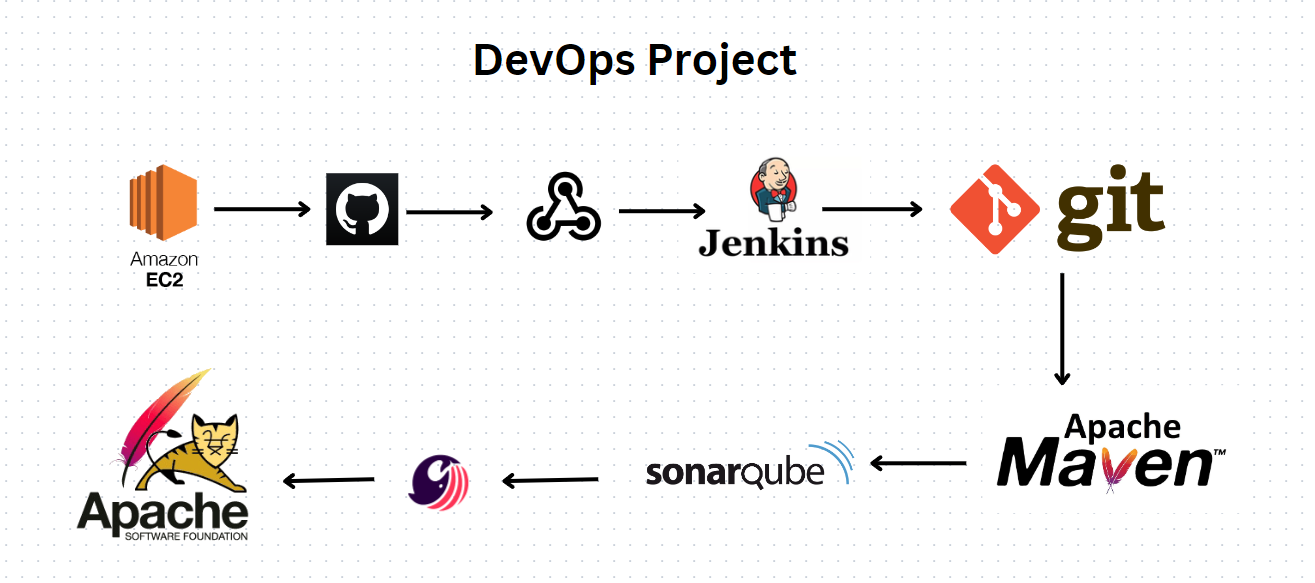
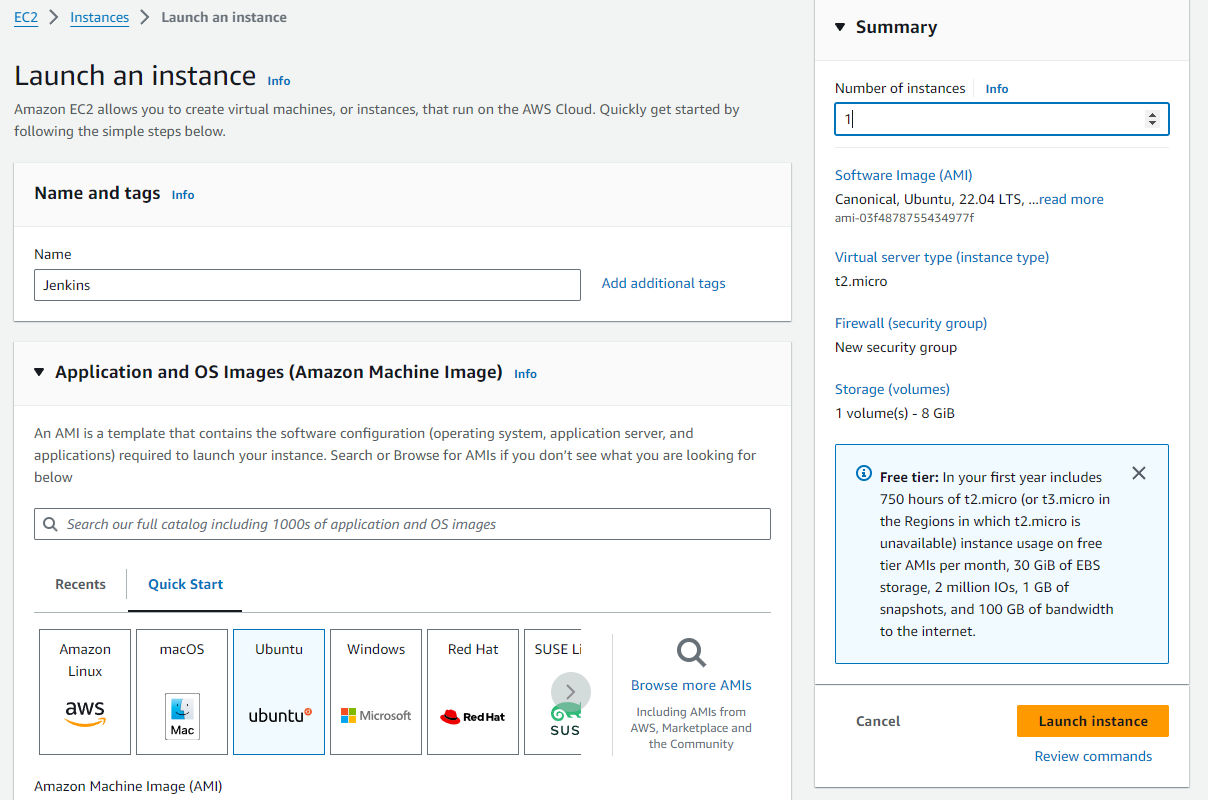
**DevOps Project: Deploying student registration page on apache tomcat.**



Git Repo URL: https://github.com/aakashbshendage/Student\_App.git

Step 1: Launch EC2 instance for Jenkins using t2.micro.



Step 2: Connect EC2 using SSH.

Step 3: Run Command to update a system: **sudo apt update**

Step 4: Install JDK:

Cmd: **sudo apt install openjdk-11-jre-headless -y**

Step 4: Download and Install Jenkins

**sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \**

**https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key**

**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**

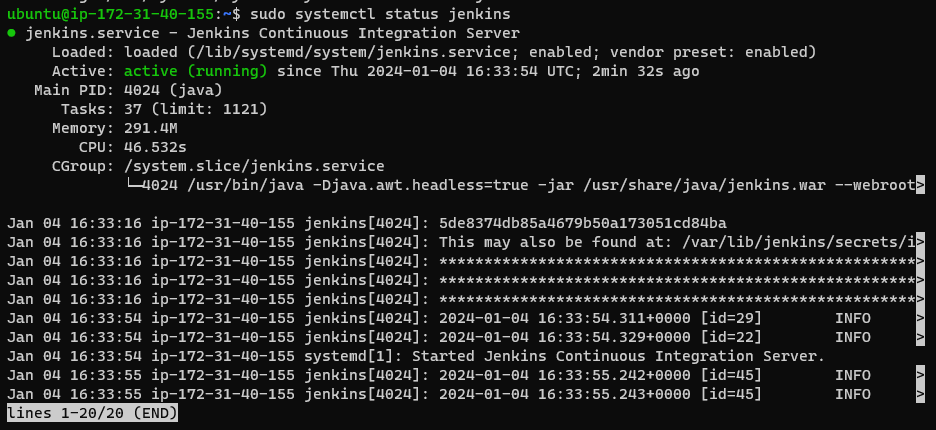
**sudo apt-get install Jenkins -y**

Step 5: Start Jenkins service.

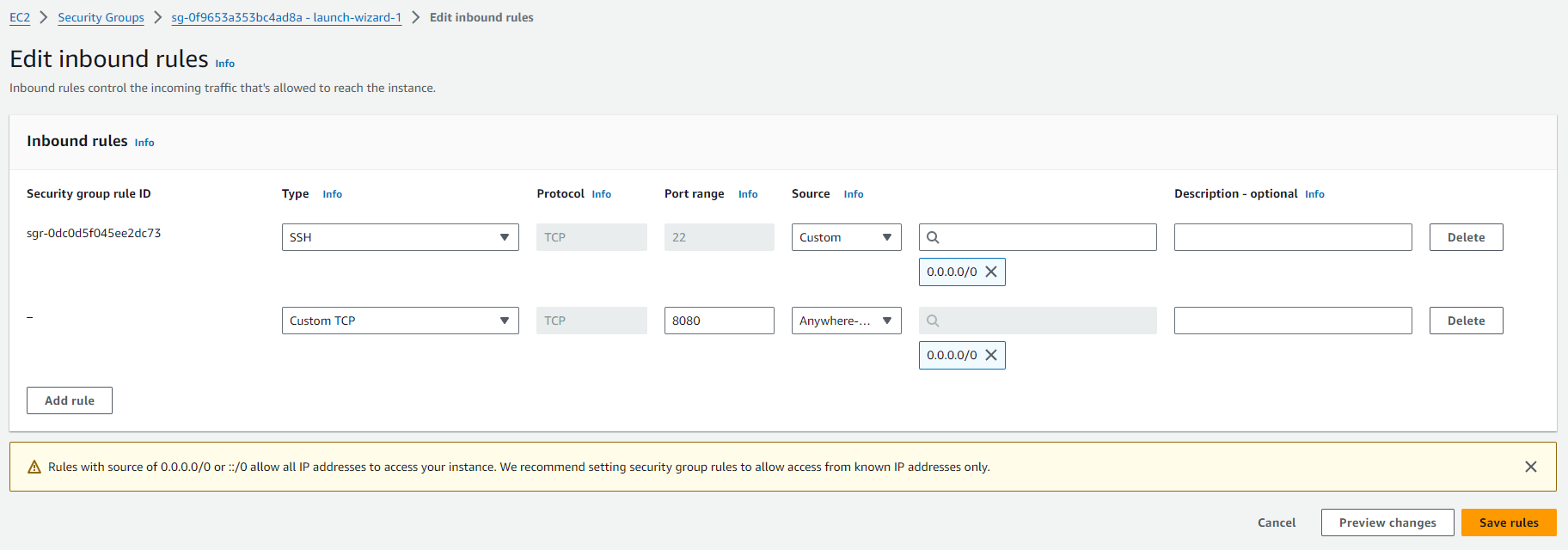
Cmd: **sudo systemctl start Jenkins   
 sudo systemctl enable Jenkins**

To check Jenkins service status

Cmd: **sudo systemctl status Jenkins**



Step 5: Now add port no 8080 in a security group.



Step 6: To access Jenkins.

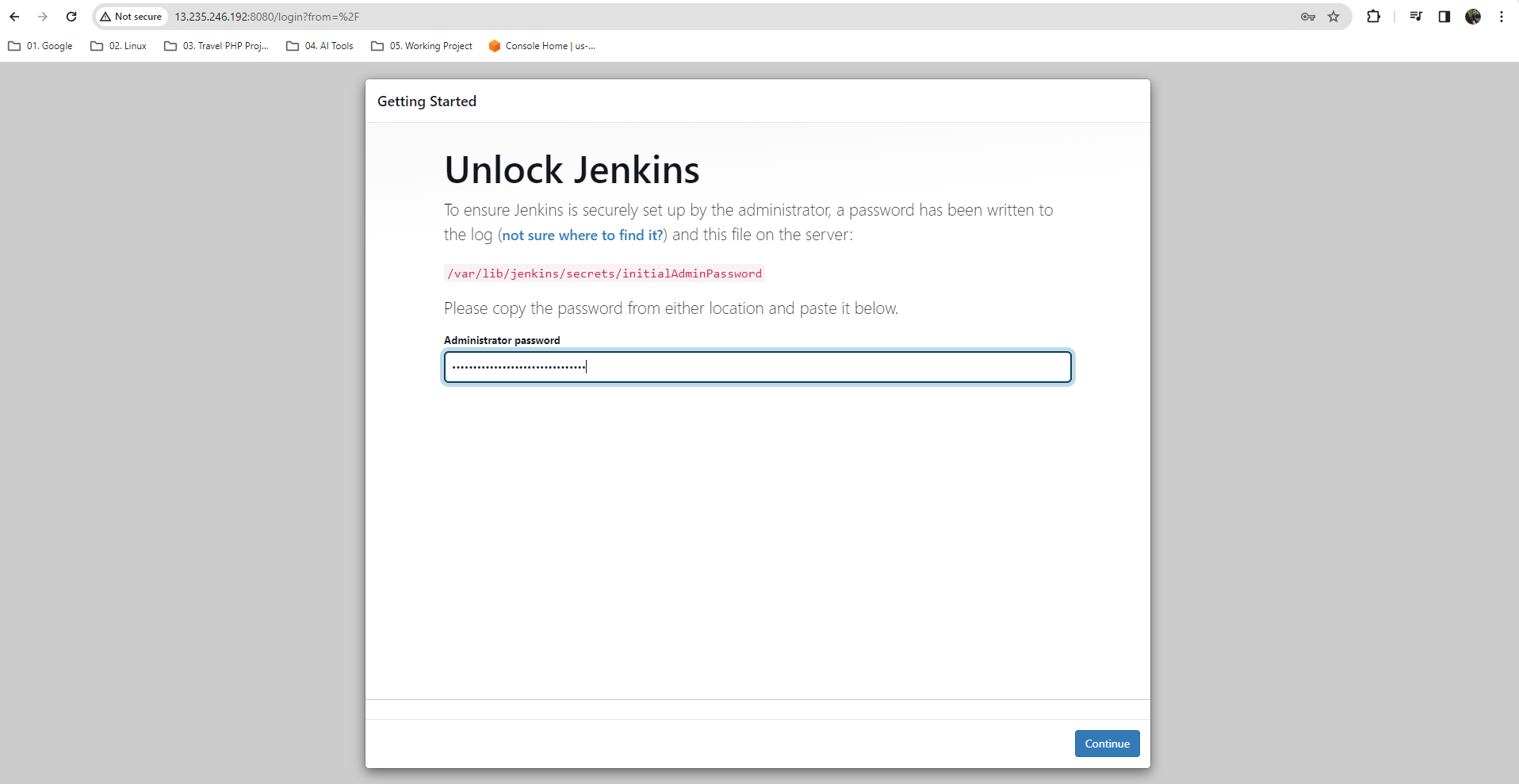
Copy instance public IP and then add listener 8080 to access Jenkins.

[**http://InstancePublicIP:8080/**](http://InstancePublicIP:8080/)

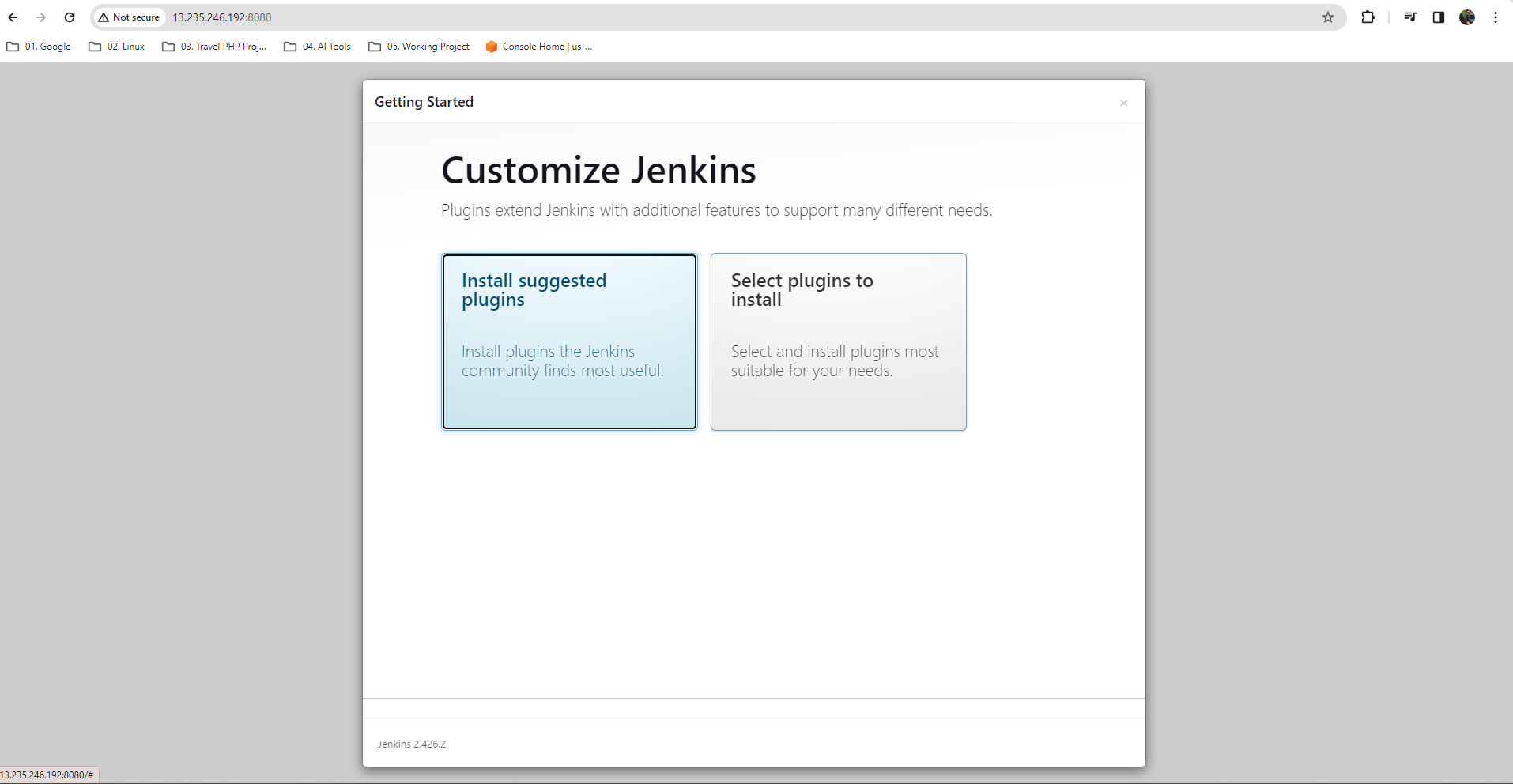
To unlock Jenkins take password from

sudo cat /var/lib/jenkins/secrets/initialAdminPassword and paste.

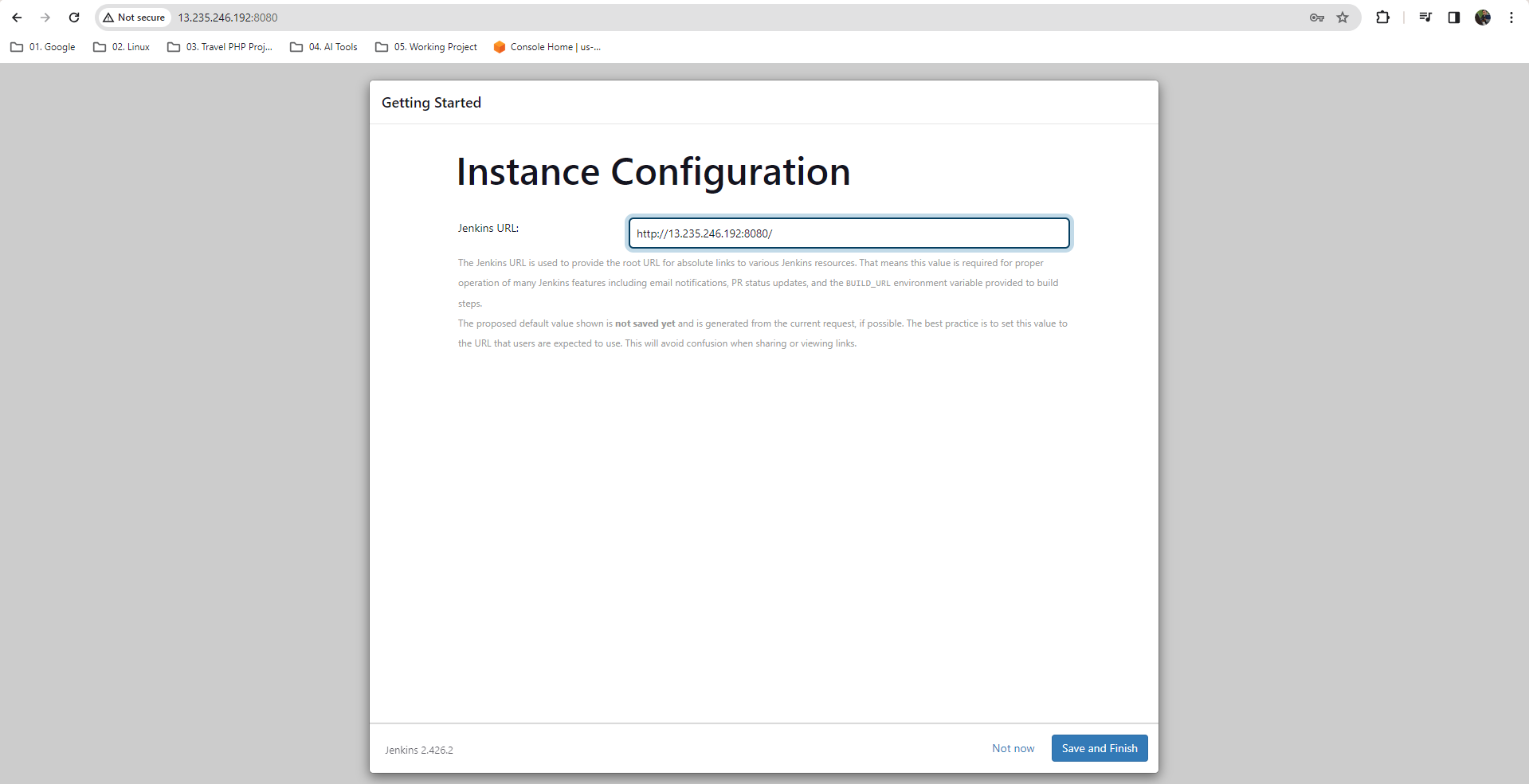




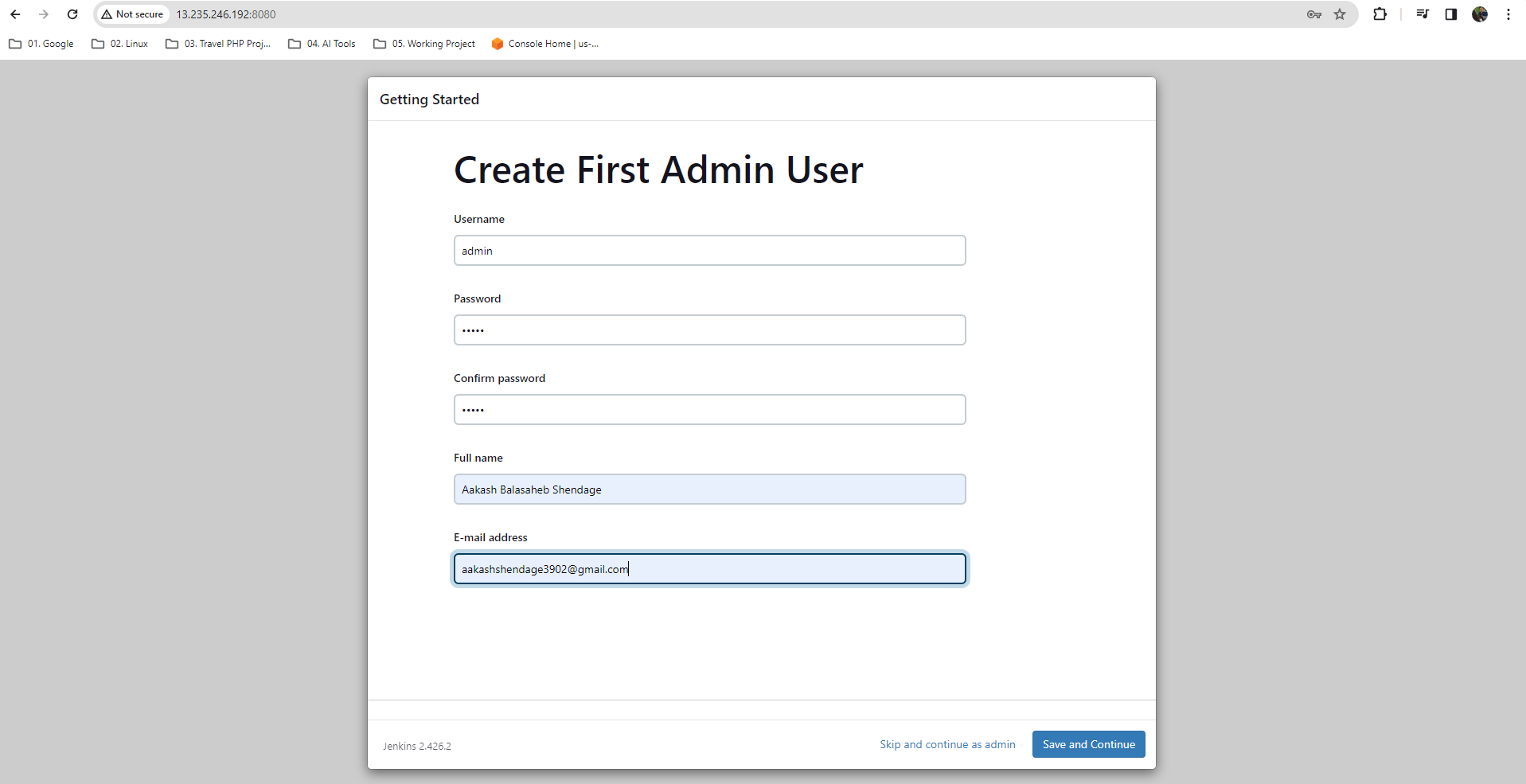
Install suggested plugins.

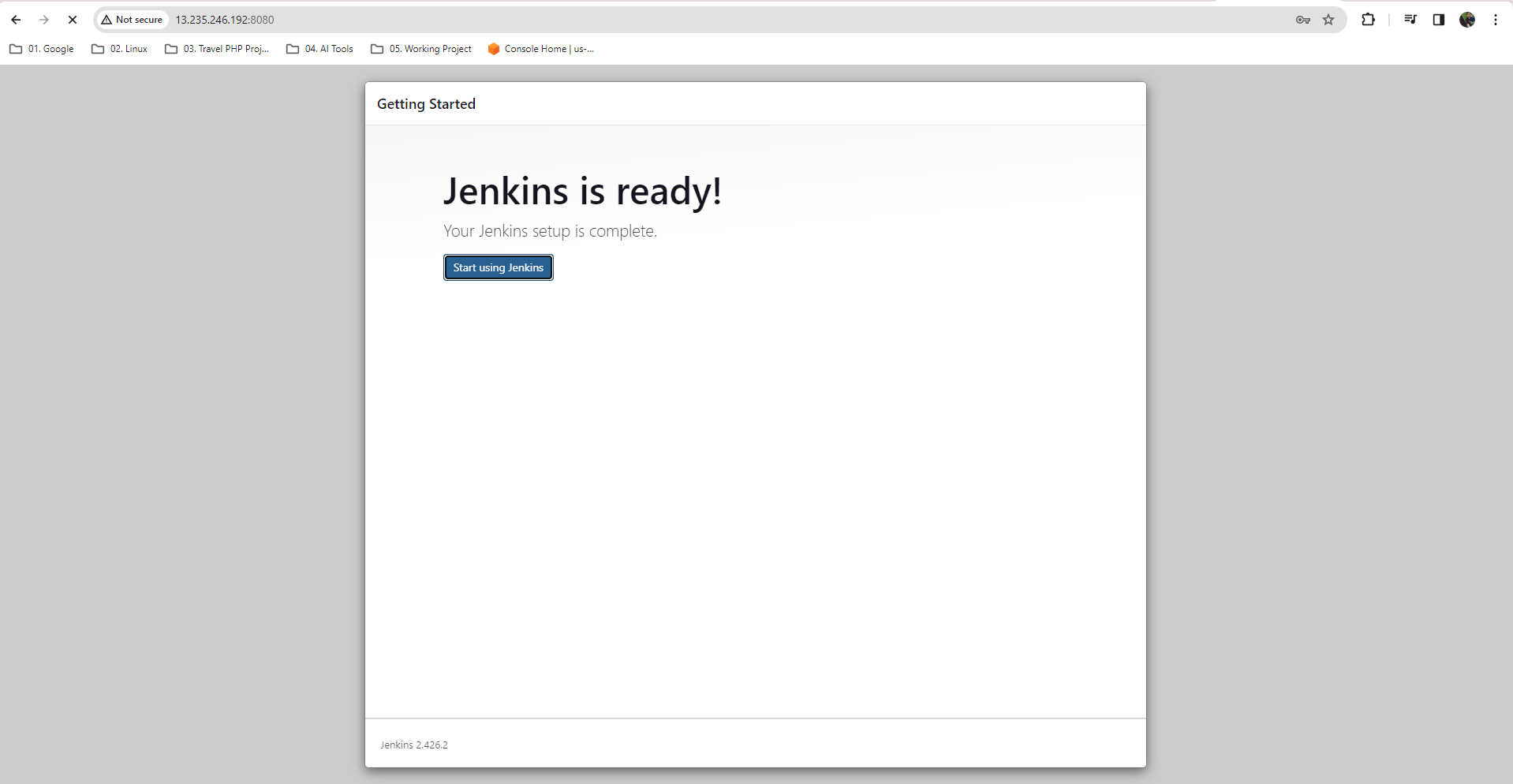


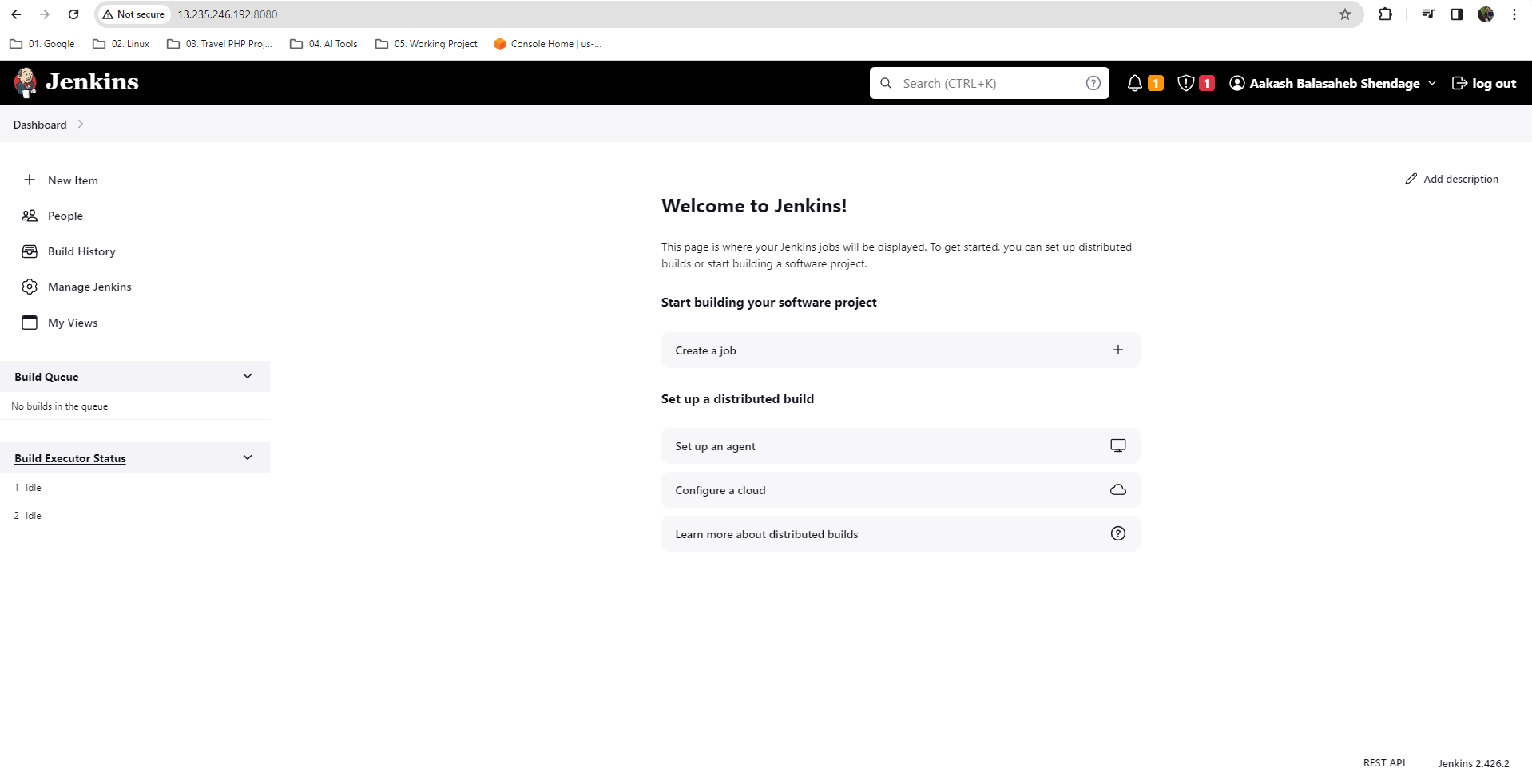
Click save and finish.



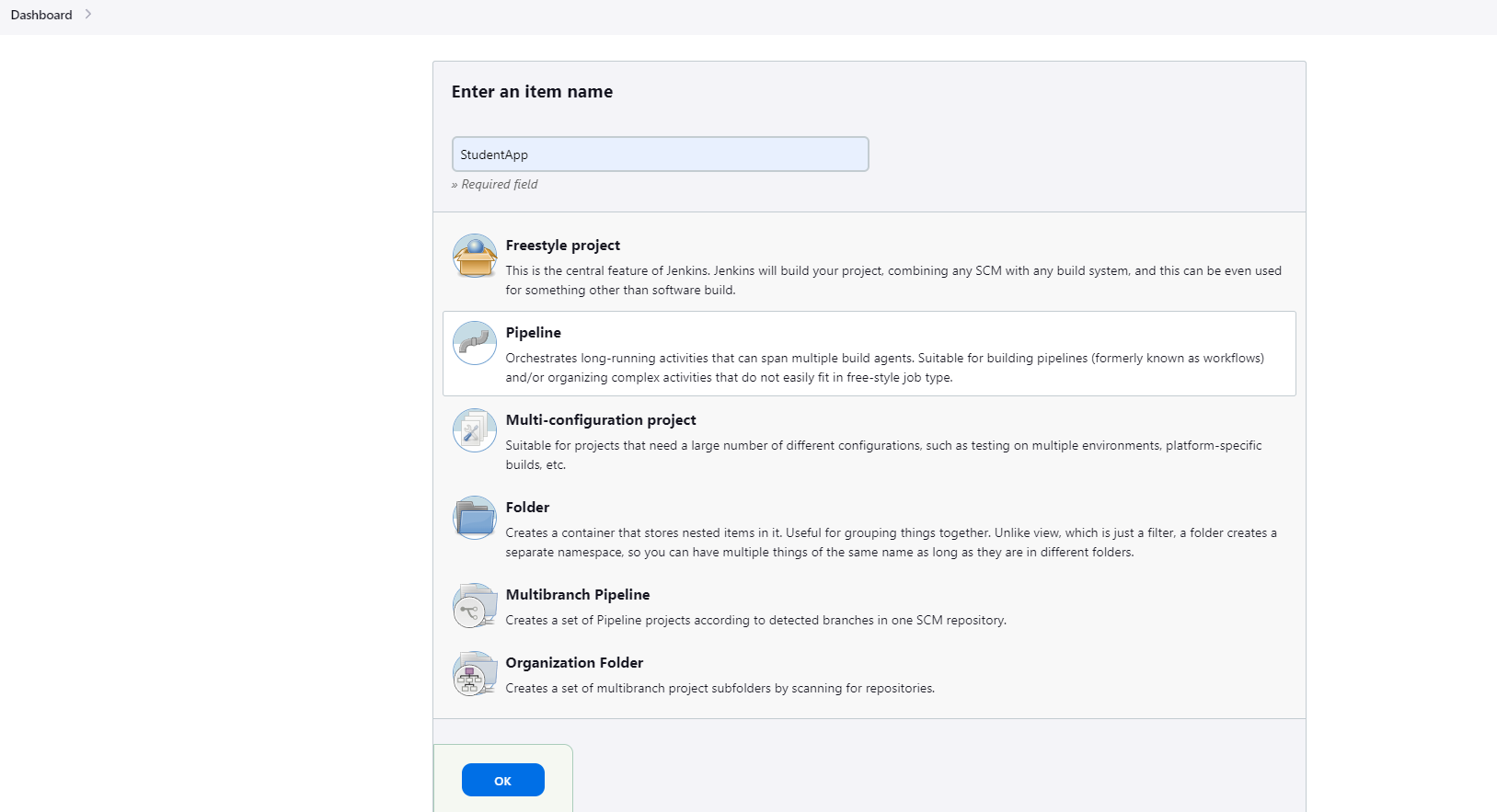
Step 7: Create first admin user in Jenkins then save and continue and start using Jenkins.



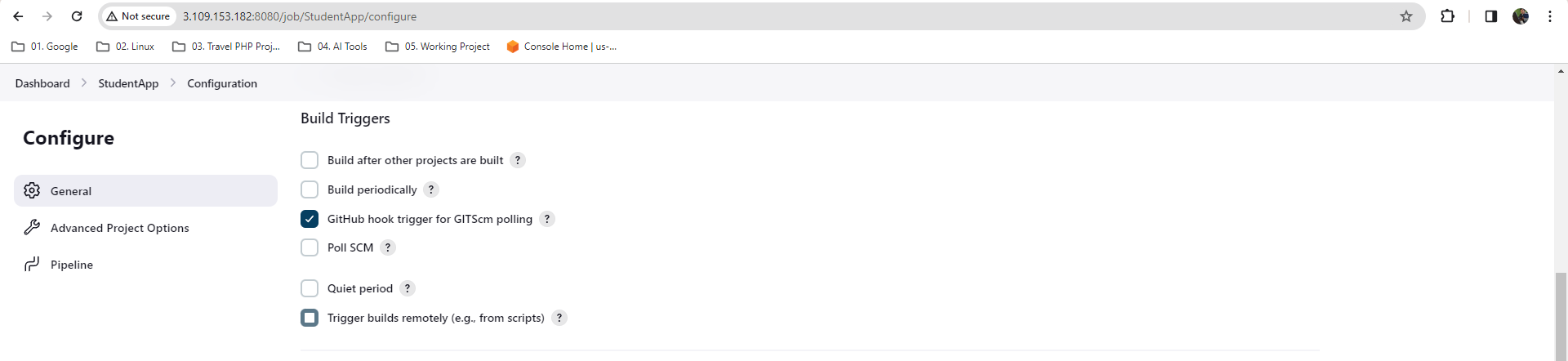


Step 8: Now create a job

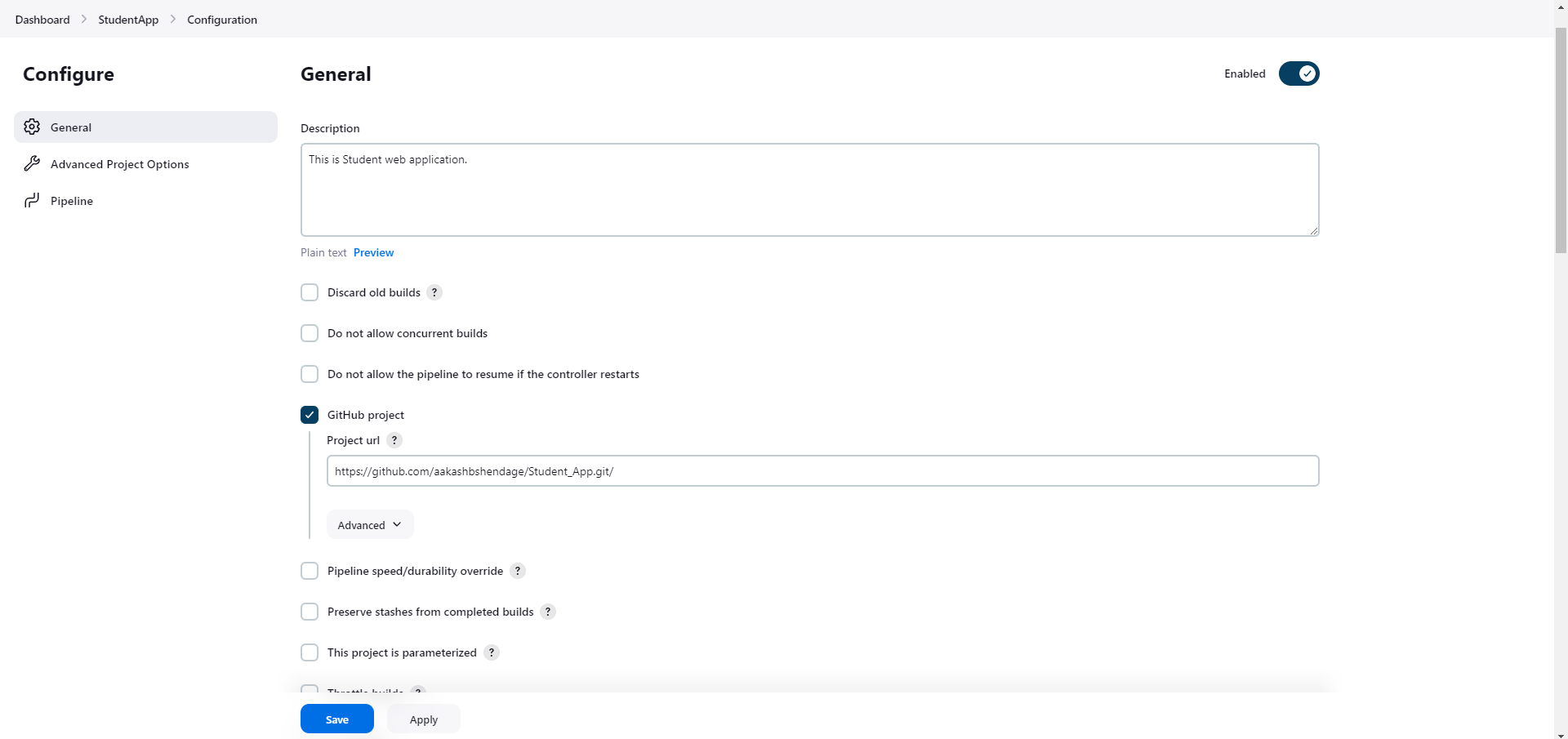
Select pipeline.

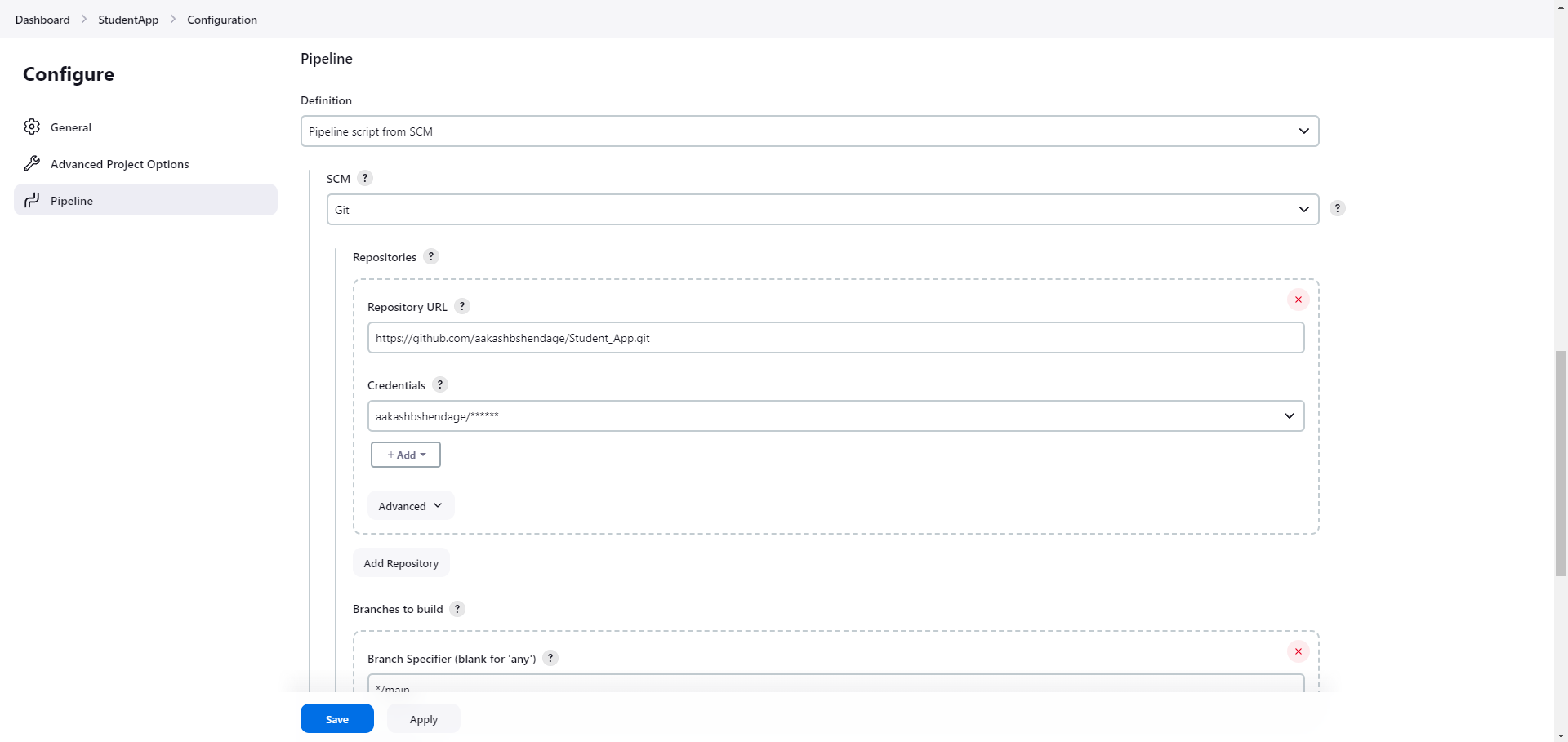


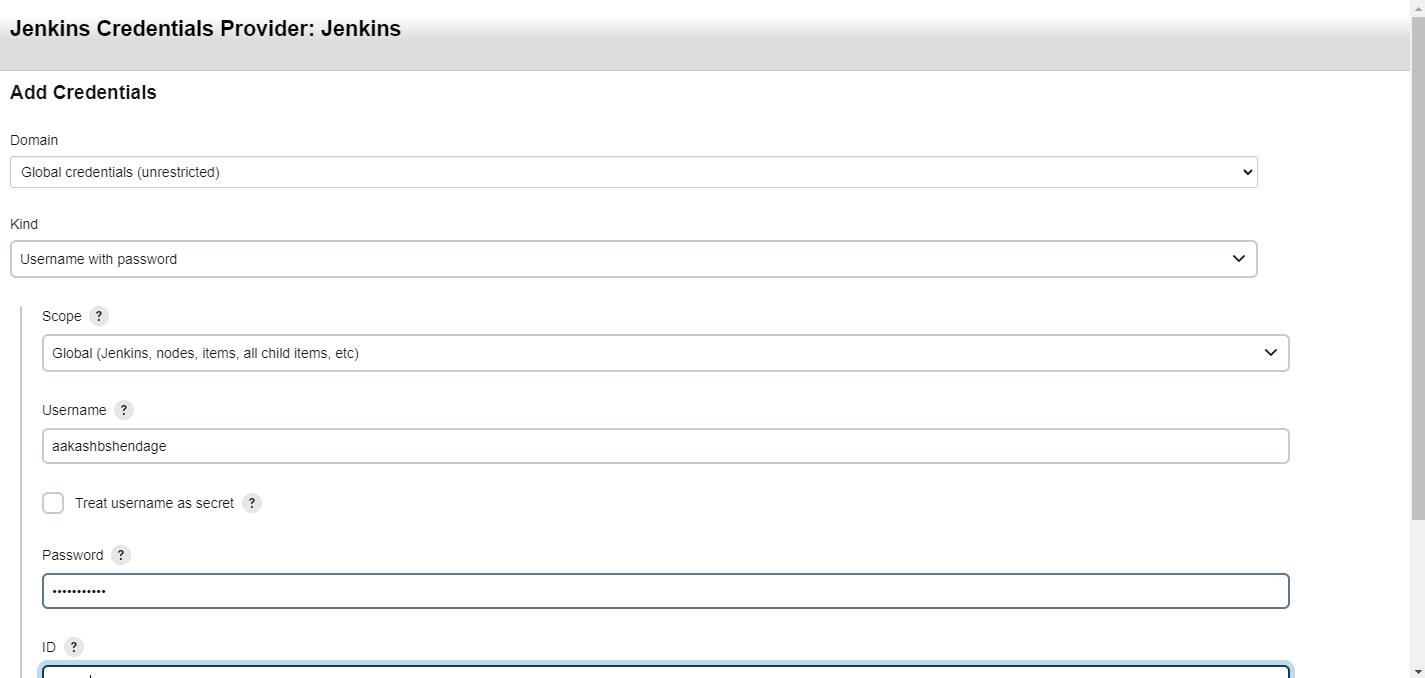
Step 9: Configure a project.



In a Pipeline Select pipeline from SCM and add GitHub credentials.



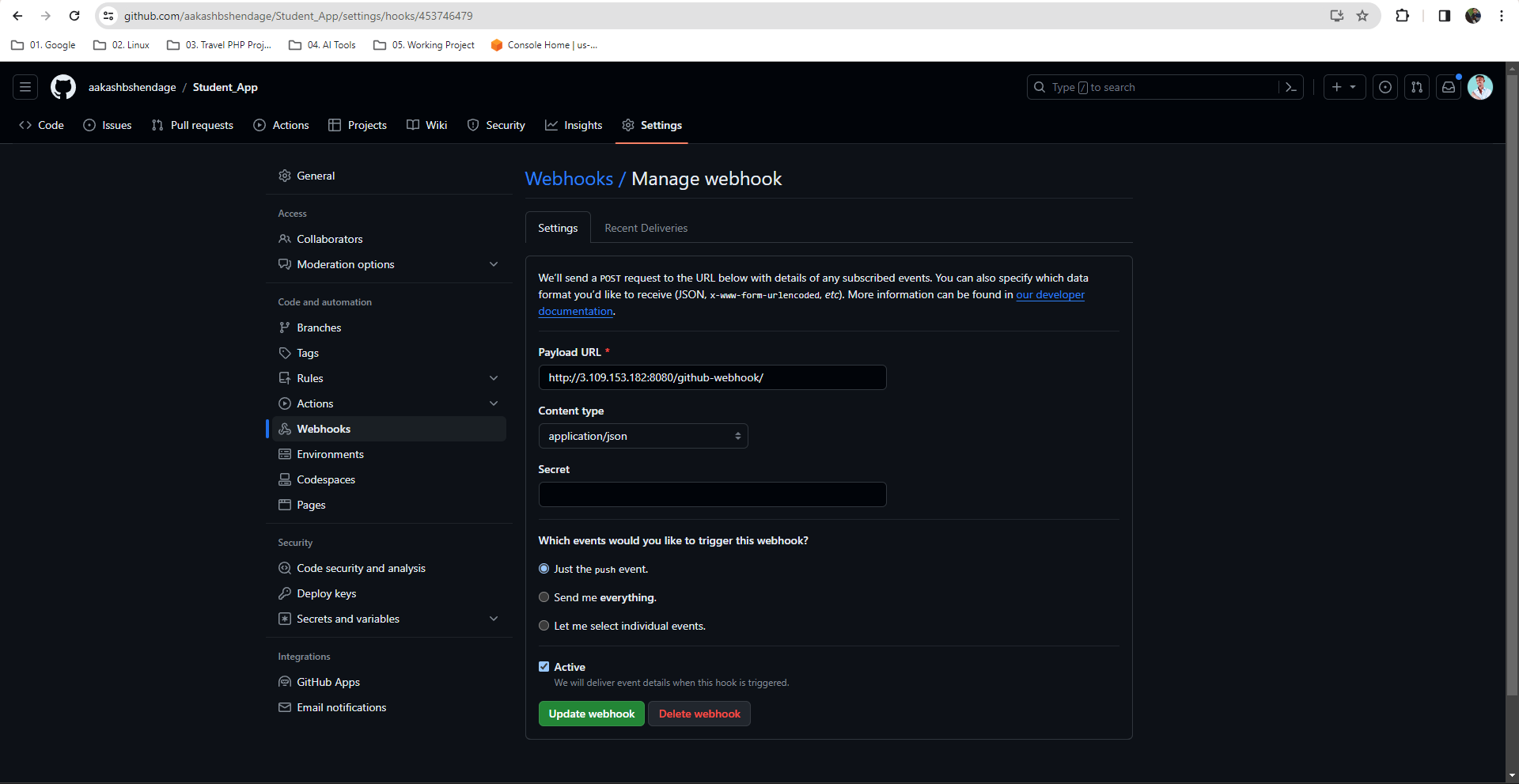




Step 10: Go to GitHub repo settings add Jenkins URL.

Jenkins URL: <http://JenkinsURL:8080/github-webhook/>

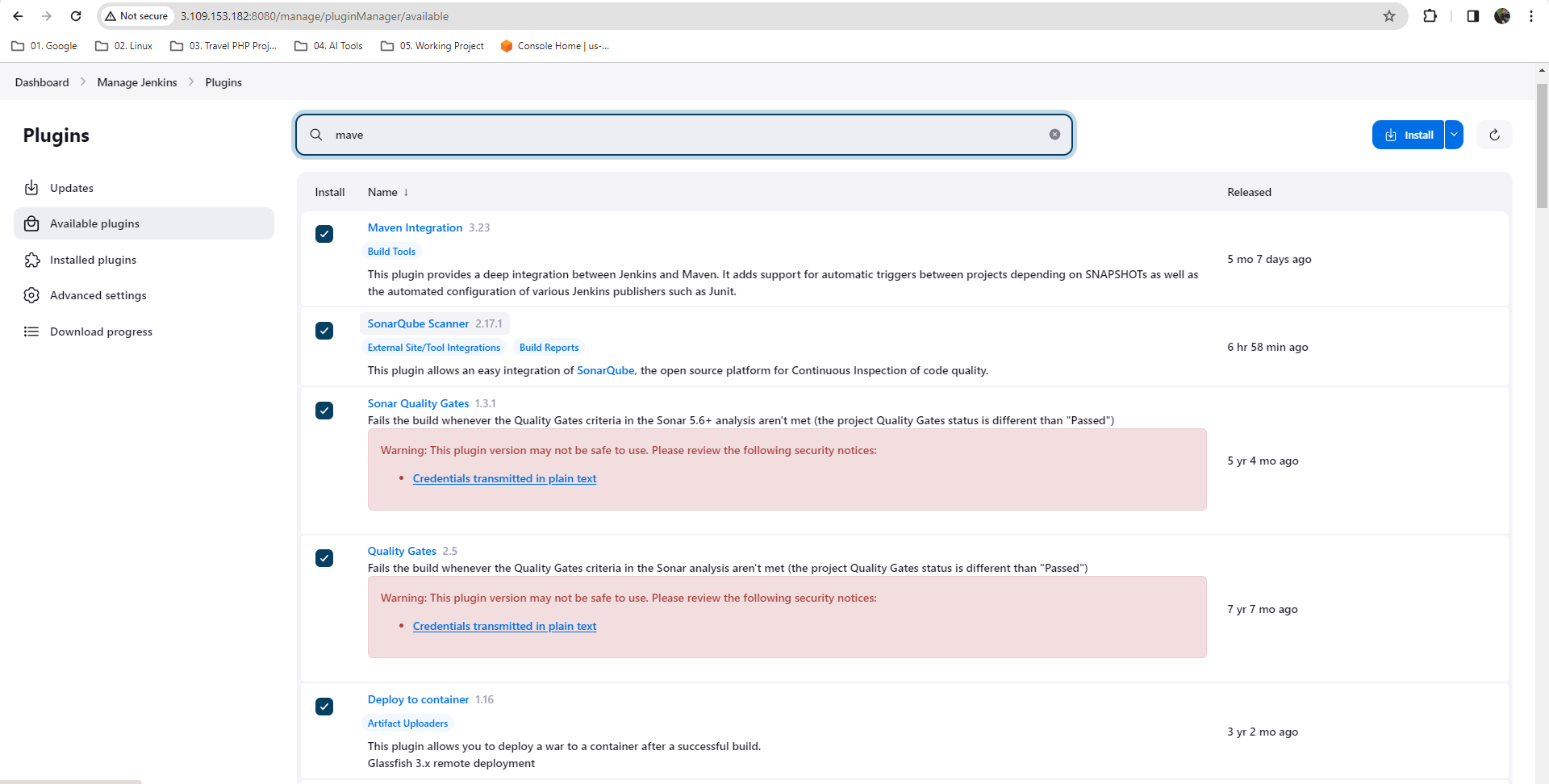
Select Content type: application/json and save.



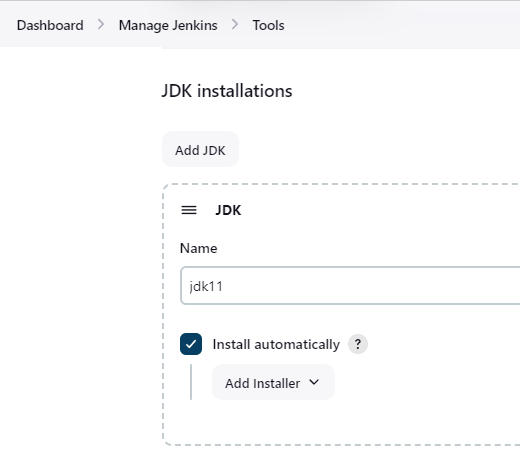
Now we added webhook successfully.

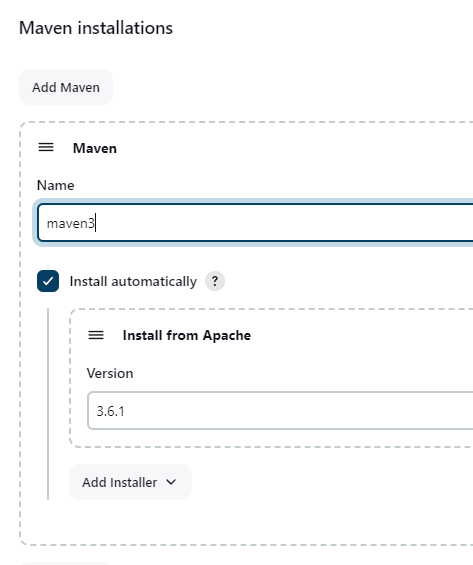
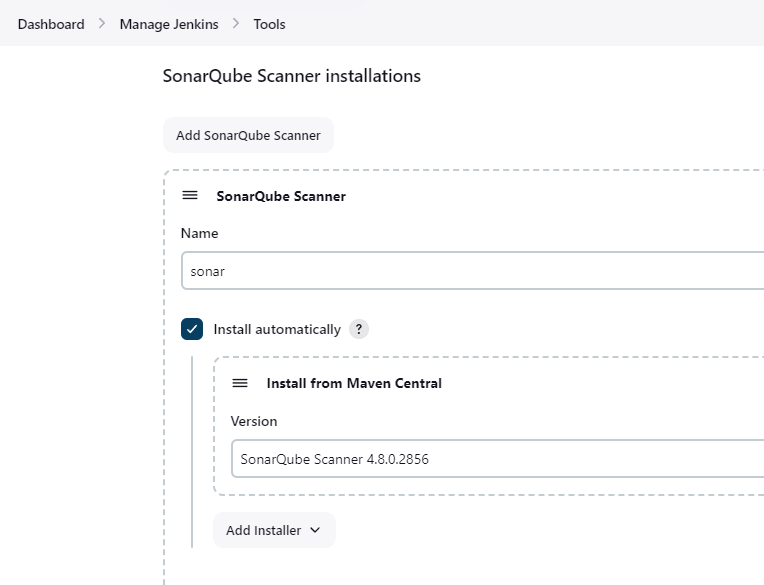
Step 11: Go to Jenkins Dashboard -> Manage Jenkins -> Plugins.

Install following plugins



Step 12: Go to Jenkins Dashboard -> Manage Jenkins -> Tools

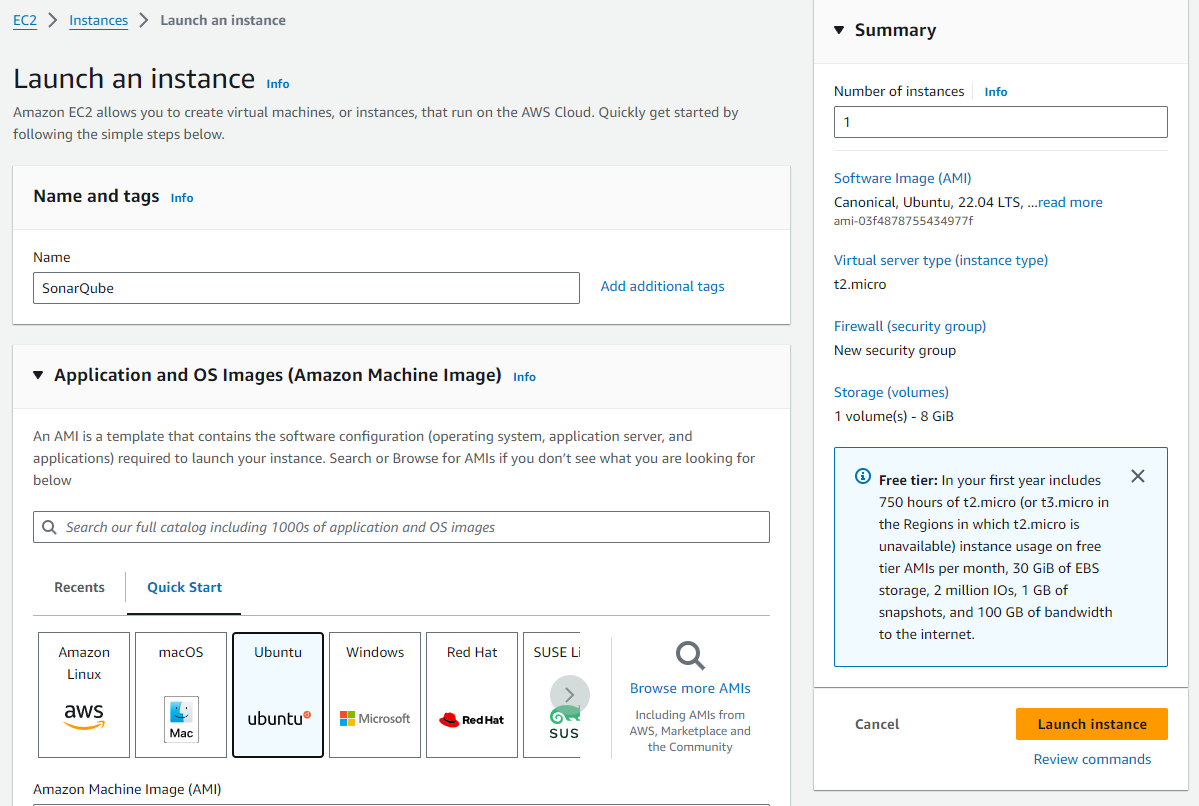


Click apply and save.

Step 13: Now launch EC2 instance for SonarQube.

With t2.medium



Connect sonarqube instance using SSH.

Then run following command to install sonarqube:

**sudo apt update**

**sudo apt install docker.io -y**

**sudo docker run -d -p 9000:9000 sonarqube:lts-community**

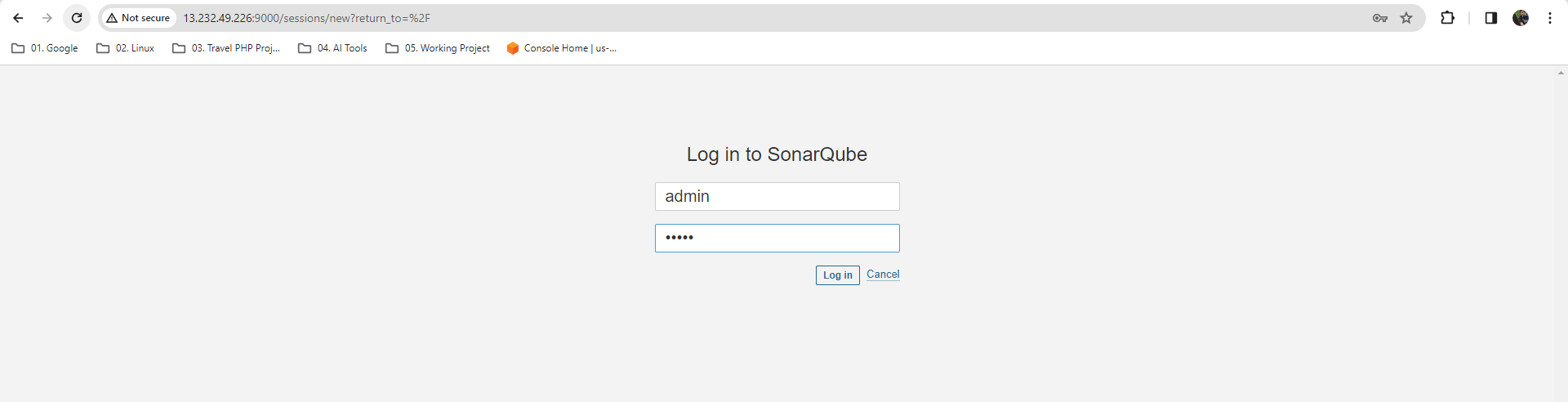
make sure to add 9000 port in your instance security group.

Step 14: Now access sonarqube server using instance public Ip

<http://SonarQubePublicIP:9000>

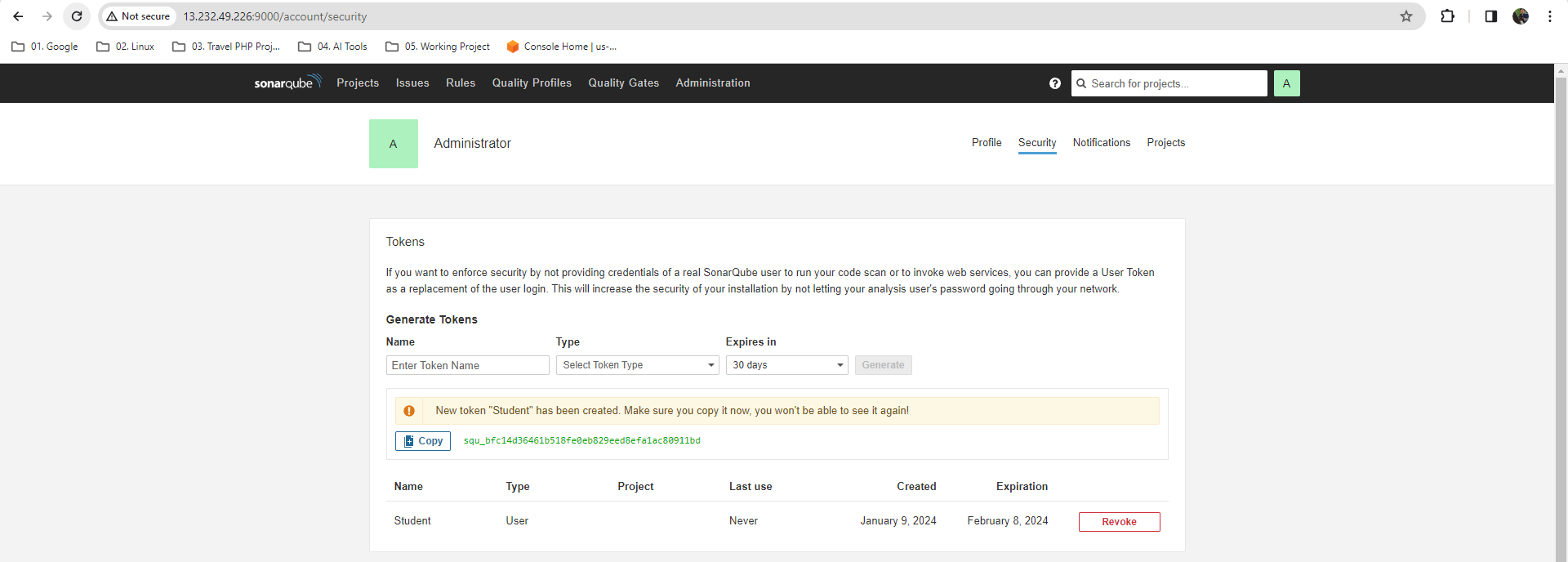
Default User is **admin** and Password is **admin** of sonarqube.

After login change the password



Now go to sonarqube Account -> Security -> Generate Tokens -> Enter Token Name -> Select User Type Token -> Select expiry -> Click Genrate.

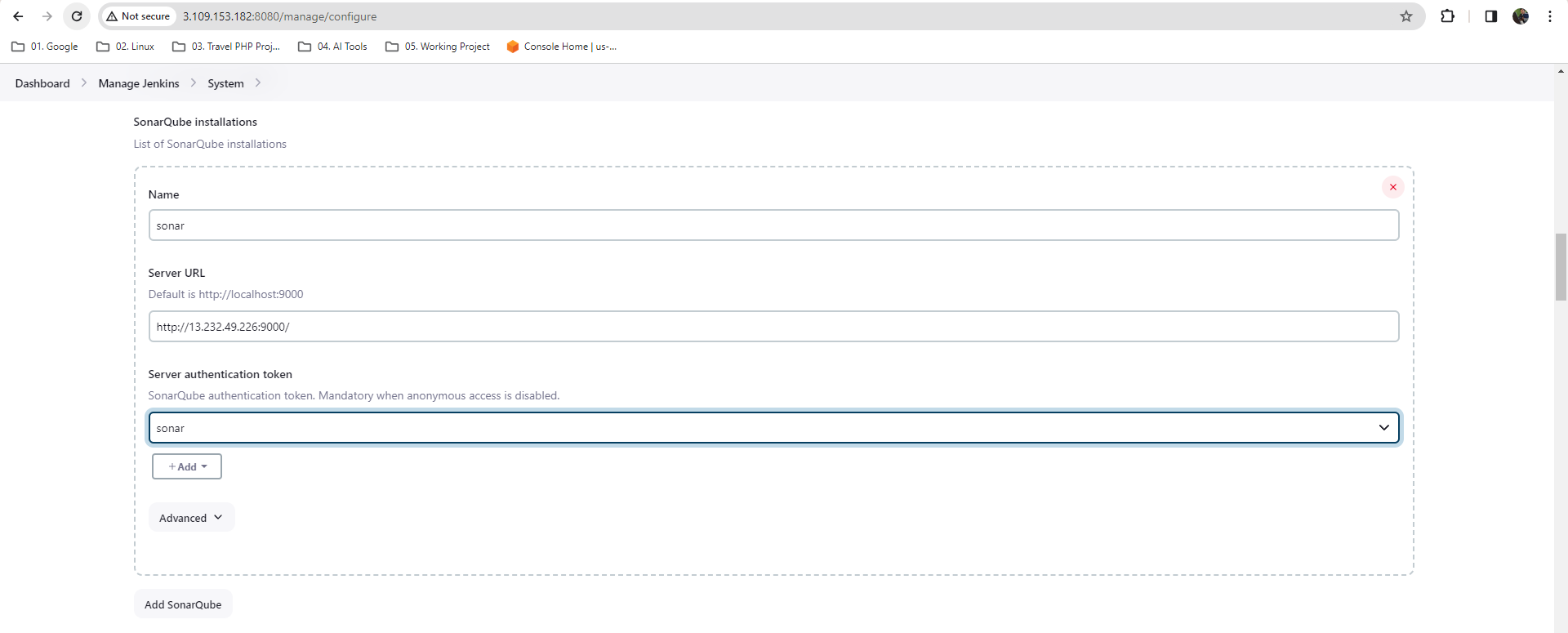
Now copy the code and note it down.



Step 15: Go to Jenkins instance.

Jenkins Dashboard -> Manage Jenkins -> System -> Add SonarQube instance -> Enter Name -> Enter Sonar URL -> Add server authentication token.

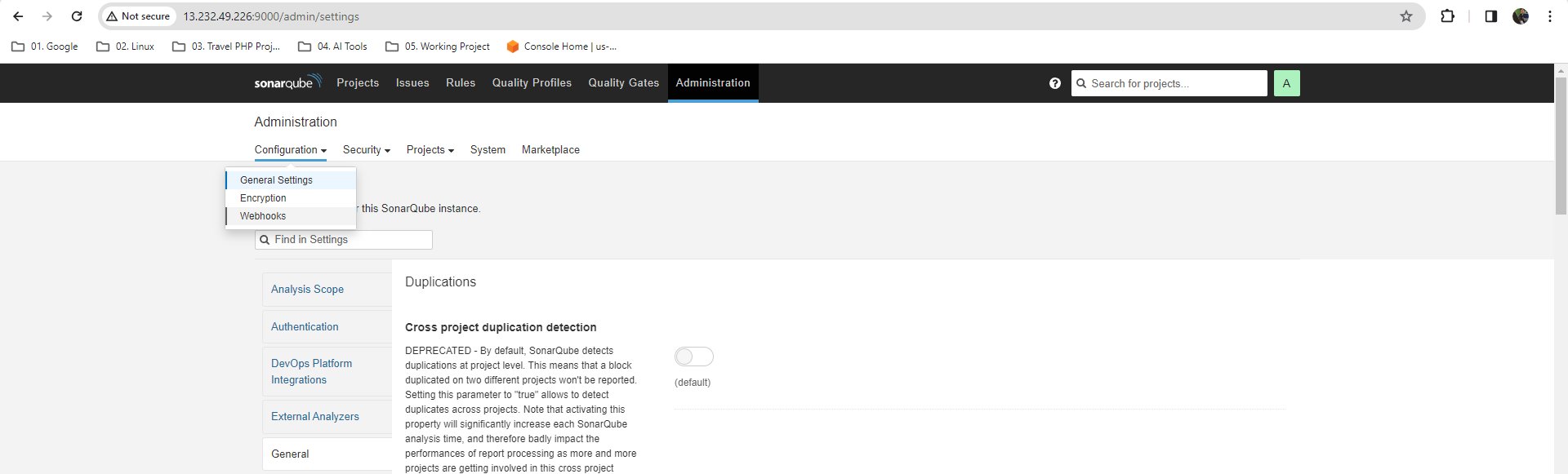
Select Add -> Secret text -> Paste the token here.

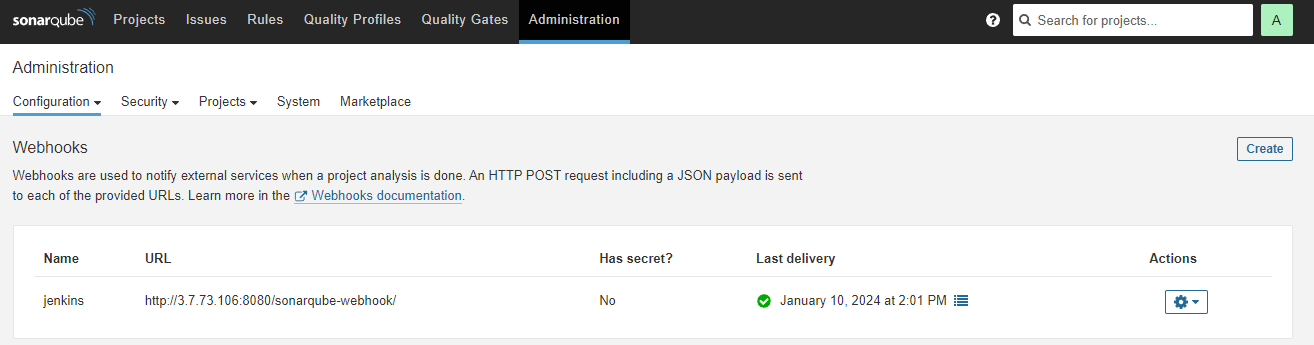


Step 16: Now again go to sonarqube instance.

Admin -> Settings -> Configuration -> Webhooks -> Enter Jenkins URL here.

<http://JenkinsURL:9000/sonarqube-webhook/>





Step 17: Now go to github update Jenkinsfile.

Code:

pipeline {

agent any

tools{

// Tools configuration add here

jdk 'jdk11'

maven 'maven3'

}

stages {

stage('Gitcheck Out') {

// Add here github source code URL

steps {

git changelog: false, poll: false, url: 'https://github.com/aakashbshendage/Student\_App.git'

echo 'Pull sucessfull'

}

}

stage('Compile') {

// Add here building stage

steps {

sh 'mvn clean package -DskipTests=true'

echo 'Build sucessfull'

}

}

stage('Sonar Analysis') {

// Add here sonarqube configuration

steps {

script{

withSonarQubeEnv(credentialsId: 'sonar')

{

sh 'mvn sonar:sonar'

echo ' Test sucessfull'

}

}

}

}

stage("Quality Gate"){

// Add here Quality Gate stage

steps {

script {

waitForQualityGate abortPipeline: false, credentialsId: 'sonar'

echo ' Quality Gate test sucessfull'

}

}

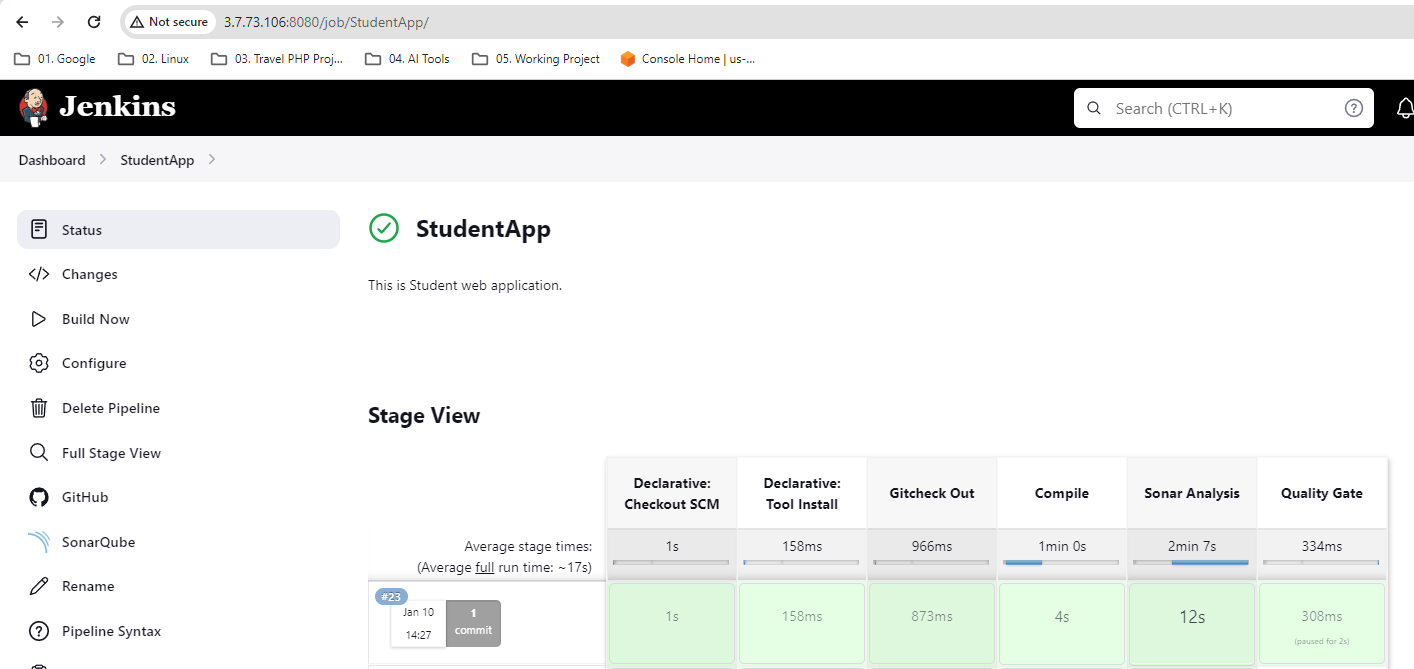
}

}

}

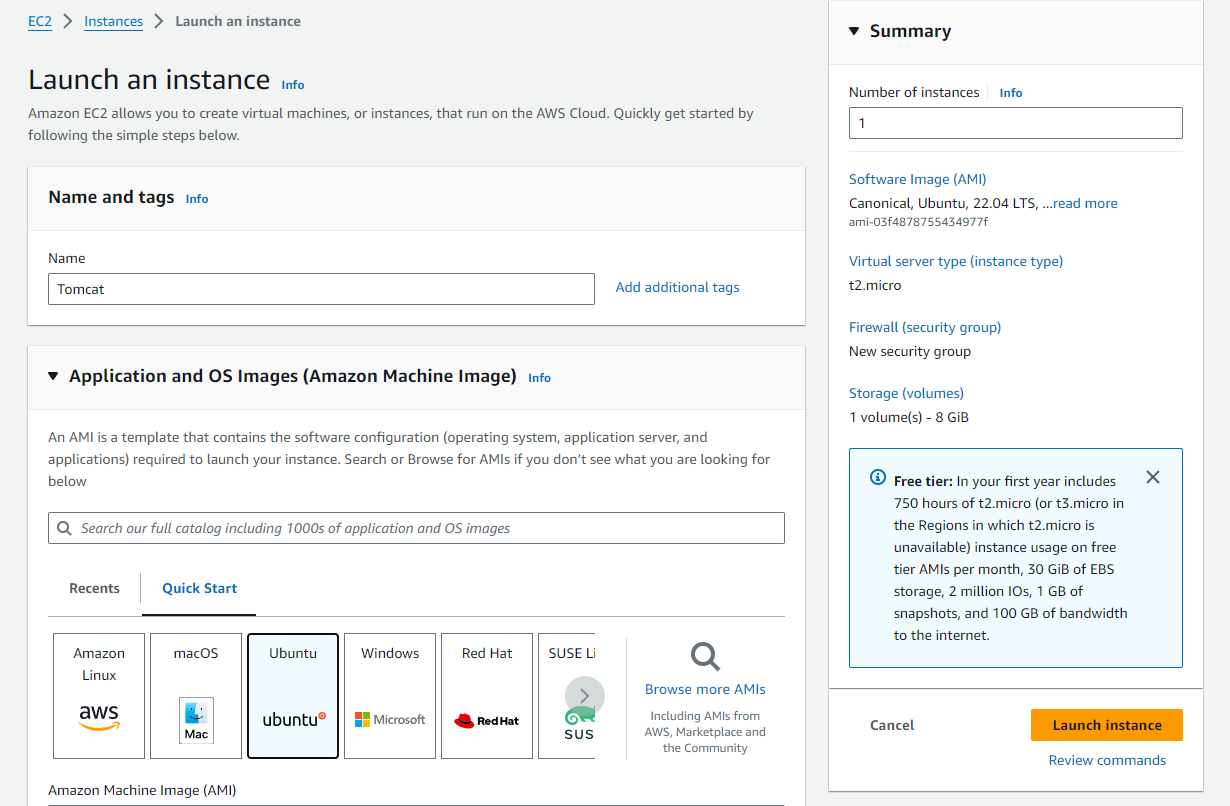
Save and Push the code.

Now check to Jenkins your project will automated building.

****

Step 18: Now deploy project on Tomcat.

Launch new EC2 instance to deploy the project

****

Connect sonarqube instance using SSH.

Then run following command to install apache tomcat:

**sudo apt update**

**sudo apt install openjdk-11-jre-headless -y**

**wget** [**https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.85/bin/apache-tomcat-9.0.85.tar.gz**](https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.85/bin/apache-tomcat-9.0.85.tar.gz)

**sudo tar -xzf apache-tomcat-9.0.85.tar.gz -C /opt**

Now go to

cd /opt/apache-tomcat-9.0.85/bin

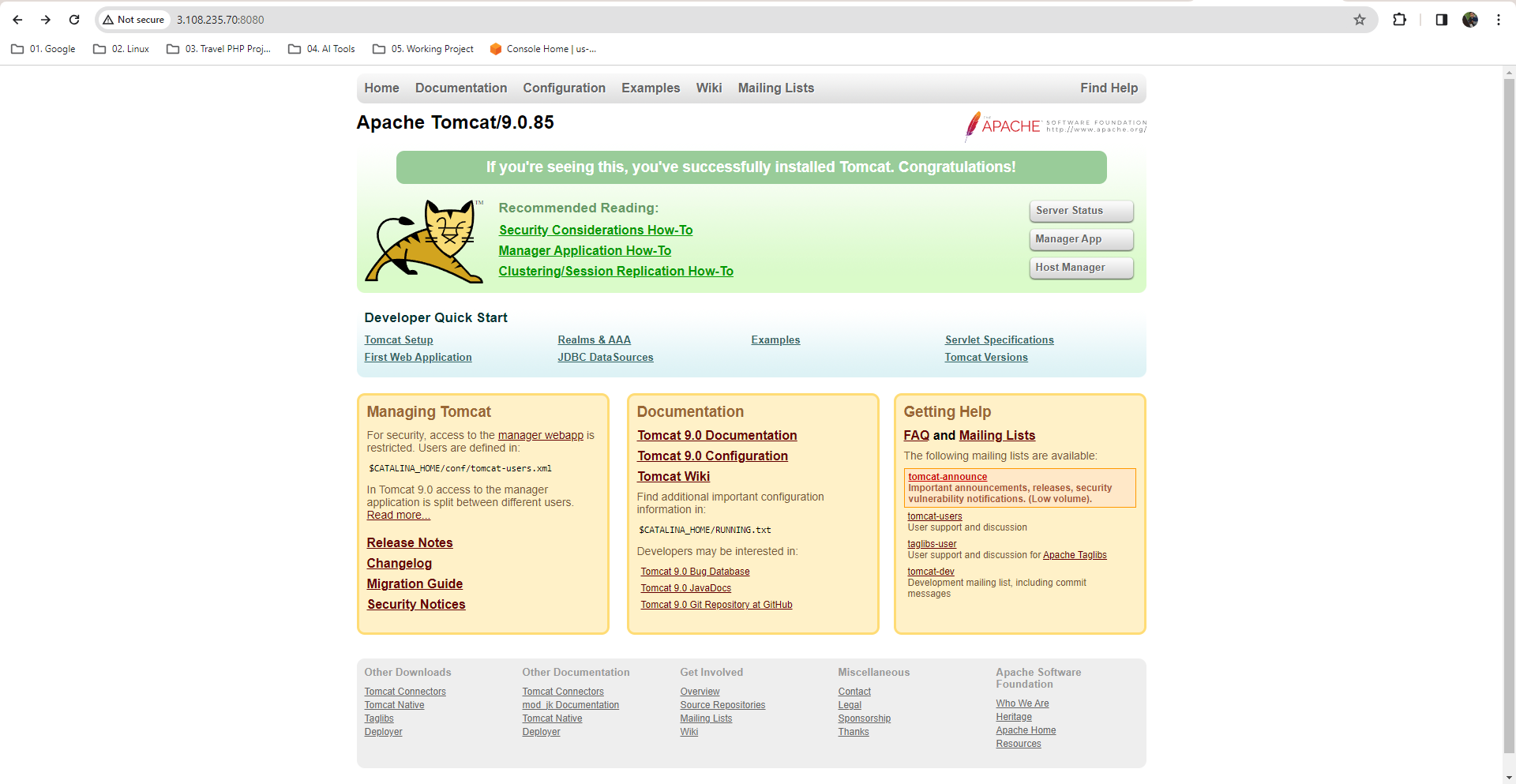
Start Catalina.sh

cmd: **./catalina.sh start**

Step 19: Now access tomcat using instance IP.

<http://TomcatURL:8080>

make sure to add port 8080 in your security group.



Step 20: Comment this lines.

Go to

1. vim /opt/ apache-tomcat-9.0.85/webapps/host-manager/META-INF/context.xml



1. vim /opt/ apache-tomcat-9.0.85/webapps/manager/META-INF/context.xml



Add this following lines in tomcat-users.xml

vim /opt/ apache-tomcat-9.0.85/conf/tomcat-users.xml

<role rolename="manager-gui"/>

<role rolename="manager-script"/>

<role rolename="manager-jmx"/>

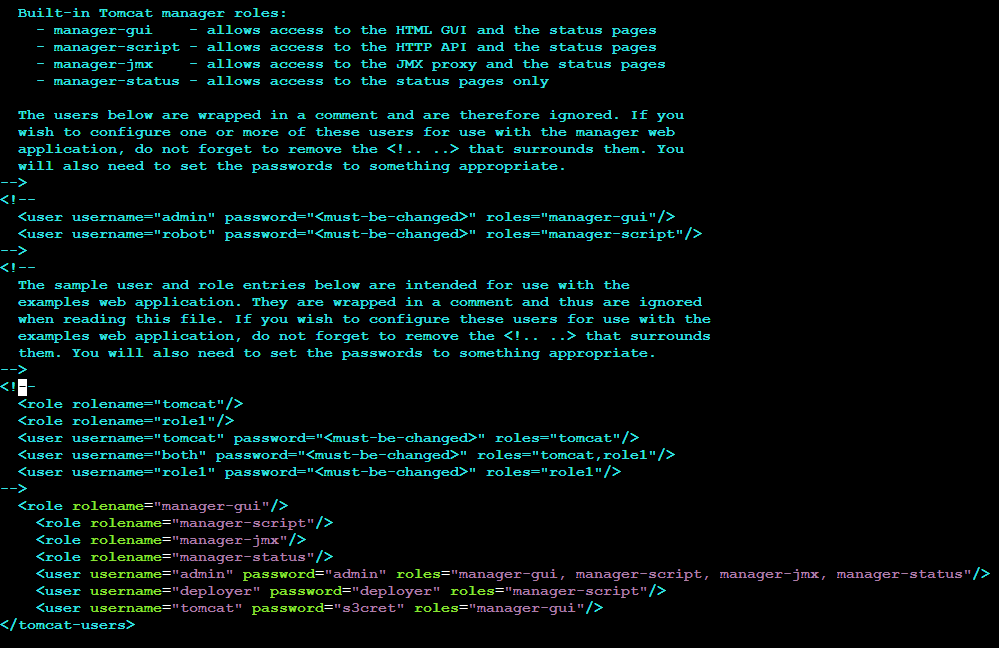
<role rolename="manager-status"/>

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

<user username="deployer" password="deployer" roles="manager-script"/>

<user username="tomcat" password="s3cret" roles="manager-gui"/>

</tomcat-users>



Step 20: Now stop and start again tomcat.

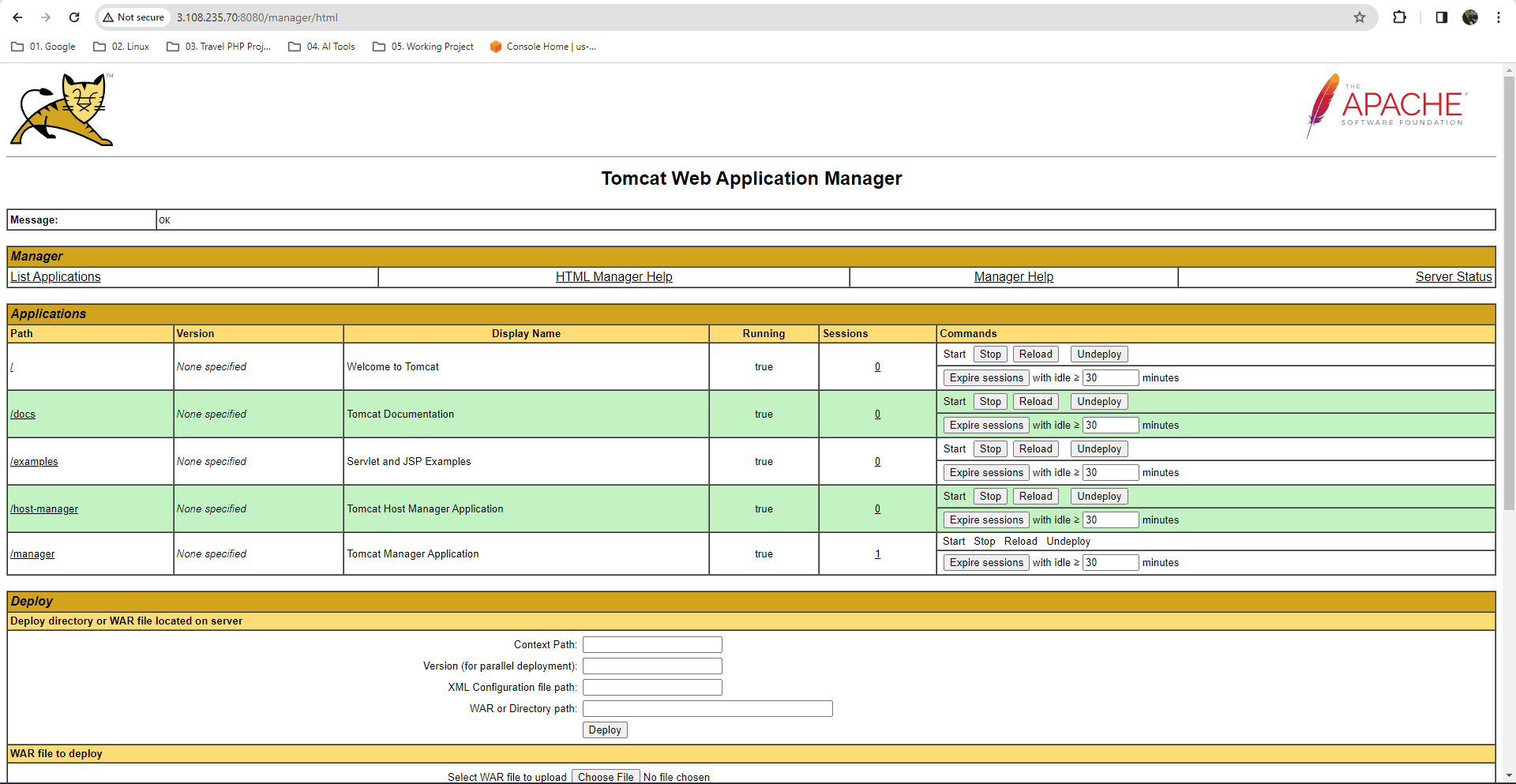
cd /opt/ apache-tomcat-9.0.85

cmd: **./catalina sh stop**

**./catalina sh start**

Step 21: Now go to tomcat browser select Manager App

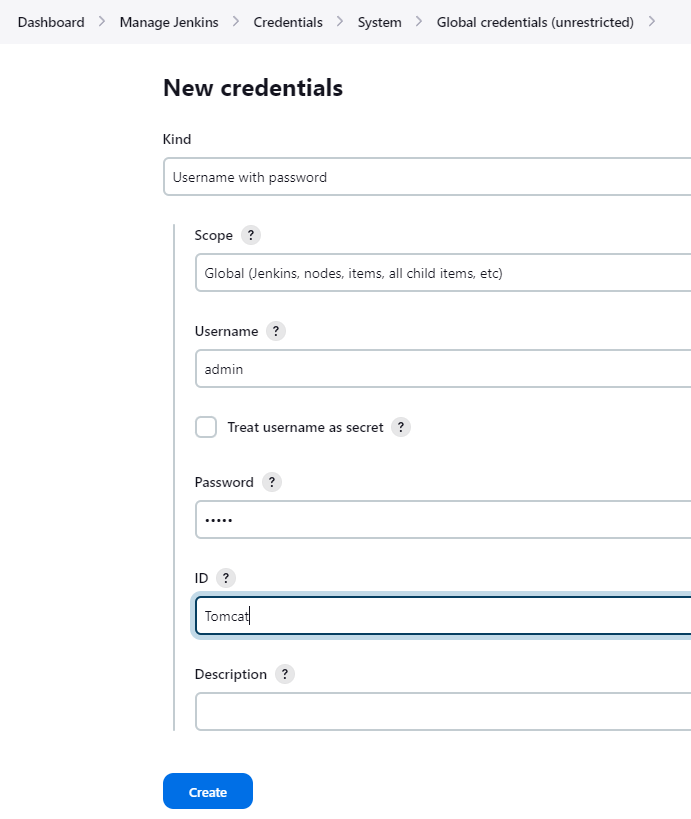
Enter username: **admin** password: **admin**



Step 22: Now go to Jenkins server. Add credentials.

Dashboard -> Manage Jenkins -> Credentials -> System -> Global credentials

Select Kind username with password -> Scope Global -> Username Enter here tomcat username -> Password Enter here tomcat password.



Final Pipeline:

pipeline {

agent any

tools{

// Tools configuration add here

jdk 'jdk11'

maven 'maven3'

}

stages {

stage('Gitcheck Out') {

// Add here github source code URL

steps {

git changelog: false, poll: false, url: 'https://github.com/aakashbshendage/Student\_App.git'

echo 'Pull sucessfull'

}

}

stage('Compile') {

// Add here building stage

steps {

sh 'mvn clean package -DskipTests=true'

echo 'Build sucessfull'

}

}

stage('Sonar Analysis') {

// Add here sonarqube configuration

steps {

script{

withSonarQubeEnv(credentialsId: 'sonar')

{

sh 'mvn sonar:sonar'

echo ' Test sucessfull'

}

}

}

}

stage("Quality Gate"){

// Add here Quality Gate stage

steps {

script {

waitForQualityGate abortPipeline: false, credentialsId: 'sonar'

echo ' Quality Gate test sucessfull'

}

}

}

stage ('Deploy') {

steps {

deploy adapters: [tomcat9(credentialsId: 'Tomcat', // Enter your tomcat credentials here

path: '', url: 'http://3.108.235.70:8080/')], // Enter here tomcat instance IP

contextPath: '/', war: '\*\*/\*.war'

echo ' Deploy sucessfull'

}

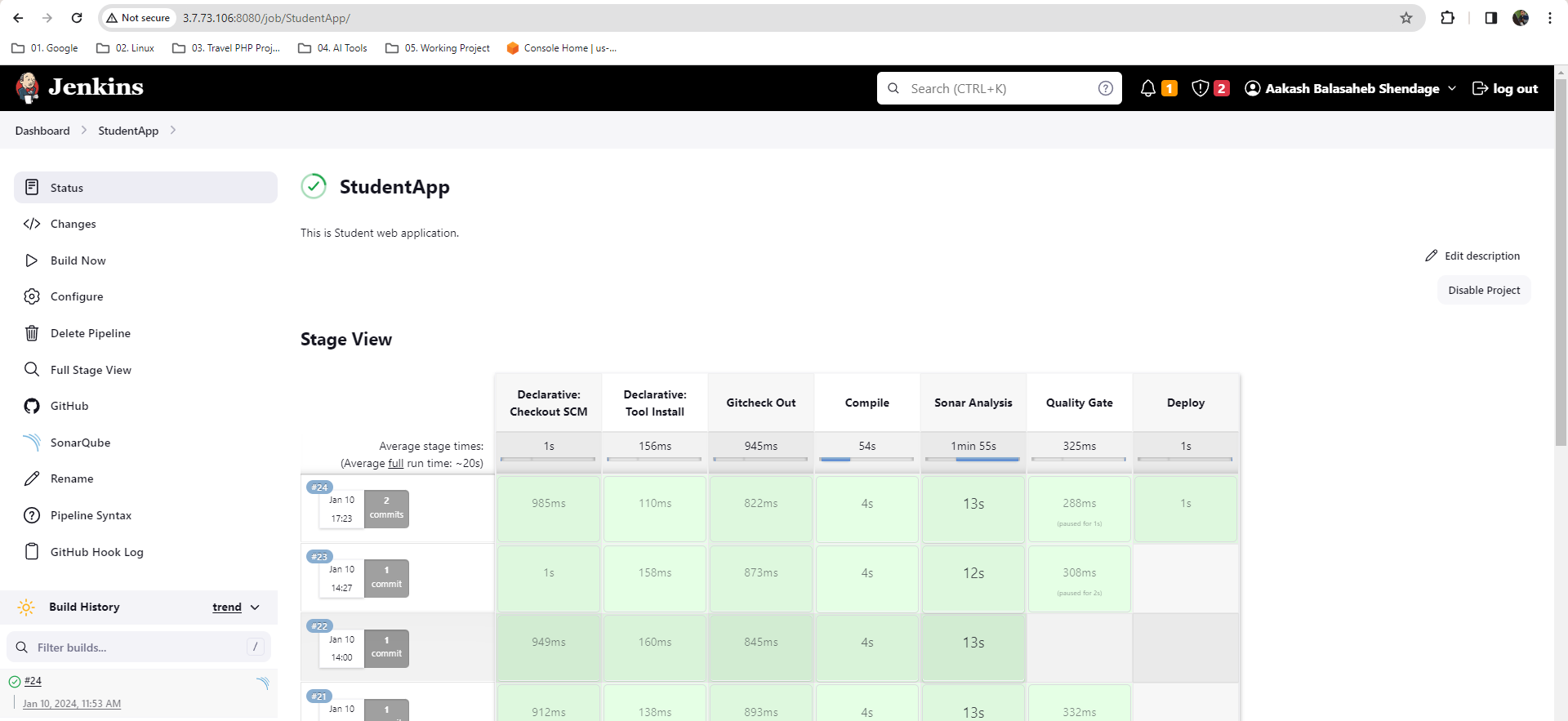
}

}

}

Push this code to github.

Step 23: Now go to Jenkins server our pipeline will triggered automatically.

****

Step 23: Once deployment is done.

Access our student registration page using tomcat url.

****

**Thank You**