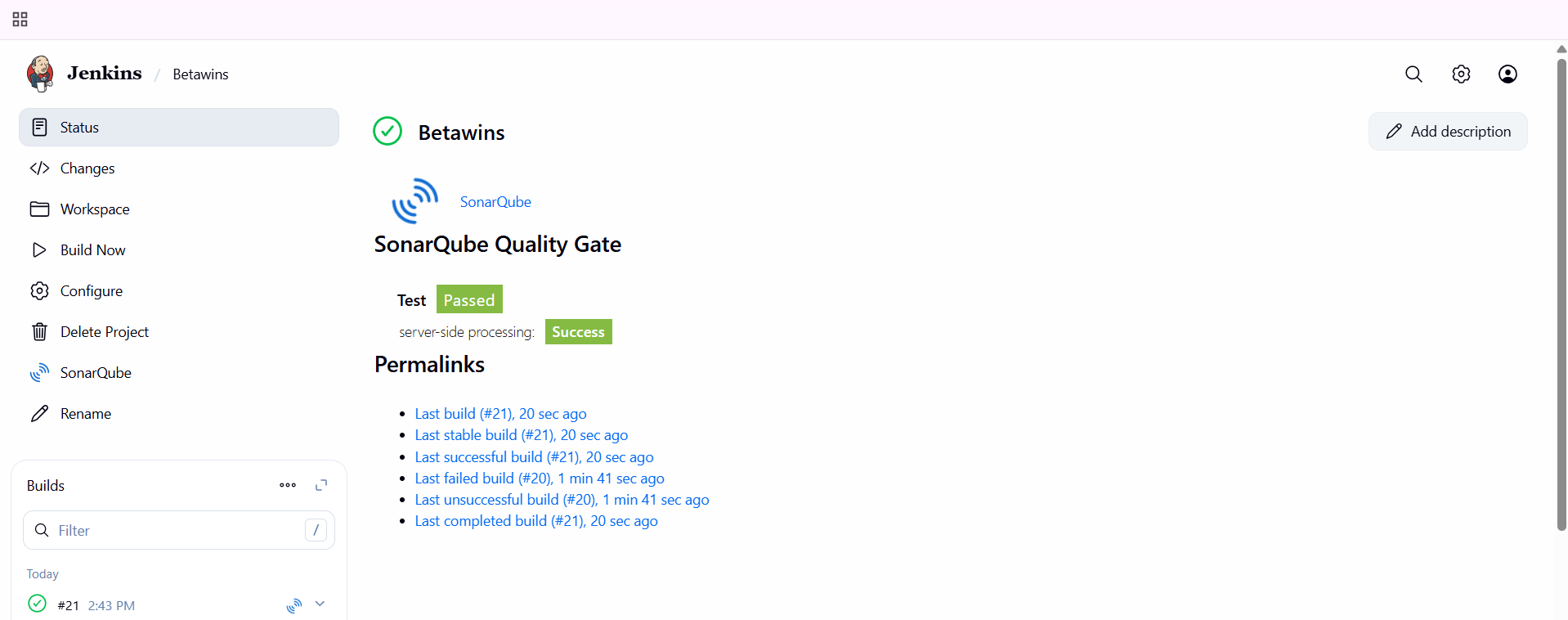
now go to system---SonarQube installations—add the URL of server and select the token and click on save..



now click on your job—configure—build steps—in analysis properties—add from repository

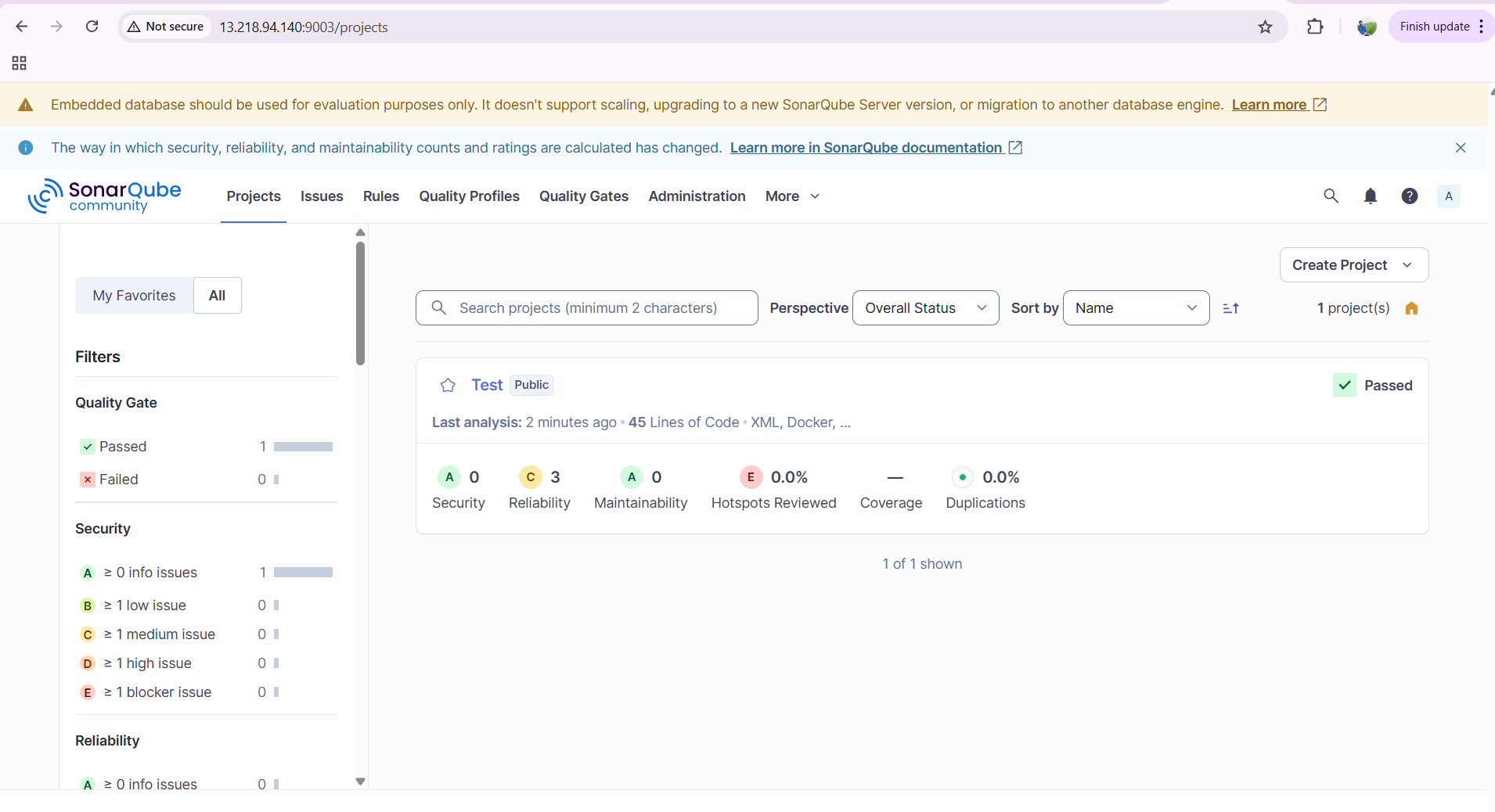


once it has been done…click on build now and you can see task has been success

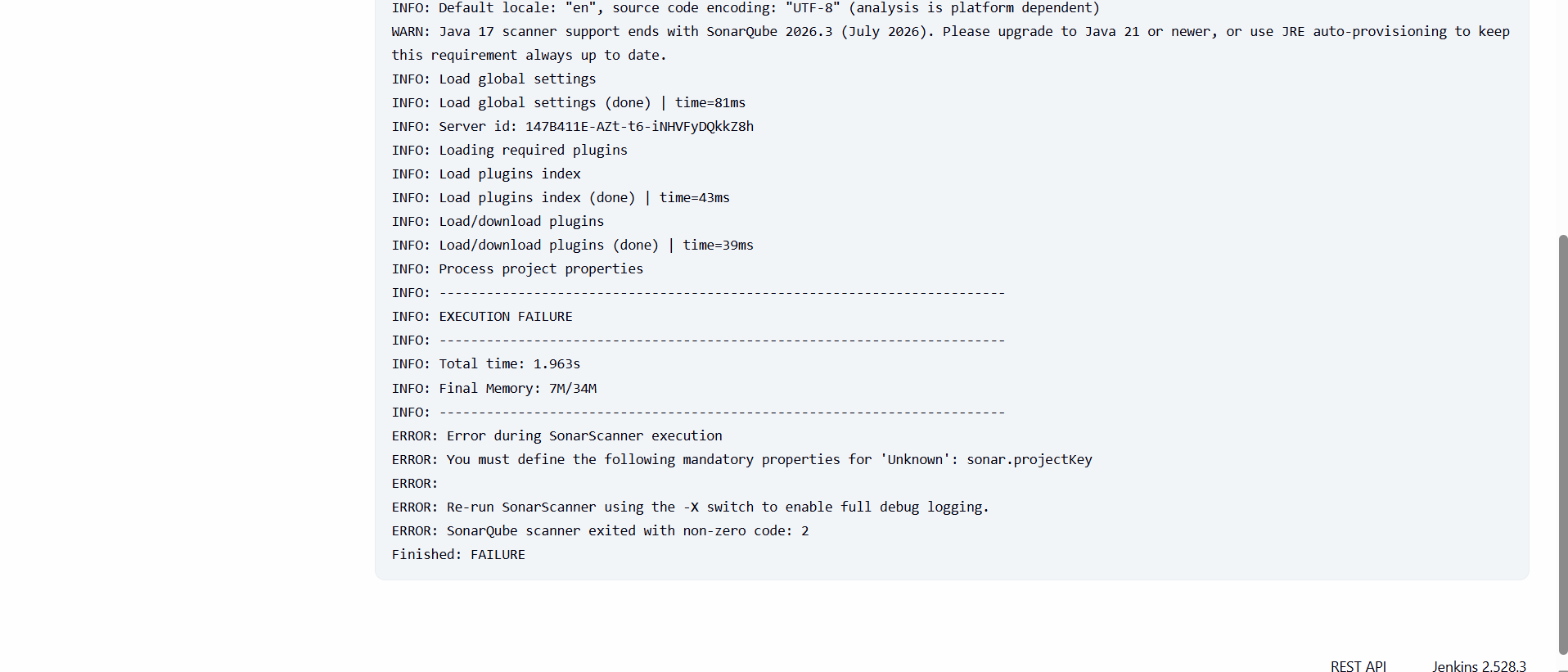


**verification steps:**

refresh the SonarQube page now and verify the project has been created



errors faced:



issue faced with sonar scanner versions….

