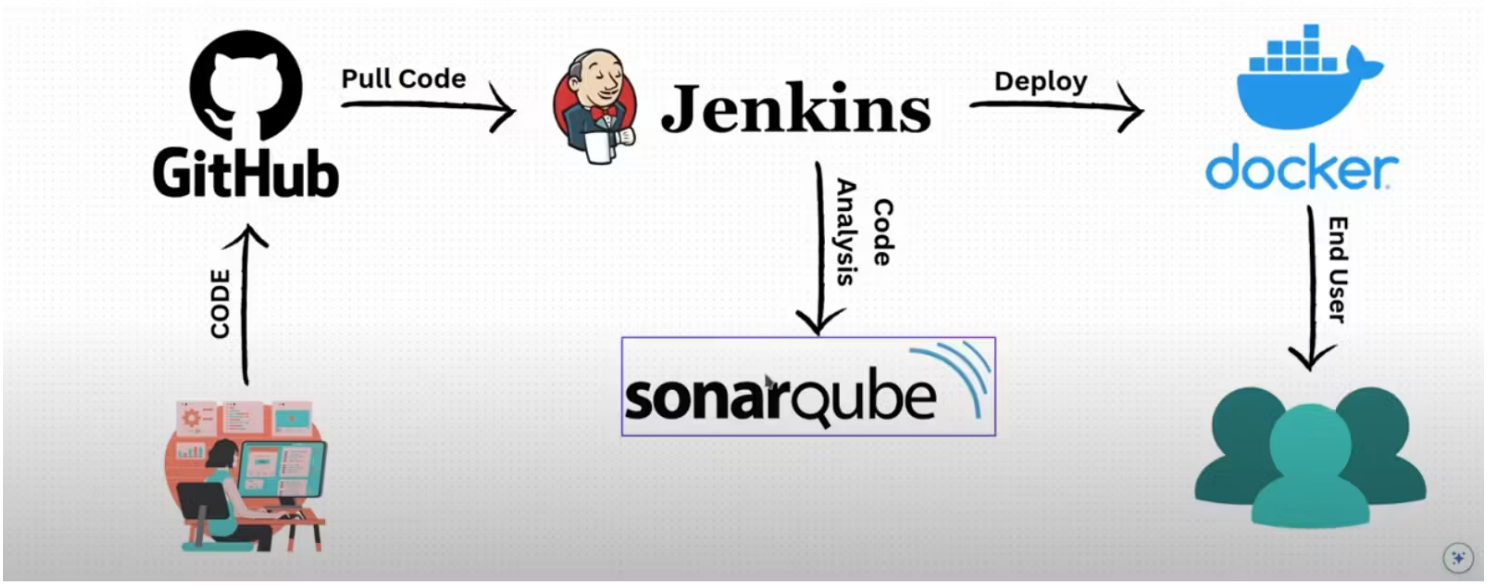
DevOps by Sagar Aulakh

Follow



Jenkins CI/CD Pipeline – SonarQube, Docker, Github Webhooks on AWS

SA Sagar Aulakh

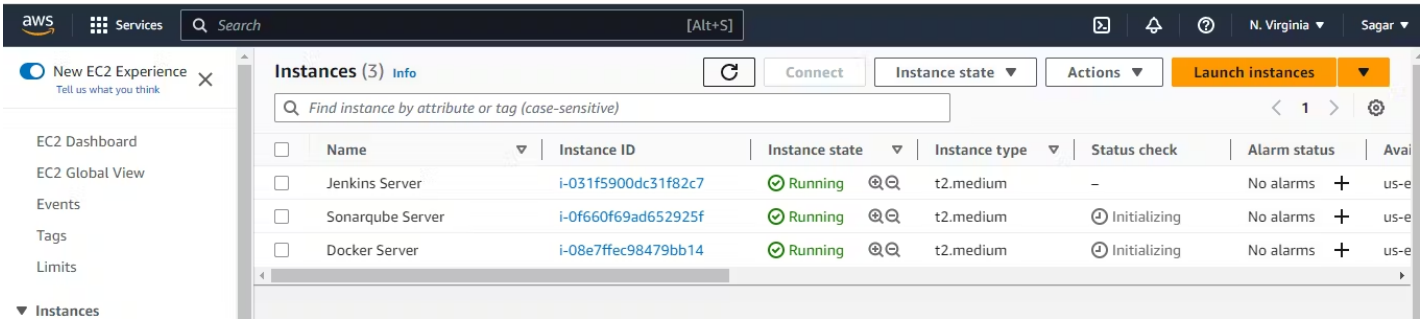
Apr 26, 2023 · 7 min read

TABLE OF CONTE

Launch Instances

I have created 3 t2.medium instances

- Instance for Jenkins
- Instance for Sonarqube
- Instance for Docker



Install Jenkins

Jenkins Server

COPY

```
sudo apt update
clear
```

Now we have to install java first before installing Jenkins

COPY

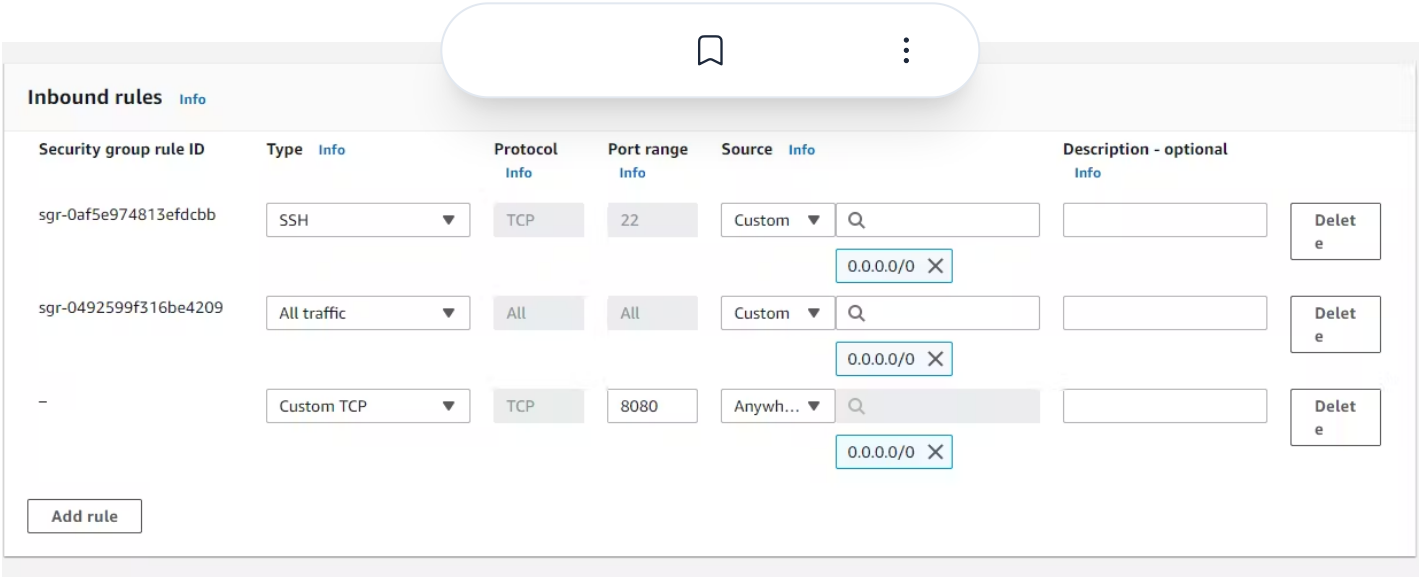
```
sudo apt install openjdk-11-jre -y
```

Now we have to paste the Jenkins commands

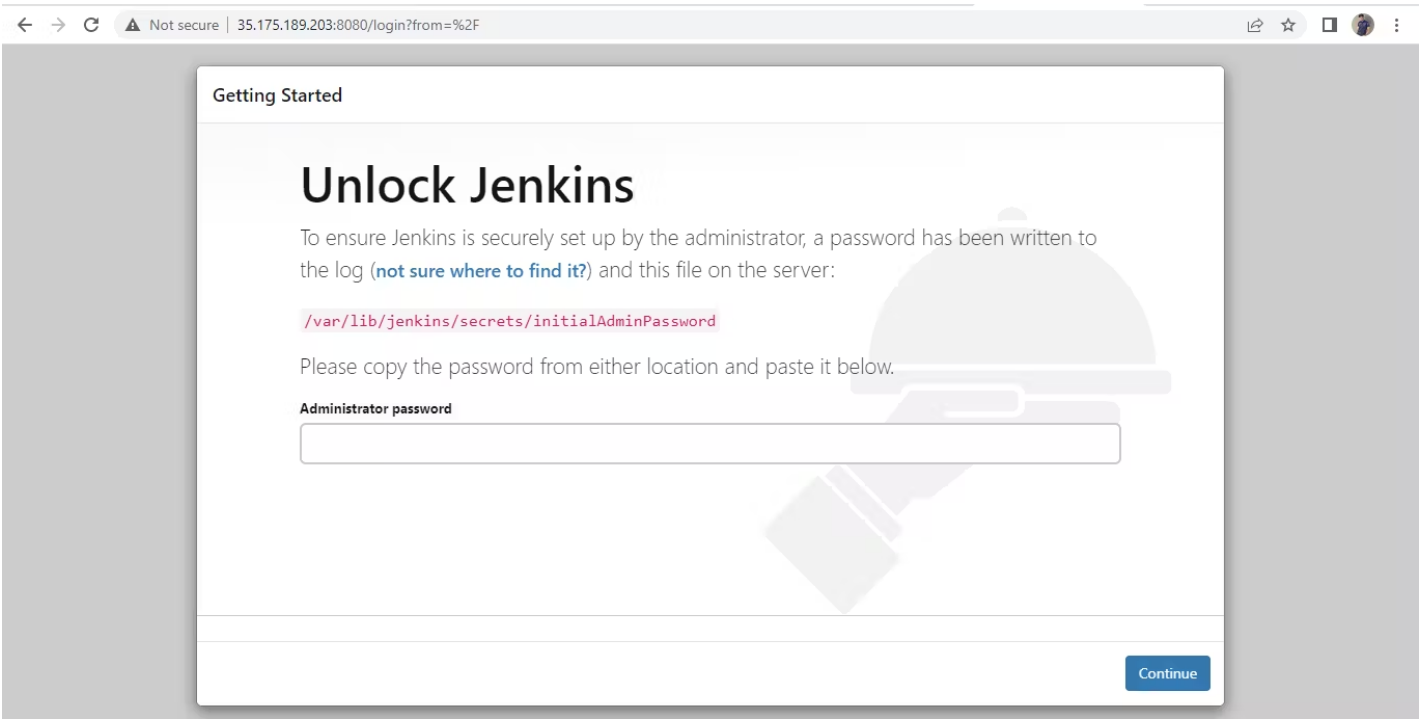
COPY

```
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
```

Now we have to allow port 8080 in the Jenkins server security group inbound settings



Now use the public URL and port 8080 to view the Jenkins server

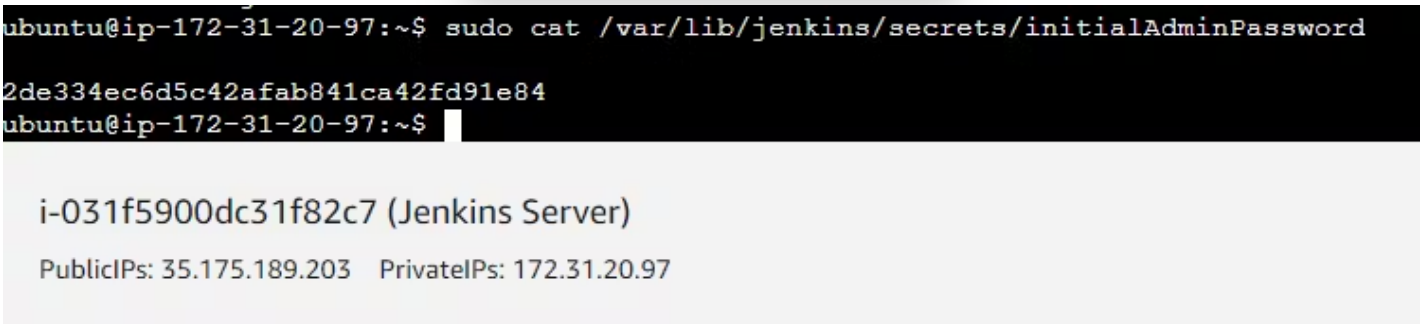


Copy the command which was highlighted in the above picture and use the command

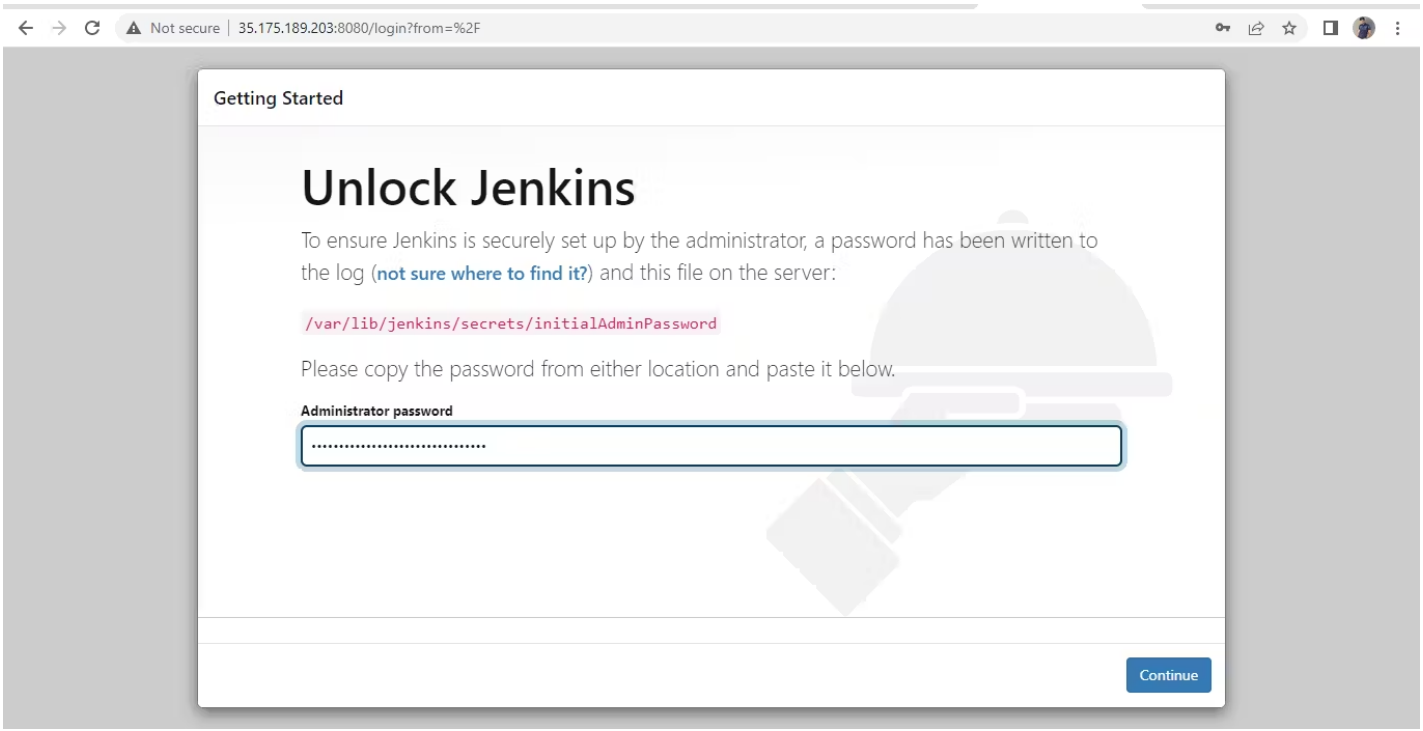
COPY

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

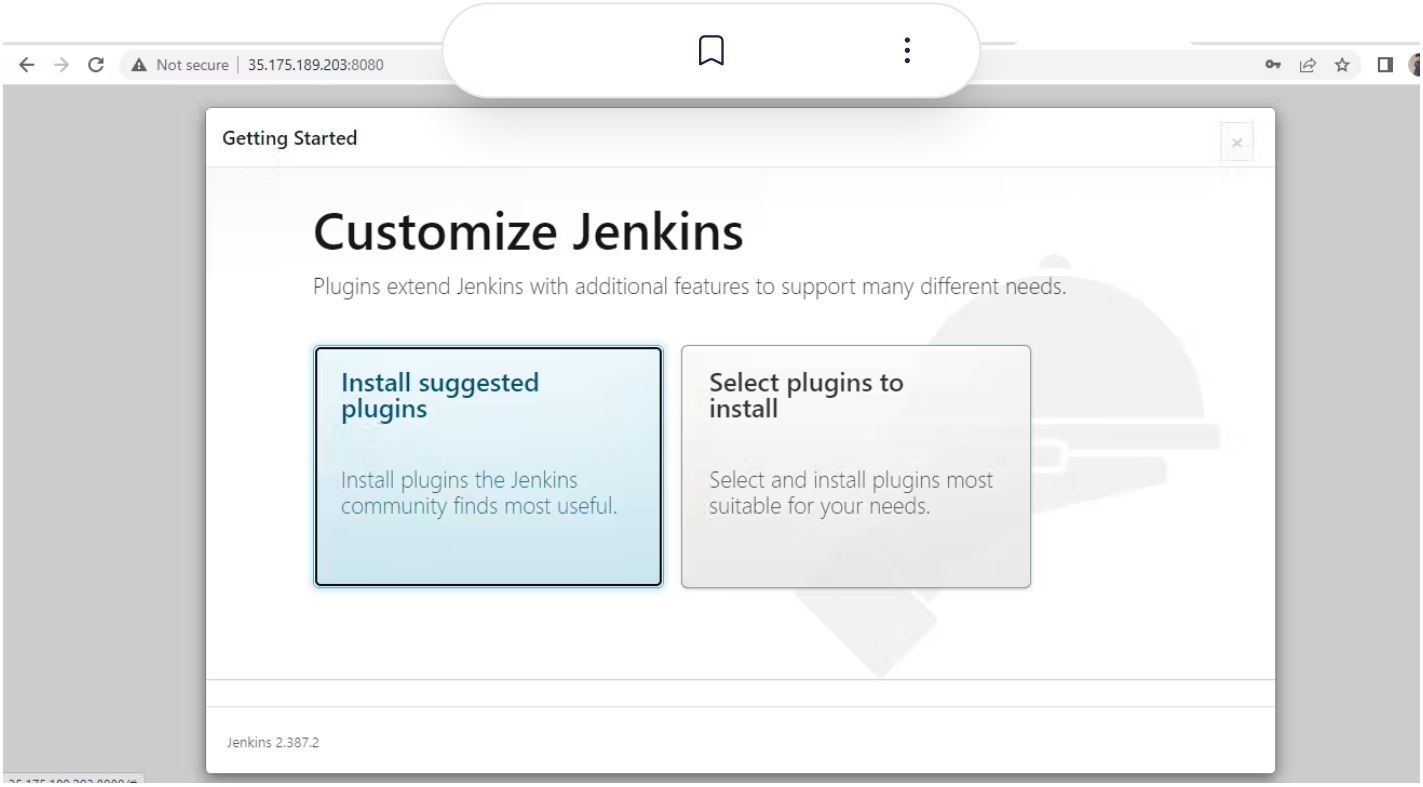
It will show the output a pa



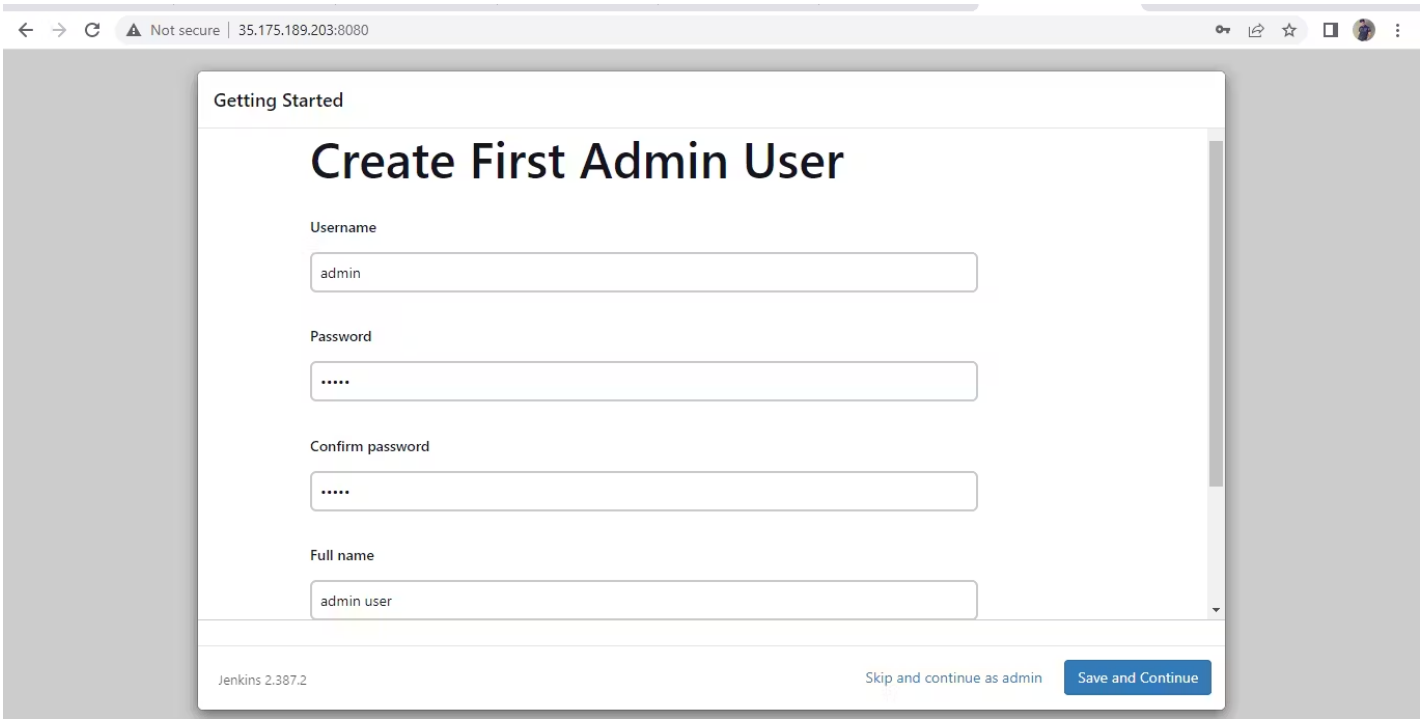
Copy the password and paste it to the Jenkins server



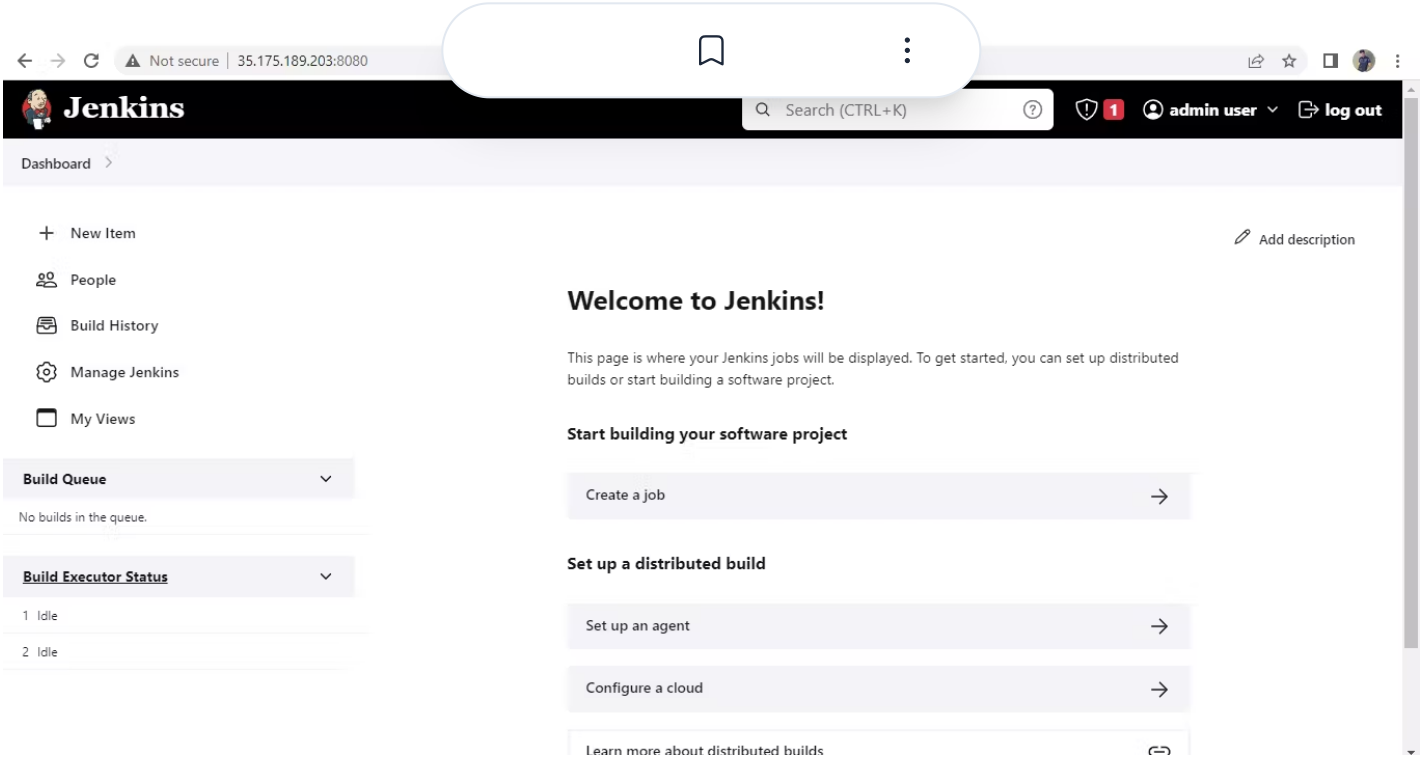
Install suggested plugins



You can set admin credentials

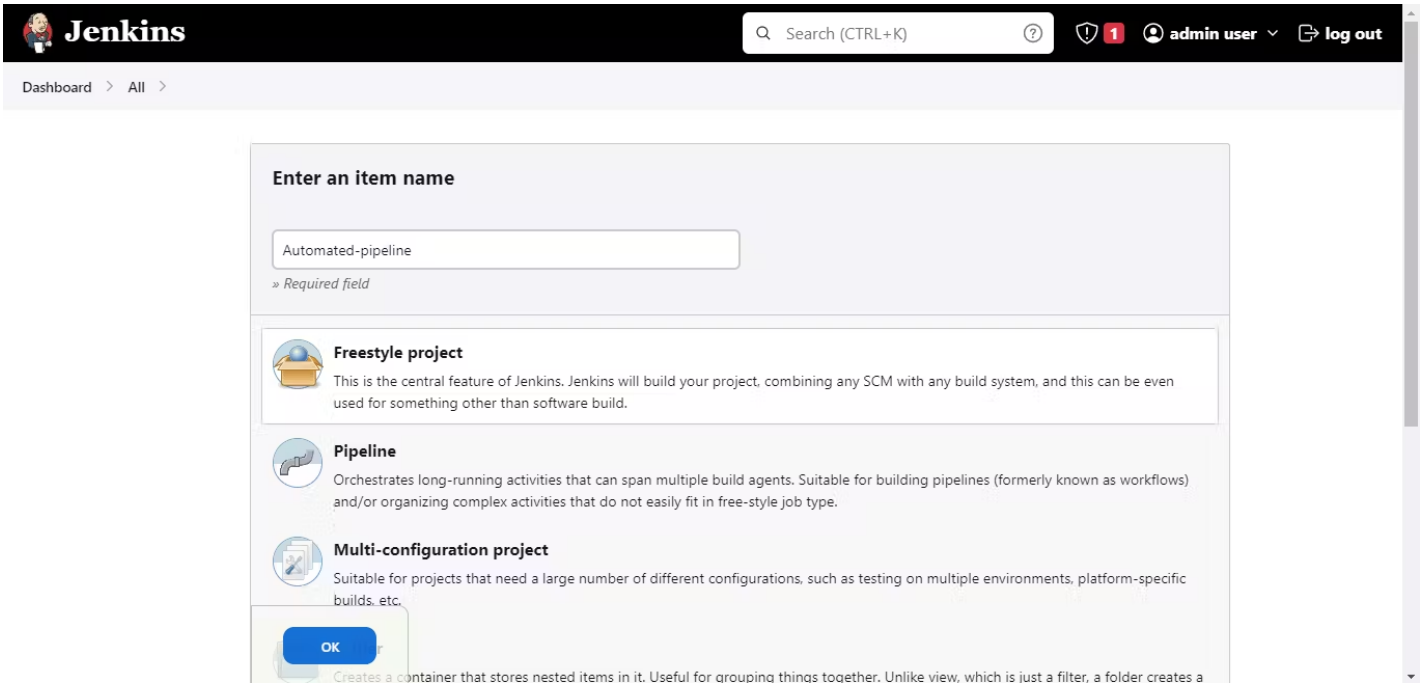


Jenkins Installation done

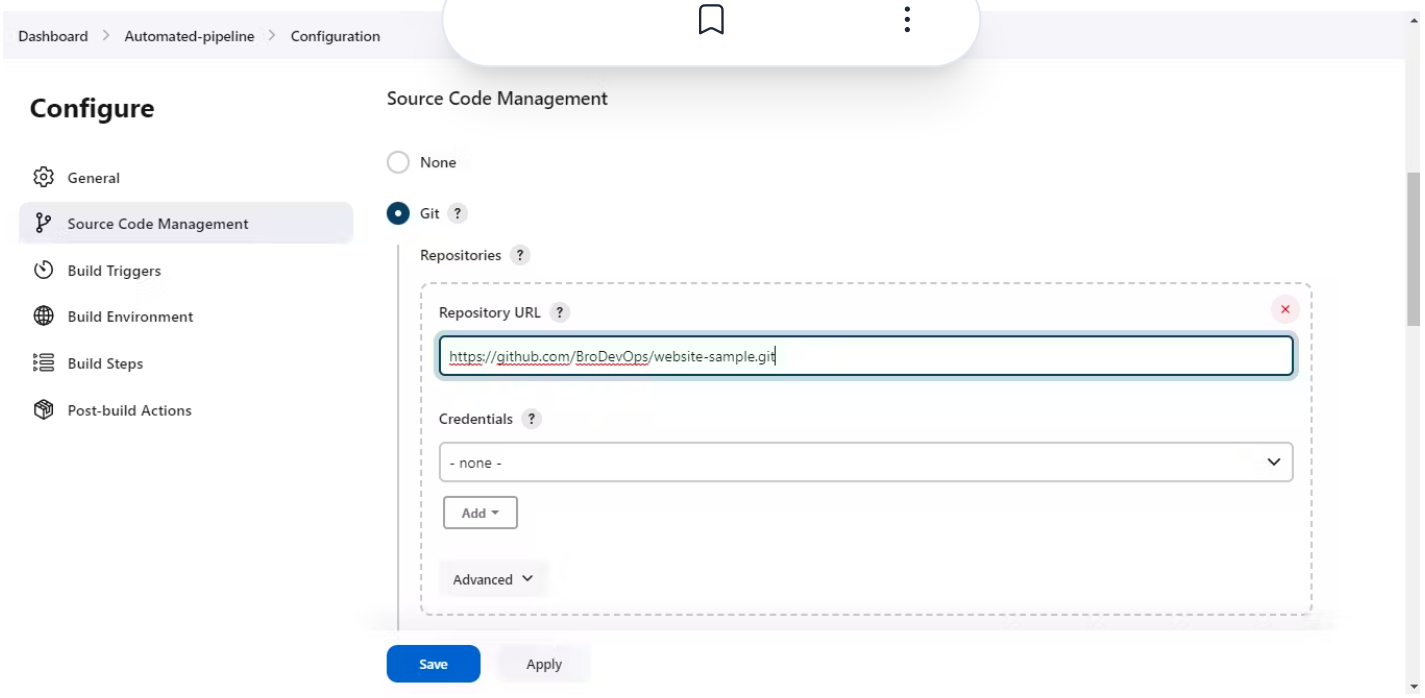


Creating a Pipeline in Jenkins Server

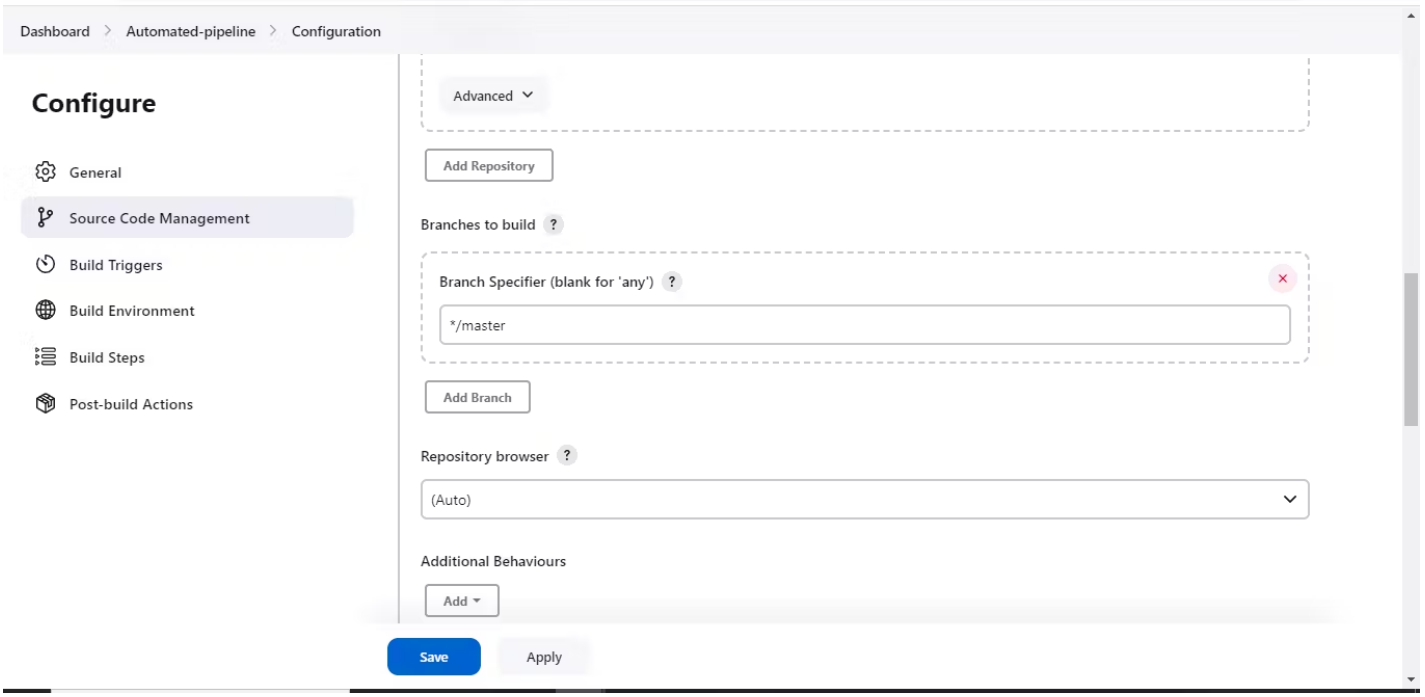
In this, I have selected Freestyle project



In the source code management, I have selected the Git option

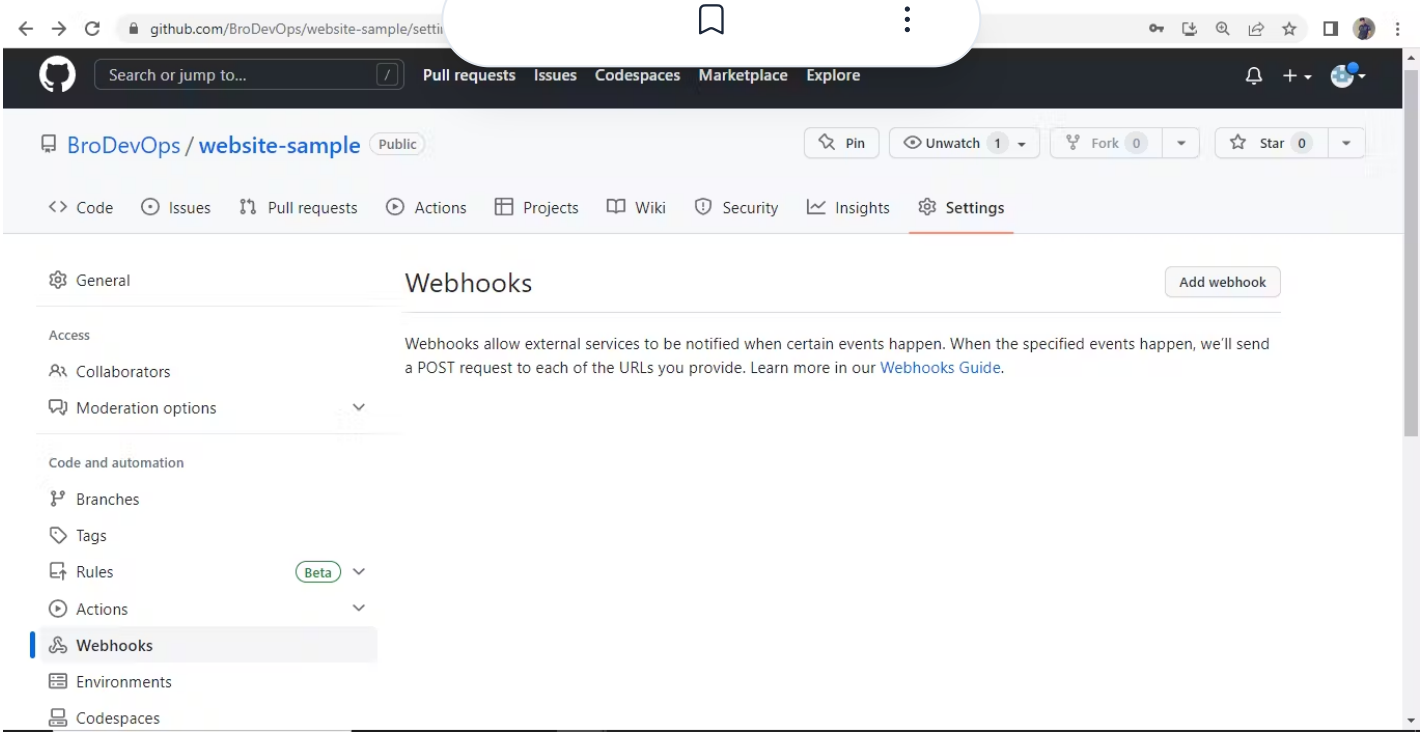


Also select branch of your git



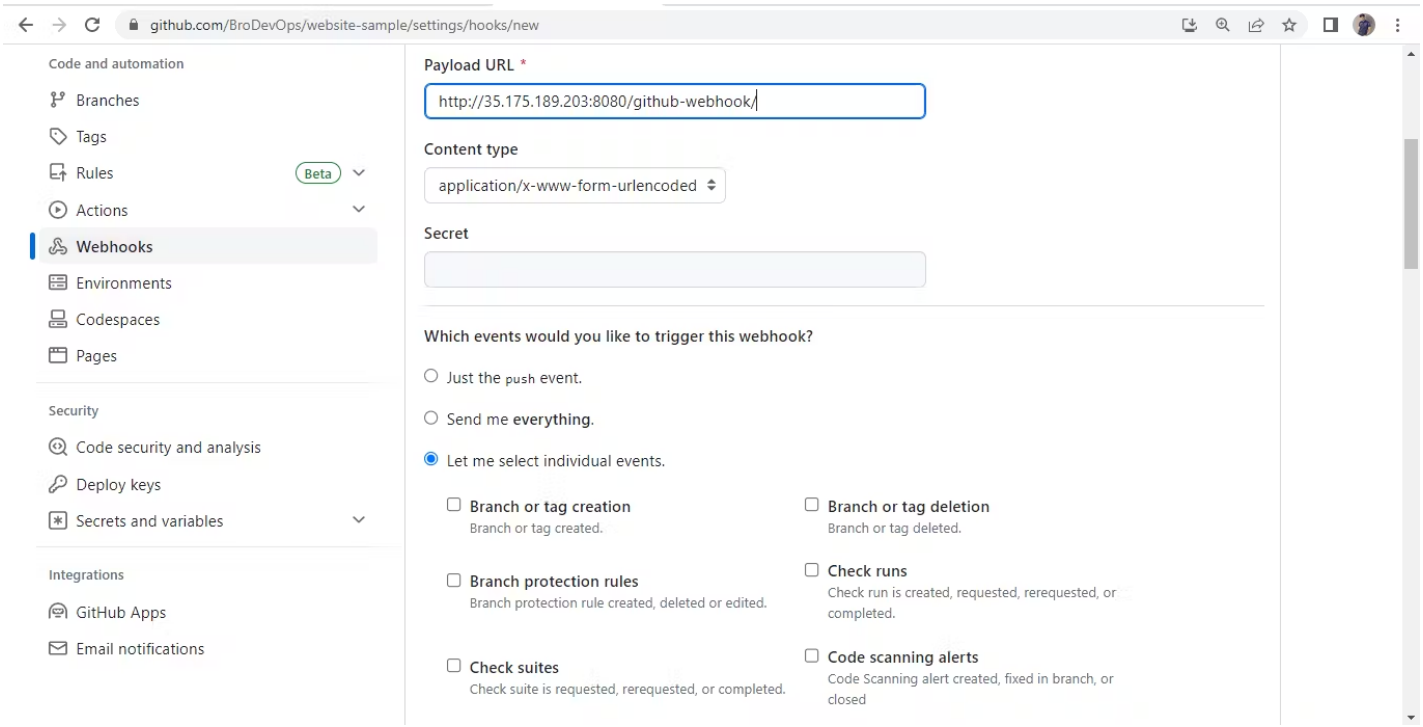
Adding a Webhooks

Now open the settings of your Git Repository, select webhooks

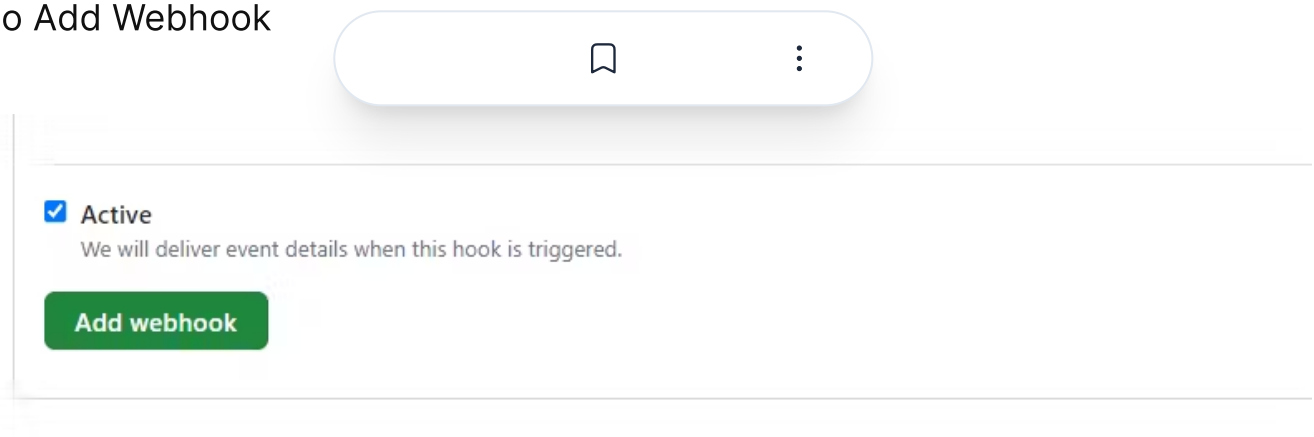


Click to Add Webhook, that time it will ask you to enter your password, once you enter your password, you're able to add Webhooks.

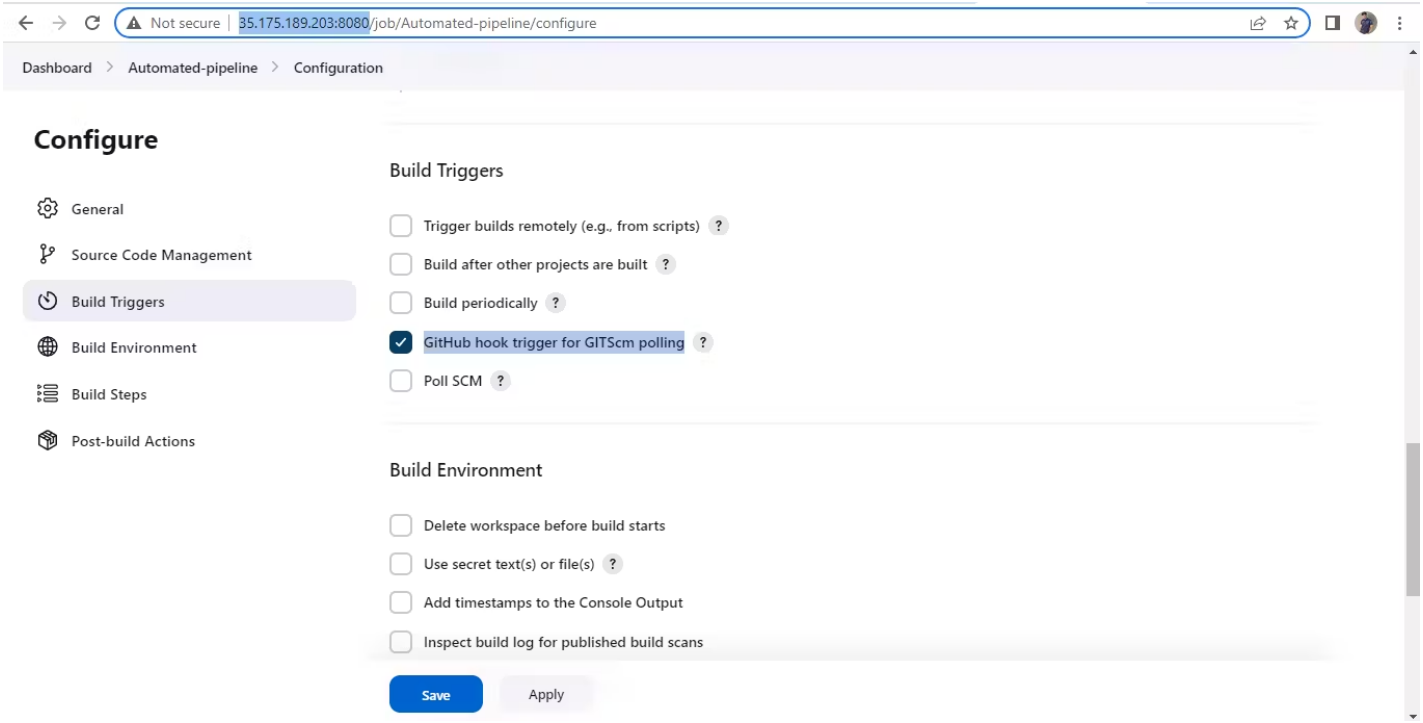
- Enter the Jenkins URL such as http://35.175.189.203:8080/
- and extension github-webhook after the URL



Click to Add Webhook

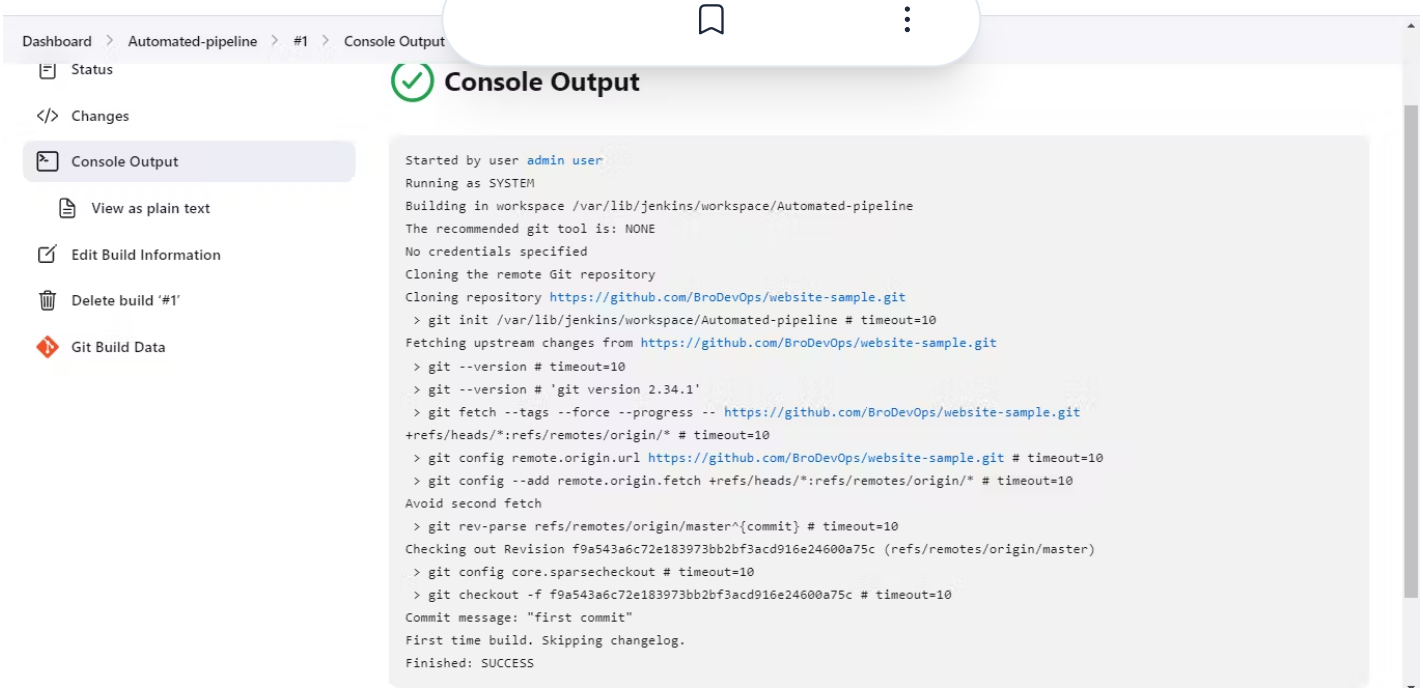


Now in Build Triggers, select the **GitHub hook trigger for GITScm polling** (because this function can trigger the pipeline automatically whenever we make changes to the repository.

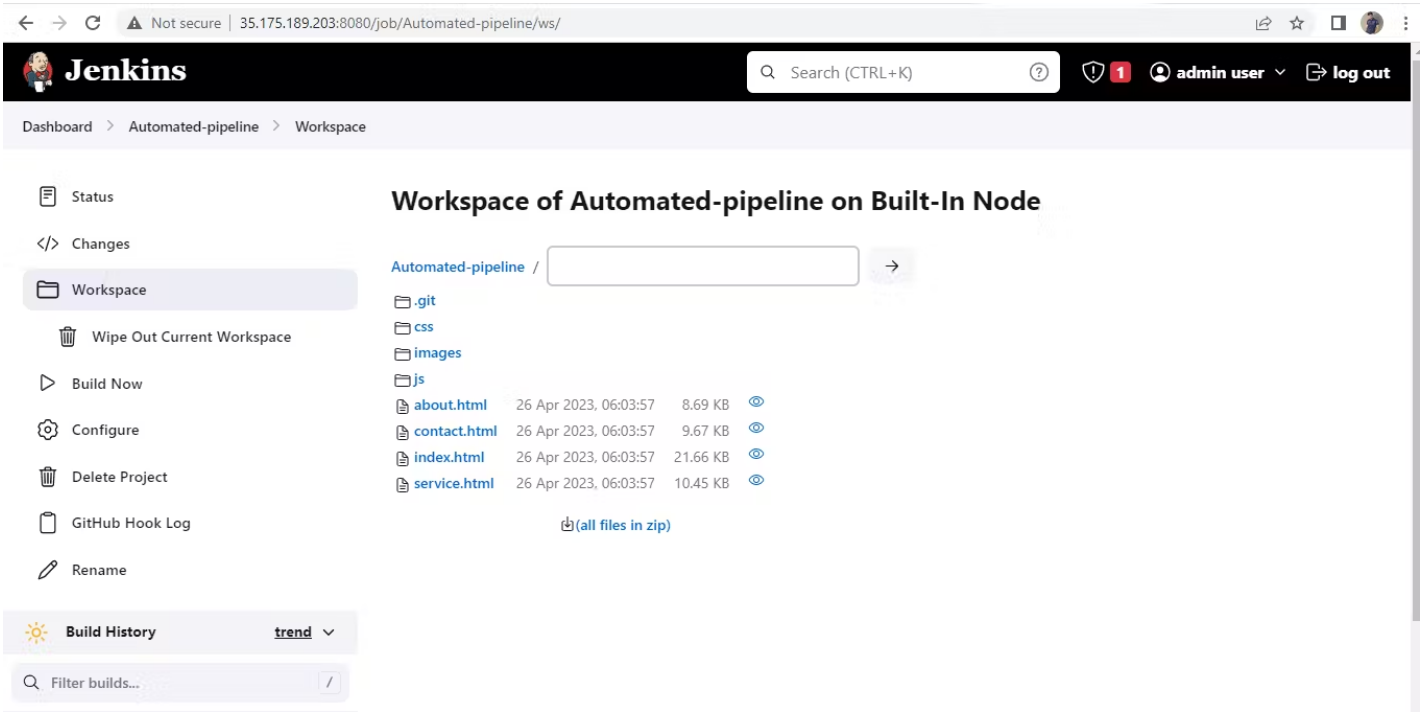


Click to save.

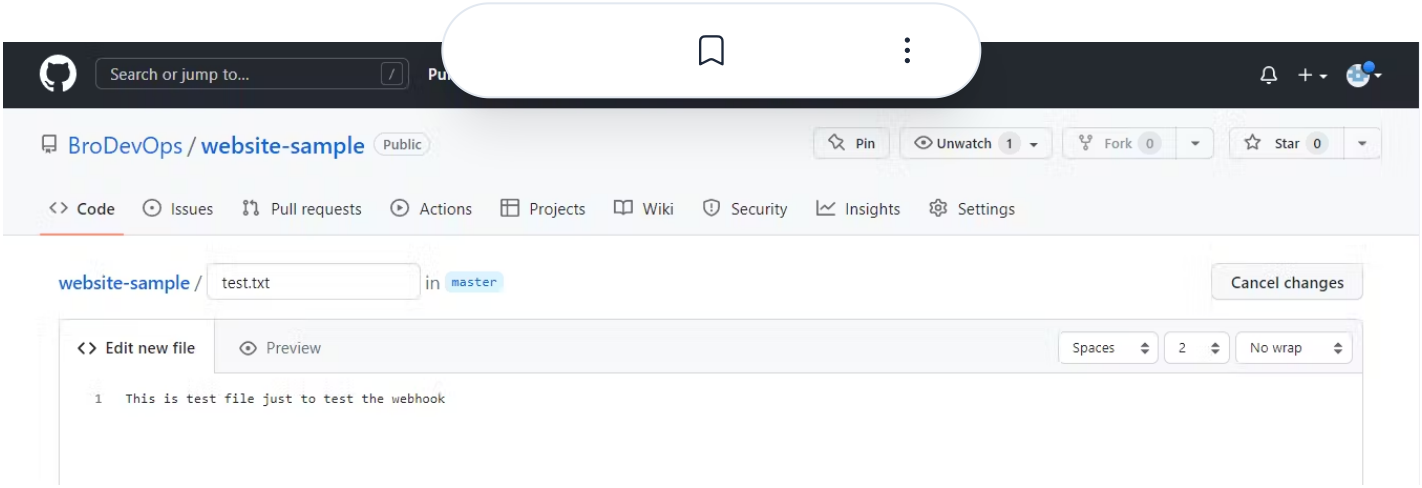
Without Webhook I have clicked to Build now and it is working perfectly fine



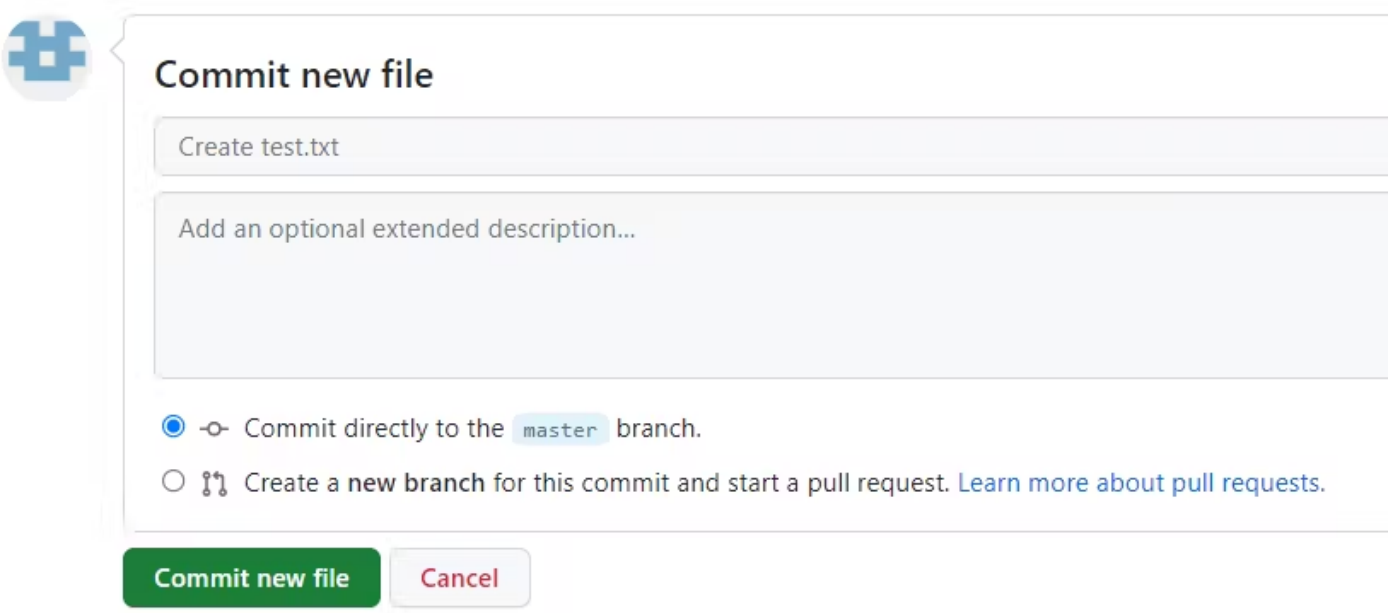
Now time to verify Webhooks, I have clicked on the workspace in Jenkins, and here text.txt file is not present which I am going to create to test the Webhook



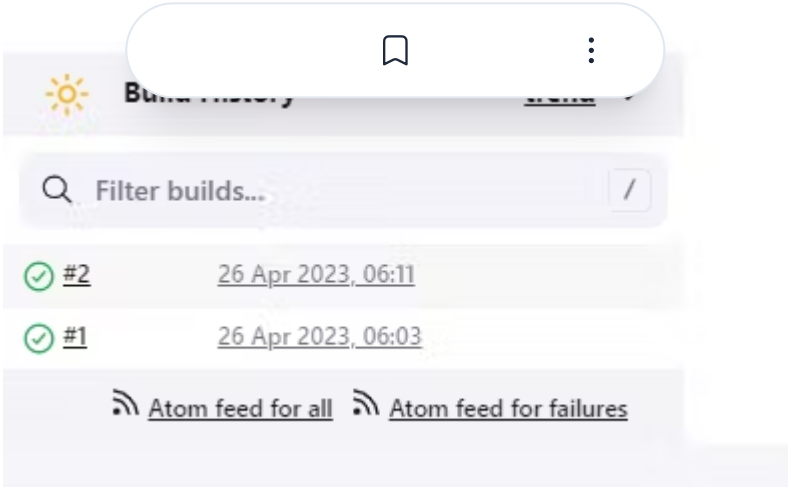
I have visited the Git Repository and created a new file by the name of test.txt

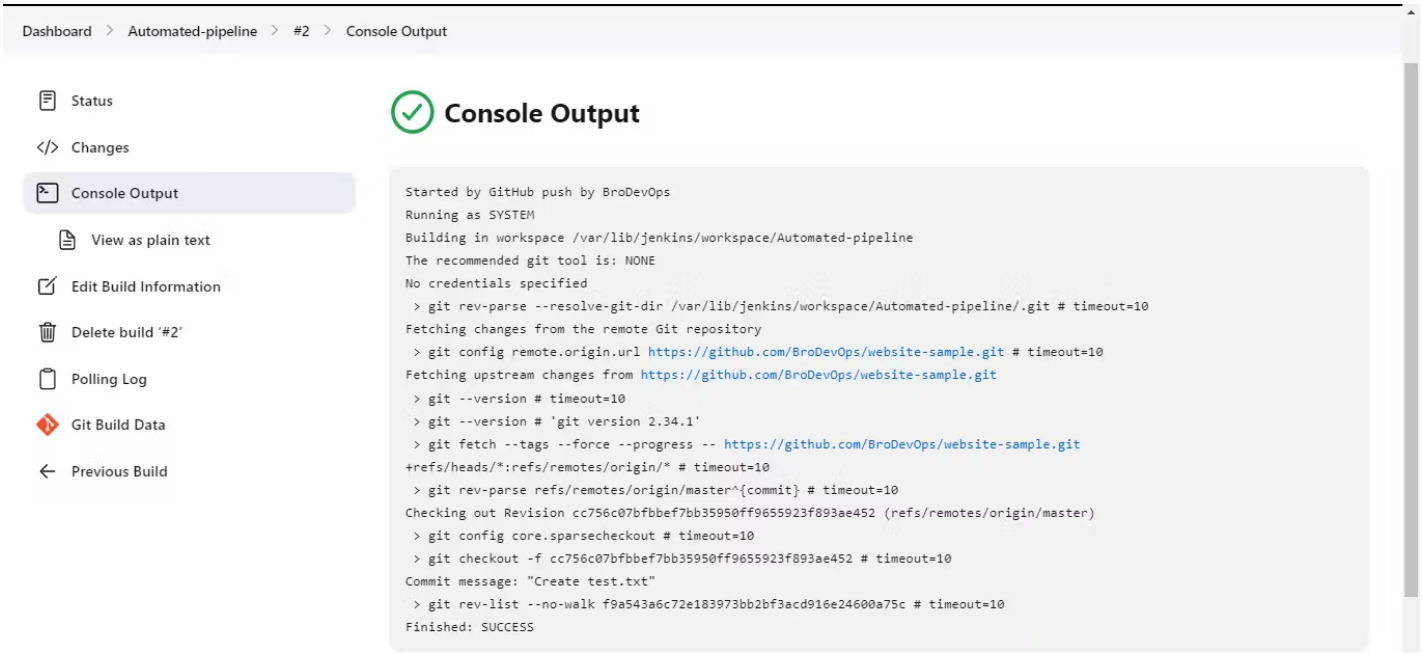


I committed the file

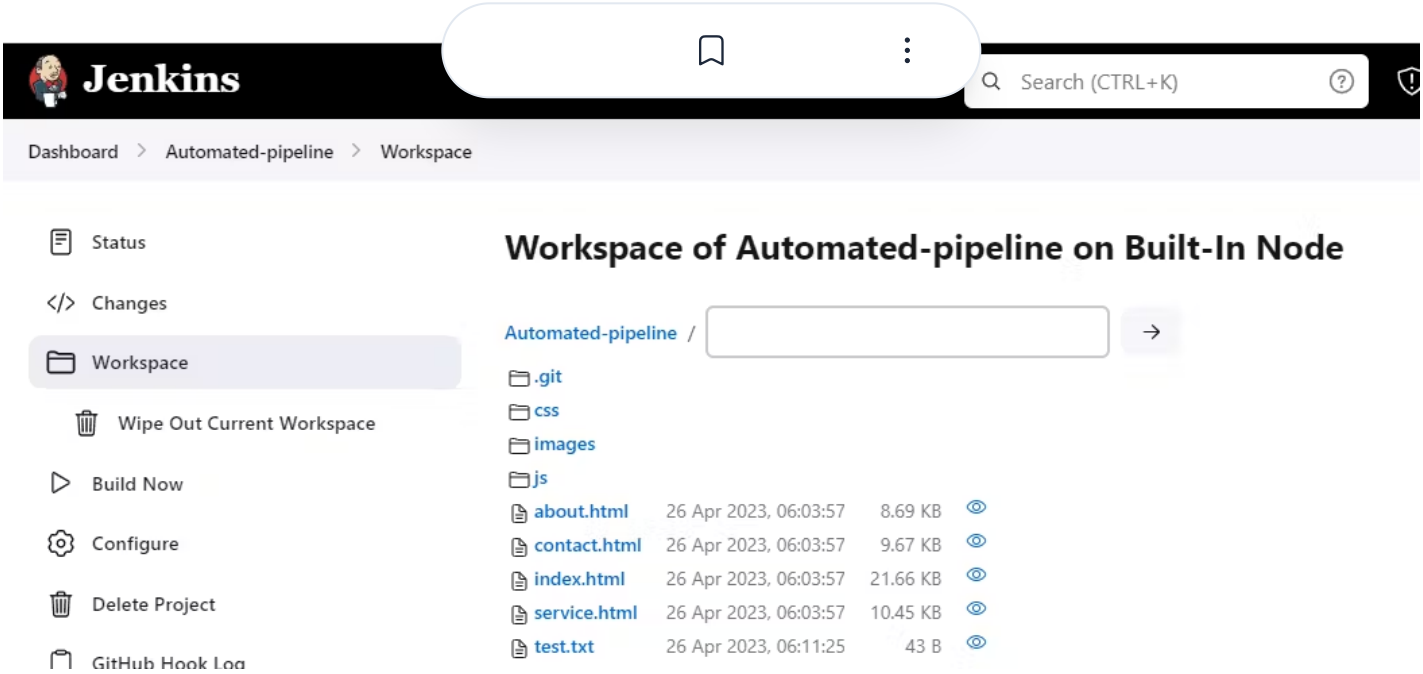


Now I am back to the Jenkins server to check #2 build auto trigger and it is working fine.





Now test.txt file is showing in the workspace



Starting a SonarQube Server

Start the **Sonarqube Instance**

COPY

```
sudo apt update
clear
```

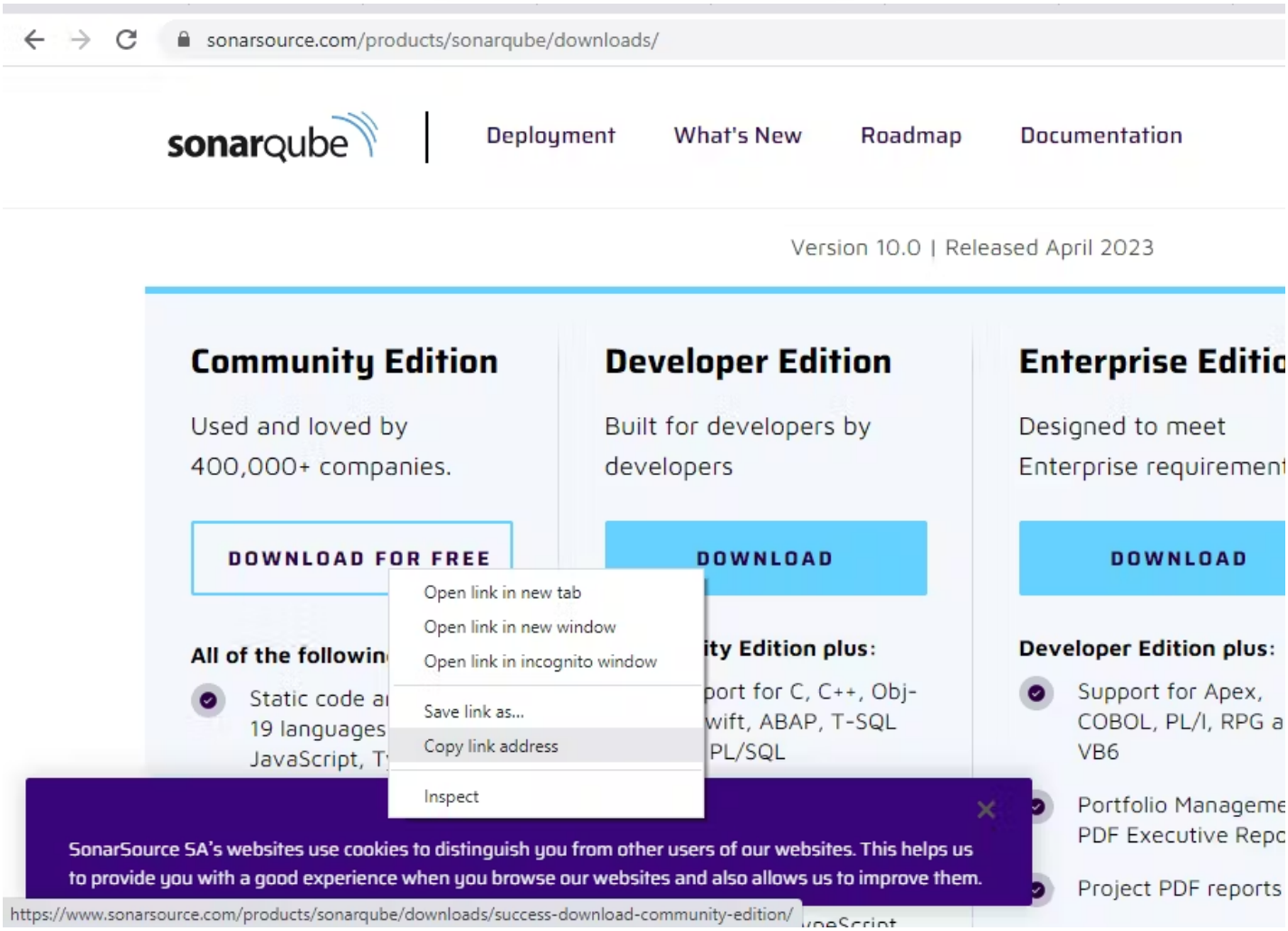
We need to install Java on the server but 17 version earlier we were using 11 version

COPY

```
sudo apt install openjdk-17-jre -y
```

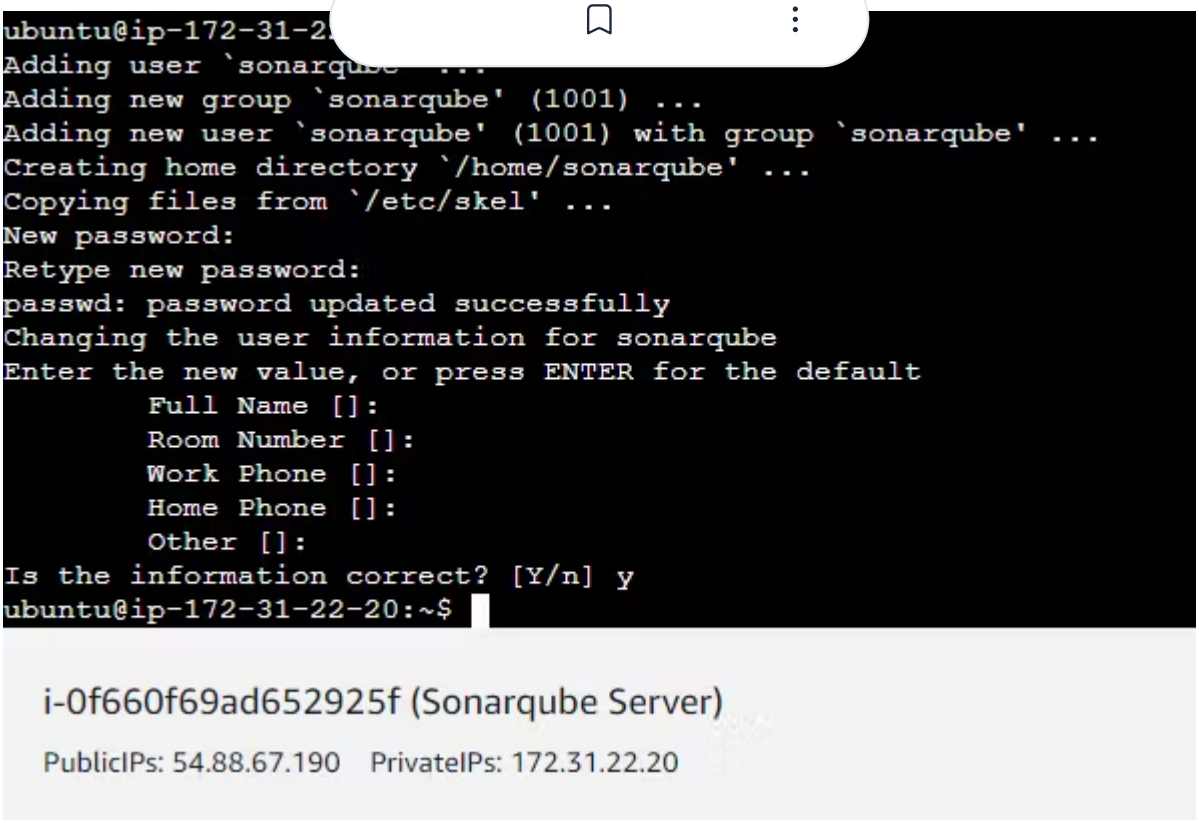
Search for the **SonarQube** website and download the community for the free version

SonarQube Download Copy the link and paste it in the terminal of the SonarQube Instance



COPY

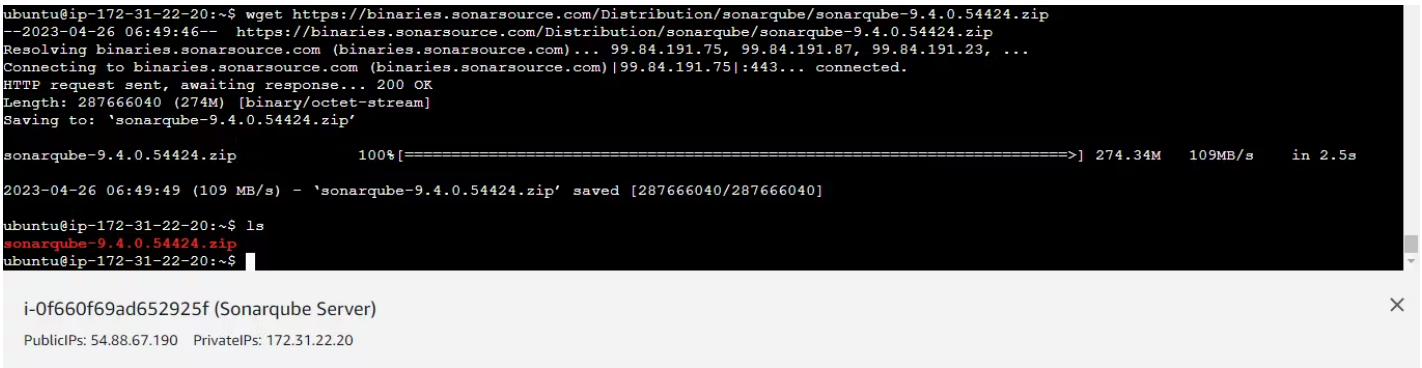
```
sudo adduser sonarqube
```



Use the command below with wget

COPY

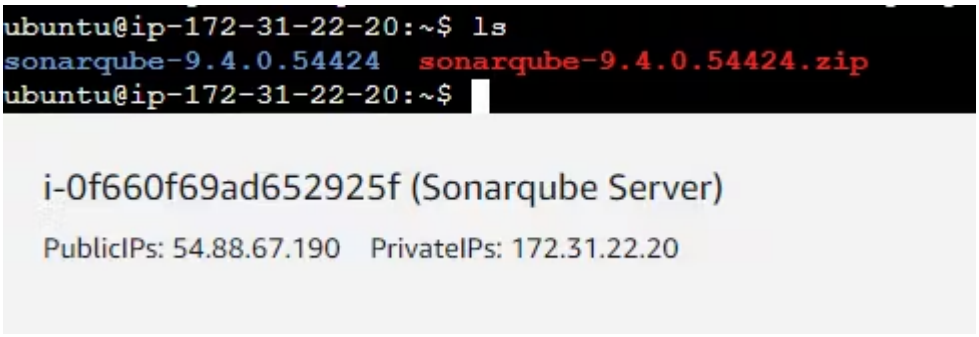
```
wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-9.4.0.5
```



COPY

```
sudo apt install unzip
unzip *
ls
```

Now the unzipped file is showing



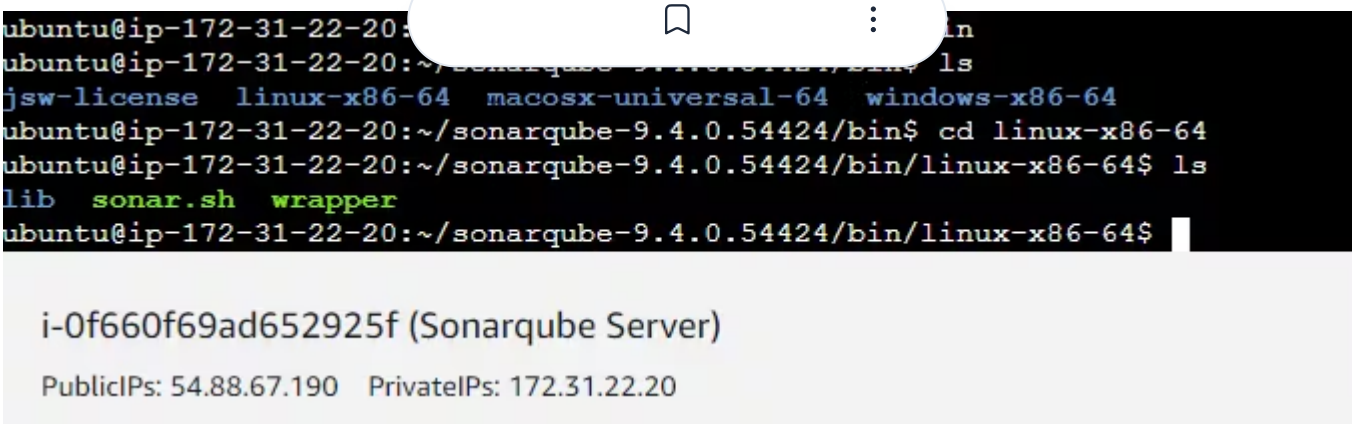
COPY

```
cd sonarqube-9.4.0.54424
ls
```

Go to the bin folder

COPY

```
cd bin
cd linux-x86-64
ls
```

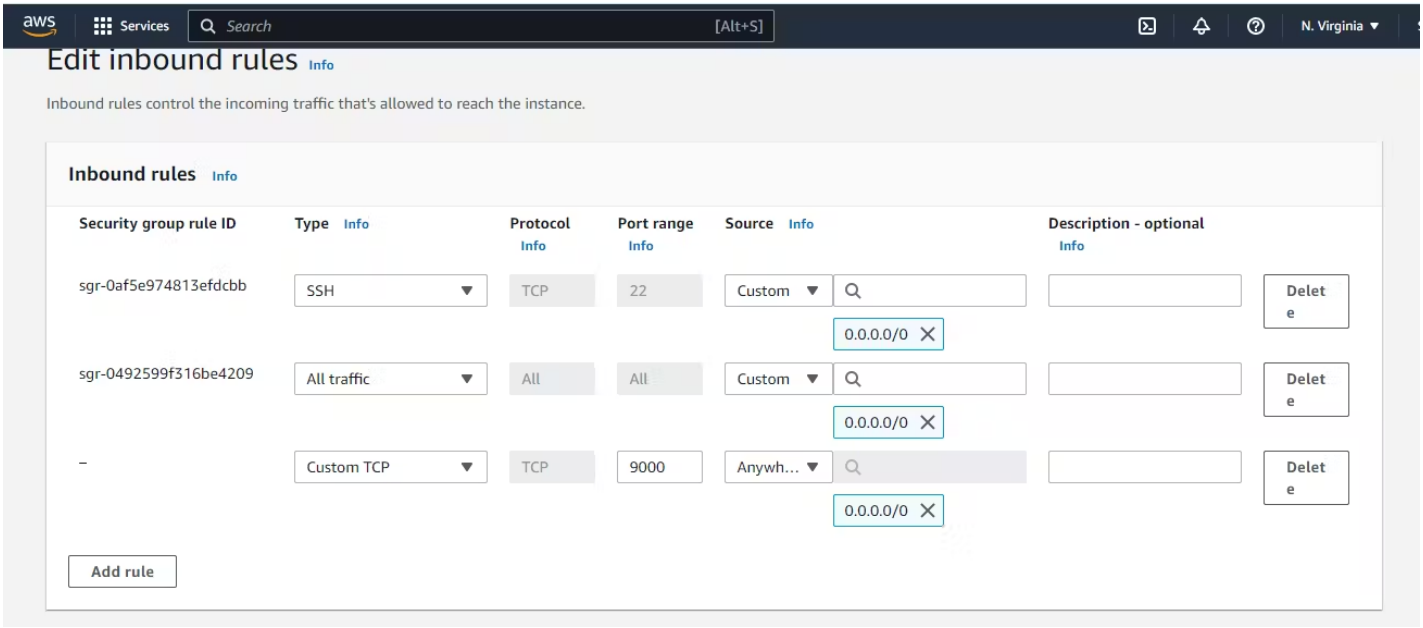


Now start the sonar server by using the command

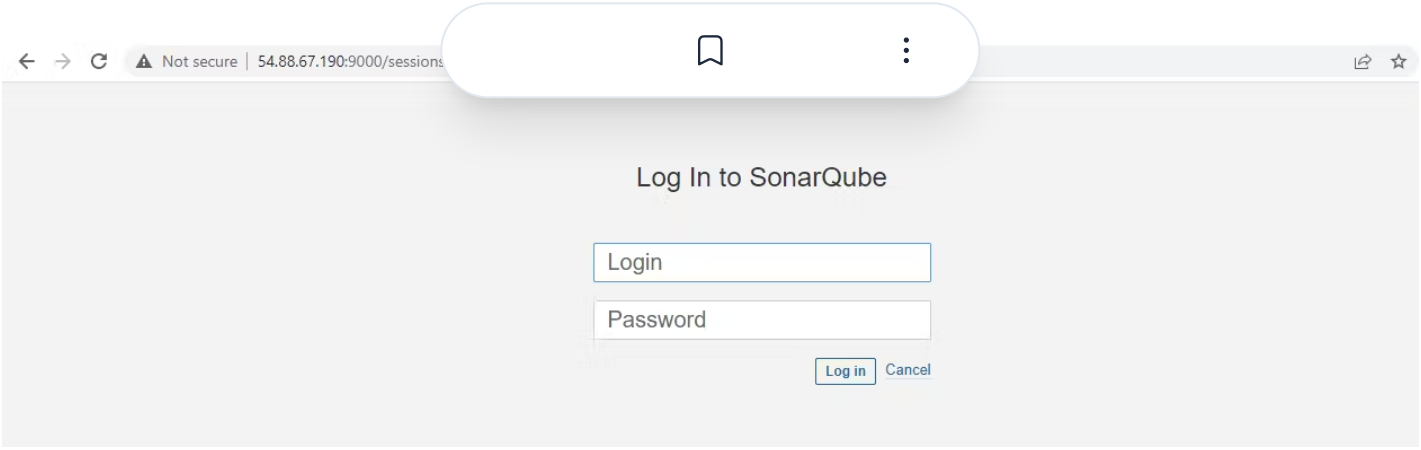
COPY

```
./sonar.sh start
```

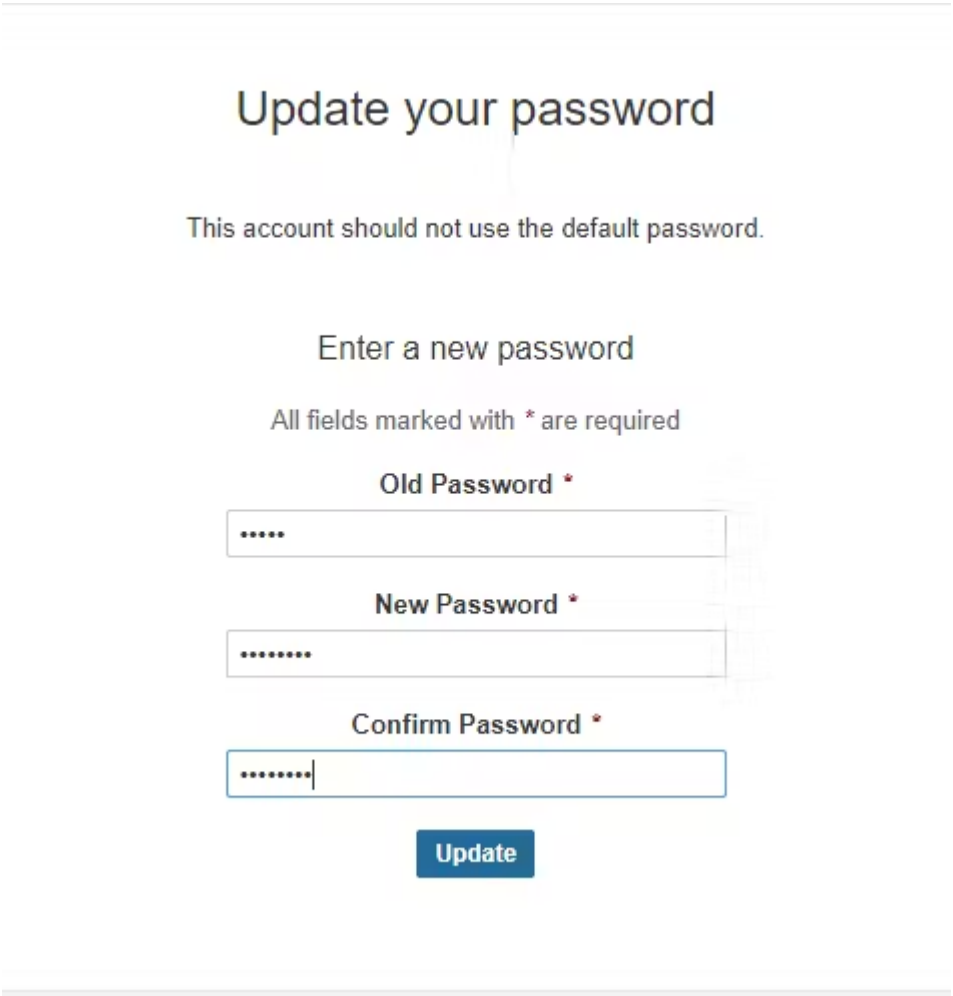
And also we need to port 9000 in the inbound setting of the security group



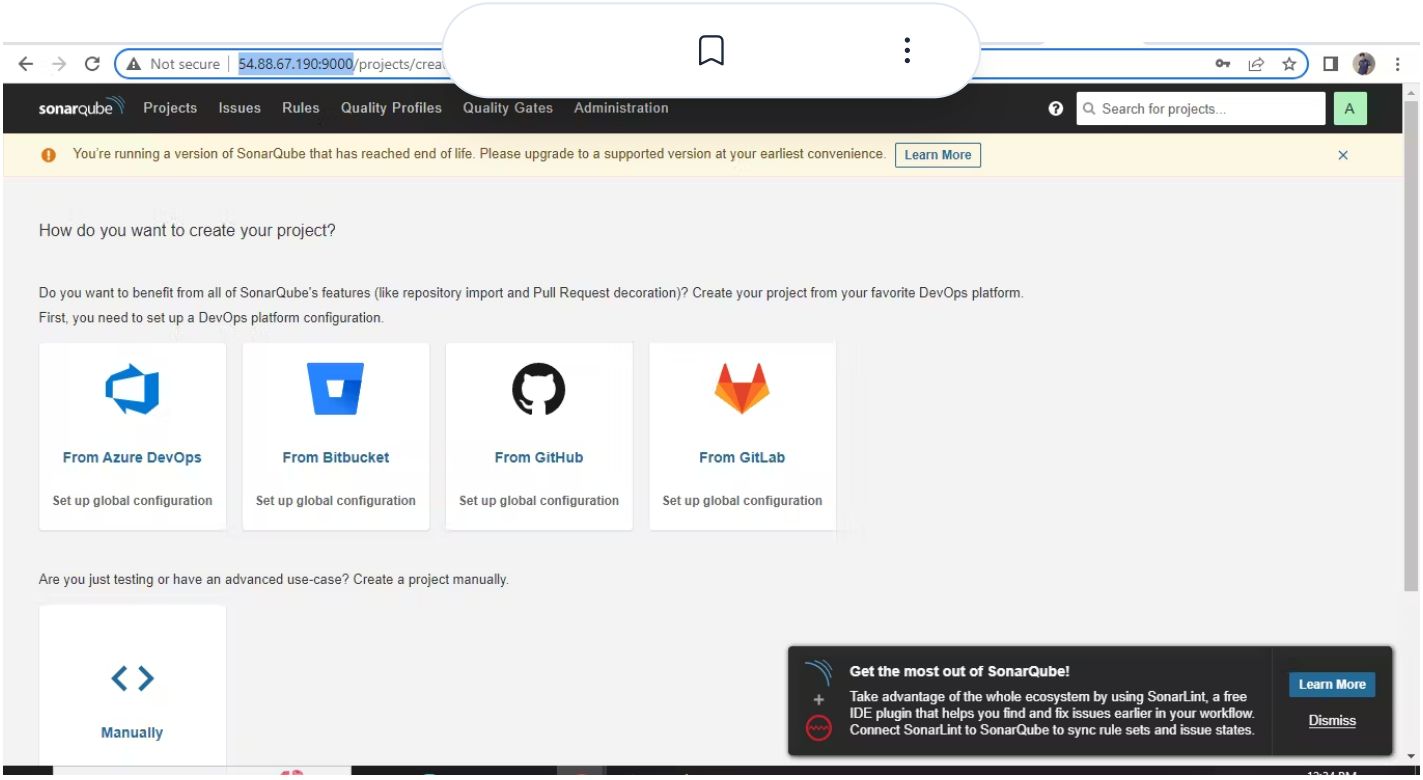
Now it's time to check the URL http://54.88.67.190:9000/



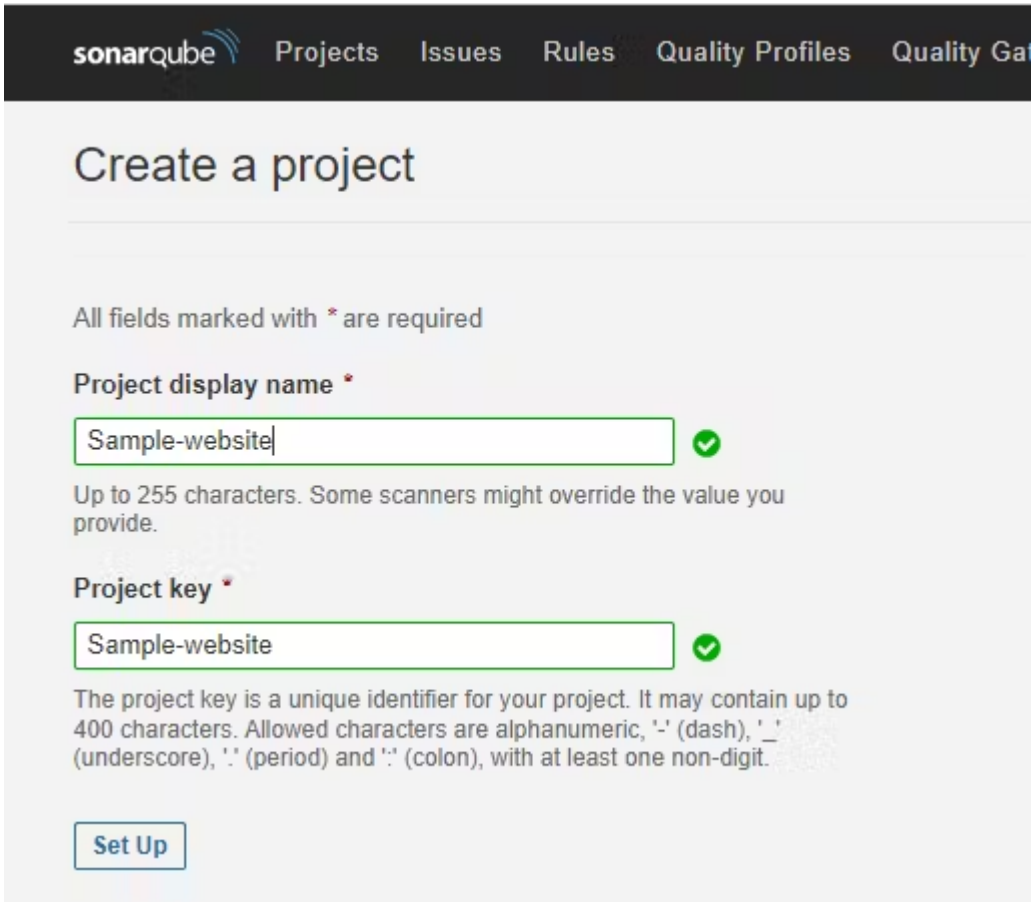
Update the password



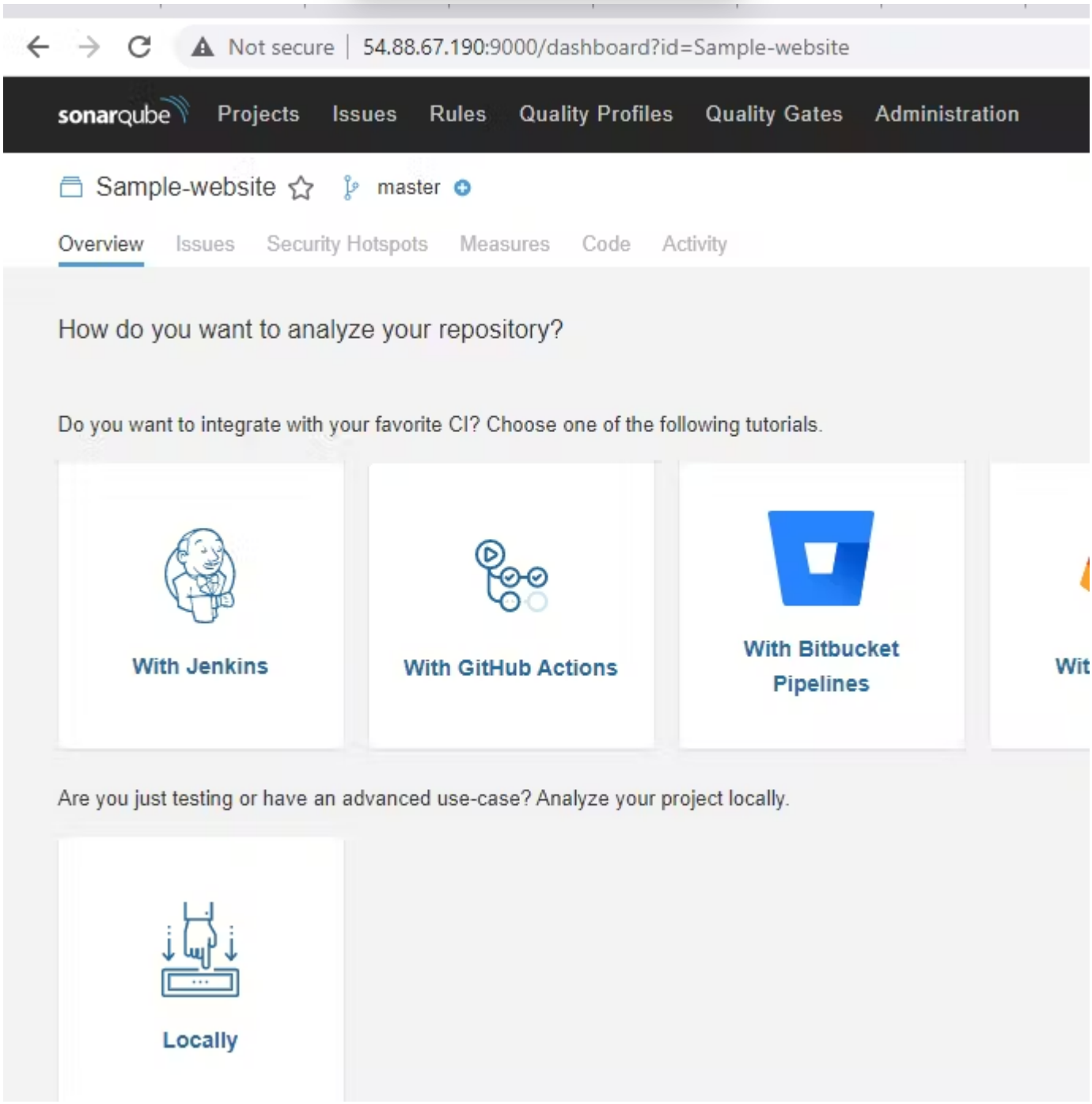
After updating the password we can access the Sonarqube



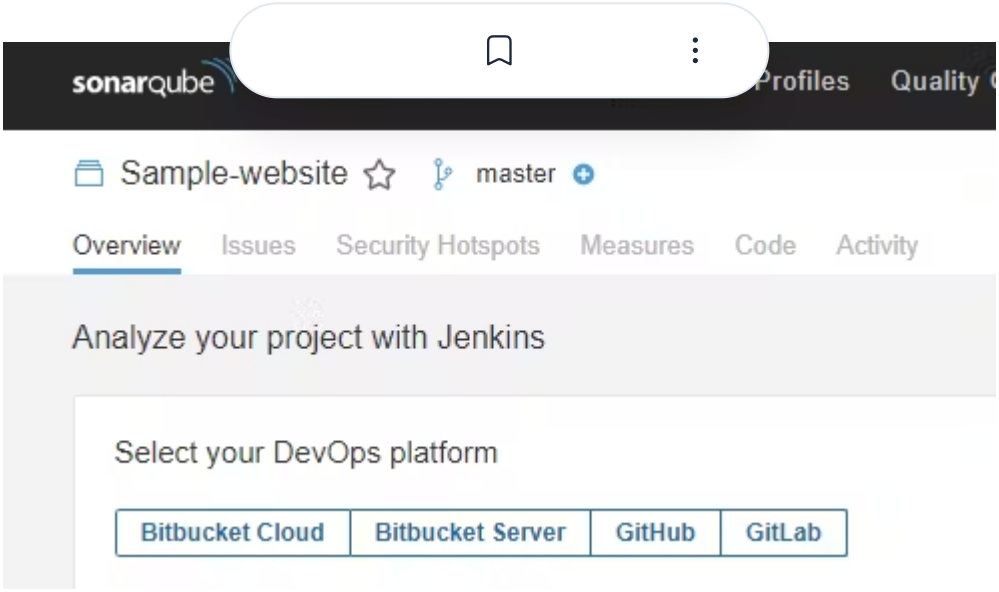
Now I have selected the project type **Manually**



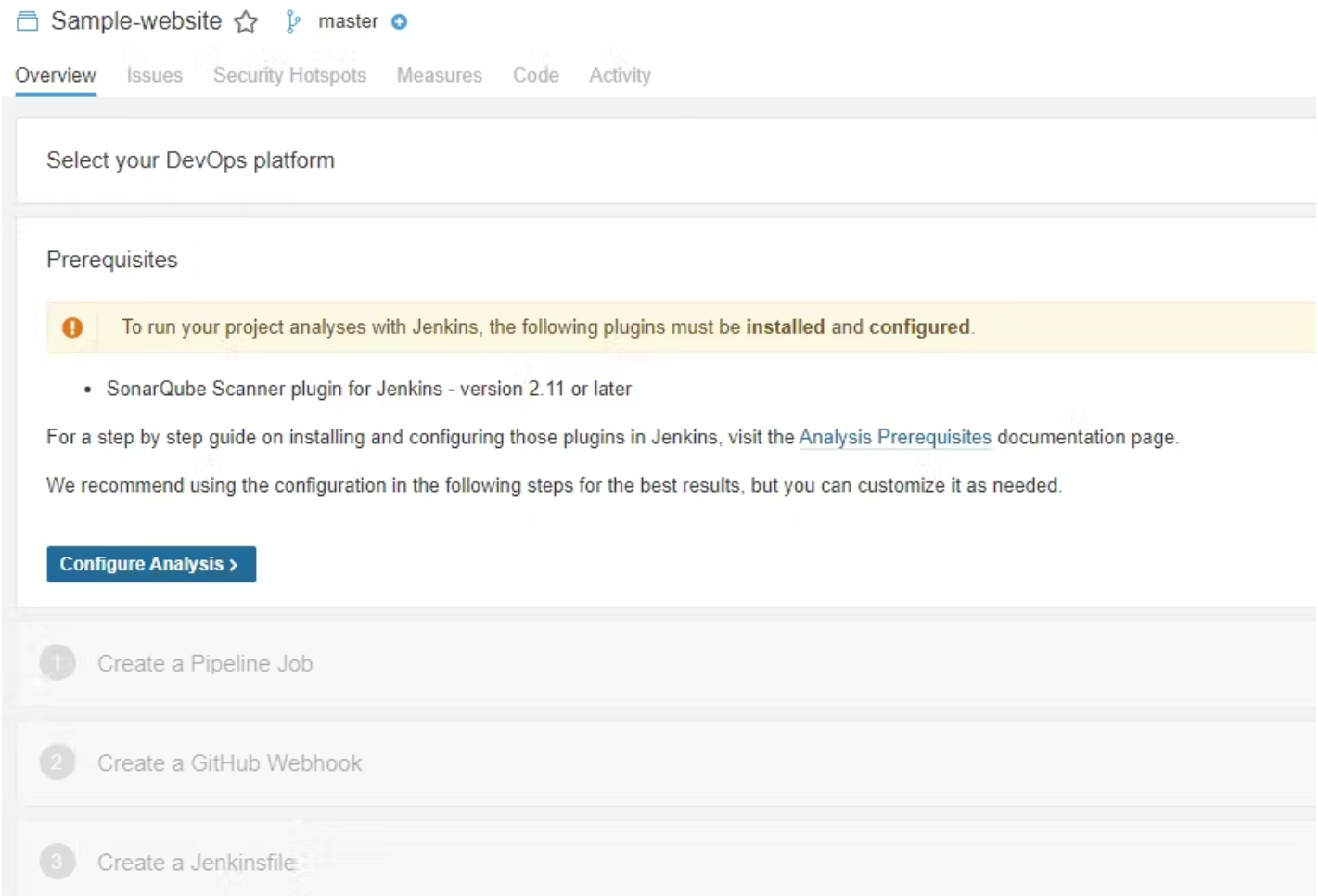
After this, I have selected th[illegible]



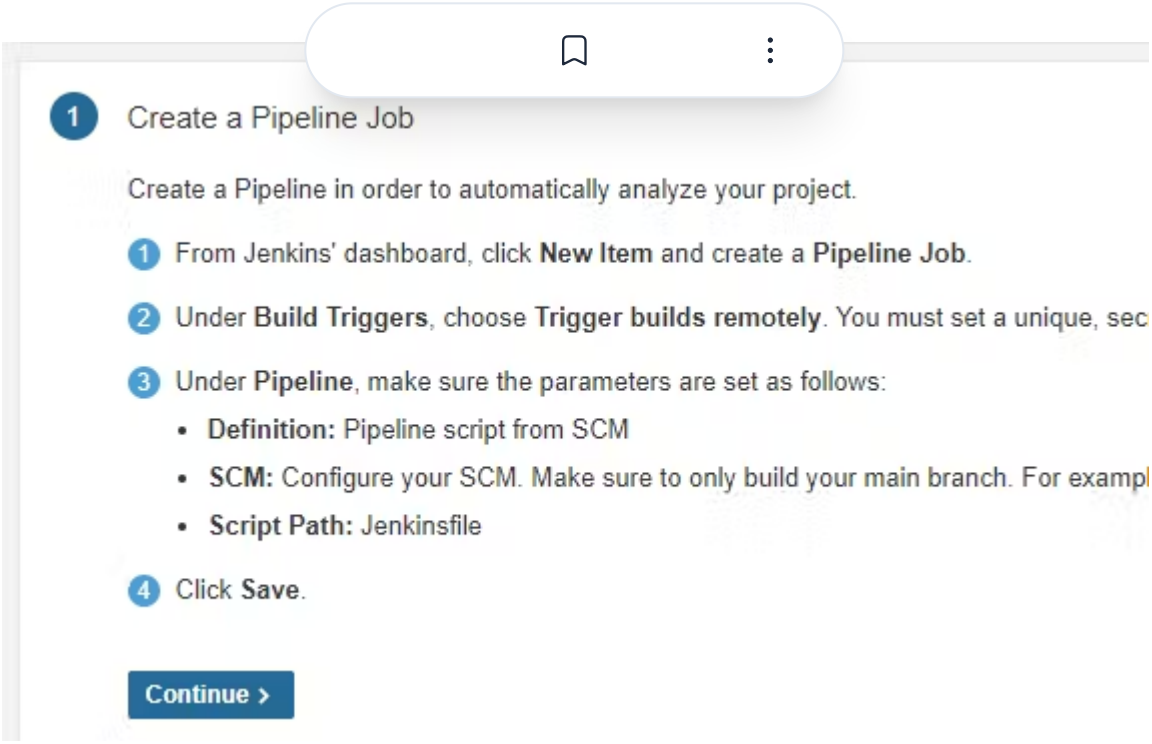
Selected the DevOps platform **GitHub**



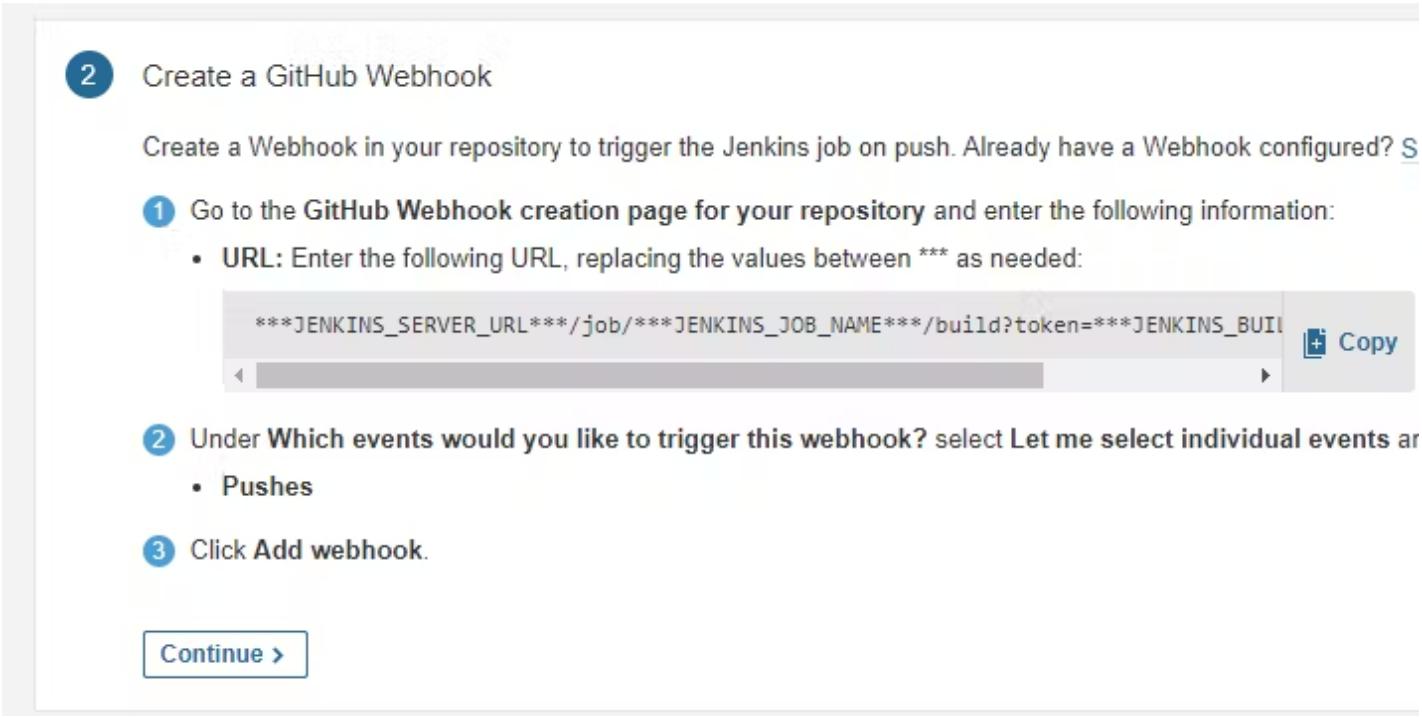
You can just simply select the options **Configure Analysis**



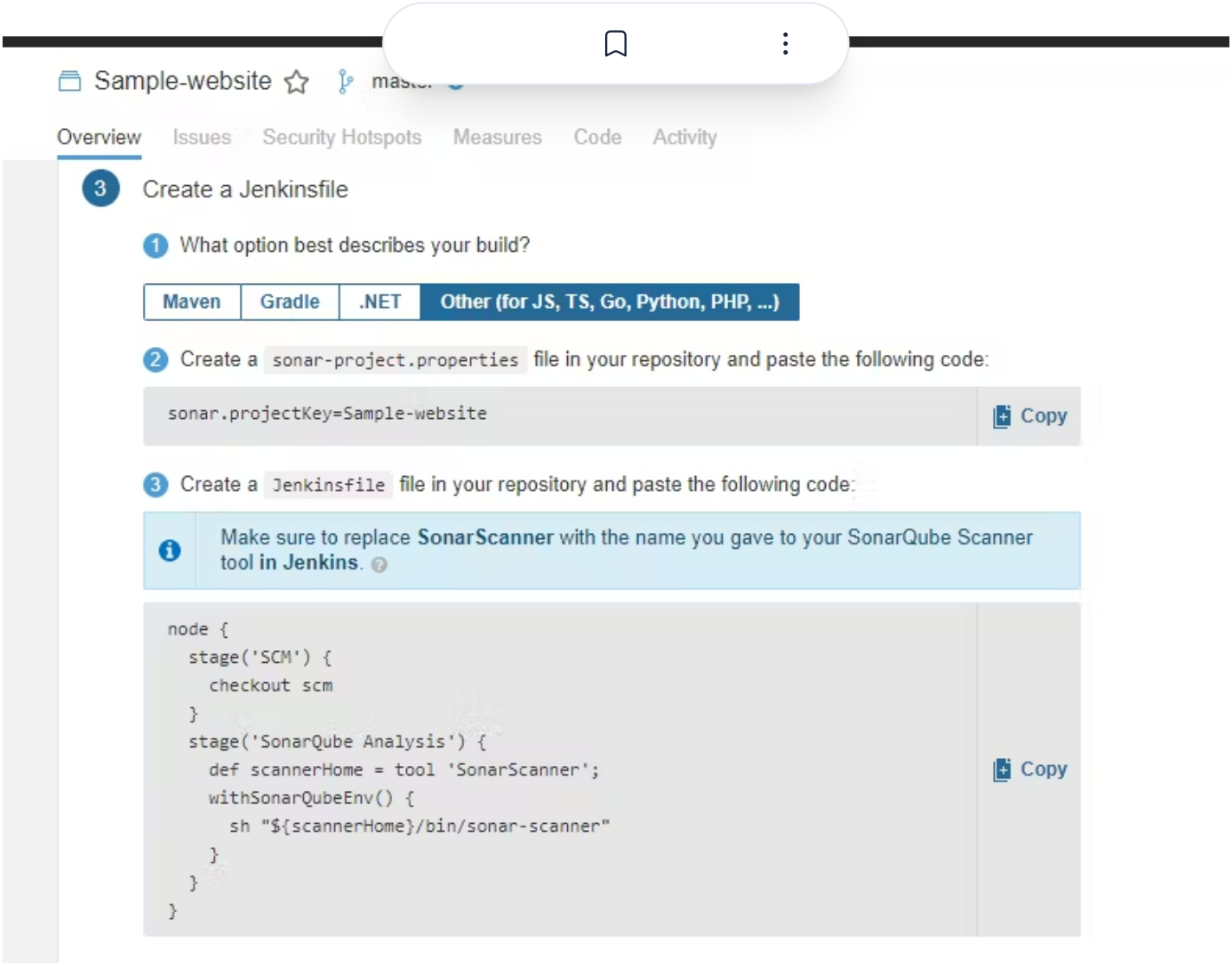
Then select **Continue**



Also, click on **Continue** here



From here I have selected the **Others**

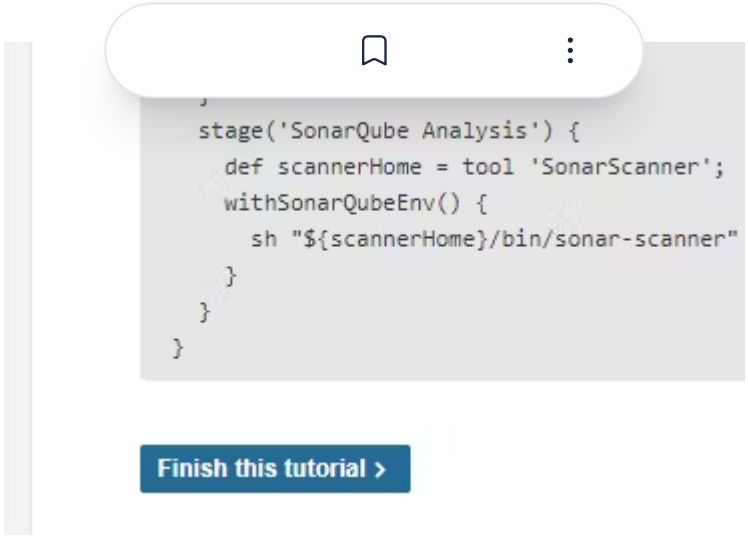


Copy the ProjectKey for now you can keep this Key in your notepad

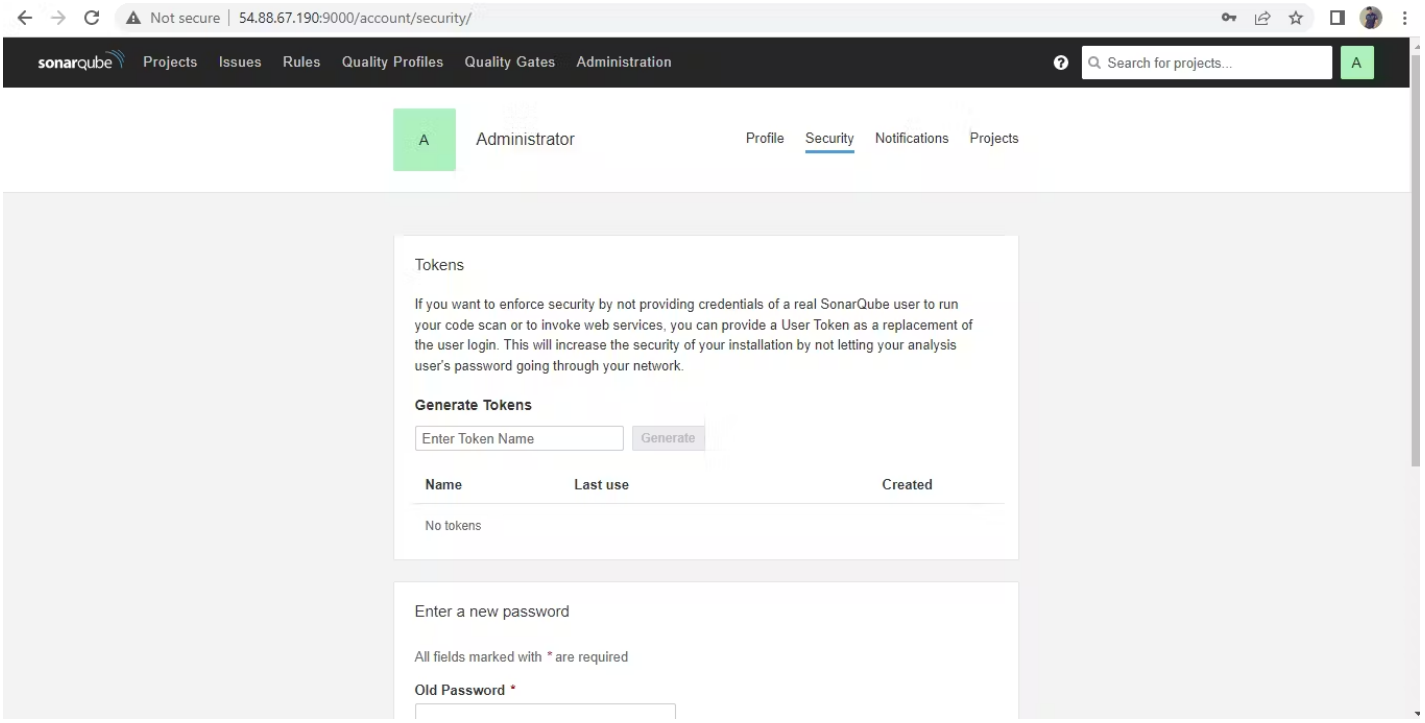
COPY

```
sonar.projectKey=Sample-website
```

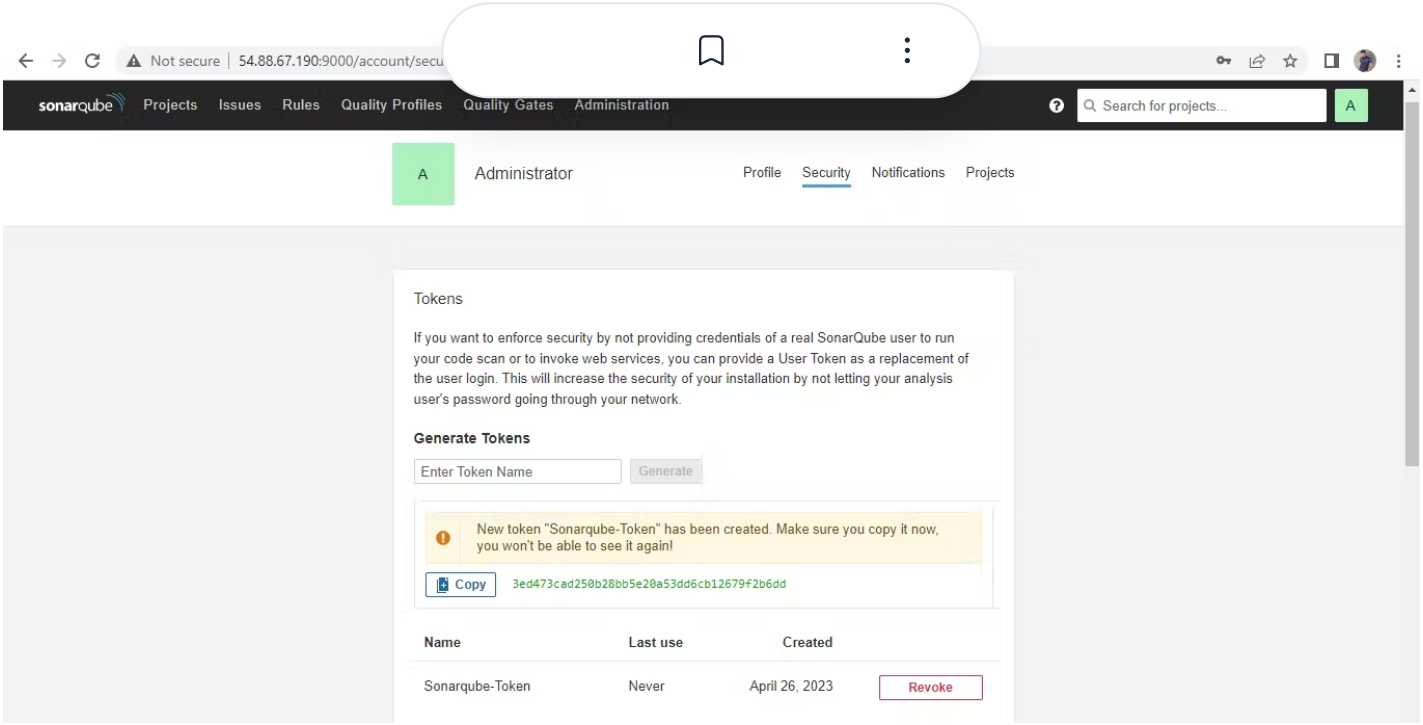
Finish the tutorial



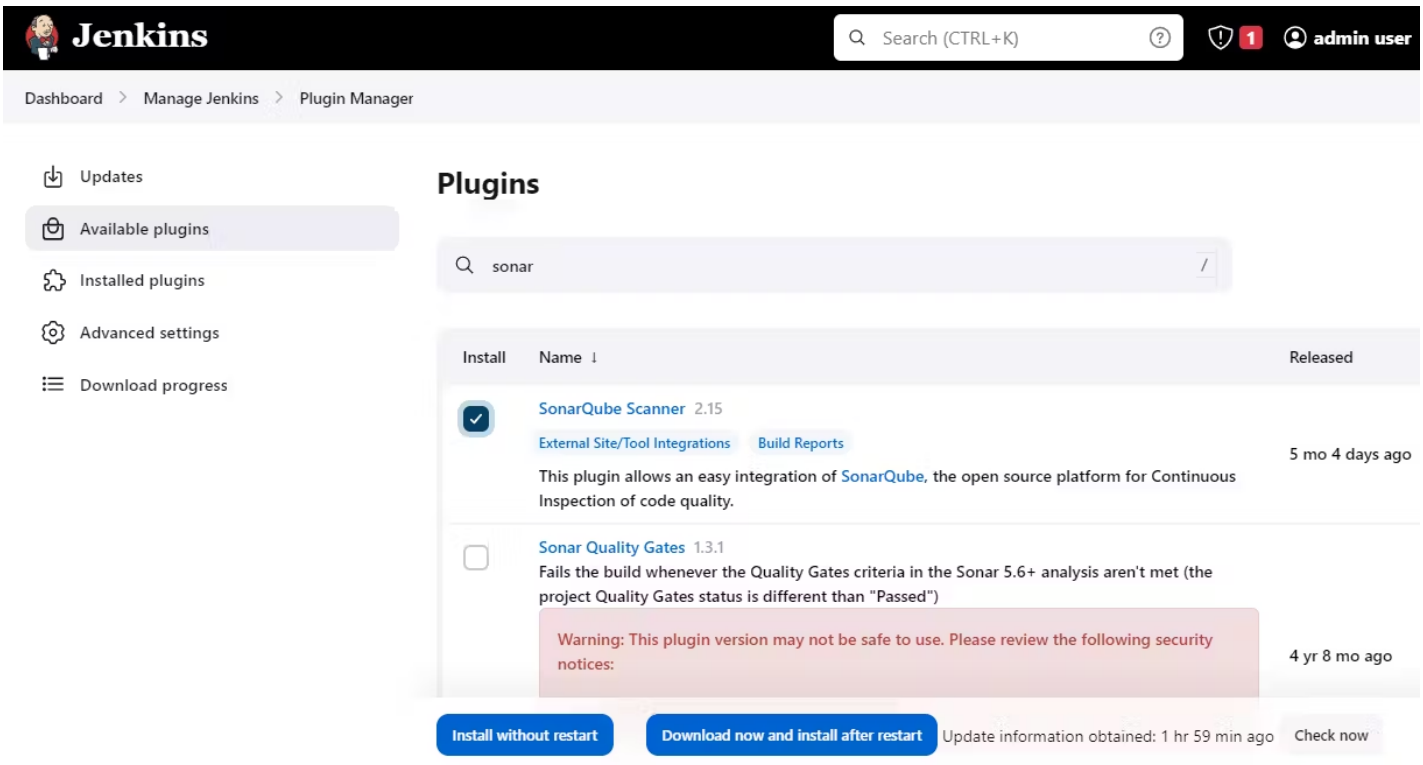
Now click to settings, we need to create token



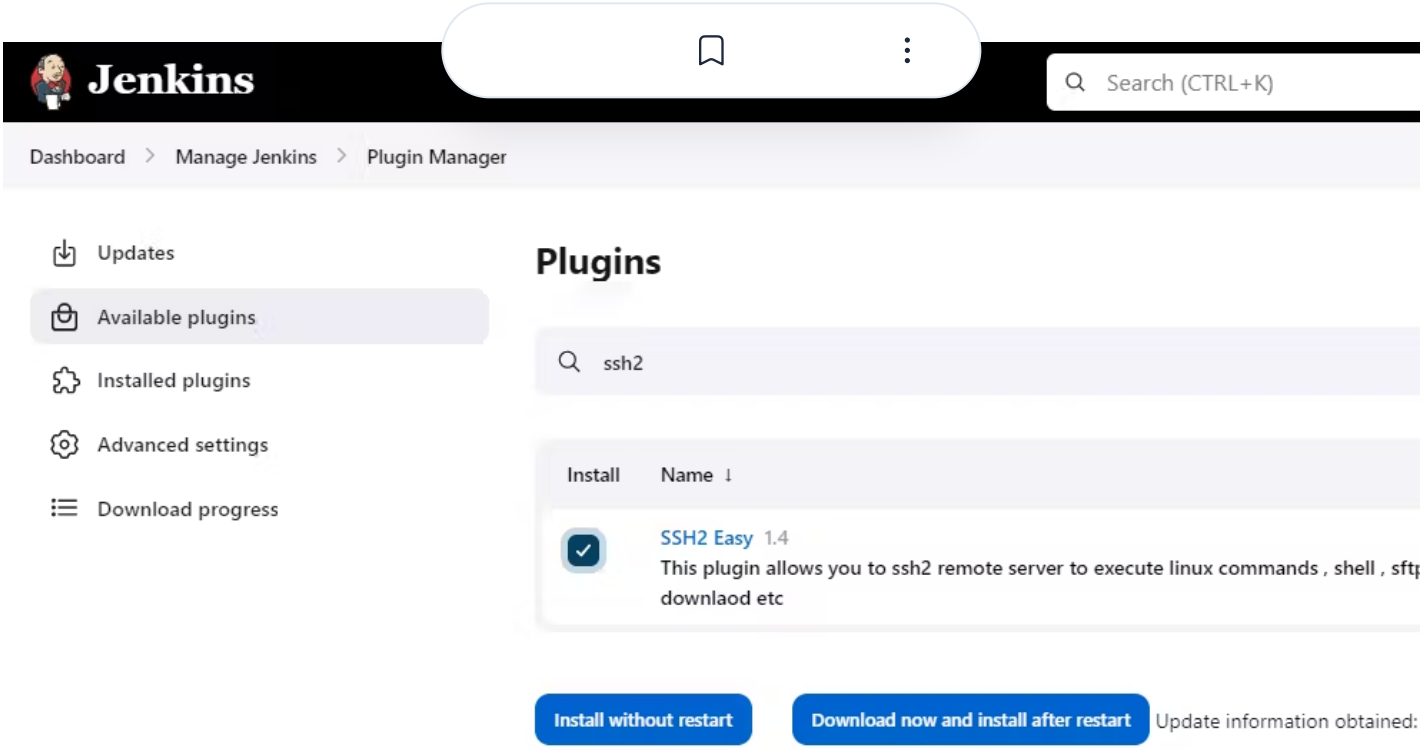
Now I have created the Token and copy the token for now you can keep this token in your notepad



Now back to Jenkins, Manage Jenkins and Install the plugin



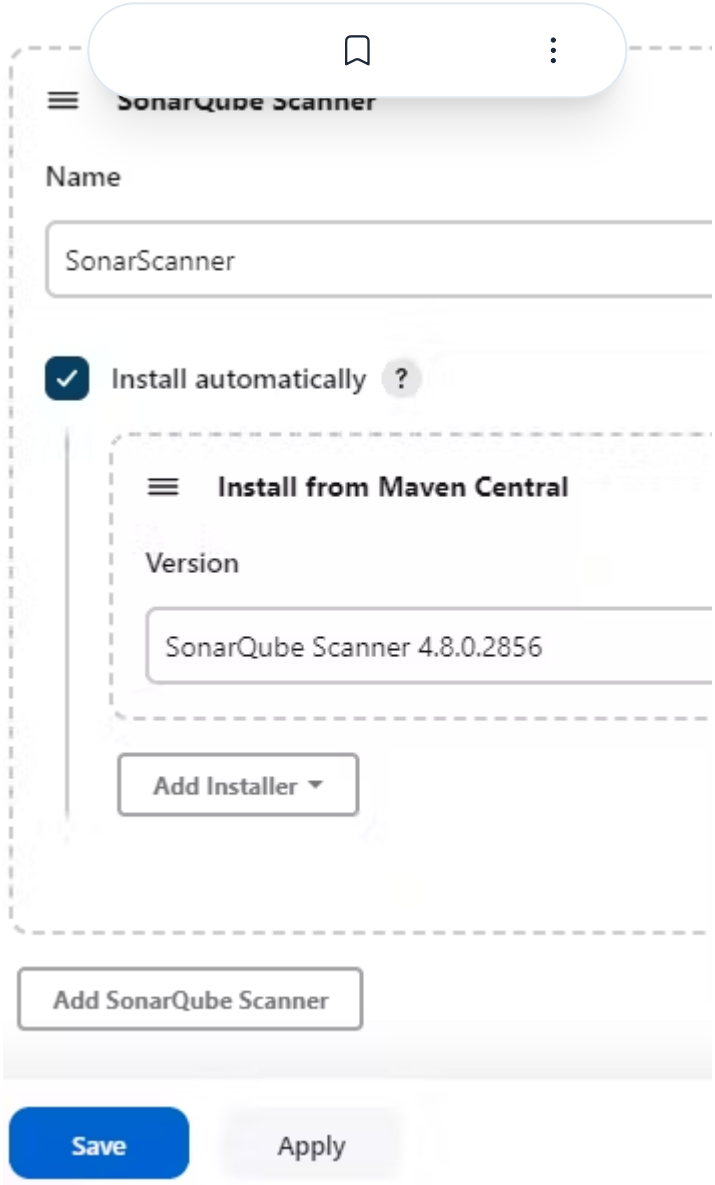
Install one more plugin



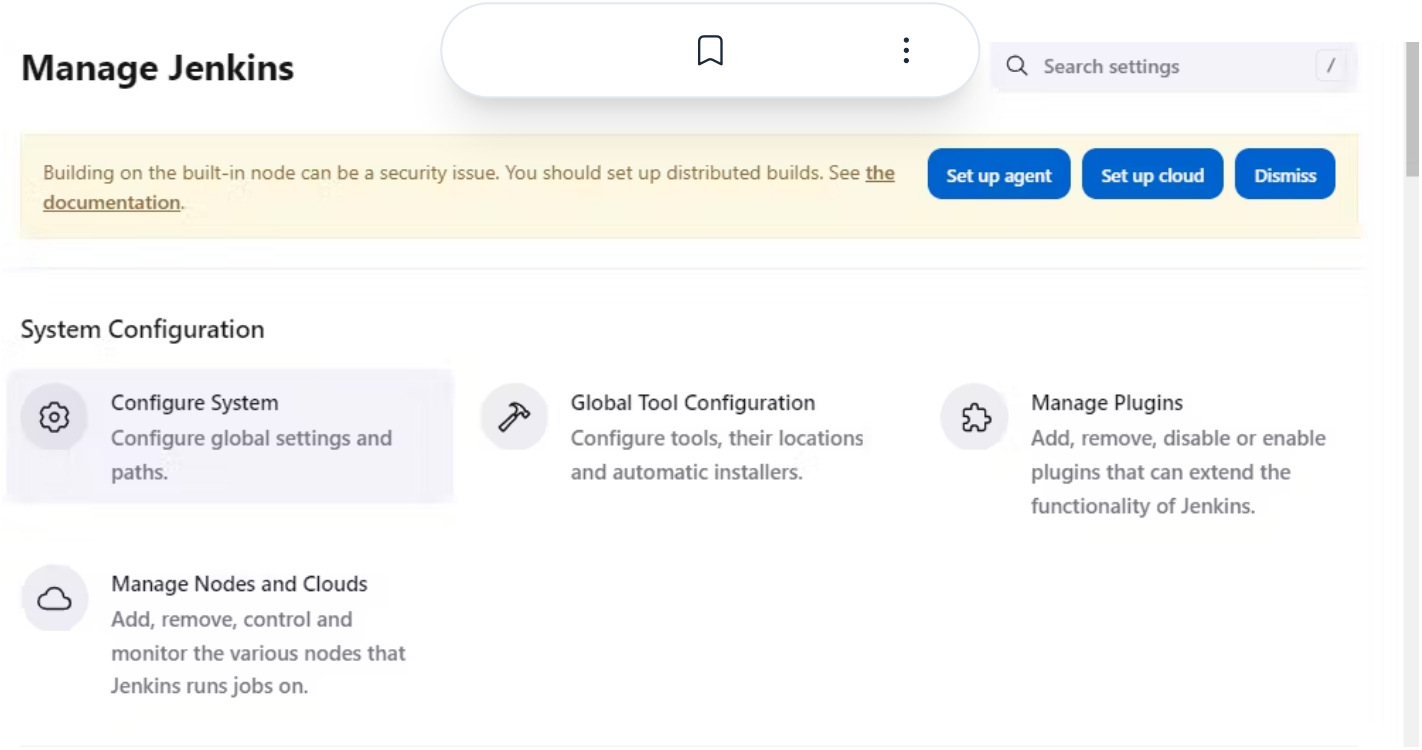
Restart Jenkins after installation of plugins

Now Go to the **Global Tool Configuration**

- Scrolling Down
- Click on SonarQube Scanner
- Give any name and rest default

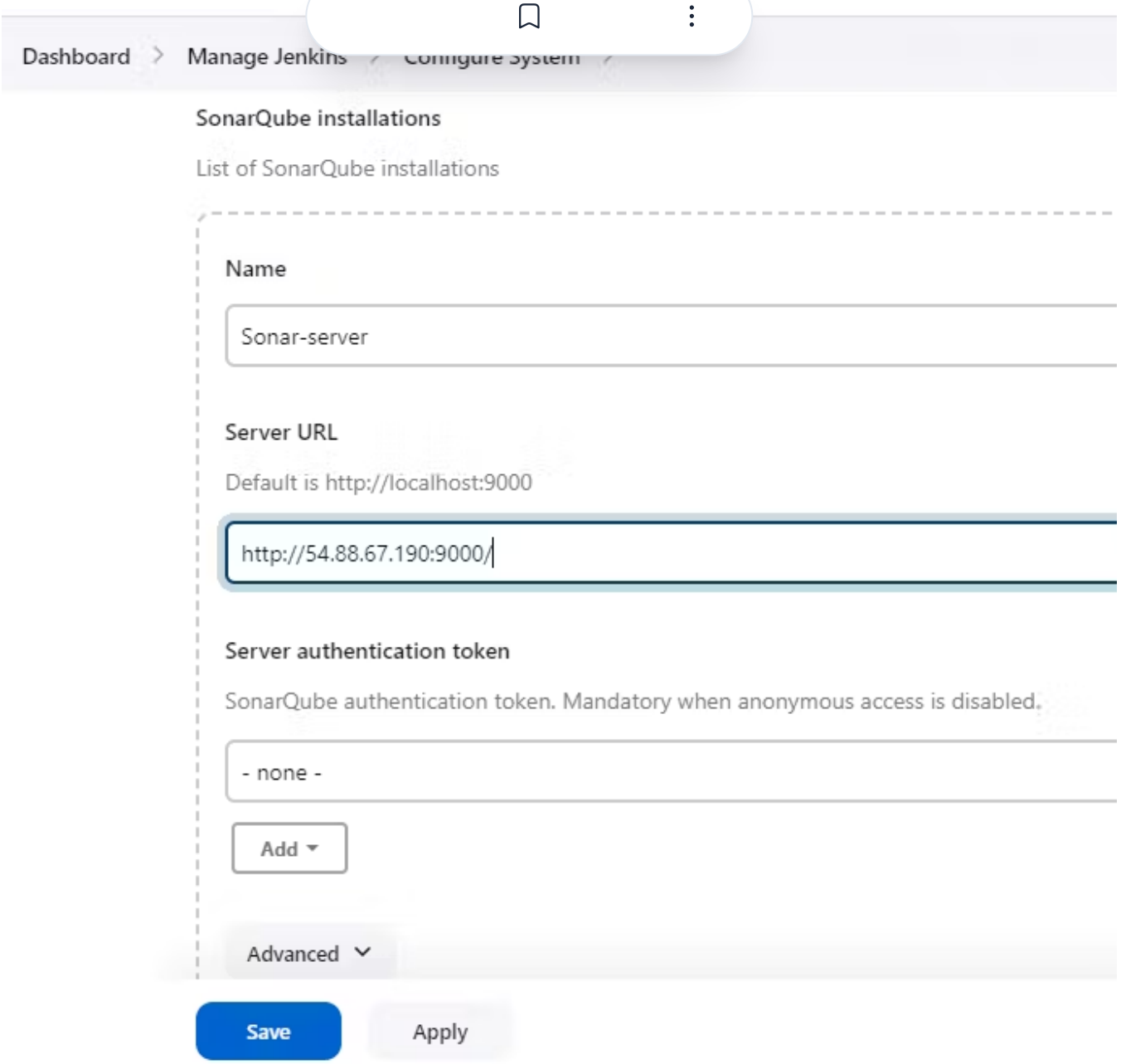


After saving it, Click to Manage Jenkins again and click on **Configure System**



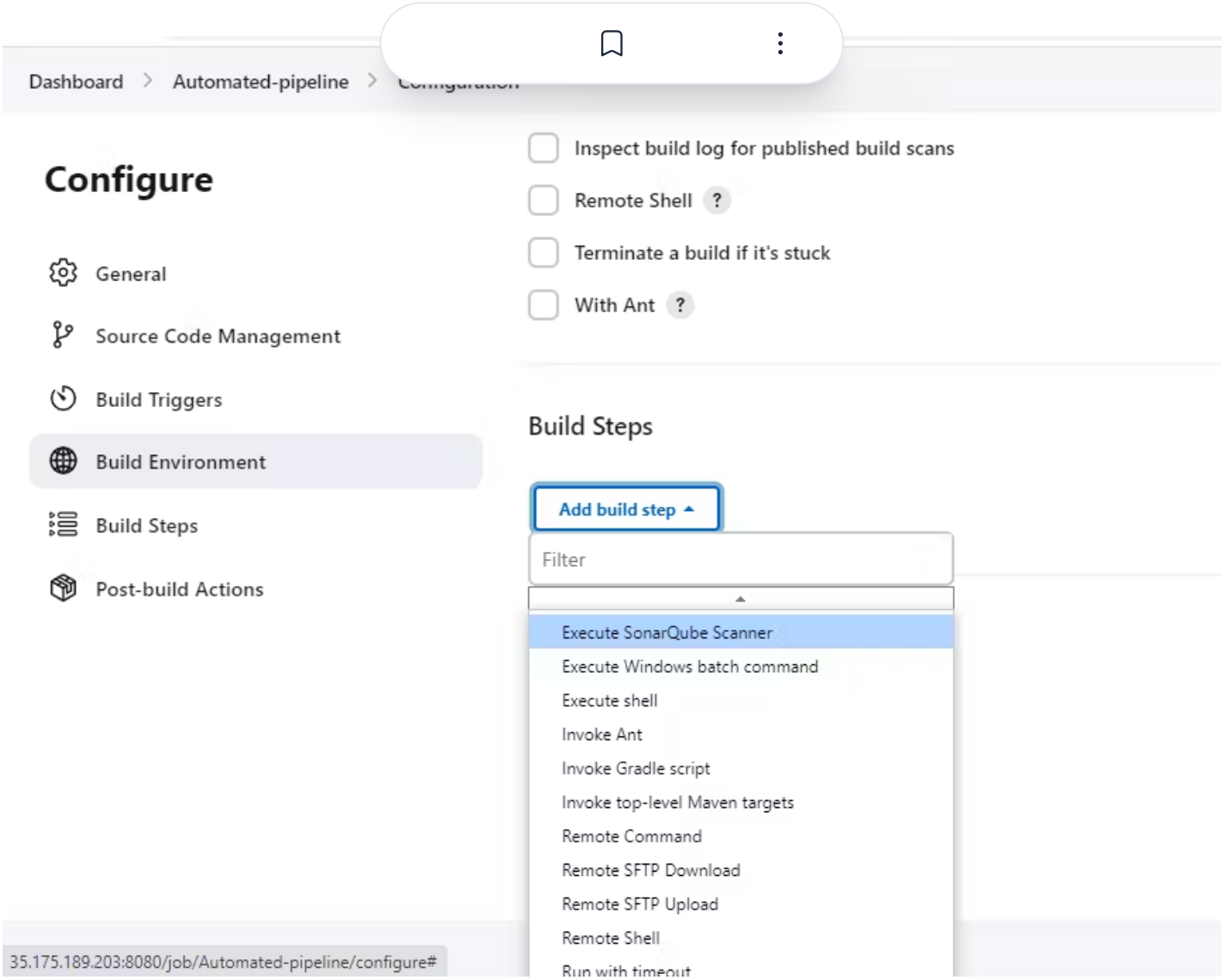
Configure System

- Scrolling down
- Click on the **SonarQube server**
- Click to add
- Give any name
- Paste the Sonarqube URL
- Save

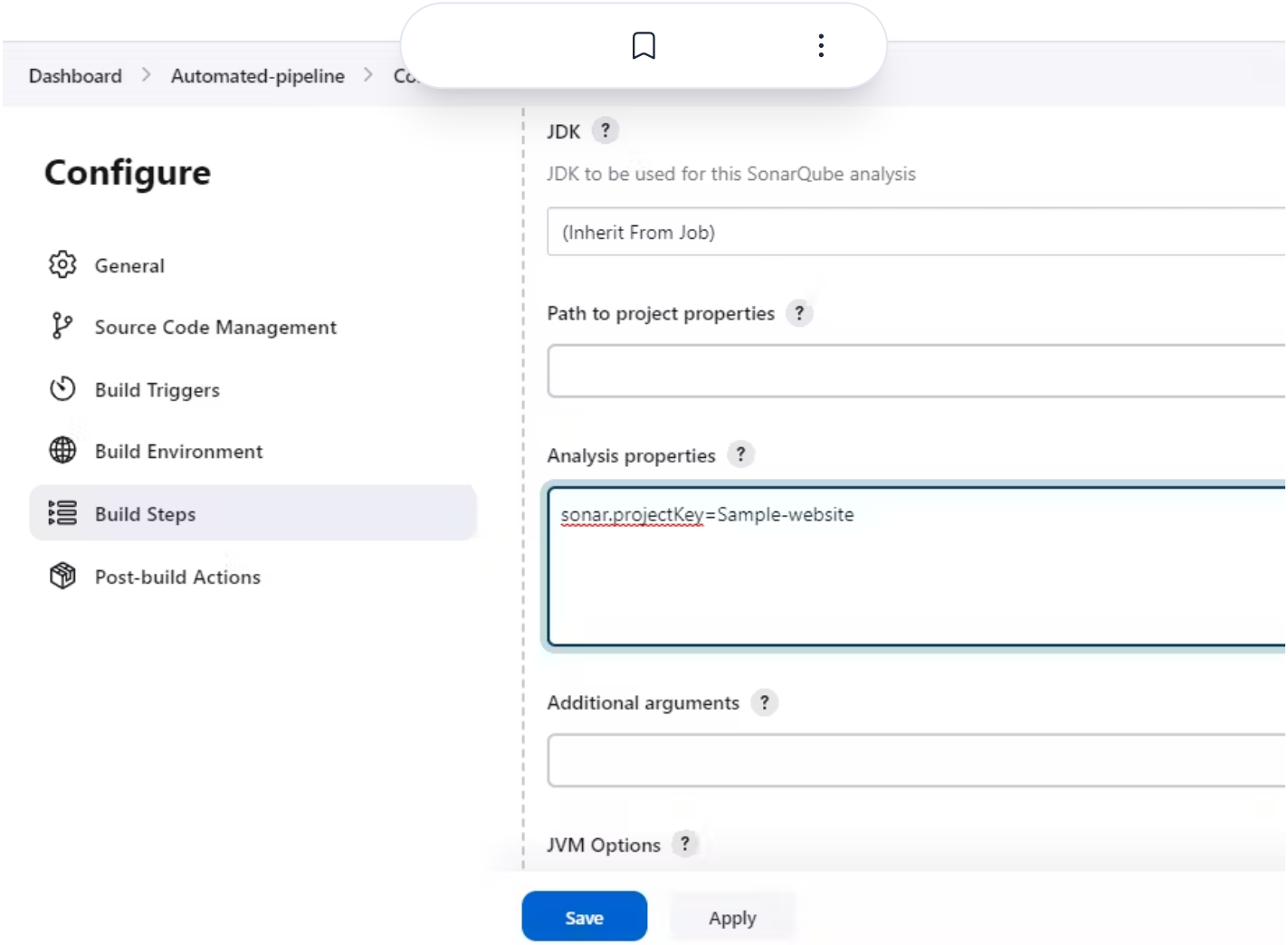
- 

After Saving, Click Configure of the Pipeline in Jenkins

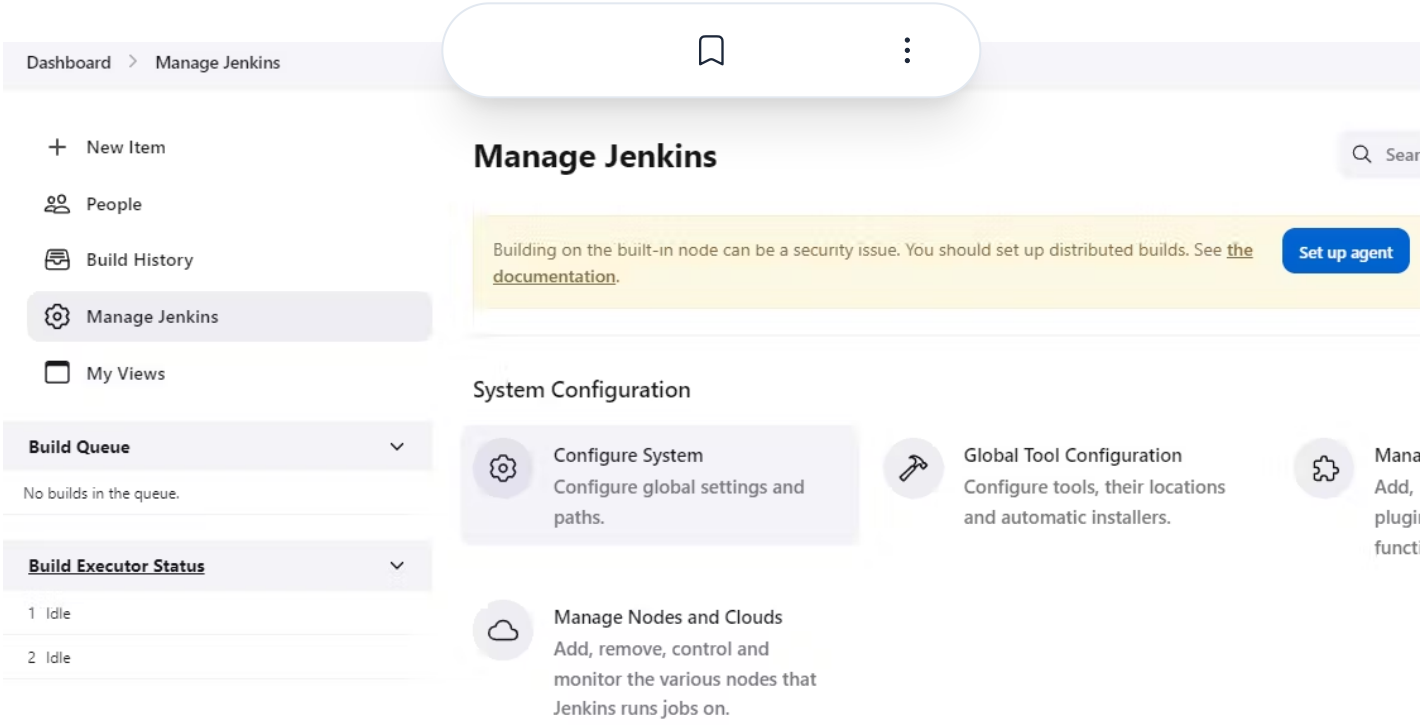
- Click to Build Environment
- Select **Execute SonarQube Scanner** in **Build Steps**



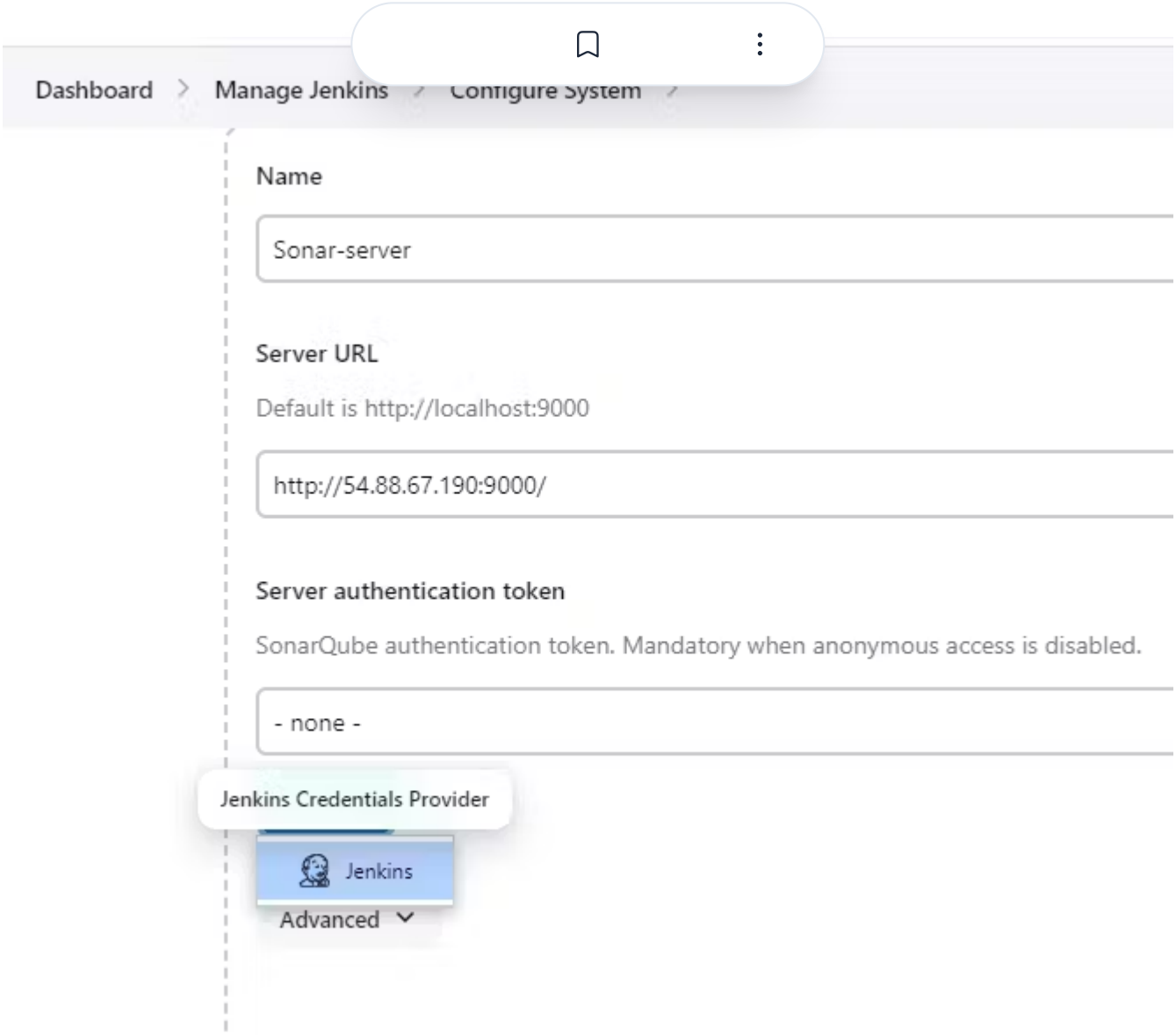
Ignore everything just paste the key here rest default and saved it



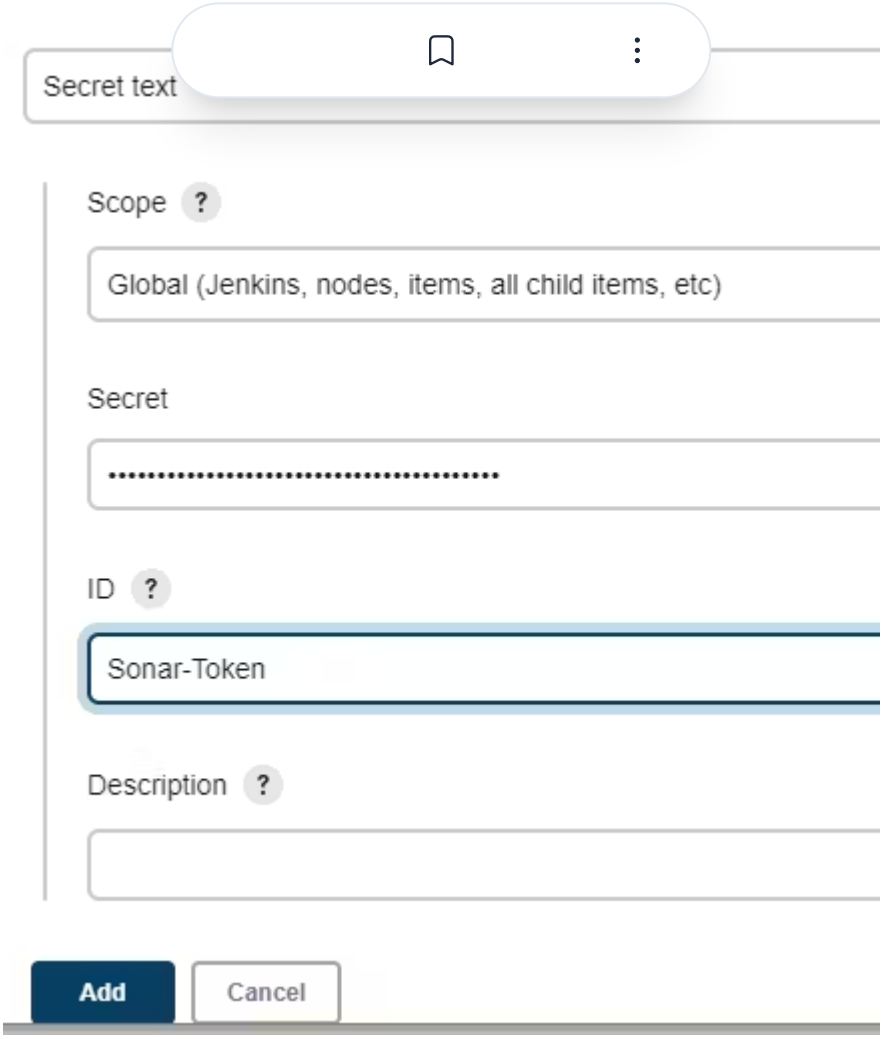
Now back to Manage Jenkins, select Configure System again



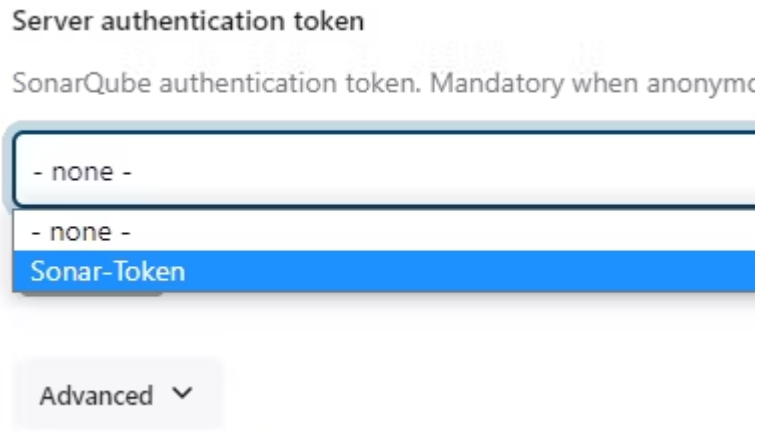
Now we have to add token here



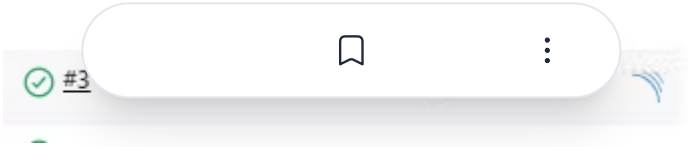
- Select Secret text
- ID - any name



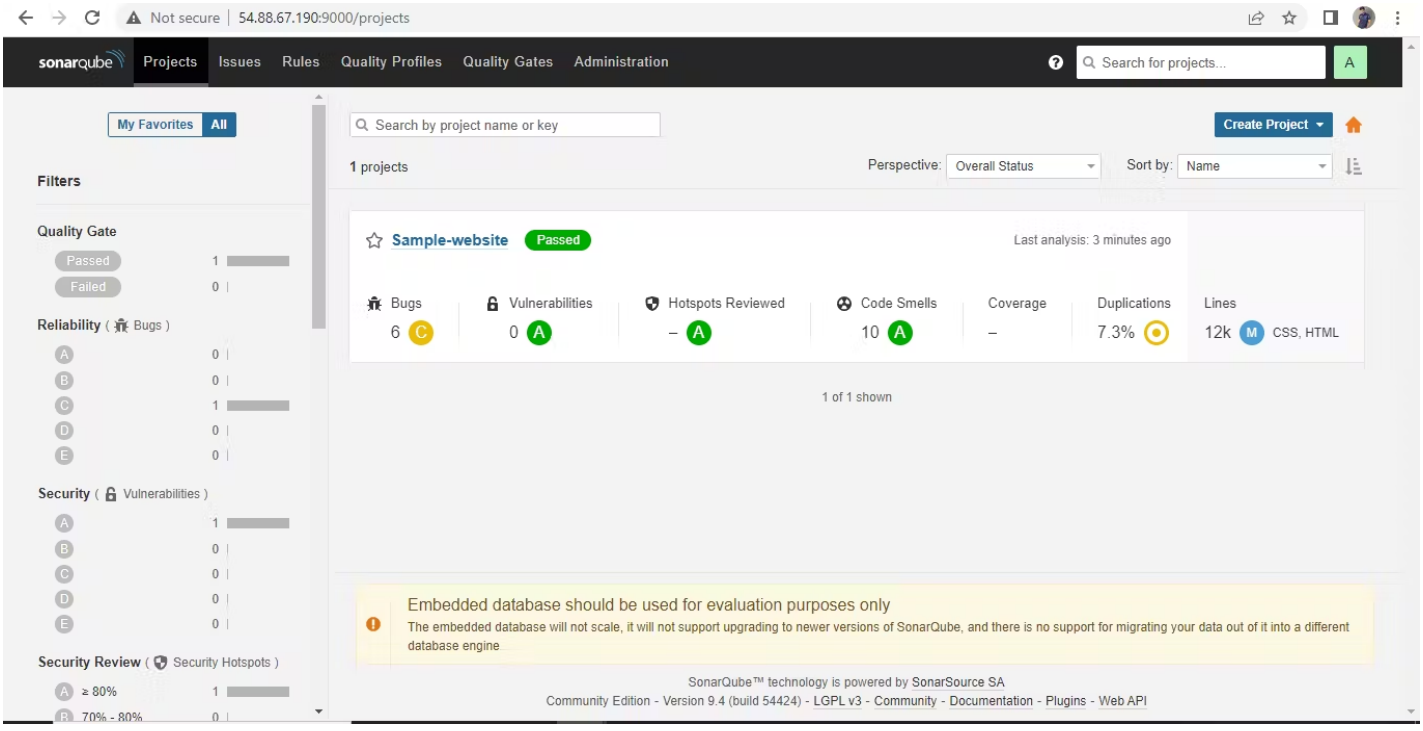
Select Token after adding and click on save



Now go back to Pipeline to verify whether it's working or not, it's absolutely working fine



Now going to check **SonarQube**, it's perfectly working fine



Once our code is passed n[illegible] Docker

Installing Docker

Started Instance

Install Docker

Update the `apt` package index and install packages to allow `apt` to use a repository over HTTPS:

COPY

```
sudo apt-get update
sudo apt-get install \
    ca-certificates \
    curl \
    gnupg
```

Add Docker's official GPG key:

COPY

```
sudo install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -
sudo chmod a+r /etc/apt/keyrings/docker.gpg
```

Use the following command to set up the repository:

COPY

```
echo \
  "deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker
  "$(. /etc/os-release && echo "$VERSION_CODENAME")" stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

Update the `apt` package index:

COPY

```
sudo apt update
```

To install the latest version, run:

COPY

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugi
```

Password-based Server Connectivity

Now I am going to make connection between Jenkins server and Docker server

After running all the commands go back to the **Jenkins server**

COPY

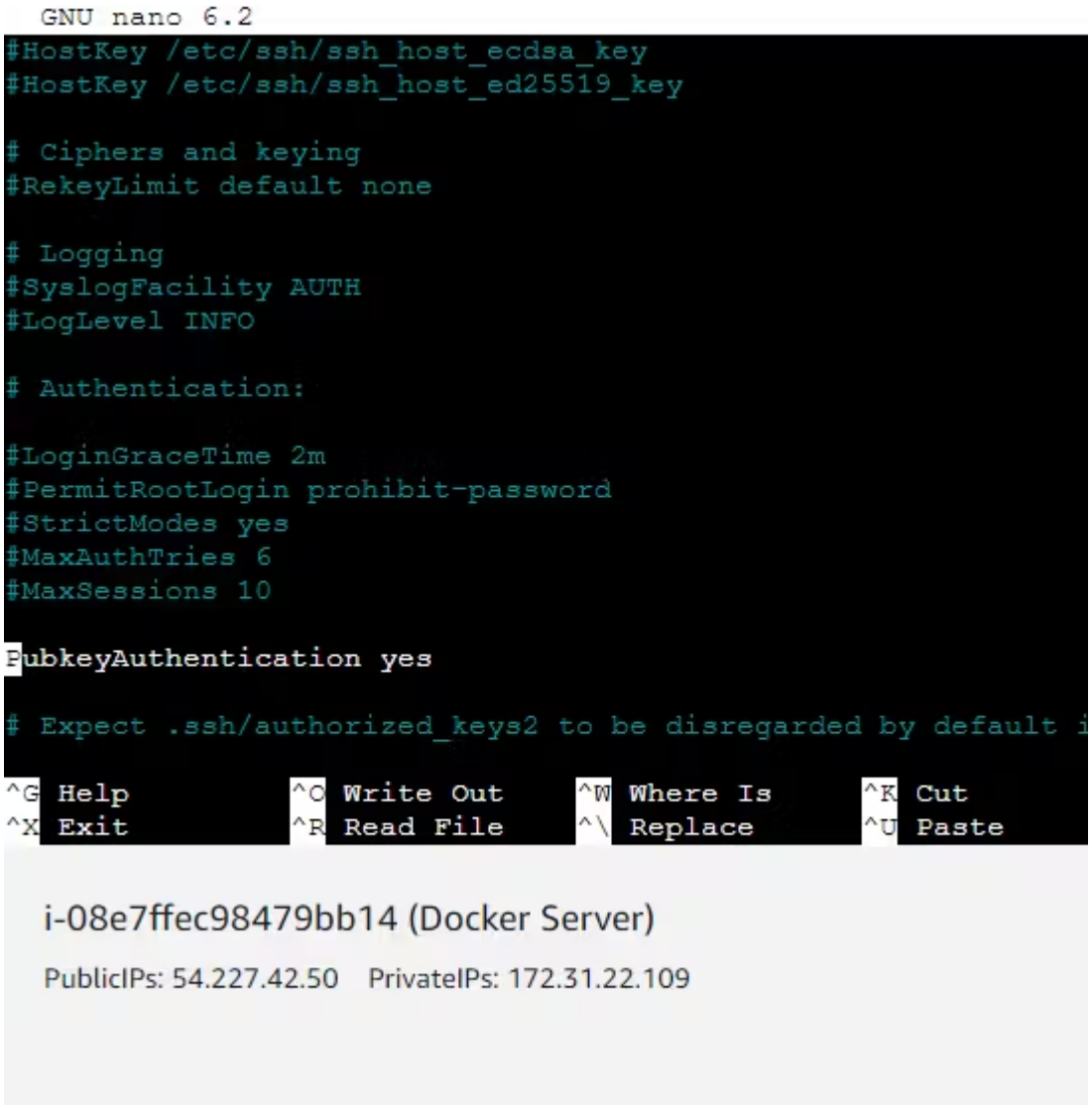
```
sudo su jenkins
```

Open **Docker server**

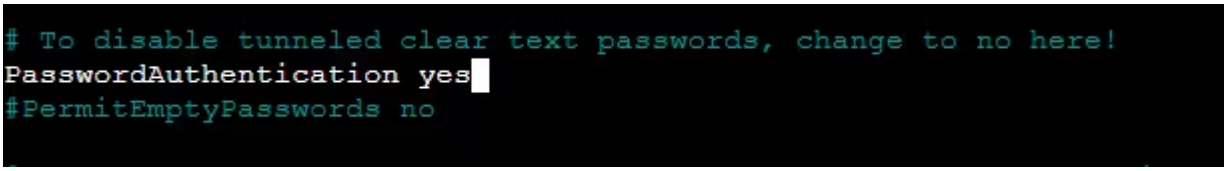
COPY

```
sudo su
nano /etc/ssh/sshd_config
```

Uncomment this first



And change the Password Authentication to Yes

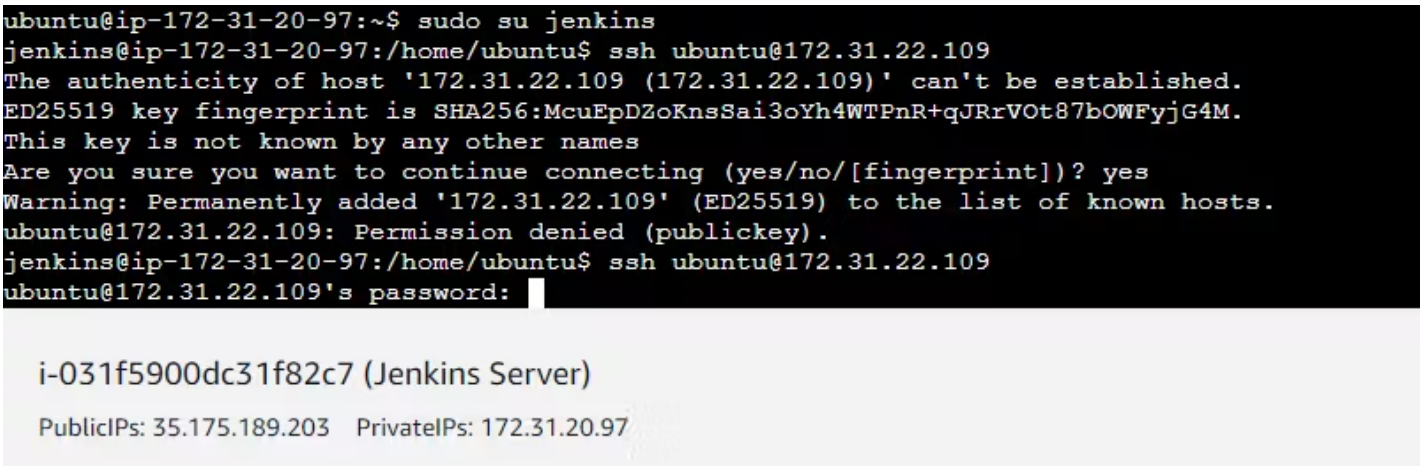


COPY

```
systemctl restart sshd
```

Back to **Jenkins Server**

Now you can see it's asking for the password earlier it's showing permission denied

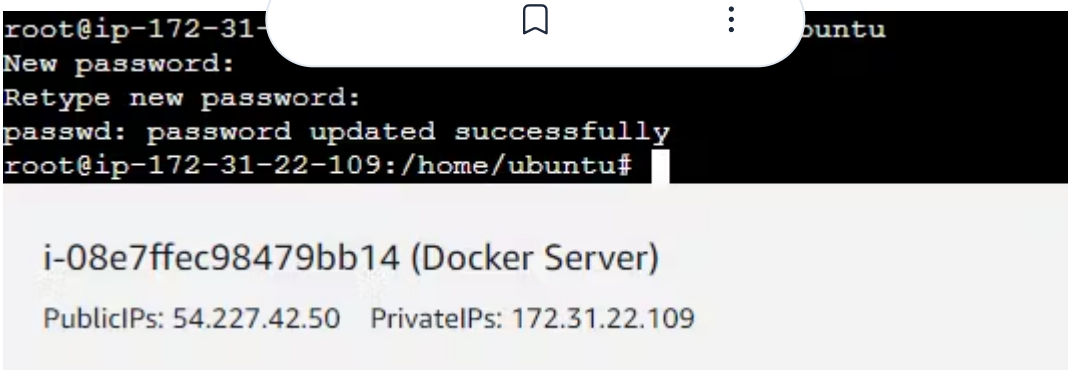


Now we have to change the password of the Docker Ubuntu user

Back to **Docker Server**

COPY

```
passwd ubuntu
```

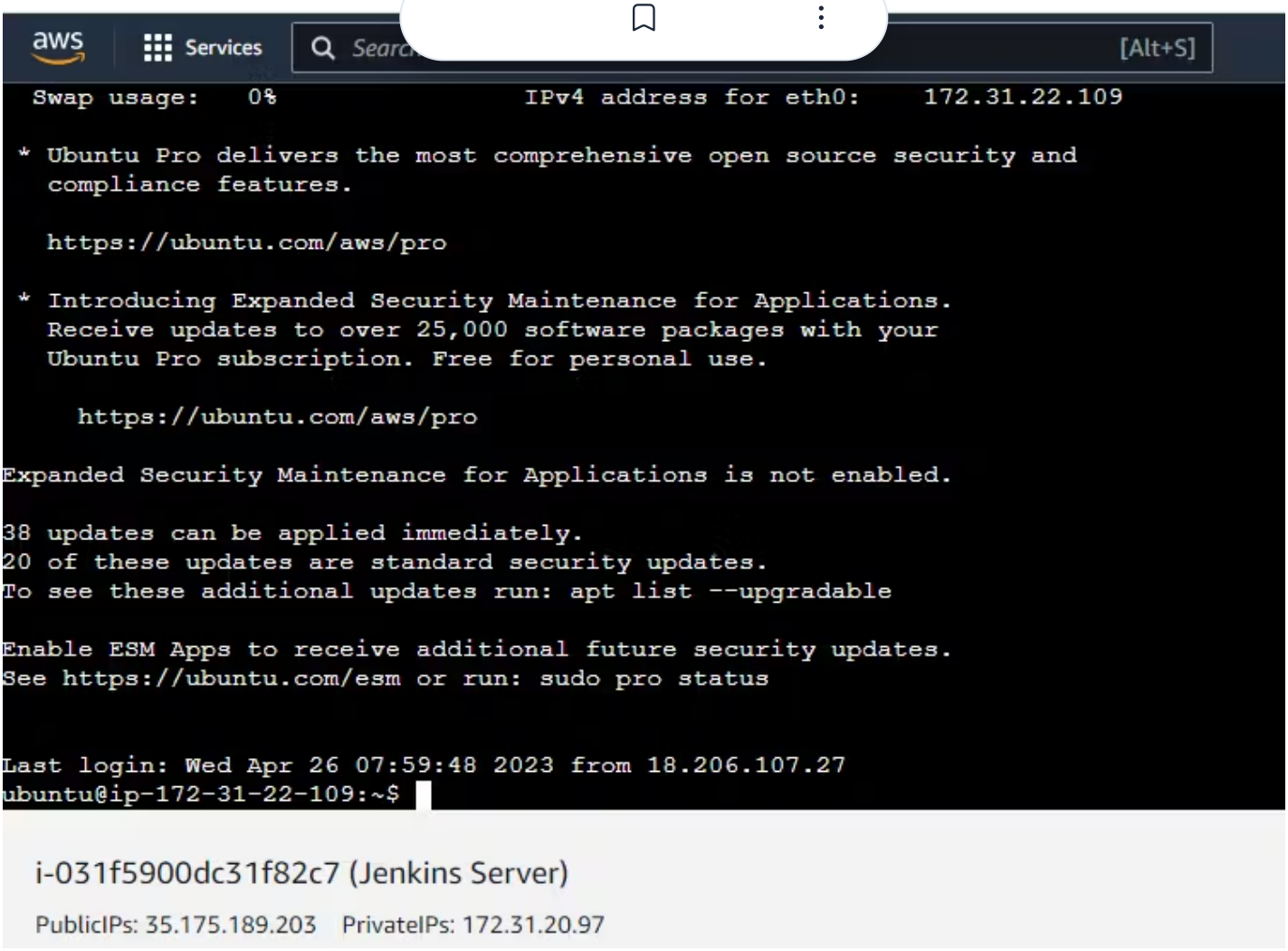


Back to **Jenkins Server**

COPY

```
ssh ubuntu@172.31.22.109
```

Access done now



Now I am going to generate SSH Key in **Jenkins Server**

```
jenkins@ip-172-31-20-97:/
Generating public/private rsa key pair.
Enter file in which to save the key (/var/lib/jenkins/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /var/lib/jenkins/.ssh/id_rsa
Your public key has been saved in /var/lib/jenkins/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:j8F79JgA138khX89oi/7RWl4n8XMc2wJacseYEqH2gk jenkins@ip-172-31-20-97
The key's randomart image is:
+---[RSA 3072]----+
|             ..  |
|          .. ...  |
|       .E.o.+.=. .|
|        += =.++=*=|
|        .S+. o=oOX|
|          B =..=.*|
|         o = o. o.|
|          . . ..  |
|            .+.   |
+----[SHA256]-----+
jenkins@ip-172-31-20-97:/home/ubuntu$
```

i-031f5900dc31f82c7 (Jenkins Server)

PublicIPs: 35.175.189.203 PrivateIPs: 172.31.20.97

After generating keygen I entered the command

COPY

```
ssh-copy-id ubuntu@172.31.22.109
```

After running the command enter the password

```
jenkins@ip-172-31-20-97:/home/ubuntu$ ssh-copy-id ubuntu@172.31.22.109
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/var/lib/jenkins/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
ubuntu@172.31.22.109's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'ubuntu@172.31.22.109'"
and check to make sure that only the key(s) you wanted were added.

jenkins@ip-172-31-20-97:/home/ubuntu$ ssh ubuntu@172.31.22.109
Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 5.15.0-1031-aws x86_64)
```

i-031f5900dc31f82c7 (Jenkins Server)

PublicIPs: 35.175.189.203 PrivateIPs: 172.31.20.97

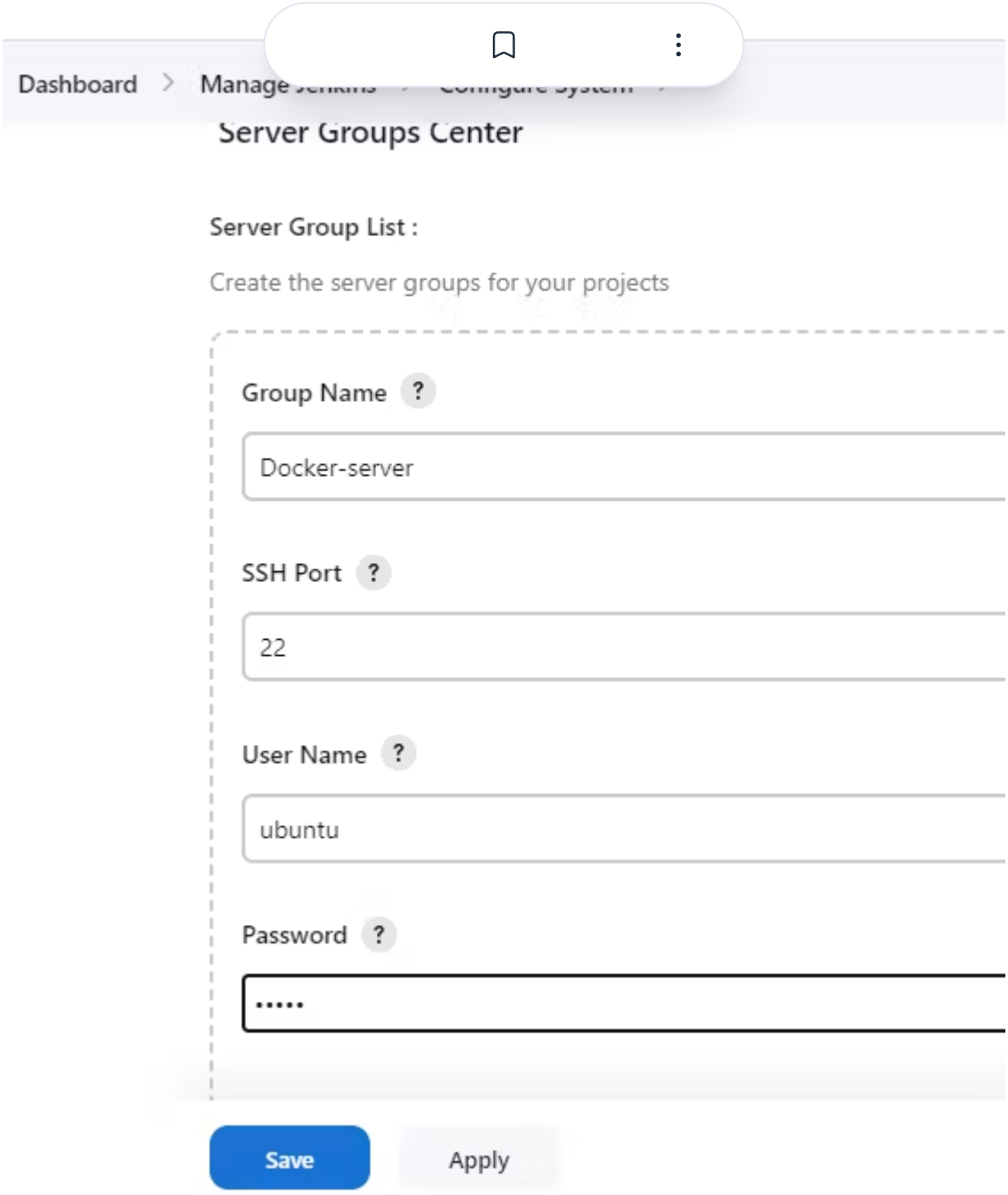
COPY

```
ssh ubuntu@172.31.22.109
```

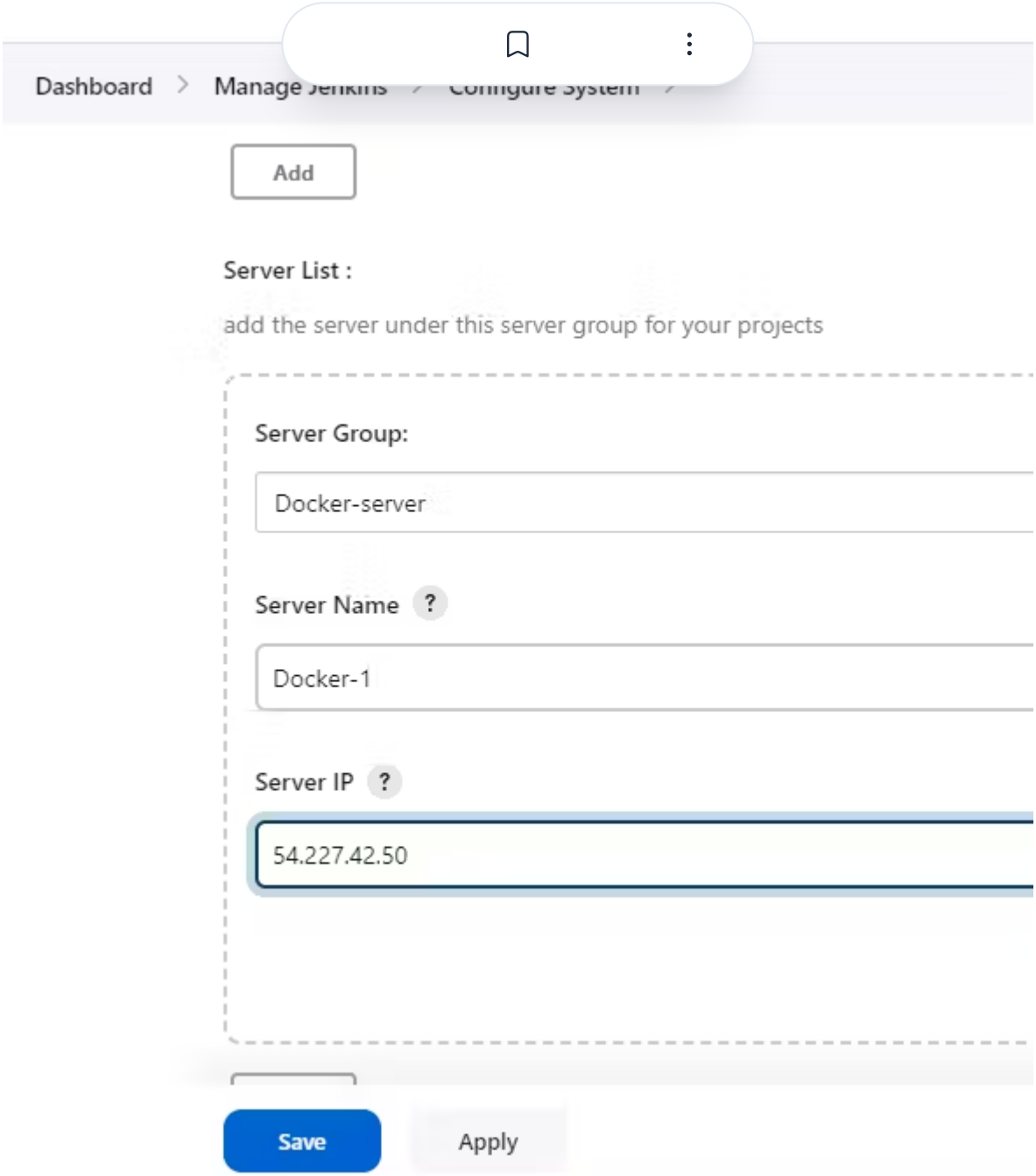
Now we don't need to password again anymore

Back to **Jenkins** again

- Manage Jenkins
- Configure system
- Server group center

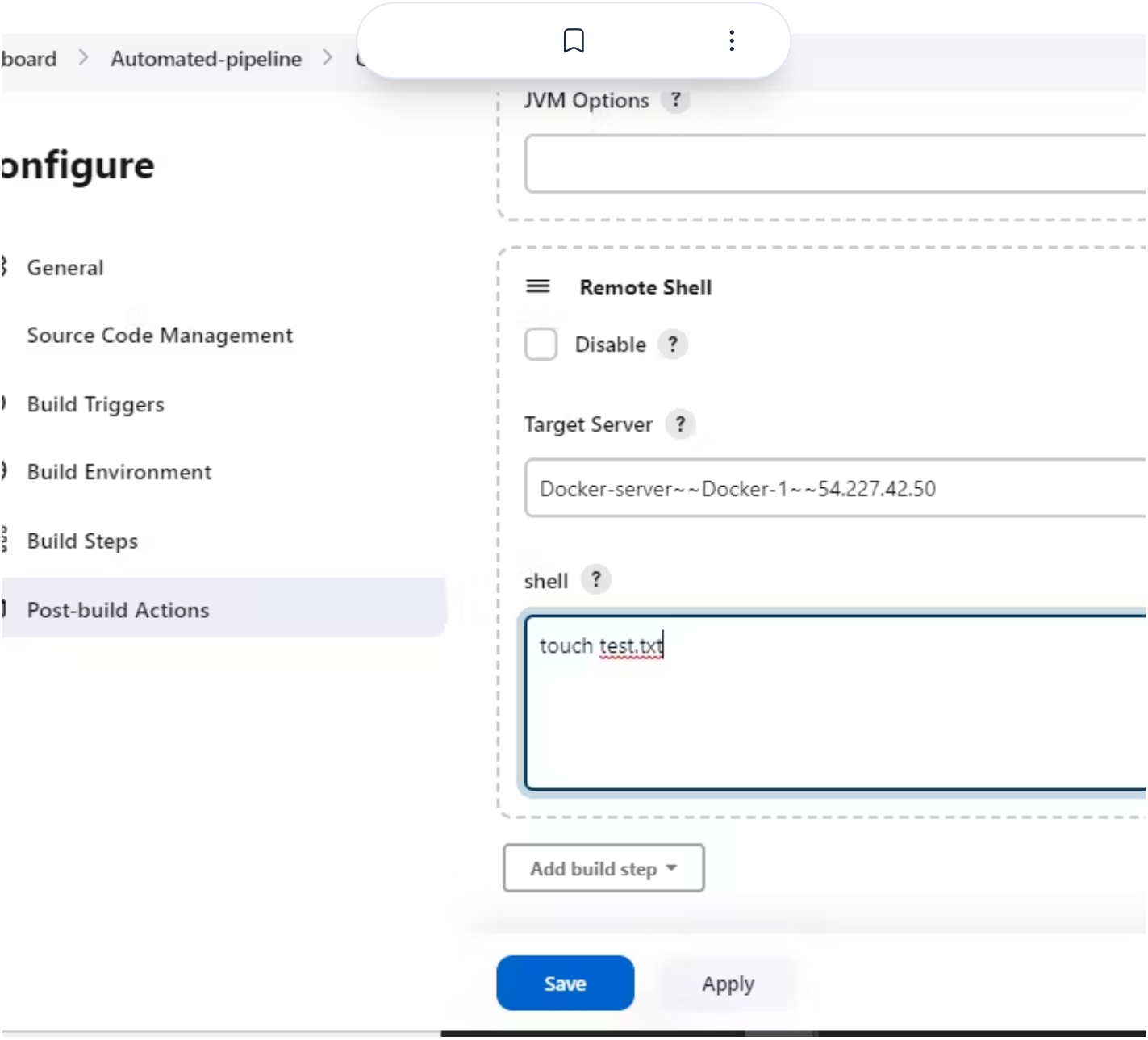


Now we have to add a server list



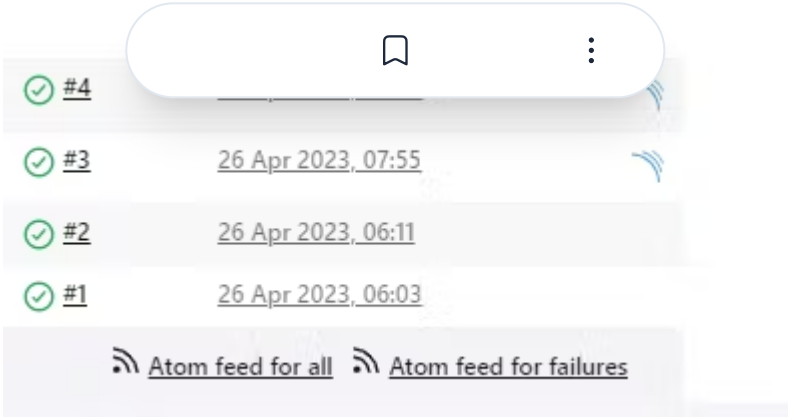
Now go to **Pipeline**

- Configure
- Post-build action
- Add build step



Now I am gonna build the pipeline to verify whether it's working or not

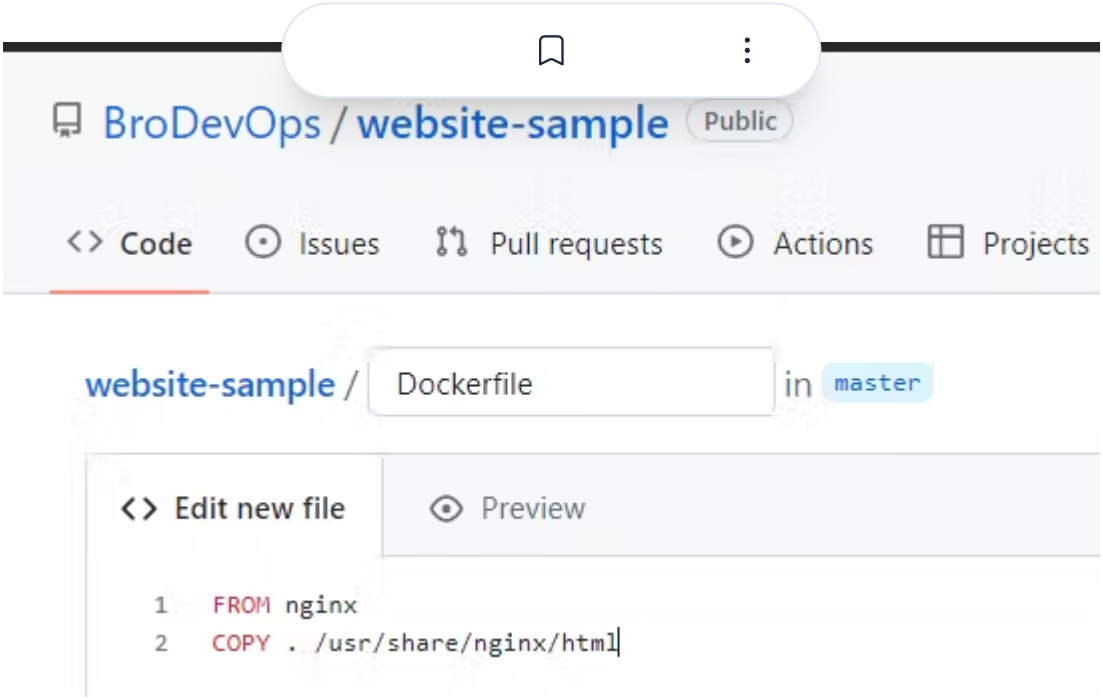
And it seems to be working



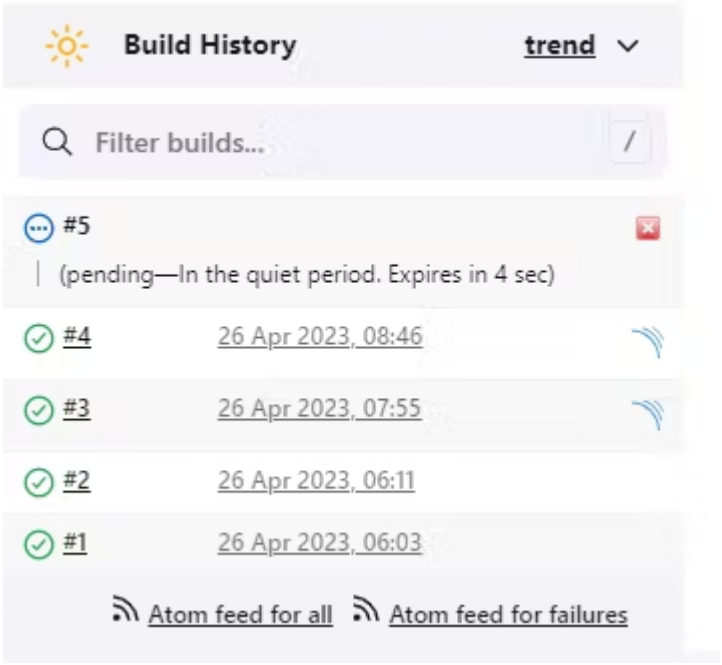
Open **Docker Server**, as we can our file got created here

Open the Git Repository now

- Create a Dockerfile
- Commit the file



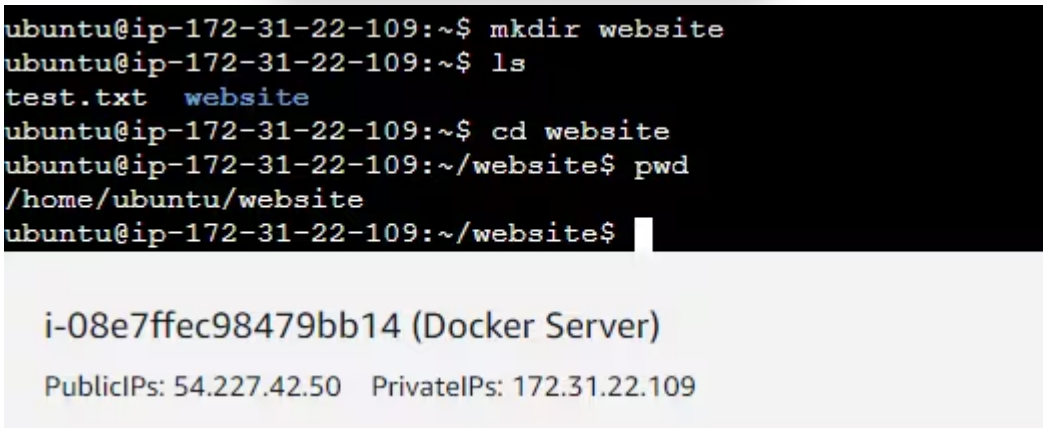
See Auto trigger started



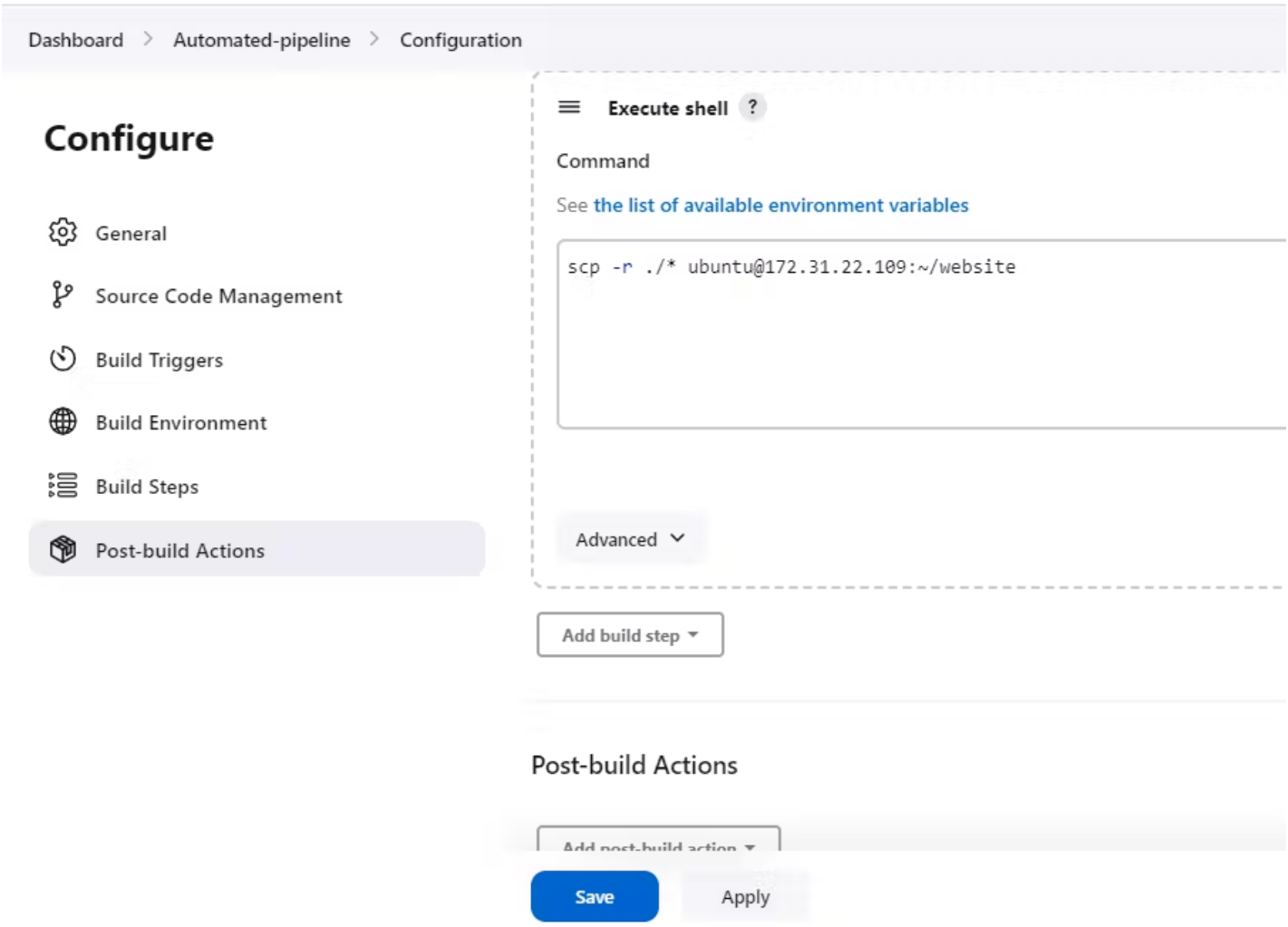
Now returned to Pipeline, configure

- I have deleted the Remote shell
- Clicked to execute shell

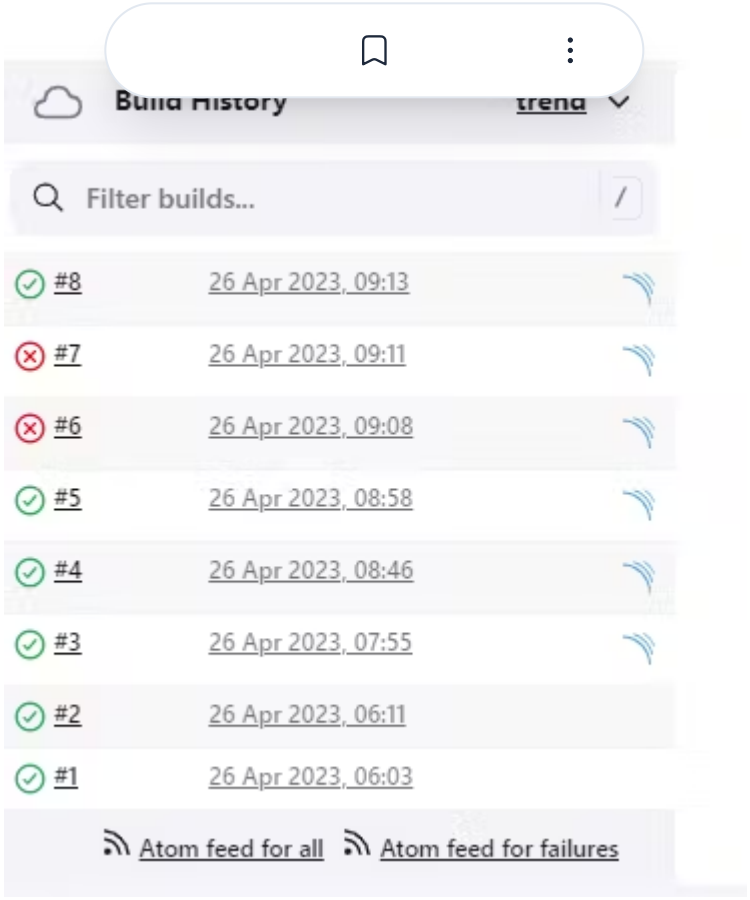
Created a folder in the **Doc**



Clicked to execute shell and fill the details, here I entered docker server IP and folder which is website



We have got the success message here and let us check our docker server



All the files copied to the Docker server

Building the Image and running the container

Back to the Docker server and need to give permission so that we can run all the commands without sudo

COPY

```
sudo usermod -aG docker ubuntu
newgrp docker
```

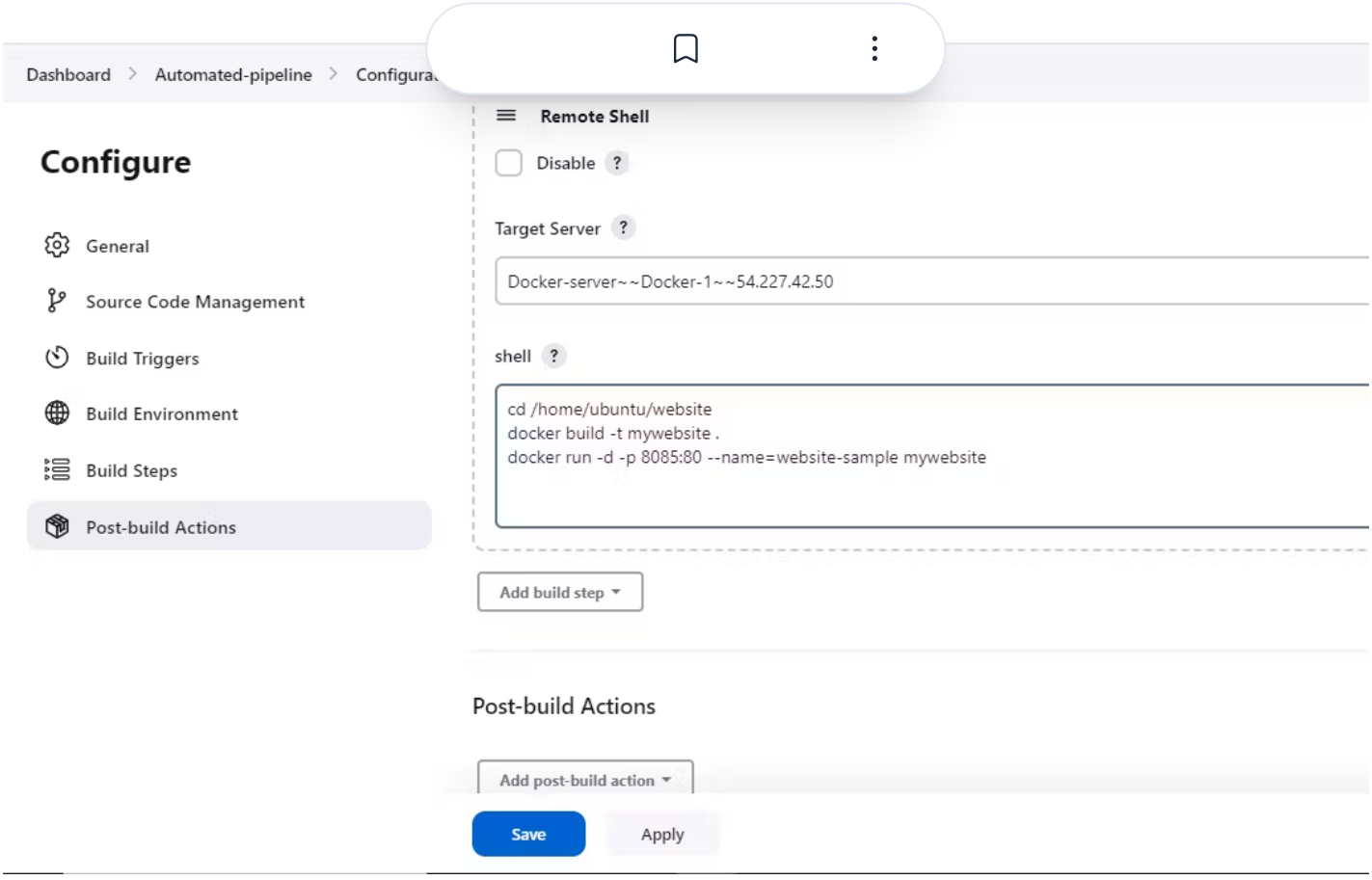
After giving the permission [illegible] without sudo

COPY

```
docker ps
```

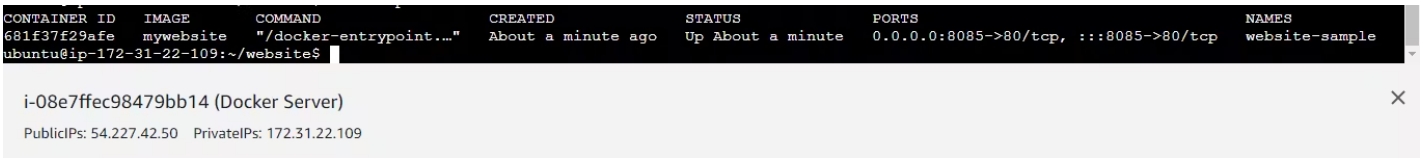
Now back to **Jenkins**

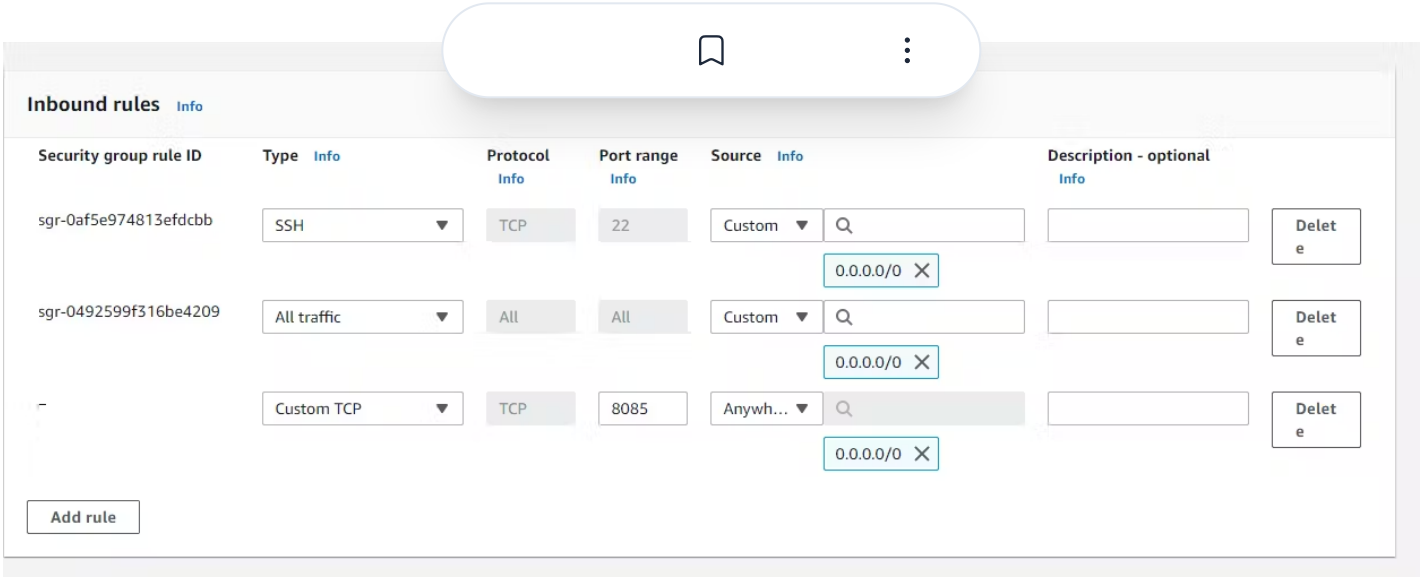
- Click to Pipeline
- Click to configure
- Click to post-build actions
- Select the Remote shell again in the build steps
- I gave any random name



Time to check our docker container got created or not

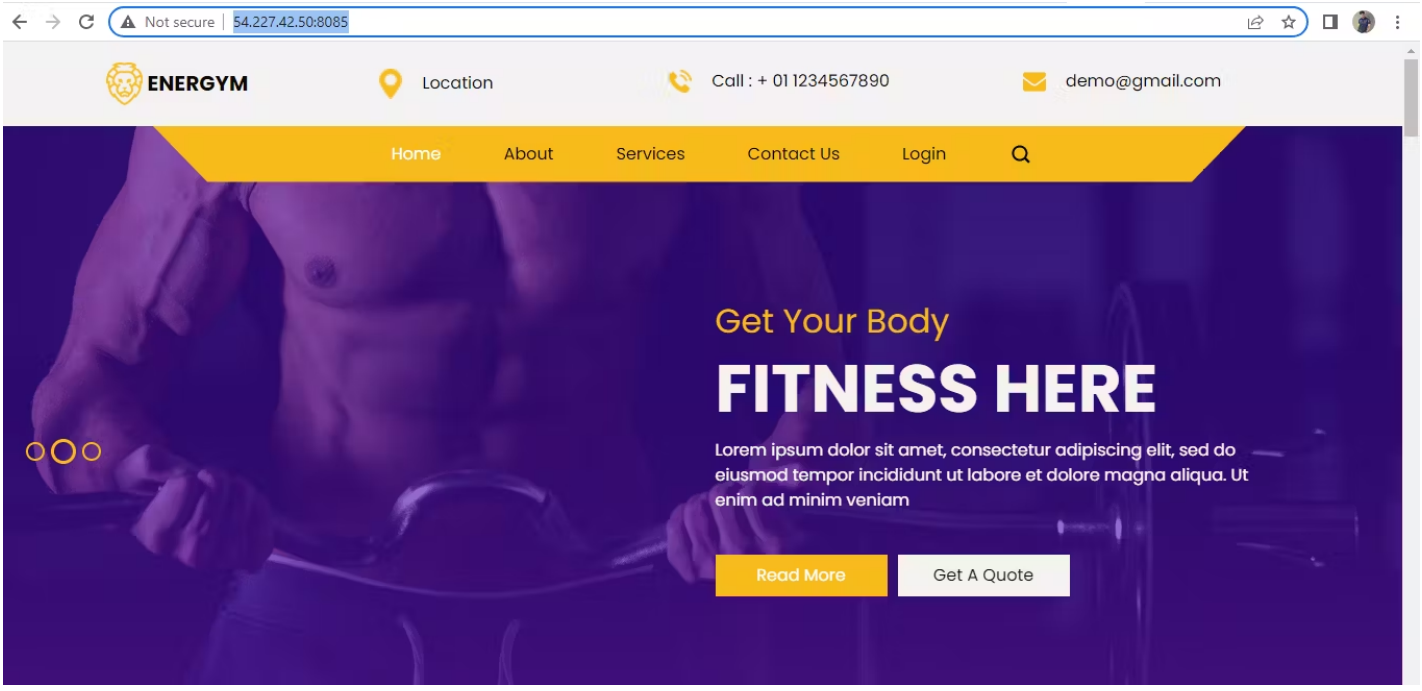
It got created but we have to add port 8085 in the inbound setting of the security group





Now time to check the public URL of the docker Instance

http://54.227.42.50:8085/ (our code is successfully deployed on docker container)



Sub[illegible]tter

Read articles from **DevOps by Sagar Aulakh** directly inside your inbox. Subscribe to the newsletter, and don't miss out.

parandhamareddy026@gmail SUBSCRIBE

Jenkins Docker Ubuntu ci-cd Linux

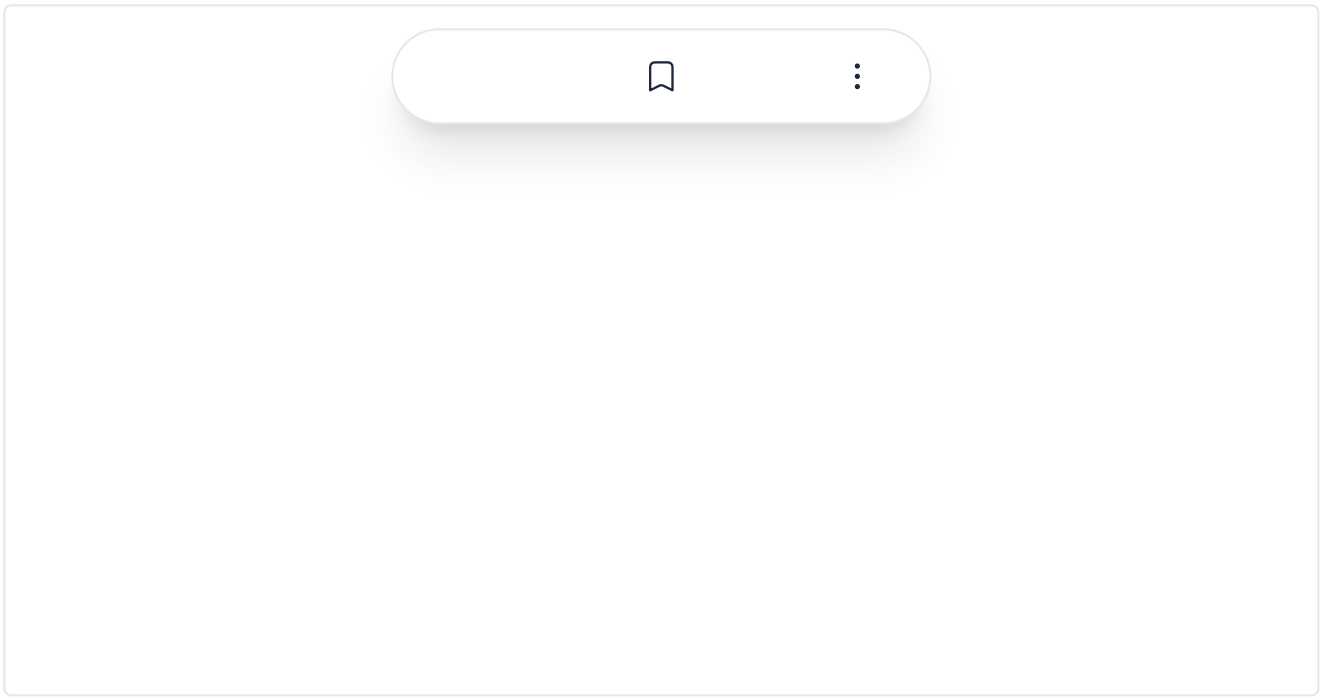
MORE ARTICLES

Sagar Aulakh

Installing Java and MySQL DB using Ansible Playbook

Launch EC2 Instance Create 1 Controller Instance Create 2 Slave Instances We need to install Ansibl...

Sagar Aulakh



How to push a local project to GitHub repository

Step: 1 Open the file manager and the path of the code Below I have created 3 sample files under Fol...

Sagar Aulakh

Jenkins CI with GitHub Integration

Launch Instance Create AWS EC2 - t2.micro sudo apt update clear Install JAVA because Jenkins is bui...