DevOps Project-1

Capstone Project

DevOps Certification Training

DevOps Certification Course

Capstone Projects

You have been Hired Sr. DevOps Engineer in Abode Software. They want to implement DevOps Lifecycle in their company. You have been asked to implement this lifecycle as fast as possible. Abode Software is a product-based company, their product is available on this GitHub link.

https://github.com/hshar/website.git
Following are the specifications of the lifecycle:

1. Install the necessary software on the machines using a configuration management tool.
2. Git Workflow has to be implemented

3.Code Build should automatically be triggered once commit is made to master branch or develop branch.

If commit is made to master branch, test and push to prod

If commit is made to develop branch, just test the product, do not push to prod

4.The Code should be containerized with the help of a Dockerfile. The Dockerfile should be built every time there is a push to Git-Hub. Use the following pre-built container for your application:

hshar/webapp

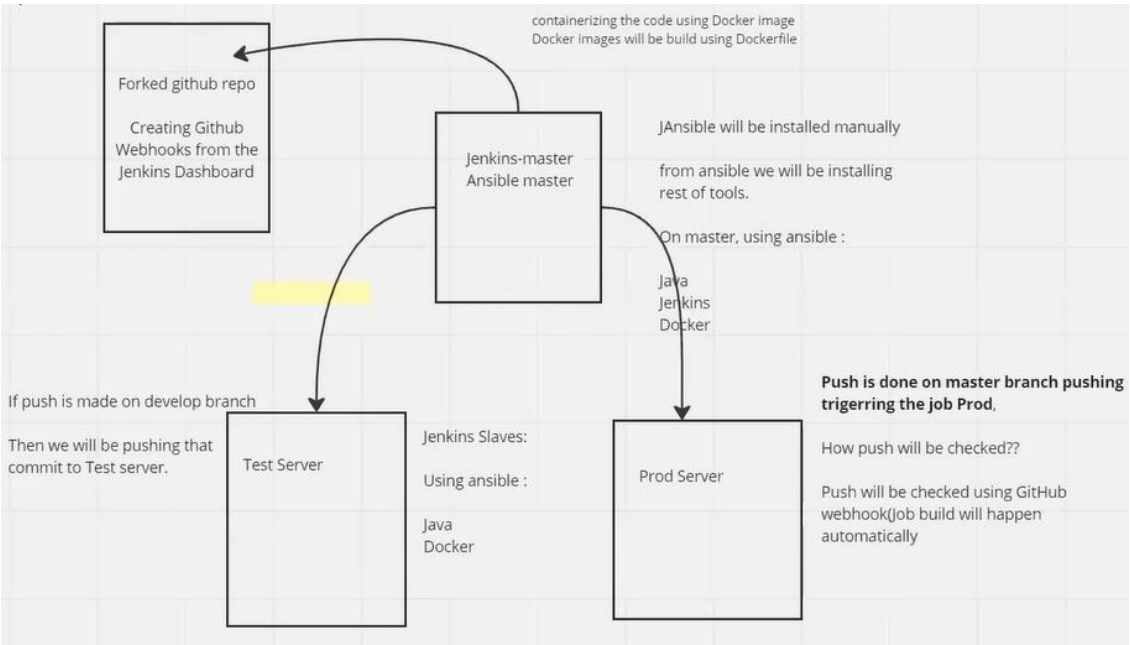
The code should reside in '/var/www/html'

5.The above tasks should be defined in a Jenkins Pipeline, with the following jobs:

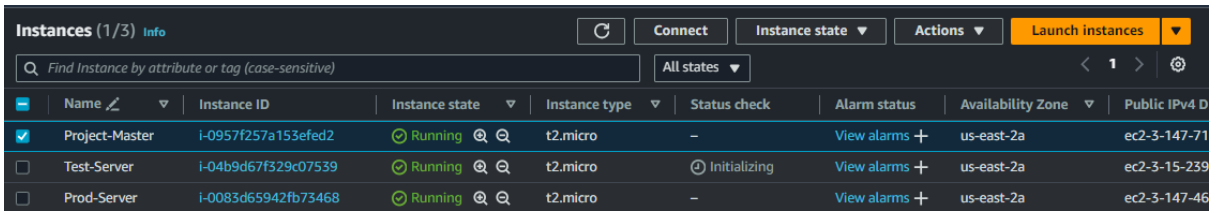
Job1 : build

Job2: test

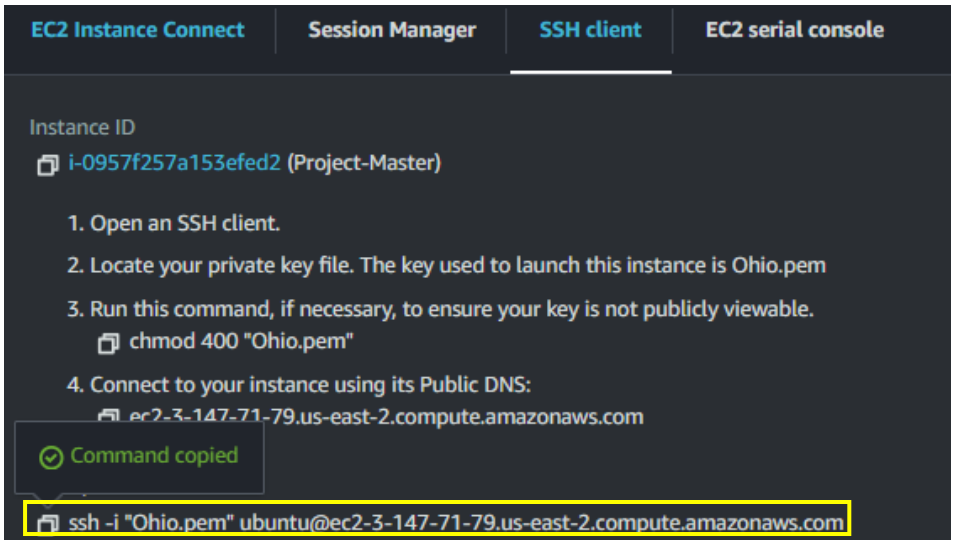
Job3 : prod



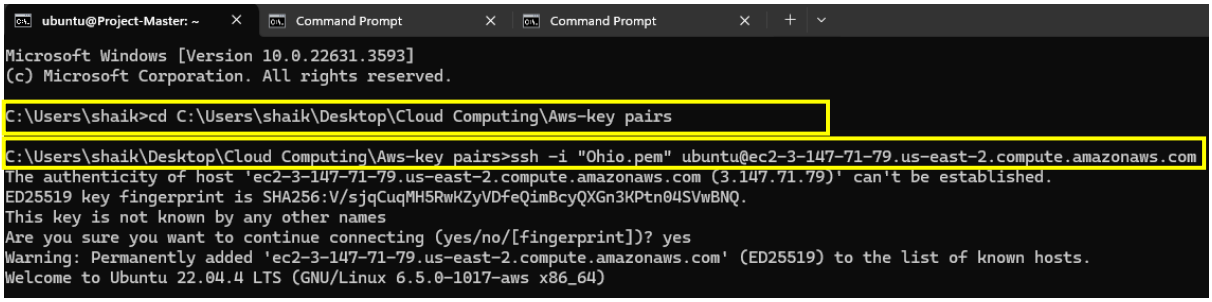
1. Launch 3 Instances Master, Test and Prod Servers



2. Copy the Master SSH Client Path Link



3. Go to Key-pair Path and Paste SSH Path and connect



4. Change Hostname

```
C:\WINDOWS\system32\cmd.   Command Prompt   Command Prompt
ubuntu@ip-172-31-5-2:~$ sudo hostnamectl set-hostname Project-Master
ubuntu@ip-172-31-5-2:~$ exit
logout
Connection to ec2-3-147-71-79.us-east-2.compute.amazonaws.com closed.

C:\Users\shaik\Desktop\Cloud Computing\Aws-key pairs>ssh -i "Ohio.pem" ubuntu@ec2-3-147-71-79.us-east-2.compute.amazonaws.com
```

5. Like How we Connect Master Server like Connect Test Server and Change Hostname for Better Understanding

```
ubuntu@Project-Master: ~   Command Prompt   Command Prompt
ubuntu@ip-172-31-11-191:~$ sudo hostnamectl set-hostname Test-Server
eubuntu@ip-172-31-11-191:~$ exit
logout
Connection to ec2-3-15-239-112.us-east-2.compute.amazonaws.com closed.

C:\Users\shaik\Desktop\Cloud Computing\Aws-key pairs>ssh -i "Ohio.pem" ubuntu@ec2-3-15-239-112.us-east-2.compute.amazonaws.com
```

6. Like How we Connect Master Server like Connect Prod Server and Change Hostname for Better Understanding

```
ubuntu@Project-Master: ~   Command Prompt   Command Prompt
ubuntu@ip-172-31-5-221:~$ sudo hostnamectl set-hostname Prod-Server
ubuntu@ip-172-31-5-221:~$ exit
logout
Connection to ec2-3-147-46-43.us-east-2.compute.amazonaws.com closed.

C:\Users\shaik\Desktop\Cloud Computing\Aws-key pairs>ssh -i "Ohio.pem" ubuntu@ec2-3-147-46-43.us-east-2.compute.amazonaws.com
```

7. Update apt in all 3 servers

```
ubuntu@Project-Master: ~   ubuntu@Test-Server: ~   ubuntu@Prod-Server: ~
ubuntu@Project-Master:~$ sudo apt update
```

```
ubuntu@Project-Master: ~   ubuntu@Test-Server: ~   ubuntu@Prod-Server: ~
ubuntu@Test-Server:~$ sudo apt update
```

```
ubuntu@Project-Master: ~   ubuntu@Test-Server: ~   ubuntu@Prod-Server: ~
ubuntu@Prod-Server:~$ sudo apt update
```

8. Install Ansible in Master server, follow the command

```
ubuntu@Project-Master: ~   ubuntu@Test-Server: ~   ubuntu@Prod-Server: ~
ubuntu@Project-Master:~$ sudo nano ansible.sh
```

Link: https://docs.ansible.com/ansible/latest/installation_guide/installation_distros.html

```
sudo apt update
sudo apt install software-properties-common
sudo add-apt-repository --yes --update ppa:ansible/ansible
sudo apt install ansible -y
```

9. Paste the commands and save it.

```
ubuntu@Project-Master: ~   ubuntu@Test-Server: ~   ubuntu@Prod-Server: ~
  GNU nano 6.2                                                    ansible.sh *
sudo apt update
sudo apt install software-properties-common
sudo add-apt-repository --yes --update ppa:ansible/ansible
sudo apt install ansible -y
```

10. Run the execute it.

```
ubuntu@Project-Master: ~   ubuntu@Test-Server: ~   ubuntu@Prod-Server: ~
ubuntu@Project-Master:~$ sudo nano ansible.sh
ubuntu@Project-Master:~$ bash ansible.sh
Hit:1 http://us-east-2.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://us-east-2.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://us-east-2.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu jammy-security InRelease
Reading package lists... 73%
```

11. Generating ssh-keygen > cd .ssh copy the public IP.

```
ubuntu@Project-Master: ~/.s   ubuntu@Test-Server: ~   ubuntu@Prod-Server: ~
ubuntu@Project-Master:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ubuntu/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ubuntu/.ssh/id_rsa
Your public key has been saved in /home/ubuntu/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:j1P+DBhTi2WQAFLU+8zsIpRAdxt1GfeswUQVZnVpAgM ubuntu@Project-Master
The key's randomart image is:
+---[RSA 3072]----+
|  .o+o.ooE+*=.=o+|
|  ... + .o.+.* o.|
| . . . +  + o =  |
|  .   o  = . o   |
|   . . =S o .    |
|    o   =O       |
|   .   .+ +      |
|    . . .. +     |
|     . .    o    |
+----[SHA256]-----+
ubuntu@Project-Master:~$ cd .ssh
ubuntu@Project-Master:~/.ssh$ ls
authorized_keys  id_rsa  id_rsa.pub
ubuntu@Project-Master:~/.ssh$ sudo cat id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQCxaPbRXuPq1sWWFdm5lI25/v3fulp5xXiZaWjt2pwSBWivtXafUCQ4oWQwYyEPMSdNrh3QwW8BLaBkFnxprxlNvZ3mnRjHHggVEQRNltzoN/Vc
OYI3K3EBZeYpD9Voc28syzXQXFAhuw8fRGGXBjvPUa1oLjZMLo63IpkMvPJ0mn3Y7DlLaccIY7jY0ygGaIqUYCOpkV2JiH82OFY1KylZ1of71MGMds9pt/O6BEIYHLQZ0VY1lMTdeCm7iRot7c7/
qFE/jk68ZG0PE5xMDZsBlDY4l7AgQWZqmiUOh0yzk5kbZ9tp5OyzDAfrGvDwTcVEOAQluDUicN6c1+w4ujEYuQWXMoVAiJ2AG9NQ5yyjq9wyx+VqM69gsbiWLWuDu4VlFM/SNIud9jIRcb9U6YsL
nRyIs9aoddwuMlU+aL3U1RCvKa5Q97PKShk2cYc6je3qh8STv8p2xHE+DJm6qtakRx2AMDr1OgPw5RAioxRMhb1caNO7FGfy7hSX5XBLSZE= ubuntu@Project-Master
ubuntu@Project-Master:~/.ssh$
```

12. Come to Test-server and cd .ssh and open the authorized_keys file

```
ubuntu@Project-Master: ~/.ss   ubuntu@Test-Server: ~/.ssh   ubuntu@Prod-Server: ~/.ssh
ubuntu@Test-Server:~$ cd .ssh
ubuntu@Test-Server:~/.ssh$ ls
authorized_keys
ubuntu@Test-Server:~/.ssh$ nano authorized_keys
```

13. Paste it and save it

```
ubuntu@Project-Master: ~/.ss   ubuntu@Test-Server: ~/.ssh   ubuntu@Prod-Server: ~/.ssh
  GNU nano 6.2                                          authorized_keys *
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCjN3/+BsHOIJyV1/+l/XAiEC/fV2qP/tbYGI7hJDGRTzmt7v68Q262/CAD/vHcWwLb5R3LvXnEWQsI
Project-Master
```

14. Come to Prod-server and cd .ssh and open the authorized_keys file

```
ubuntu@Project-Master: ~/.ss   ubuntu@Test-Server: ~/.ssh   ubuntu@Prod-Server: ~/.ssh
ubuntu@Prod-Server:~$ cd .ssh
ubuntu@Prod-Server:~/.ssh$ ls
authorized_keys
ubuntu@Prod-Server:~/.ssh$ nano authorized_keys
```

15. Paste it and save it

```
ubuntu@Project-Master: ~/.ss   ubuntu@Test-Server: ~/.ssh   ubuntu@Prod-Server: ~/.ssh
  GNU nano 6.2                                   authorized_keys *
vdUVe7FJuYBapY88y+eyjO4LooXEL05gyFw4WUqS6k3WIk6ESnw0rjHzRwbYM3P9ZC5xr8q3QATst+u/XXG6xn8KRYTQtnXVbyP Ohio
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQCxaPbRXuPq1sWWFdm5lI25/v3fulp5xXiZaWjt2pwSBWivtXafUCQ4oWQwYyEPMSdNrh3QwW8BLaBkFnxprxlNvZ3mnRjHHggVEQRNltzoN/V
```

16. Come to Master Server create Host file using without sudo, we cant written the file

```
ubuntu@Project-Master: /etc,   ubuntu@Test-Server: ~/.ssh   ubuntu@Prod-Server: ~/.ssh
ubuntu@Project-Master:~$ cd /etc/ansible
ubuntu@Project-Master:/etc/ansible$ ls
ansible.cfg  hosts  roles
ubuntu@Project-Master:/etc/ansible$ nano hosts
```

17. Use sudo in ansible folder

```
ubuntu@Project-Master: /etc,   ubuntu@Test-Server: ~/.ssh   ubuntu@Prod-Server: ~/.ssh
ubuntu@Project-Master:~$ cd /etc/ansible
ubuntu@Project-Master:/etc/ansible$ ls
ansible.cfg  hosts  roles
ubuntu@Project-Master:/etc/ansible$ nano hosts
ubuntu@Project-Master:/etc/ansible$ sudo nano hosts
```

18. Copy the Private IP of Test Server.

Test-Server	i-04b9d67f329c07539	Running	t2.micro	2/2 checks passed	View alarms +	us-east-2a
Prod-Server	i-0083d65942fb73468	Running	t2.micro	2/2 checks passed	View alarms +	us-east-2a

i-04b9d67f329c07539 (Test-Server)

Instance ID	Public IPv4 address	Private IPv4 addresses
i-04b9d67f329c07539 (Test-Server)	3.15.239.112 | open address	172.31.11.191

19. Copy the Private IP of Prod Server.

Test-Server	i-04b9d67f329c07539	Running	t2.micro	2/2 checks passed	View alarms +	us-
Prod-Server	i-0083d65942fb73468	Running	t2.micro	2/2 checks passed	View alarms +	us-

i-0083d65942fb73468 (Prod-Server)

Instance ID	Public IPv4 address	Private IPv4 addresses
i-0083d65942fb73468 (Prod-Server)	3.147.46.43 | open address	172.31.5.221

20. Paste In the host file

```
ubuntu@Project-Master: /etc,   ubuntu@Test-Server: ~/.ssh   ubuntu@Prod-Server: ~/.ssh
  GNU nano 6.2                                   hosts *
[group]
Test ansible_host=172.31.11.191
Prod ansible_host=172.31.5.221
```

21. Now ping all, its connected with both the server.

```
ubuntu@Project-Master: /etc,   ubuntu@Test-Server: ~/.ssh   ubuntu@Prod-Server: ~/.ssh
ubuntu@Project-Master:/etc/ansible$ ansible -m ping all
The authenticity of host '172.31.11.191 (172.31.11.191)' can't be established.
ED25519 key fingerprint is SHA256:PKrXyfP54xOG6anRzNobvR7B0B3hNmUZm32gp/ZSpPk.
This key is not known by any other names
The authenticity of host '172.31.5.221 (172.31.5.221)' can't be established.
ED25519 key fingerprint is SHA256:uNrMb8ePBDjm9Ie0rFWZqv1PPULSgDUZojJInUsCOBE.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Test | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
yes
Prod | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
ubuntu@Project-Master:/etc/ansible$
```

22. Now Create sudo nano play.yaml, it's a playbook

```
ubuntu@Project-Master: /etc,   ubuntu@Test-Server: ~/.ssh   ubuntu@Prod-Server: ~/.ssh
ubuntu@Project-Master:/etc/ansible$ sudo nano play.yaml
```

23. Here we executing a script in localhost and all hosts

In localhost executing JJD.sh > Installing Jenkins,java,Dockers

In All(Host file > IP's) executing JD.sh > Installing Java,Dockers

```
ubuntu@Project-Master: /etc,   ubuntu@Test-Server: ~/.ssh   ubuntu@Prod-Server: ~/.ssh
 GNU nano 6.2                                                    play.yaml *
---
- name: executing script on master
  hosts: localhost
  become: true
  tasks:
    - name: executing JJD.sh script on master
      script: JJD.sh

- name: executing script on test and prod
  hosts: all
  become: true
  tasks:
    - name: executing JD.sh script on test and prod
      script: JD.sh
```

- name: executing script on master

  hosts: localhost

  become: true

  tasks:

```
  - name: executing JJD.sh script on master
    script: JJD.sh

- name: executing script on test and prod
  hosts: all
  become: true
  tasks:
    - name: executing JD.sh script on test and prod
      script: JD.sh
```

24. Now I am Creating JJD.sh Script file

```
ubuntu@Project-Master: /etc,   ubuntu@Test-Server: ~/.ssh   ubuntu@Prod-Server: ~/.ssh
ubuntu@Project-Master:/etc/ansible$ sudo nano JJD.sh
```

25. Pasting the commands which are going to install Java,Dockers, Jenkins in localhost

```
ubuntu@Project-Master: /etc,   ubuntu@Test-Server: ~/.ssh   ubuntu@Prod-Server: ~/.ssh
  GNU nano 6.2                                                    JJD.sh *
sudo apt update
sudo apt install openjdk-17-jdk -y
sudo apt install docker.io -y
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
```

```
sudo apt update                                   #update the package
sudo apt install openjdk-17-jdk -y                #Installing java
sudo apt install docker.io -y                     #install dockers
```

Jenkins Installation link : https://www.jenkins.io/doc/book/installing/linux/

```
sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \          #Installing Jenkins
  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
```

26. Now Creating JD.sh script file

```
ubuntu@Project-Master: /etc,   ubuntu@Test-Server: ~/.ssh   ubuntu@Prod-Server: ~/.ssh
ubuntu@Project-Master:/etc/ansible$ sudo nano JJD.sh
ubuntu@Project-Master:/etc/ansible$ ls
JJD.sh  ansible.cfg  hosts  play.yaml  roles
ubuntu@Project-Master:/etc/ansible$ sudo nano JD.sh
```

27. Paste the Commands for install java, dockers in JD script file

```
ubuntu@Project-Master: /etc,   ubuntu@Test-Server: ~/.ssh   ubuntu@Prod-Server: ~/.ssh
  GNU nano 6.2                                                    JD.sh *
sudo apt update
sudo apt install openjdk-17-jdk -y
sudo apt install docker.io -y
```

28. Now this is Playbook which we written

```
ubuntu@Project-Master: /etc,   ubuntu@Test-Server: ~/.ssh   ubuntu@Prod-Server: ~/.ssh
ubuntu@Project-Master:/etc/ansible$ cat play.yaml
---
 - name: executing script on master
   hosts: localhost
   become: true
   tasks:
     - name: executing JJD.sh script on master
       script: JJD.sh

 - name: executing script on test and prod
   hosts: all
   become: true
   tasks:
     - name: executing JD.sh script on test and prod
       script: JD.sh
```

29. Syntax check passed, and dry run also passed

```
ubuntu@Project-Master: /etc,   ubuntu@Test-Server: ~/.ssh   ubuntu@Prod-Server: ~/.ssh
ubuntu@Project-Master:/etc/ansible$ ansible-playbook play.yaml --syntax-check

playbook: play.yaml
ubuntu@Project-Master:/etc/ansible$ ansible-playbook play.yaml --check

PLAY [executing script on master] *********************************************************************
TASK [Gathering Facts] ********************************************************************************
ok: [localhost]

TASK [executing JJD.sh script on master] **************************************************************
skipping: [localhost]

PLAY [executing script on test and prod] **************************************************************

TASK [Gathering Facts] ********************************************************************************
ok: [Test]
ok: [Prod]

TASK [executing JD.sh script on test and prod] ********************************************************
skipping: [Test]
skipping: [Prod]

PLAY RECAP ********************************************************************************************
Prod                       : ok=1    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
Test                       : ok=1    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
localhost                  : ok=1    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0

ubuntu@Project-Master:/etc/ansible$
```

30. Executing the scripts according Assignment to servers, Now its installed successfully.

```
ubuntu@Project-Master: /etc,   ubuntu@Test-Server: ~/.ssh   ubuntu@Prod-Server: ~/.ssh

ubuntu@Project-Master:/etc/ansible$ ansible-playbook play.yaml

PLAY [executing script on master] ***********************************************************************************
TASK [Gathering Facts] ***********************************************************************************************
ok: [localhost]

TASK [executing JJD.sh script on master] ****************************************************************************
changed: [localhost]

PLAY [executing script on test and prod] ****************************************************************************
TASK [Gathering Facts] ***********************************************************************************************
ok: [Test]
ok: [Prod]

TASK [executing JD.sh script on test and prod] **********************************************************************
changed: [Test]
changed: [Prod]

PLAY RECAP ***********************************************************************************************************
Prod                       : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
Test                       : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
localhost                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

ubuntu@Project-Master:/etc/ansible$
```

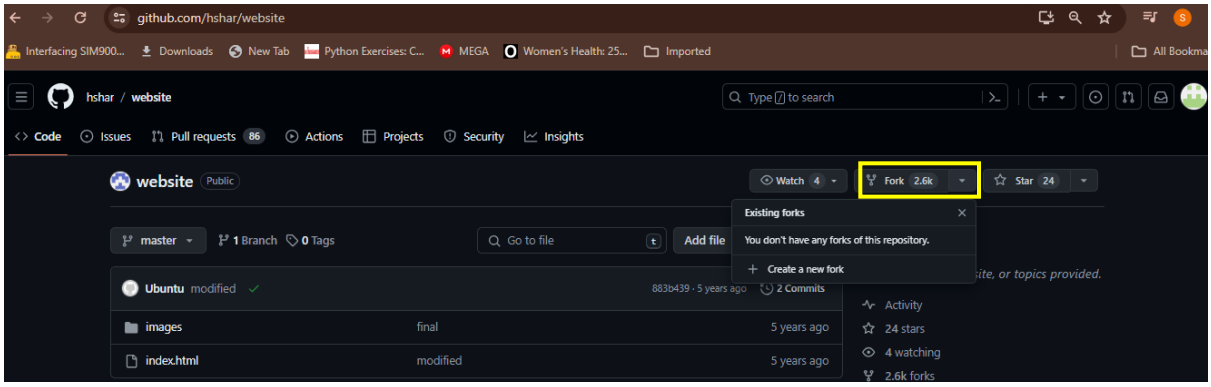
31. We can see dockers and java available in Test server.

```
ubuntu@Project-Master: /etc/   ubuntu@Test-Server: ~   ubuntu@Prod-Server: ~

ubuntu@Test-Server:~$ docker --version
Docker version 24.0.7, build 24.0.7-0ubuntu2~22.04.1
ubuntu@Test-Server:~$ java --version
openjdk 17.0.11 2024-04-16
OpenJDK Runtime Environment (build 17.0.11+9-Ubuntu-122.04.1)
OpenJDK 64-Bit Server VM (build 17.0.11+9-Ubuntu-122.04.1, mixed mode, sharing)
ubuntu@Test-Server:~$
```

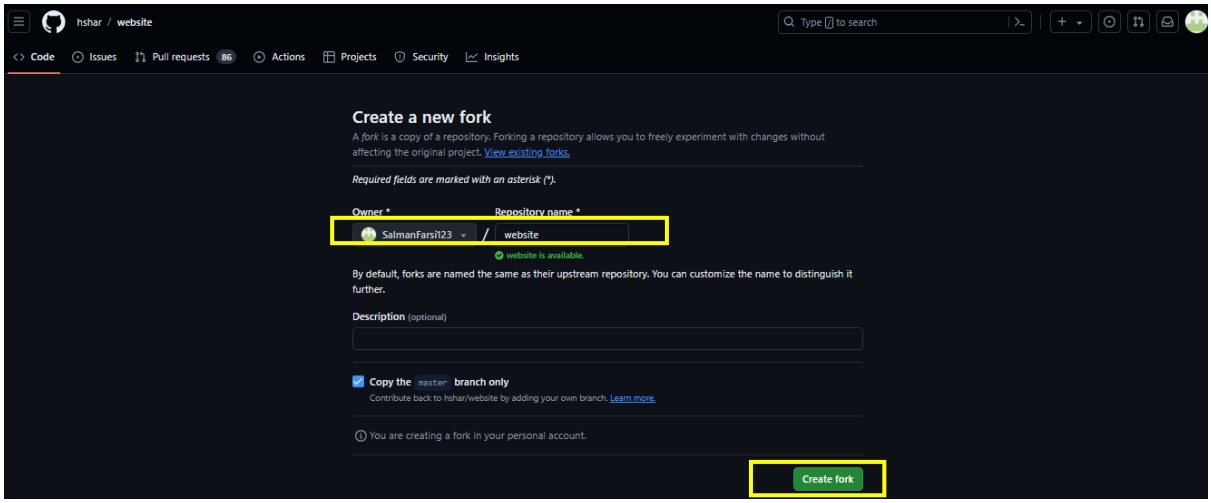
32. **We can see dockers and java available in Prod server.**

```
ubuntu@Project-Master: /etc/   ubuntu@Test-Server: ~   ubuntu@Prod-Server: ~

ubuntu@Prod-Server:~$ docker --version
Docker version 24.0.7, build 24.0.7-0ubuntu2~22.04.1
ubuntu@Prod-Server:~$ java --version
openjdk 17.0.11 2024-04-16
OpenJDK Runtime Environment (build 17.0.11+9-Ubuntu-122.04.1)
OpenJDK 64-Bit Server VM (build 17.0.11+9-Ubuntu-122.04.1, mixed mode, sharing)
ubuntu@Prod-Server:~$
```

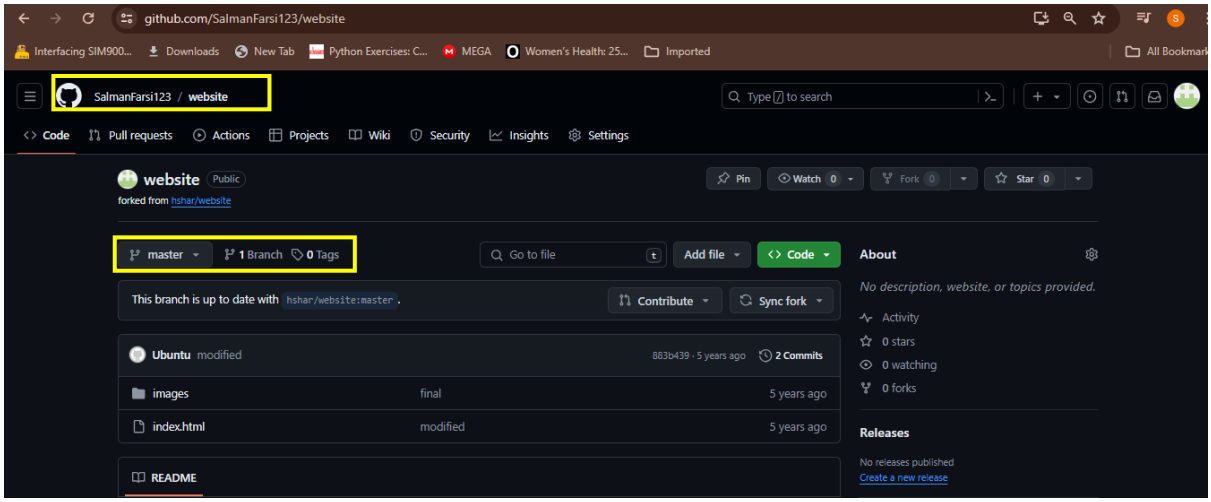
33. Go to Github: https://github.com/hshar/website.git and Click Fork



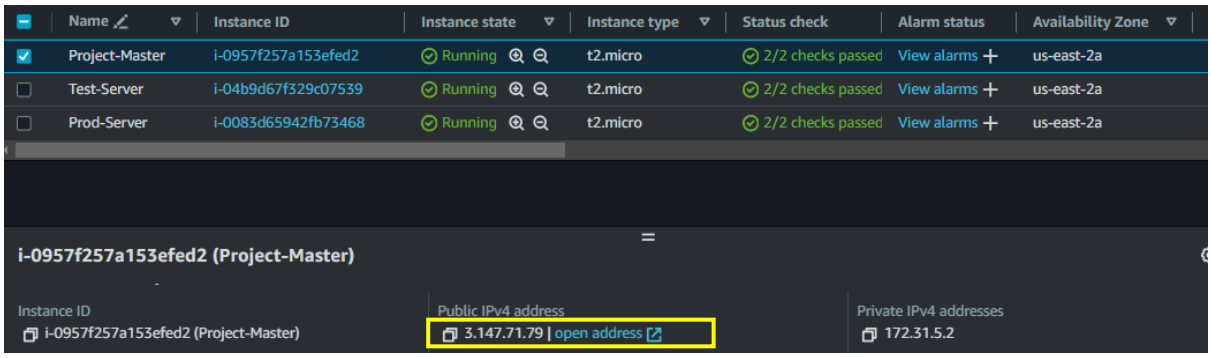
34. I am keeping Repo name as website only and Click create fork



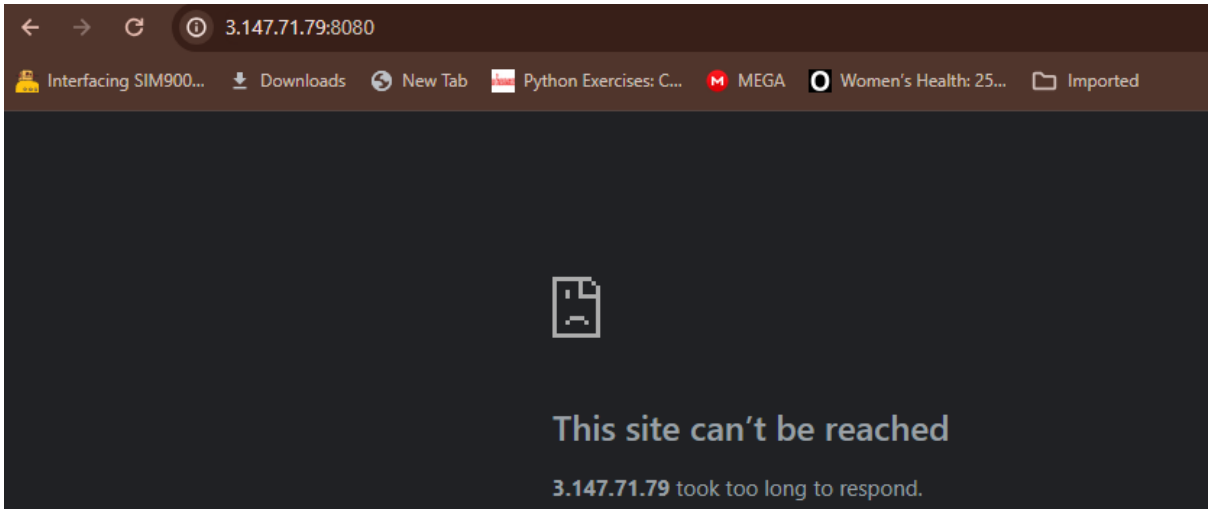
35. Now the Repo Available in my Git-Hub Account.



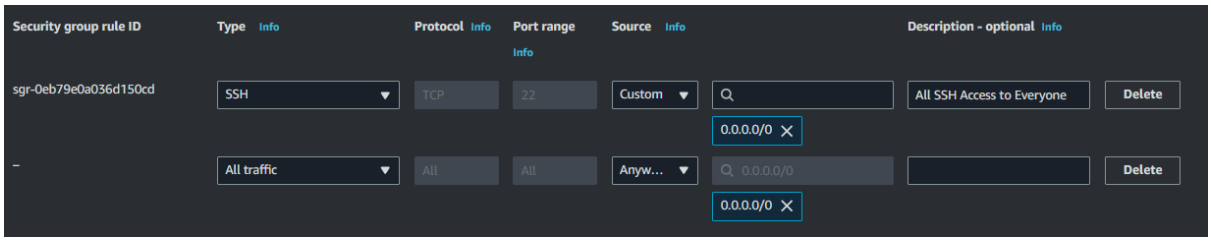
36. Copy Project-Master Public IP



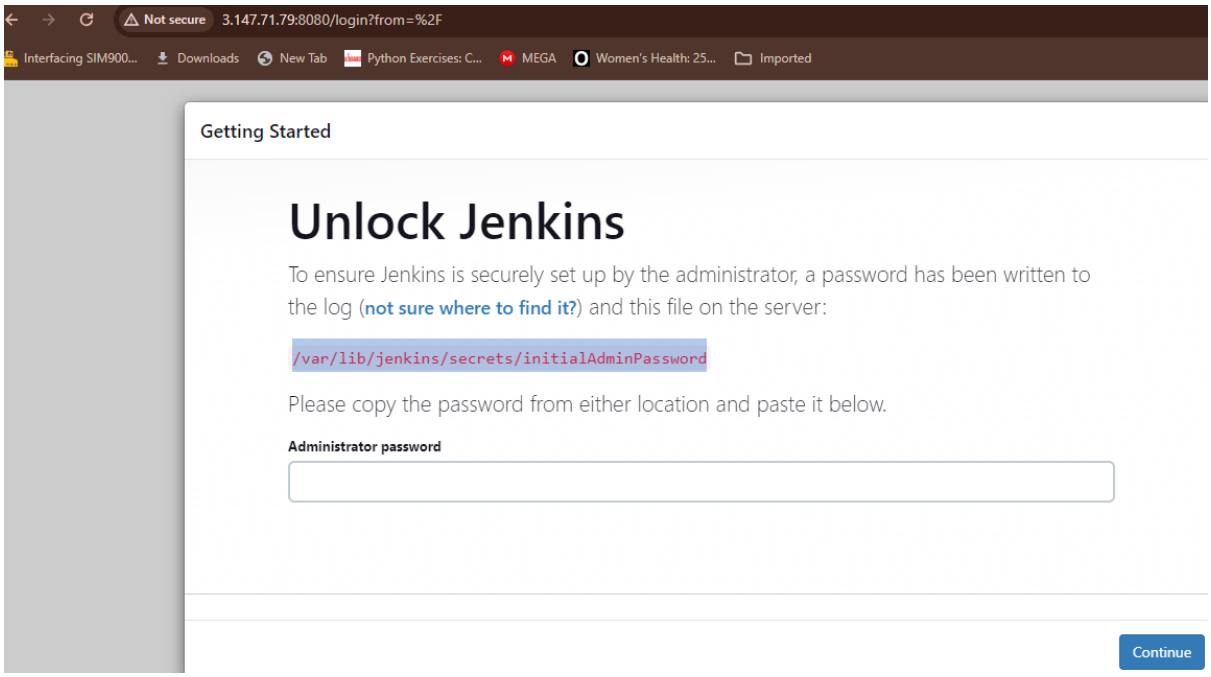
37. And Paste Public IP in the browser to login Jenkins, But its Not working 🧐 🤔



38. Allow TCP/8080 port in Security group

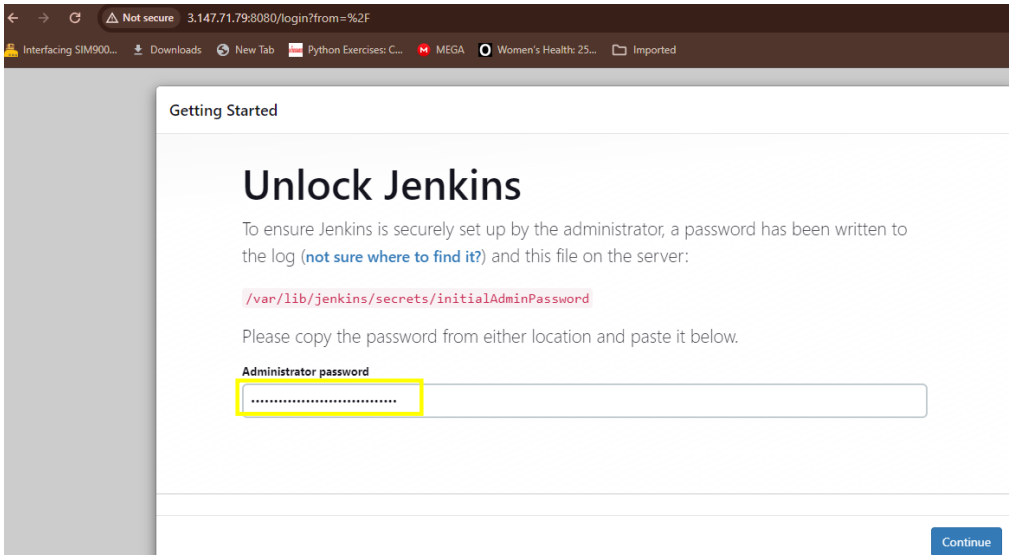


39. Now its working fine, copy the path

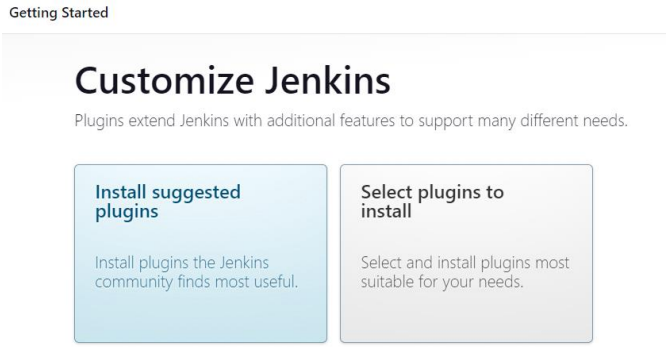


40. Copy the Administrator password

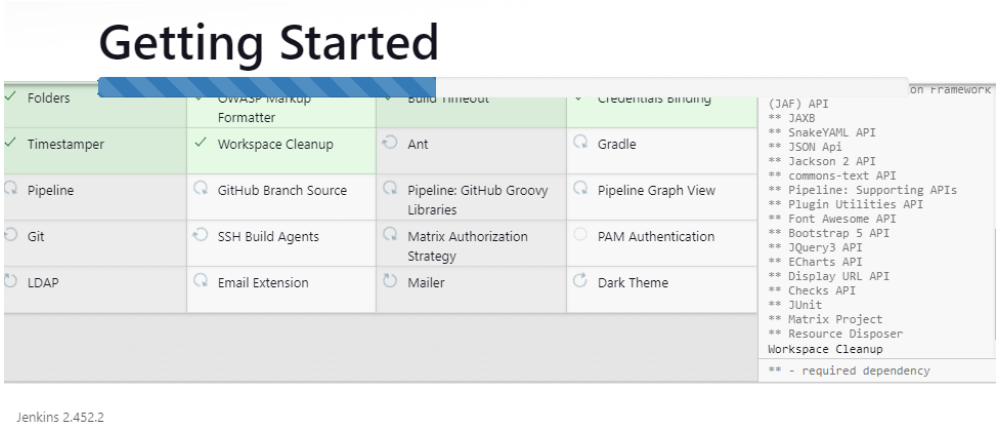
41. paste and continue



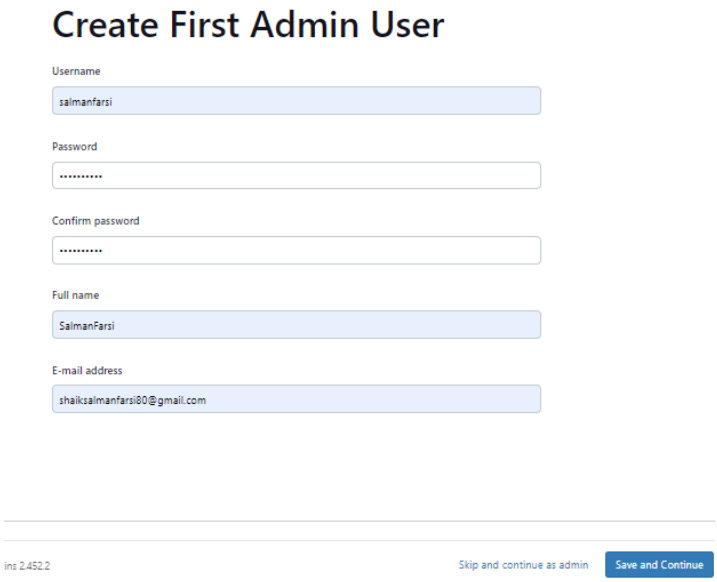
42. install suggested plugins



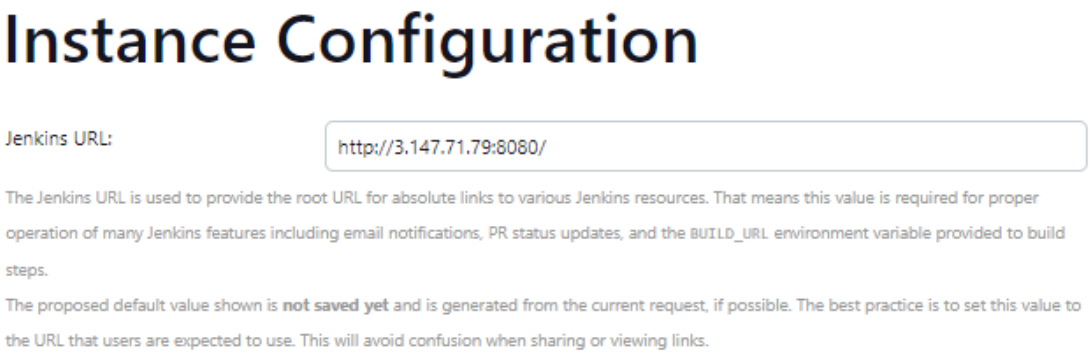
43. Its installing



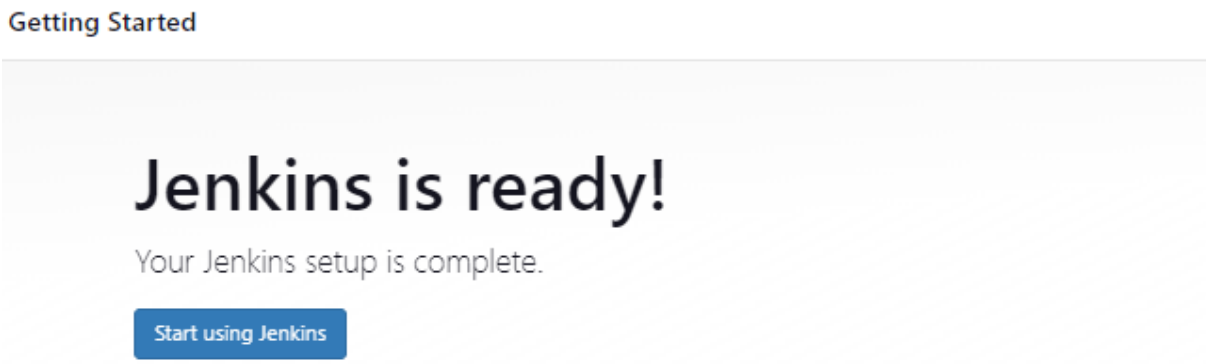
44. Create user



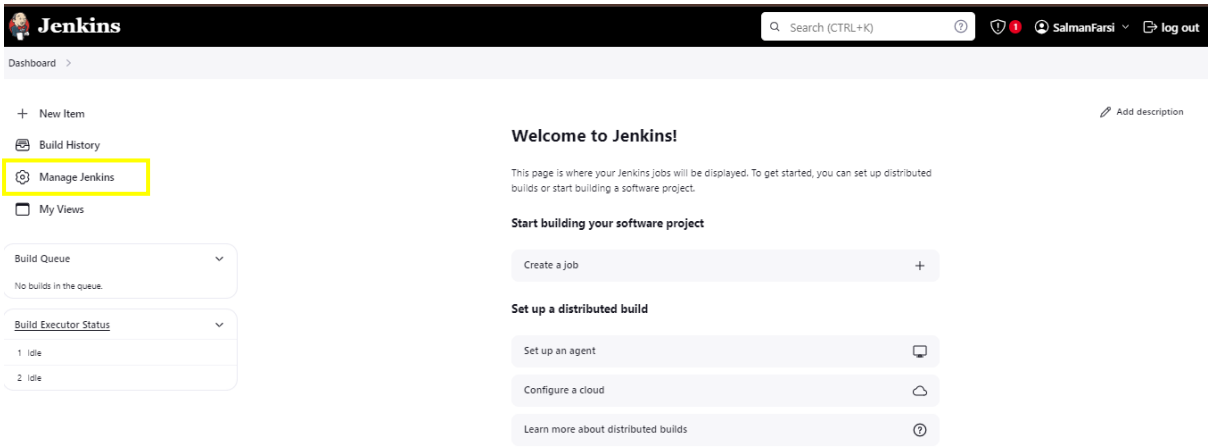
45. Continue



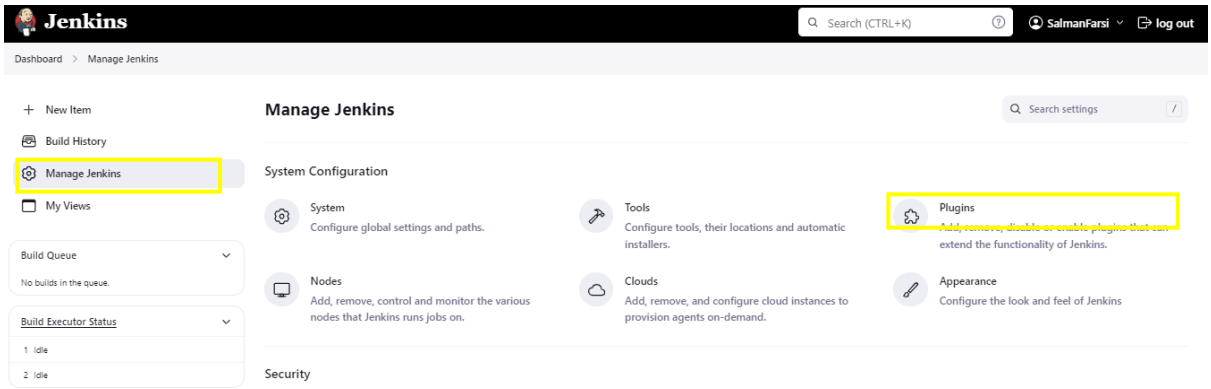
46. Start



47. Dashboard View > Click Manage Jenkins

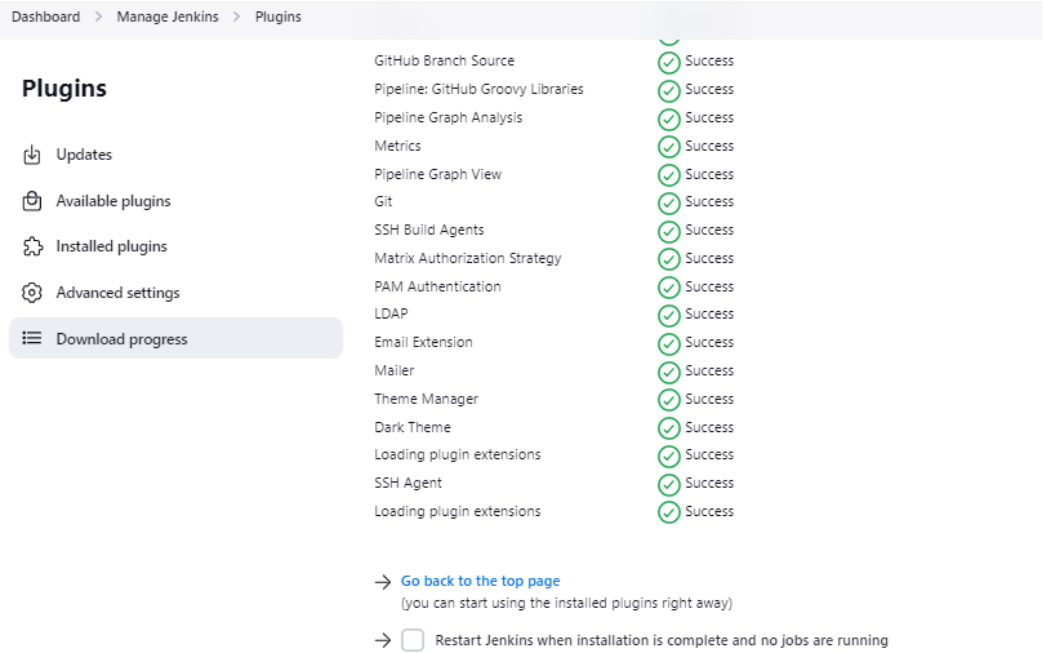


48. Go to Plugins

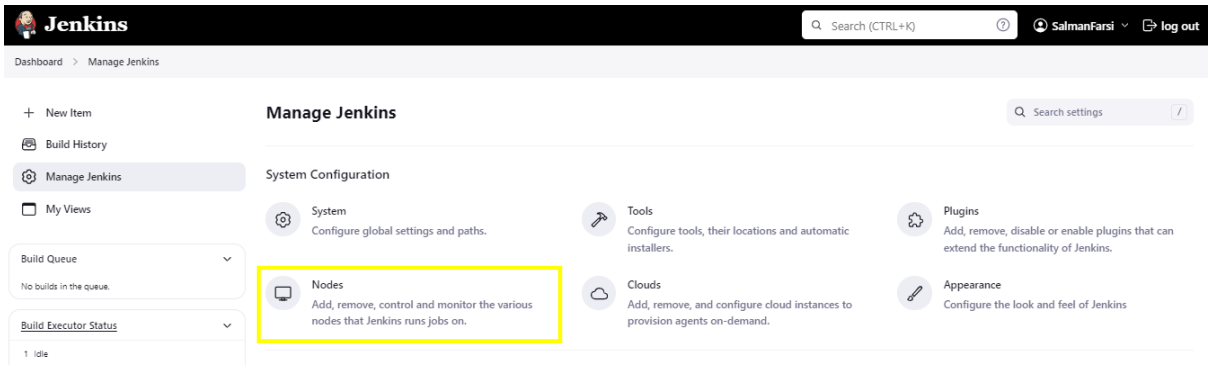


49. Click Available Plugins and Search SSH Agent and install it.

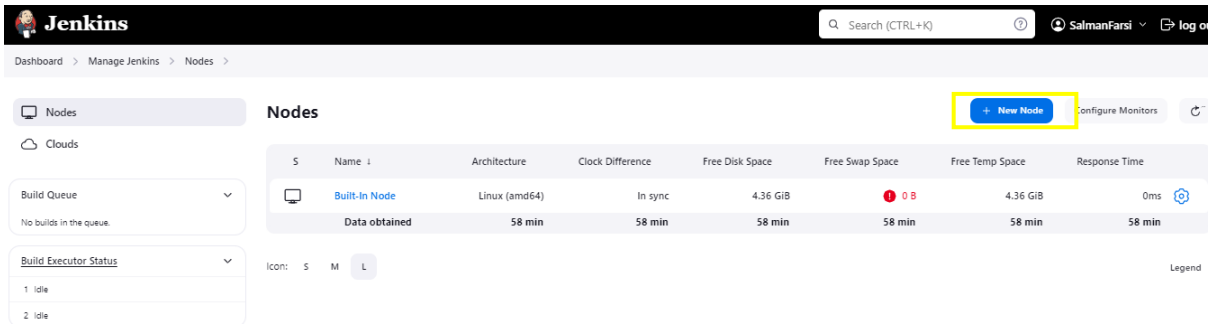
50. its downloaded



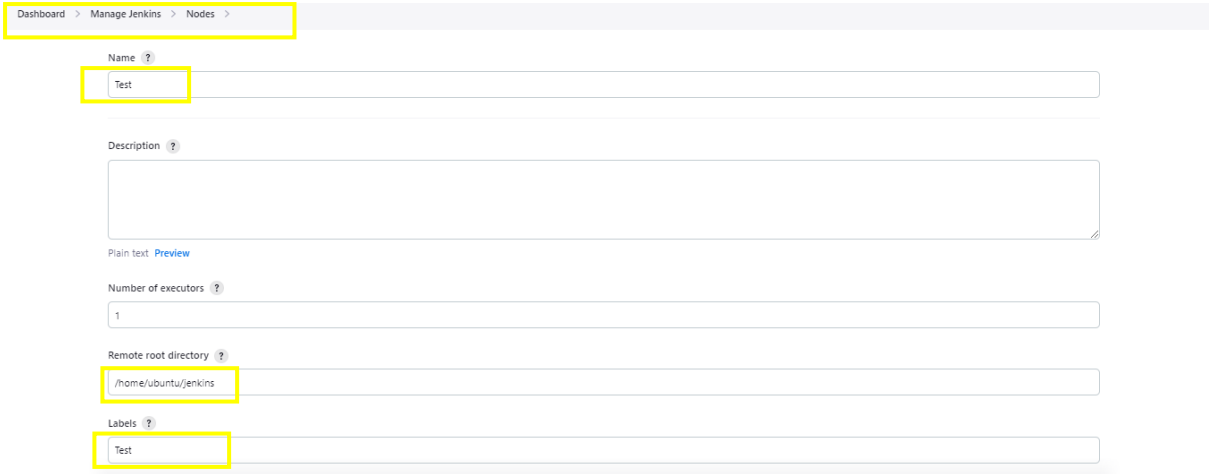
51. Come Back and Manage Jenkins > Click Nodes



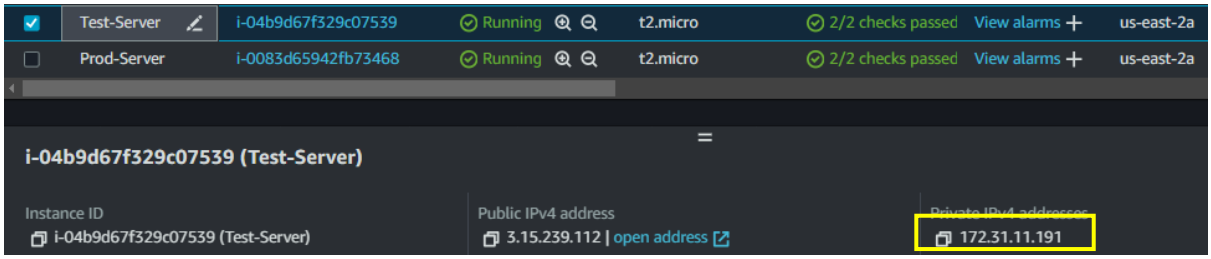
52. Currently there is no slave servers connected, Click New node



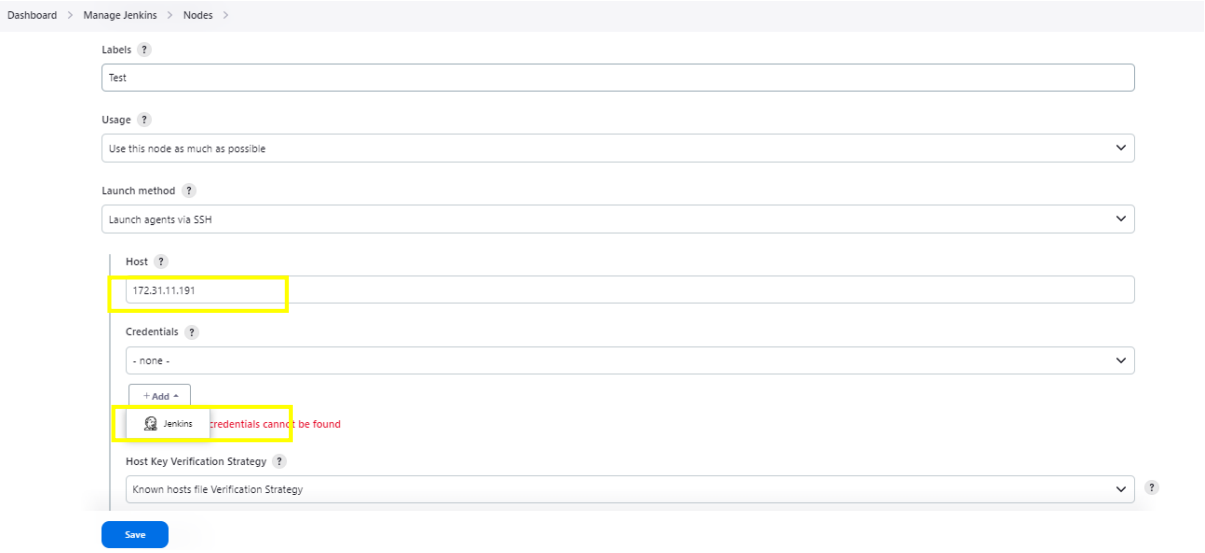
53. Giving name As Test and remote root directory /home/ubuntu/Jenkins > After executing the data going to save in this location.



54. Copying the private IP of Test Server

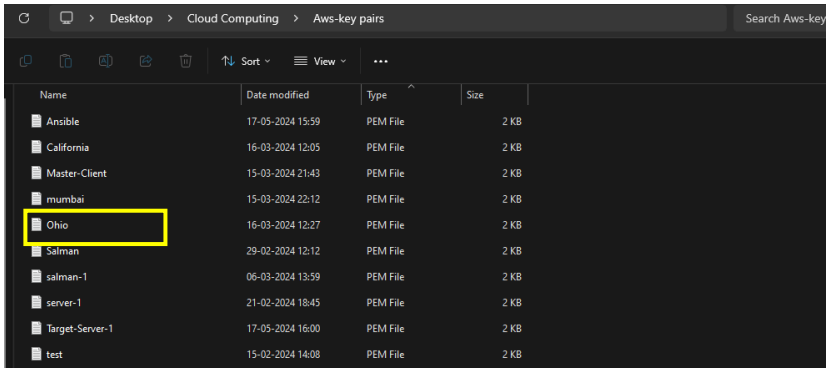


55. Launch Method is launch agents via ssh and Host : private IP and Credentials is our private key pair Click Add



56. Kind: SSH Username with Private Key and Scroll

57. This is My Personal laptop which I stored all key pairs in one folder



58. Copy the key which attached with the Test server, I used the Ohio key pair for all 3 instances

59. Username was ubuntu and pasting the private key-pair and click Add

Jenkins Credentials Provider: Jenkins

Username

ubuntu

Treat username as secret ?

Private Key

Enter directly

Key

Enter New Secret Below

5MT9AoGBALzG+ImiuPmCvnH687oKIXqiowWzjSmoQl+cyLHYvmfs+RBDqvJJutqa
j4xu6vVzrENpTSGw+18TV0BitkA7CcZNLHv0NQw9XzBqGKuEMNZ6J7oweU3Cizs5
ZgioQCnQ6tJ4W69LNDIJNCgKkn7LwwXIzWYuTm9+e0/YvvIiNrFd

Passphrase

Cancel Add

60. Change Host Key Verification Strategy: Non Verifying and Save it

Dashboard > Manage Jenkins > Nodes >

Usage ?

Use this node as much as possible

Launch method ?

Launch agents via SSH

Host ?

172.31.11.191

Credentials ?

ubuntu

+ Add

Host Key Verification Strategy ?

Non verifying Verification Strategy

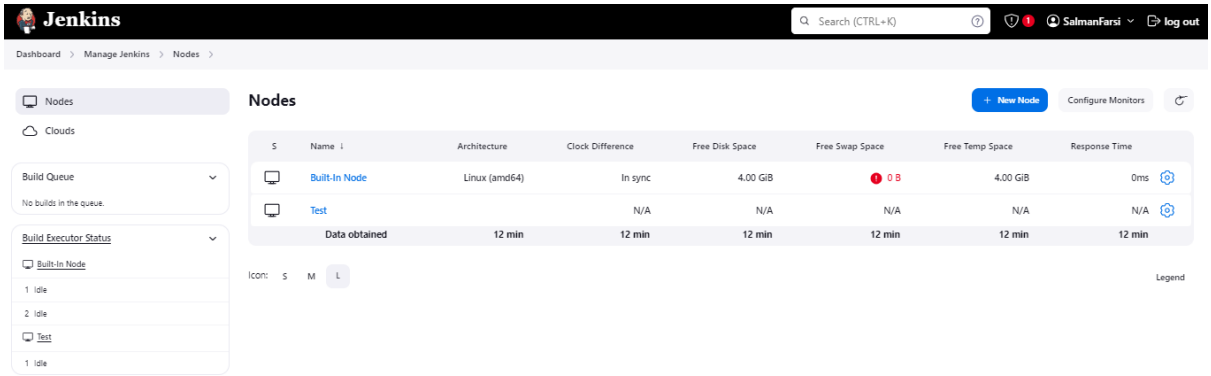
Advanced

Availability ?

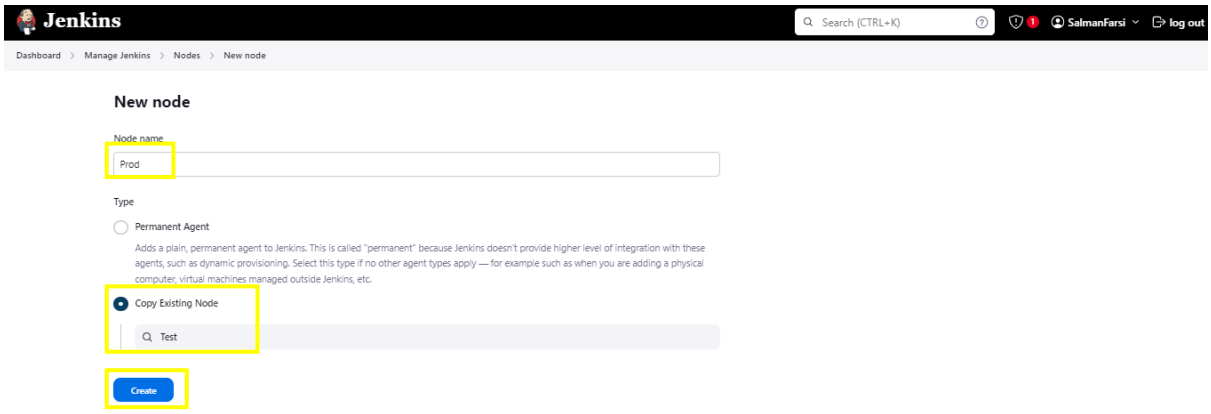
Keep this agent online as much as possible

Save

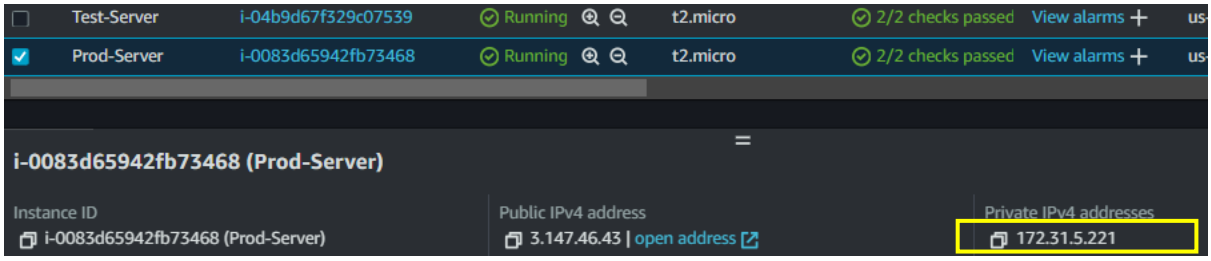
61. Its Test Server Connected Successfully.



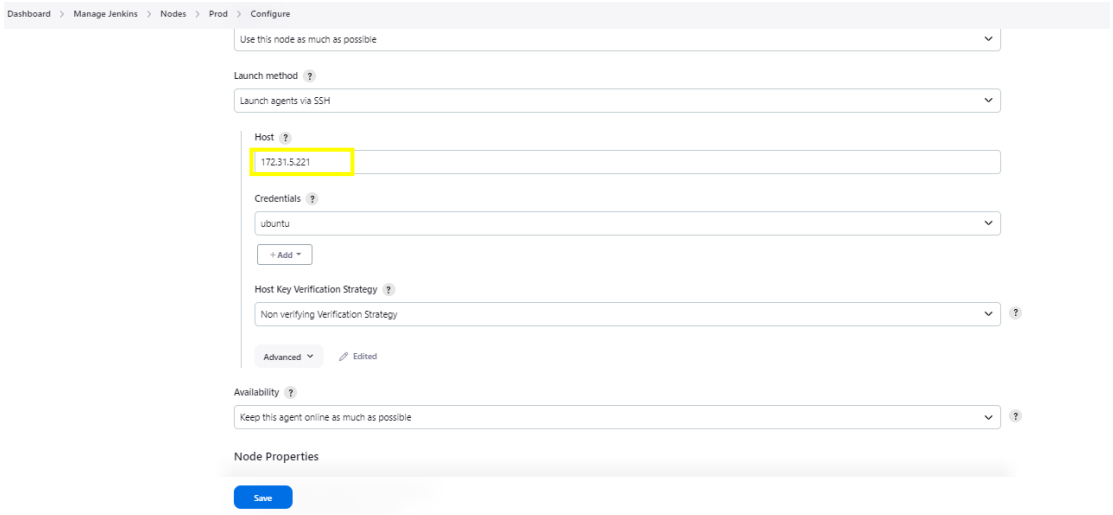
62. Now Creating Another Node Named as Prod and Copying from Existing Node and Click Create.



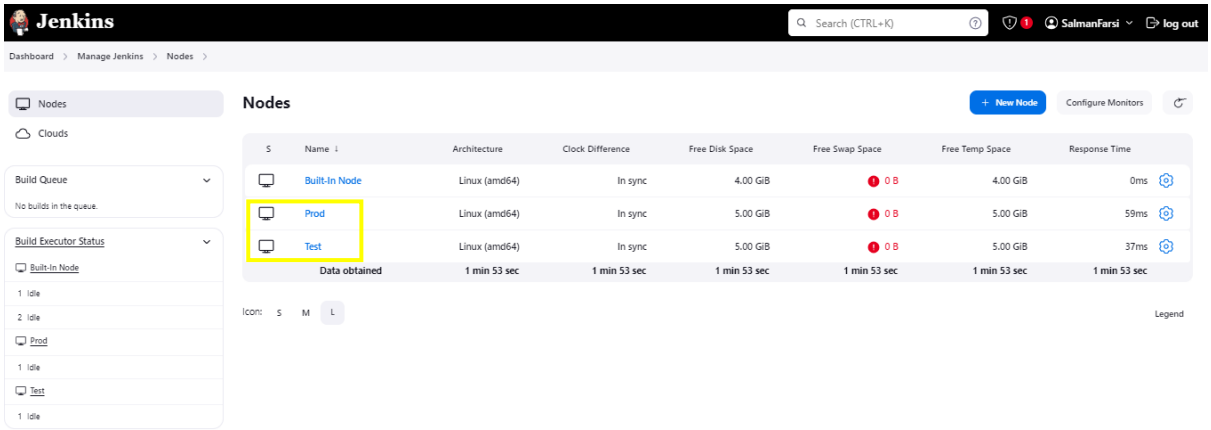
63. Copy the Prod Private IP.



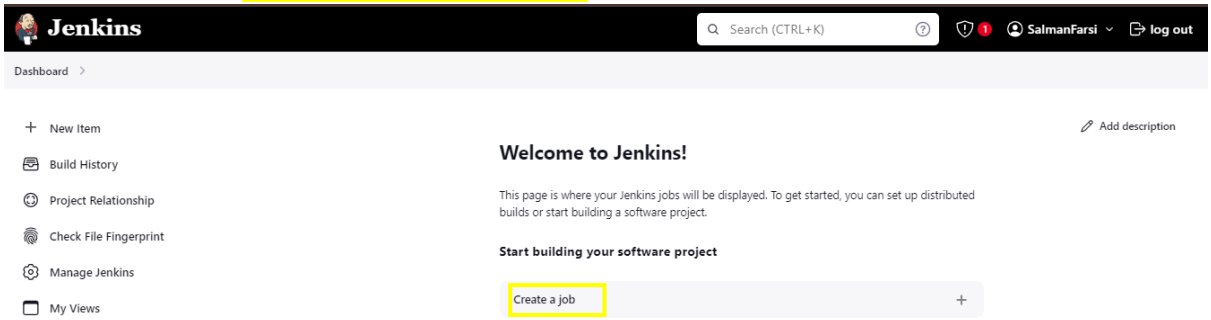
64. And Change It, If u give a label on Test-server and then Here(Prod-server)also U need to change.



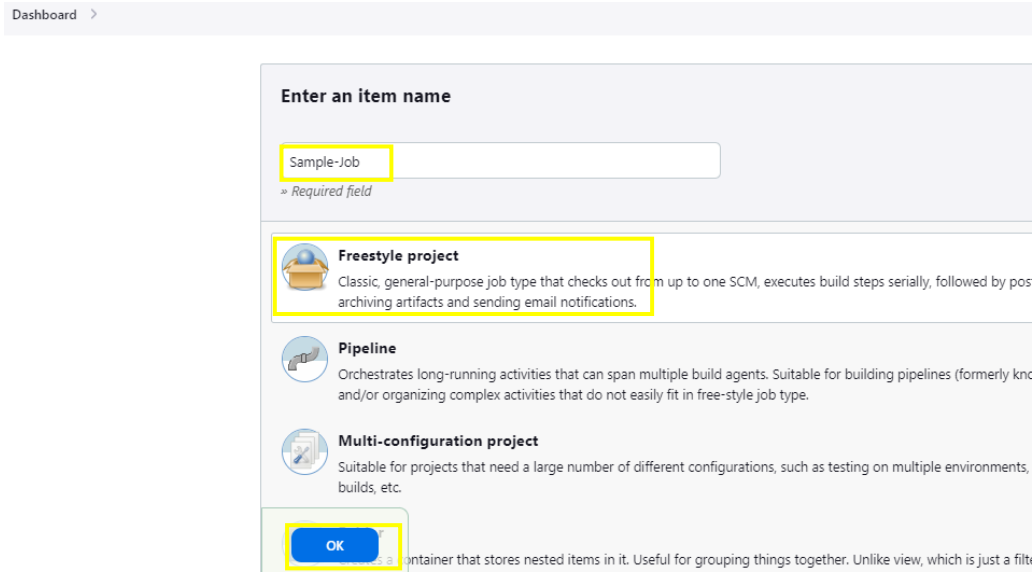
65. Now Prod-Server also Connected



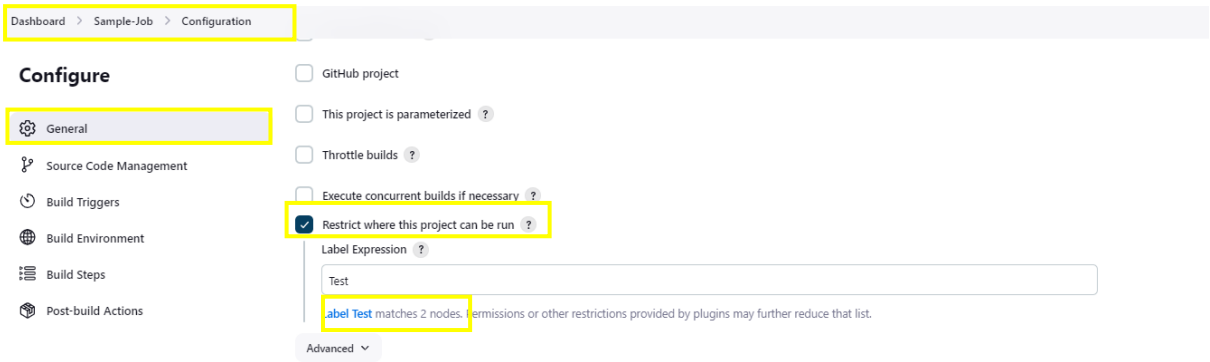
66. Now Creating a Sample-Job, Click Create Job



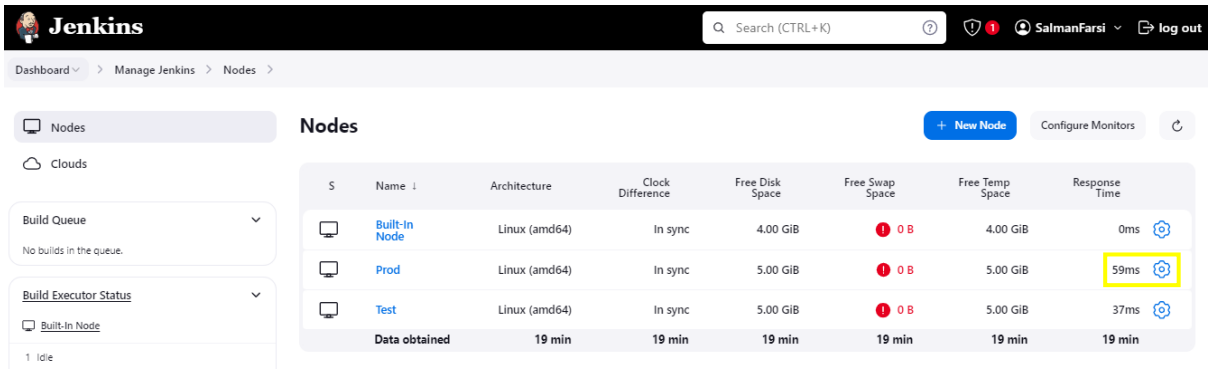
67. Give Name and Selecting Freestyle Project and Click Ok to Create



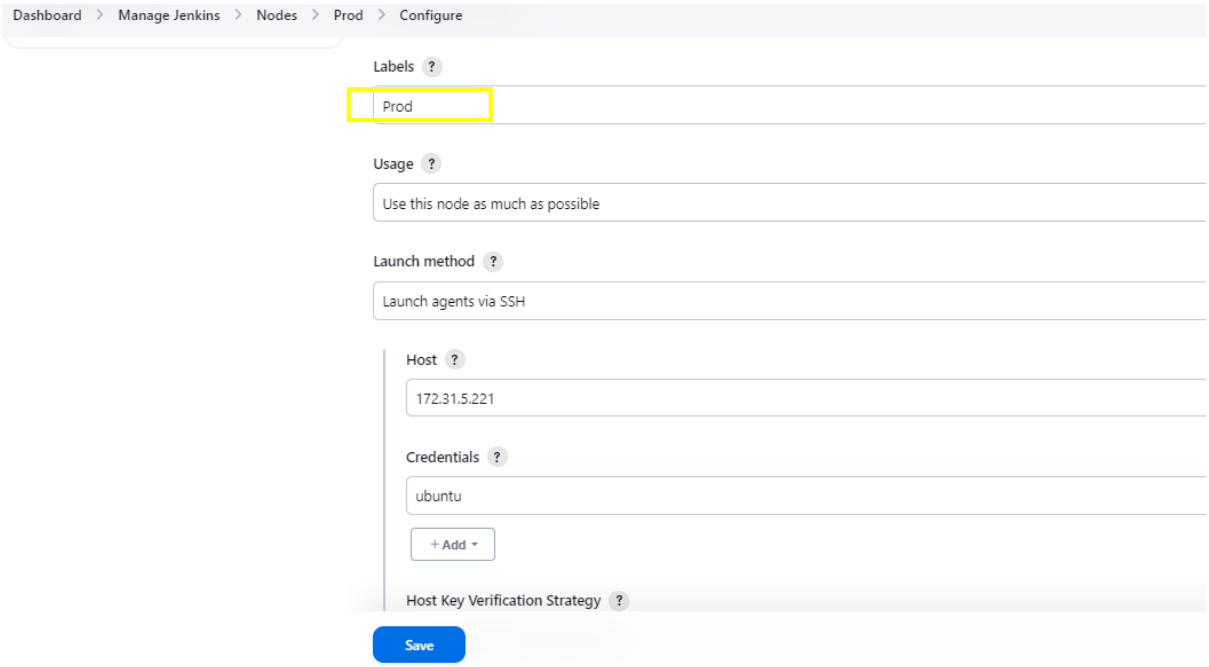
68. Sample-JoB > Configuration > General, I Selected Restrict where this Project can be run Here Telling that I have label name as Test with 2 nodes, I copied the node but not change that label on prod.



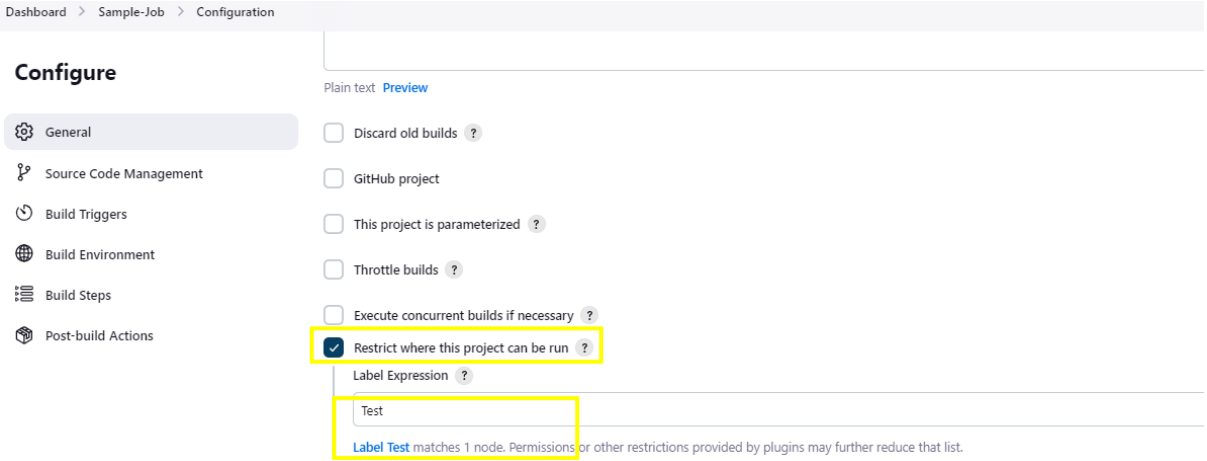
69. Go to Nodes and Click settings symbol of Prod-server



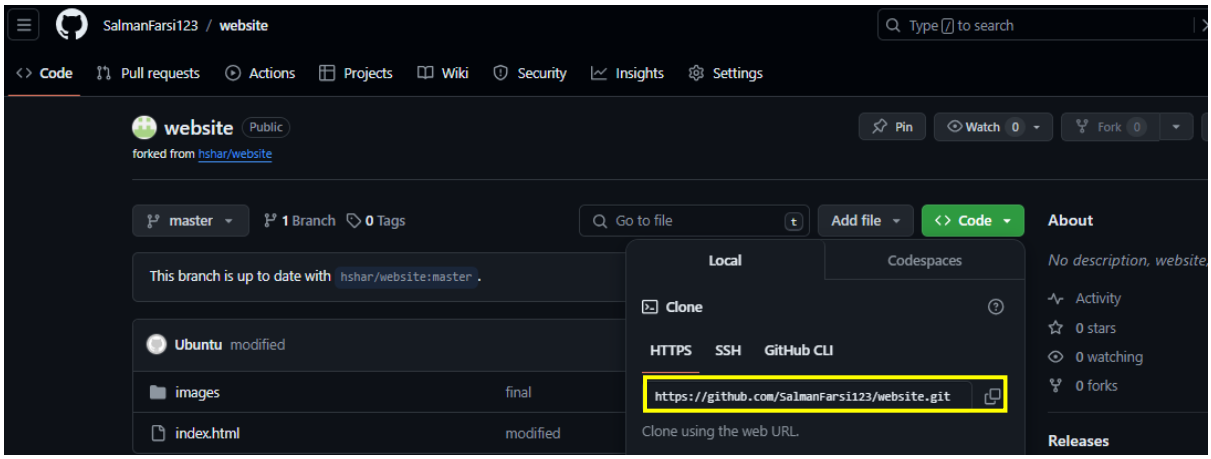
70. Change Label name Test to Prod and Save it



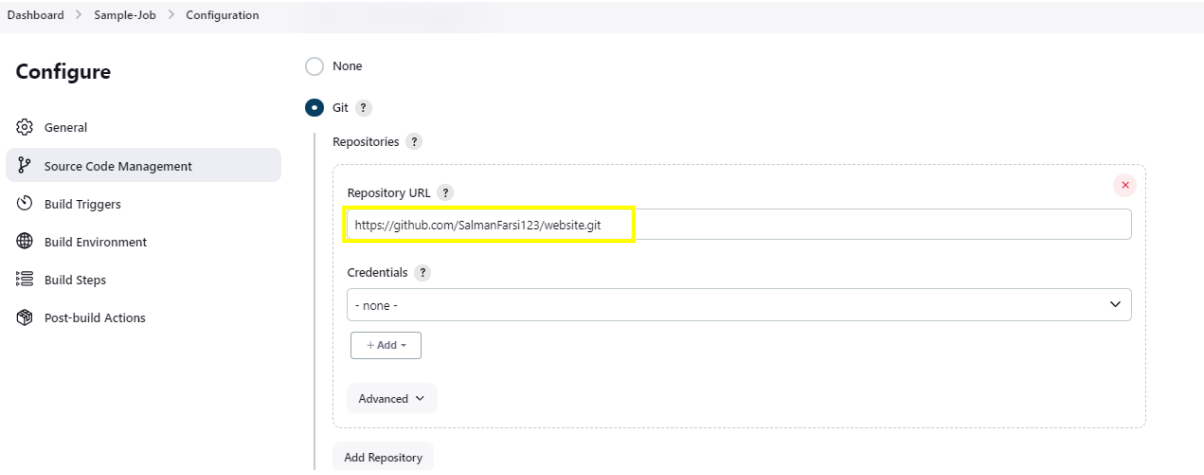
71. Now I am Selecting Test label, now its showing Mathces 1 node



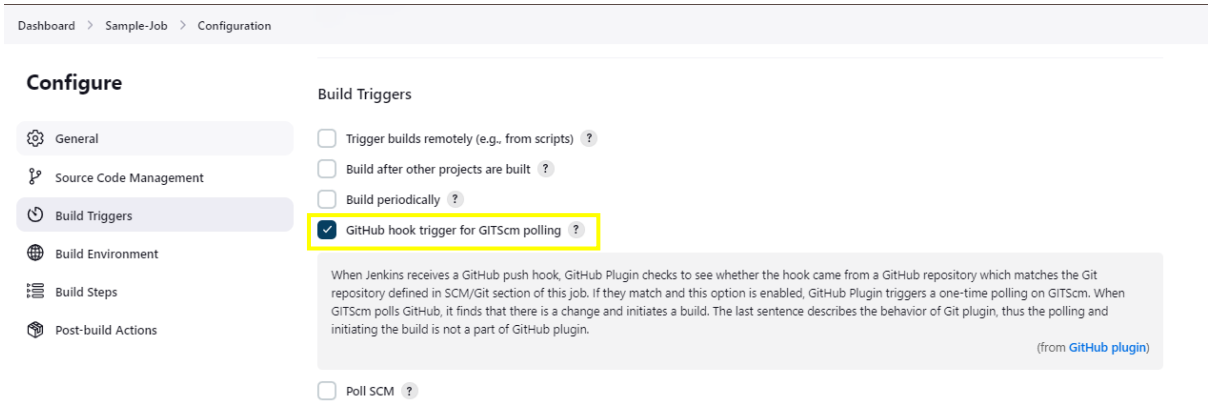
73. Come to Git-Hub and Copy the Website Repo Link, Keep it.



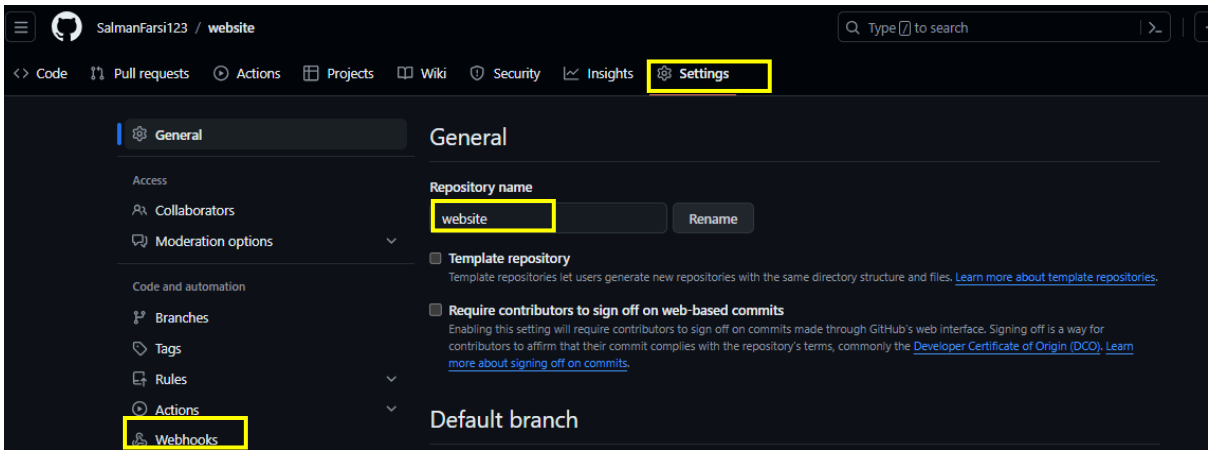
73. Paste the Git-Hub Repo link



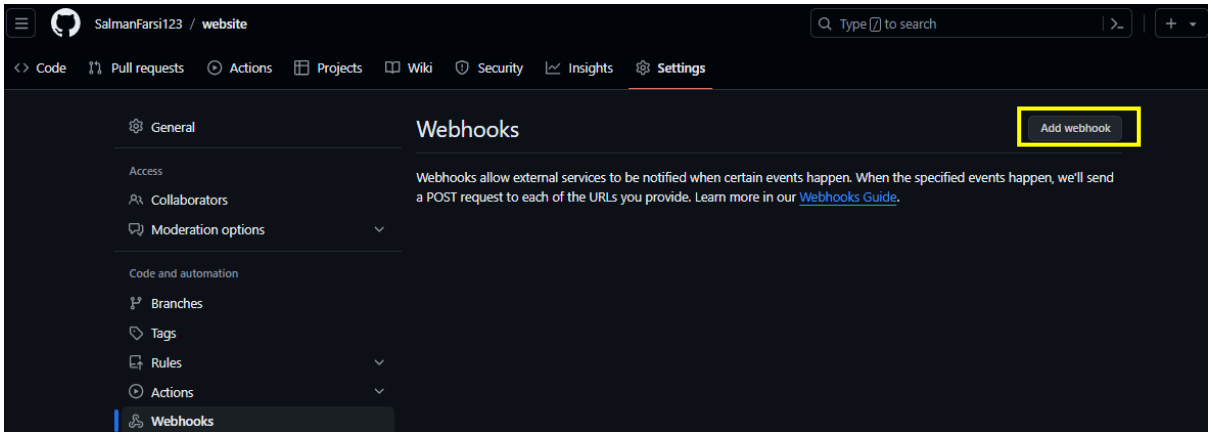
74. Enable Git-Hub Trigger for Automate Job Build for that go to github



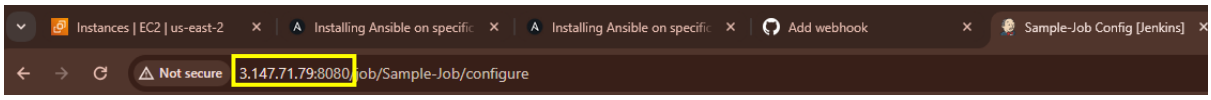
75. Click on Settings > Below the Left > Webhooks



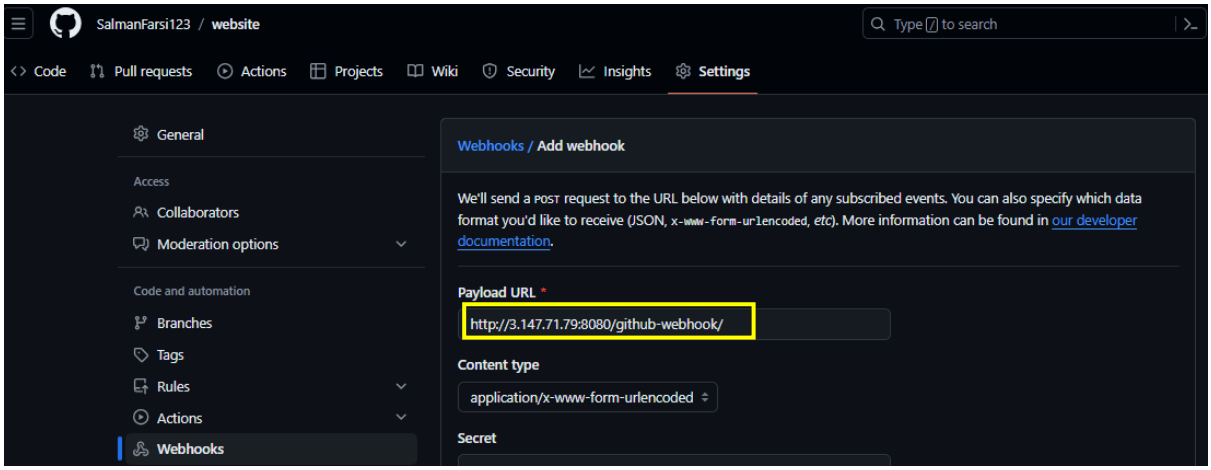
76. Click Add Webhook and right corner



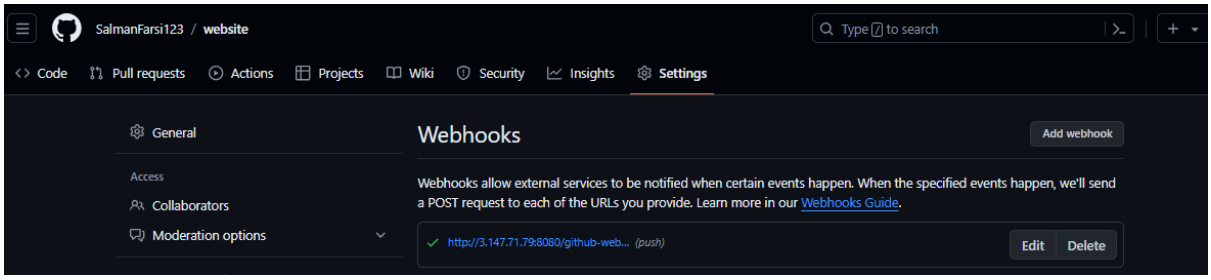
77. Copy the Jenkins Public IP Address



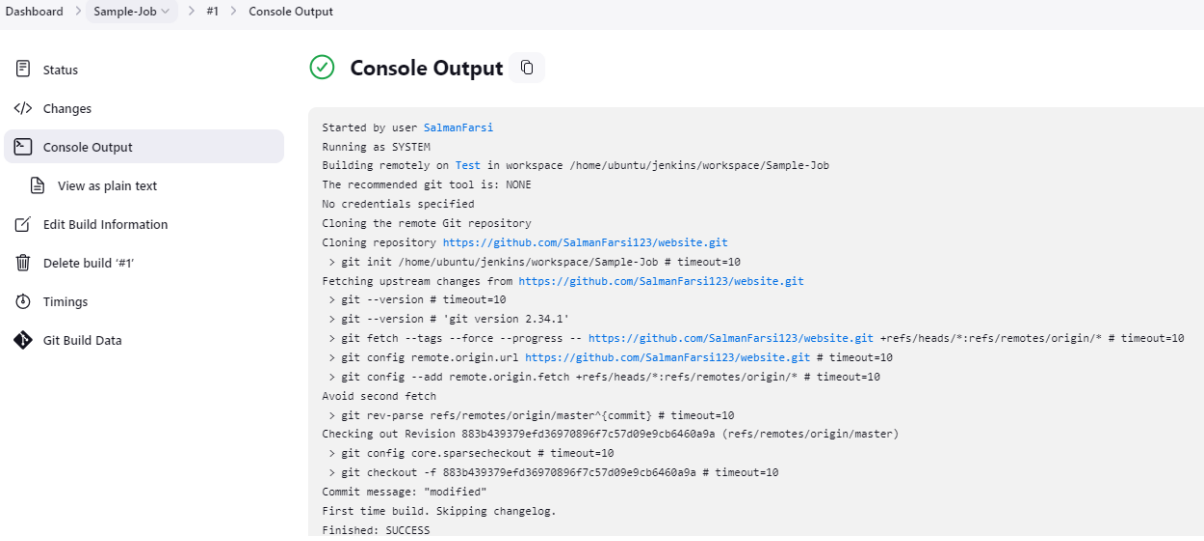
78. Paste it and Add /github/webhook/



79. Refresh it.



80. The Job Ran Automatically, and its success console output, Go to Test server



81. See the files stored structure

```
ubuntu@Project-Master: ~   ubuntu@Test-Server: ~/jenkir   ubuntu@Prod-Server: ~
ubuntu@Test-Server:~$ ls
jenkins
ubuntu@Test-Server:~$ cd jenkins
ubuntu@Test-Server:~/jenkins$ ls
remoting  remoting.jar  workspace
ubuntu@Test-Server:~/jenkins$ cd workspace
ubuntu@Test-Server:~/jenkins/workspace$ ls
Sample-Job
ubuntu@Test-Server:~/jenkins/workspace$ cd Sample-Job
ubuntu@Test-Server:~/jenkins/workspace/Sample-Job$ ks
ks: command not found
ubuntu@Test-Server:~/jenkins/workspace/Sample-Job$ ls
images  index.html
ubuntu@Test-Server:~/jenkins/workspace/Sample-Job$
```

82. Now, I am running in Built-In(Jenkins-server/Master-server)

Dashboard > Sample-Job > Configuration

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

This project is parameterized ?

Throttle builds ?

Execute concurrent builds if necessary ?

Restrict where this project can be run ?

Label Expression ?

built-in

Label built-in matches 1 node. Permissions or other restrictions provided by plugins may further reduce that list.

Advanced

83. Build Now and its success and Copy the location.

Jenkins Search (CTRL+K) SalmanFarsi log out

Dashboard > Sample-Job > #2 > Console Output

- Status
- Changes
- Console Output
- View as plain text
- Edit Build Information
- Delete build '#2'
- Timings
- Git Build Data
- Previous Build

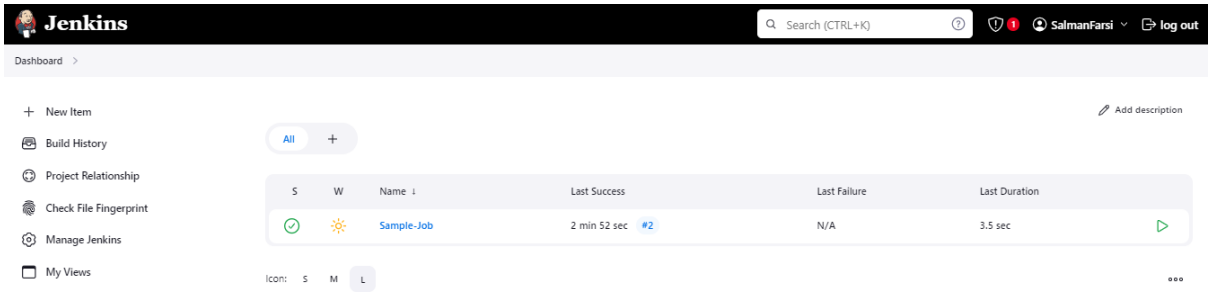
Console Output

```
Started by user SalmanFarsi
Running as SYSTEM
Building on the built-in node in workspace /var/lib/jenkins/workspace/Sample-Job
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/SalmanFarsi123/website.git
 > git init /var/lib/jenkins/workspace/Sample-Job # timeout=10
Fetching upstream changes from https://github.com/SalmanFarsi123/website.git
 > git --version # timeout=10
 > git --version # 'git version 2.34.1'
 > git fetch --tags --force --progress -- https://github.com/SalmanFarsi123/website.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://github.com/SalmanFarsi123/website.git # timeout=10
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision 883b439379efd36970896f7c57d09e9cb6460a9a (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 883b439379efd36970896f7c57d09e9cb6460a9a # timeout=10
Commit message: "modified"
 > git rev-list --no-walk 883b439379efd36970896f7c57d09e9cb6460a9a # timeout=10
Finished: SUCCESS
```

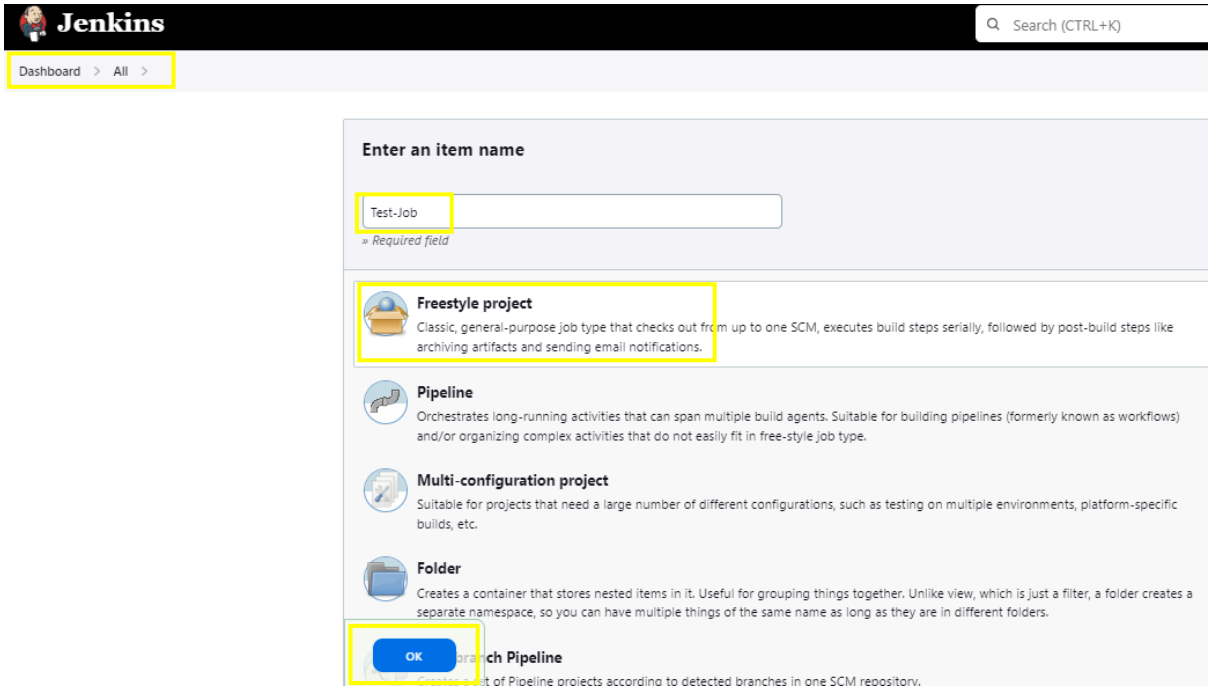
84. Paste it and we can the see the files in Master server

```
ubuntu@Project-Master: /var,   ubuntu@Test-Server: ~/jenkin:   ubuntu@Prod-Server: ~
ubuntu@Project-Master:~$ cd /var/lib/jenkins/workspace/Sample-Job
ubuntu@Project-Master:/var/lib/jenkins/workspace/Sample-Job$ ls
images  index.html
ubuntu@Project-Master:/var/lib/jenkins/workspace/Sample-Job$
```

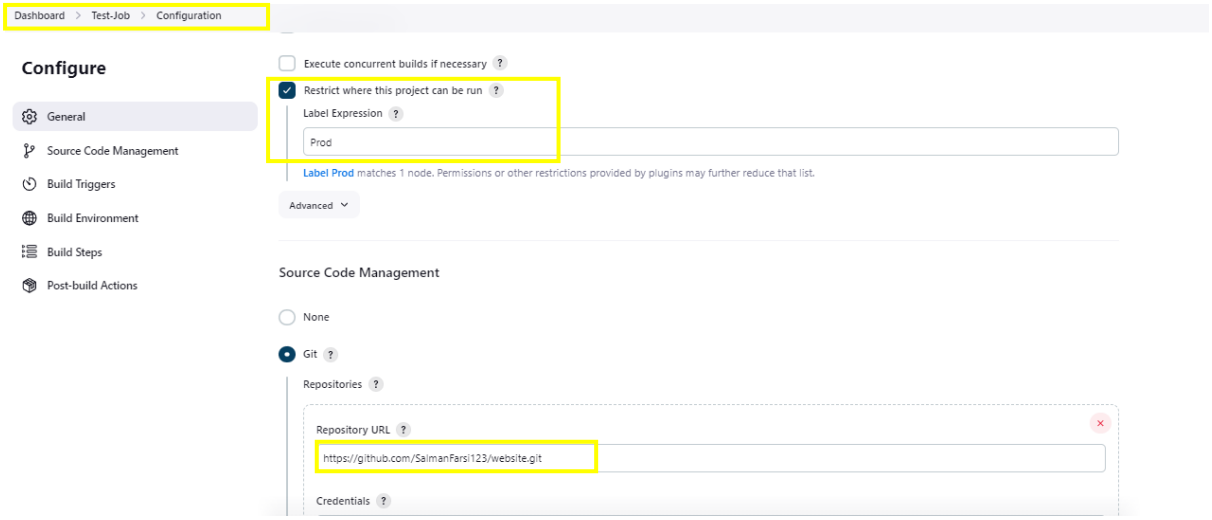
85. Now Creating Jobs for Actually Assignment Task



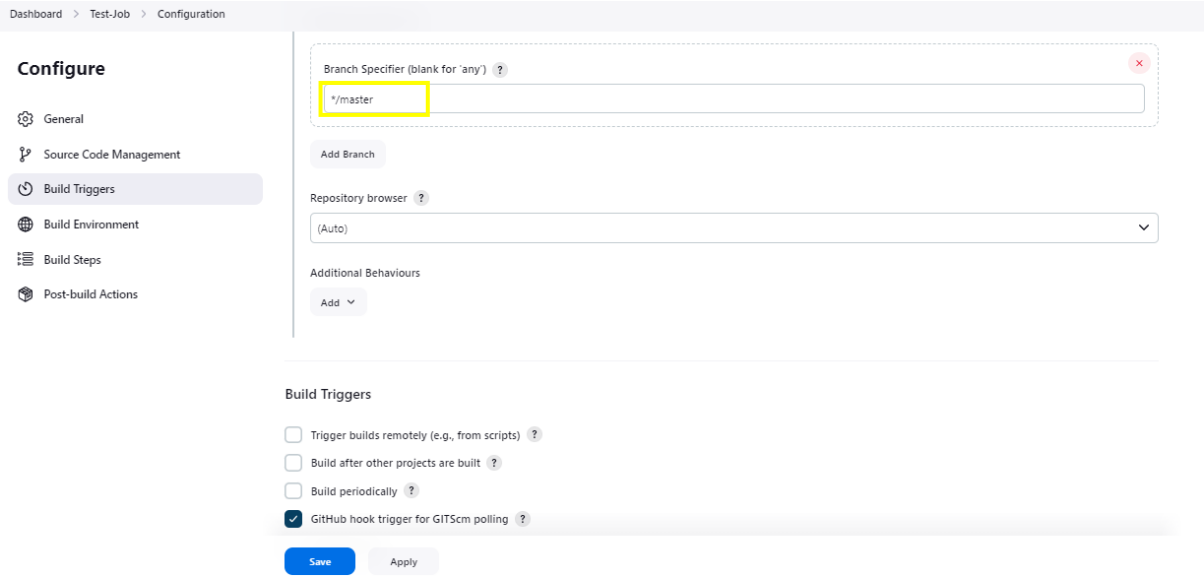
86. Naming as a Test-Job > Freestyle project > ok



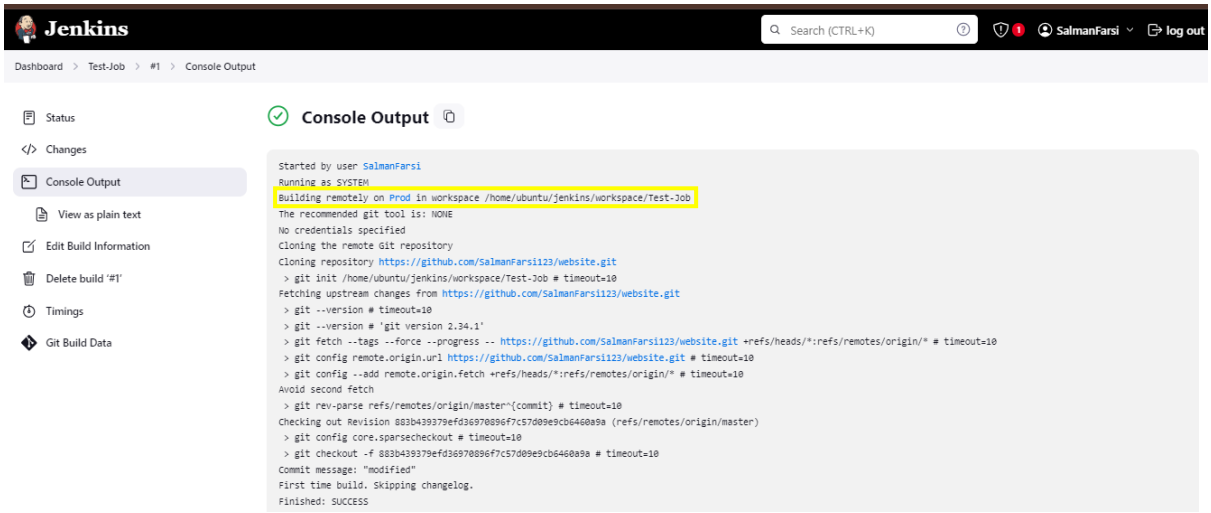
87. Selecting Prod server and Same Repo



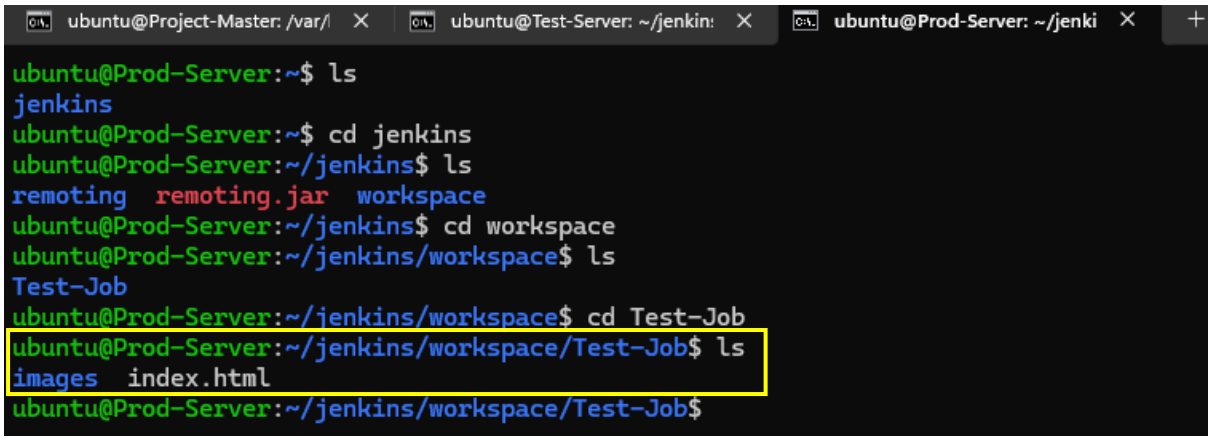
88. Branch:master and using github hook trigger



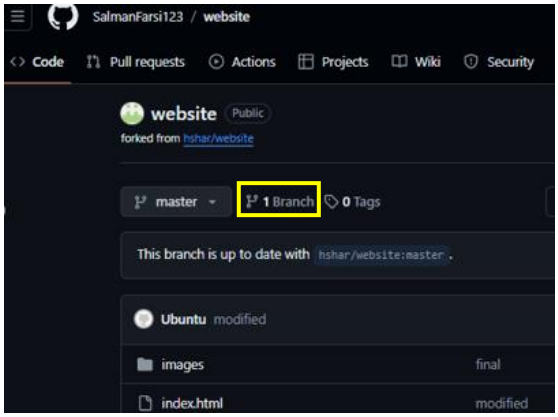
89. Click on Build Now and its ran successfully at Prod-Server



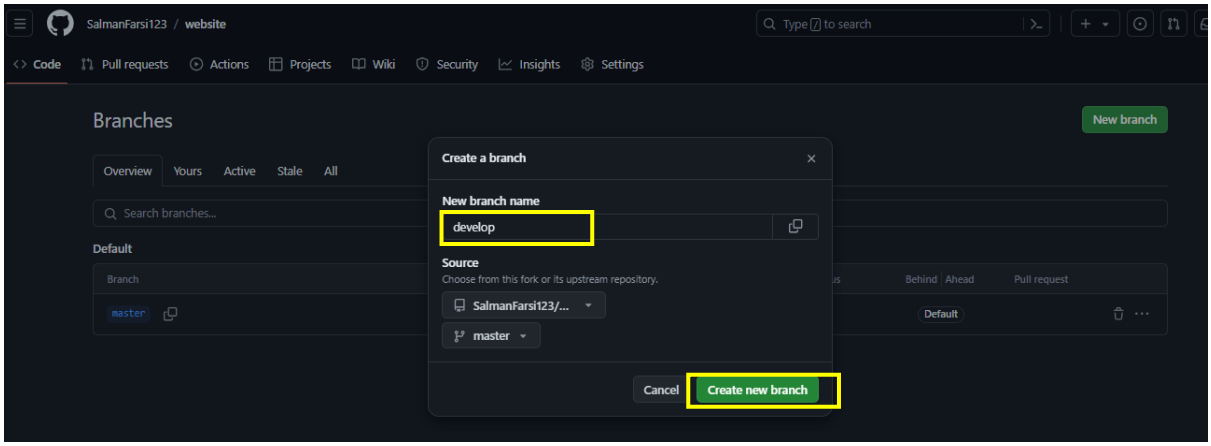
90. Executed and stored the data at given path



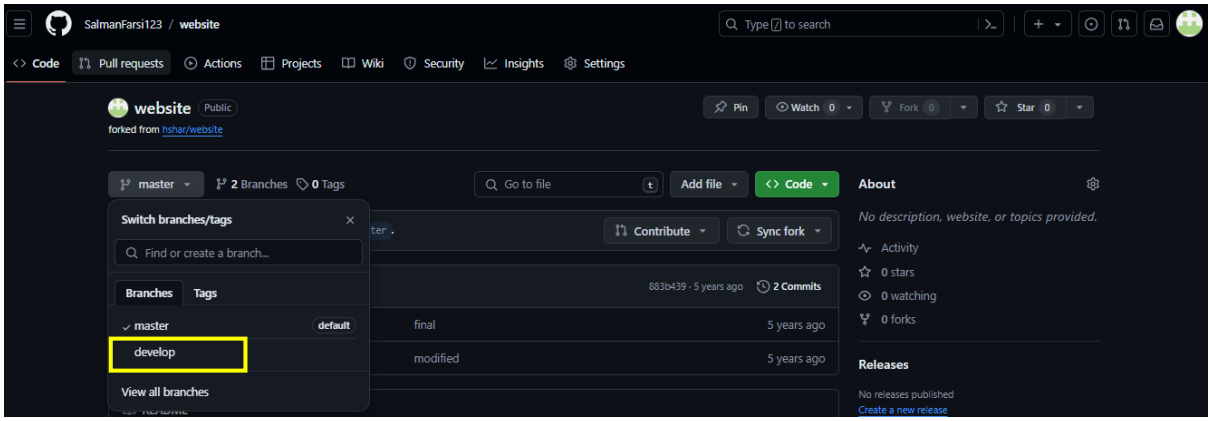
91. Click on Branch



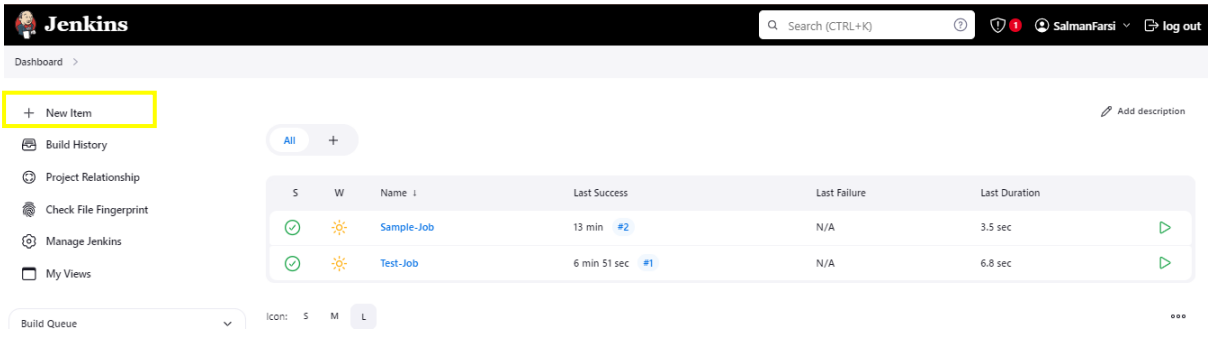
92. Create a new branch Named as develop



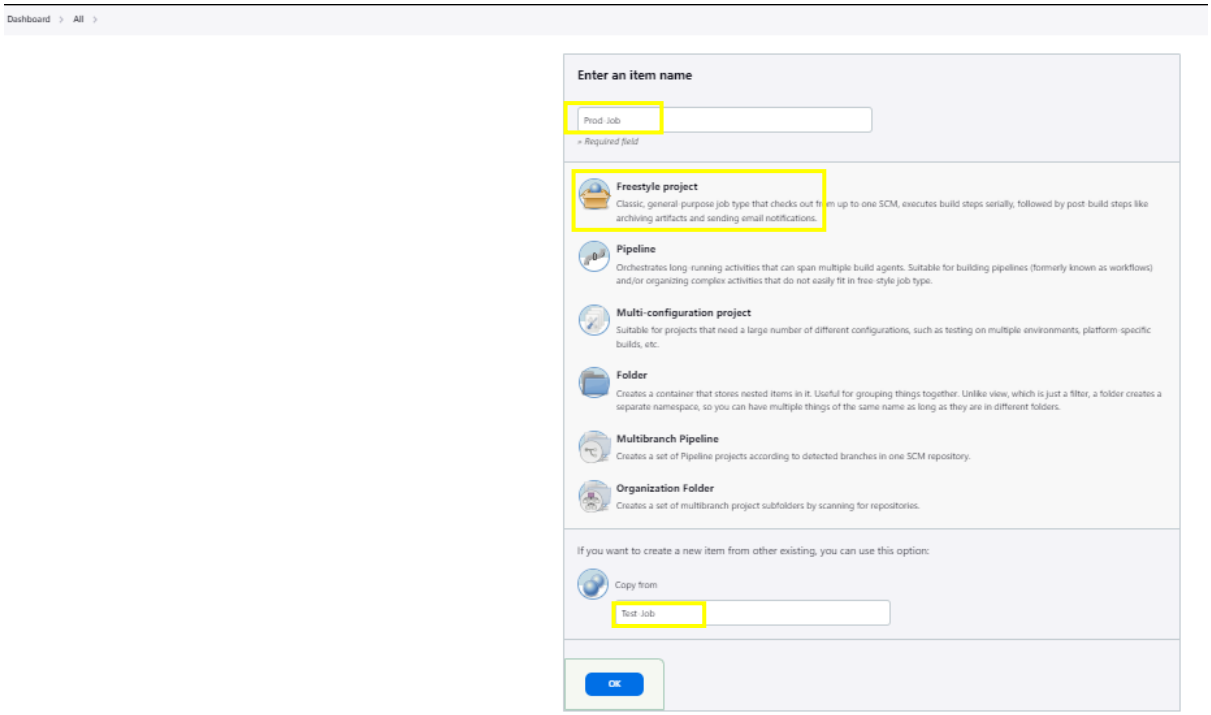
93. Now its available



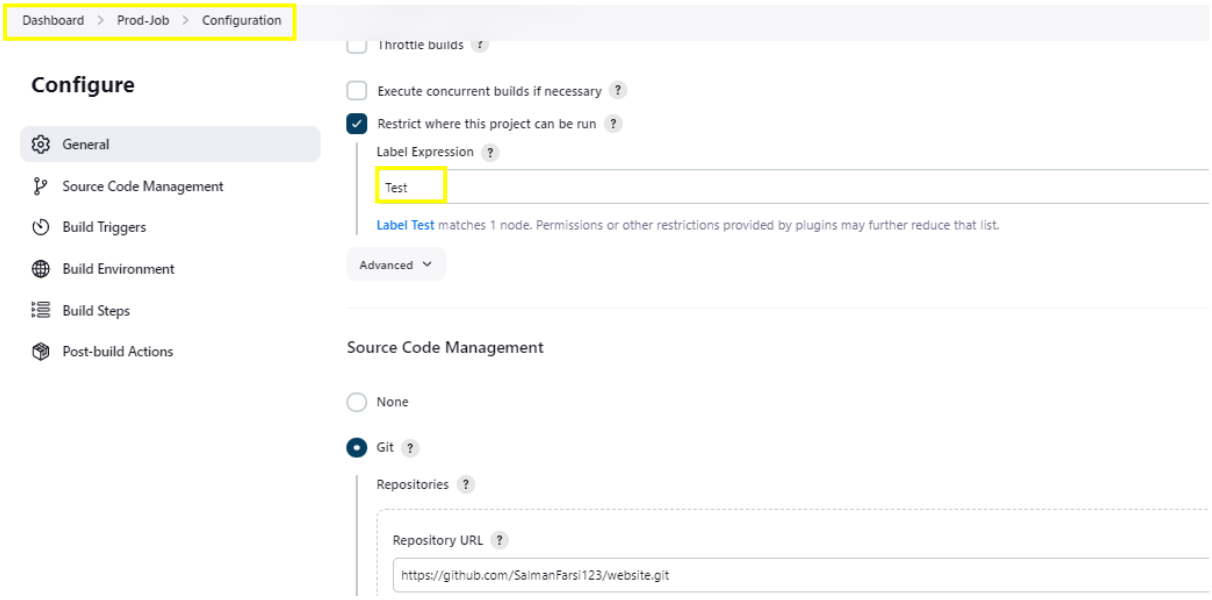
94. Now, Creating a Prod-Job



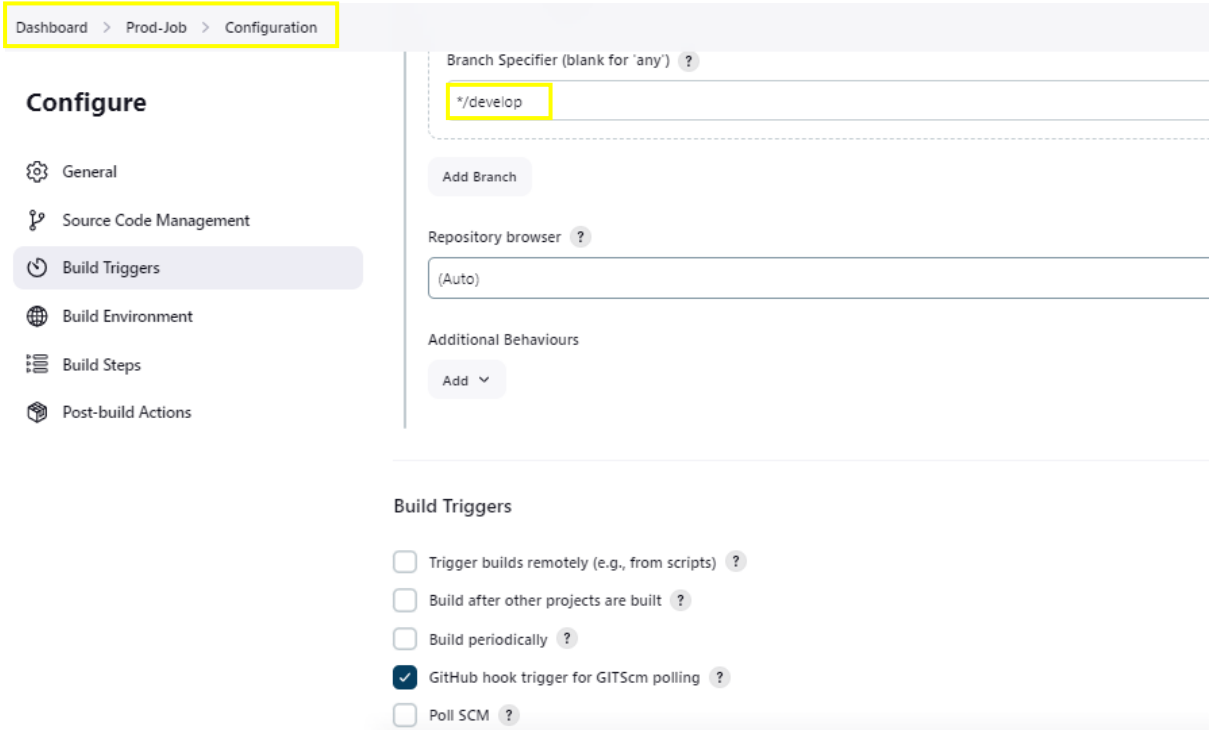
95. Prod-Job and copying from Test-Job, some what we modifying it.



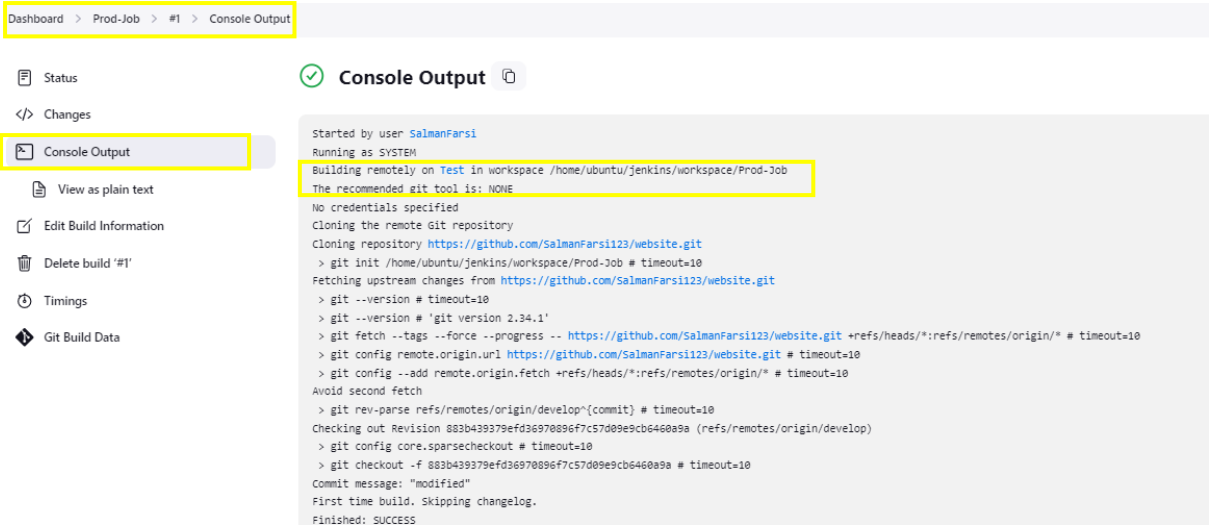
96. Changing into Test Server(Data-Storing)



97. Branch name is develop, that's it and otherwise there is no changing



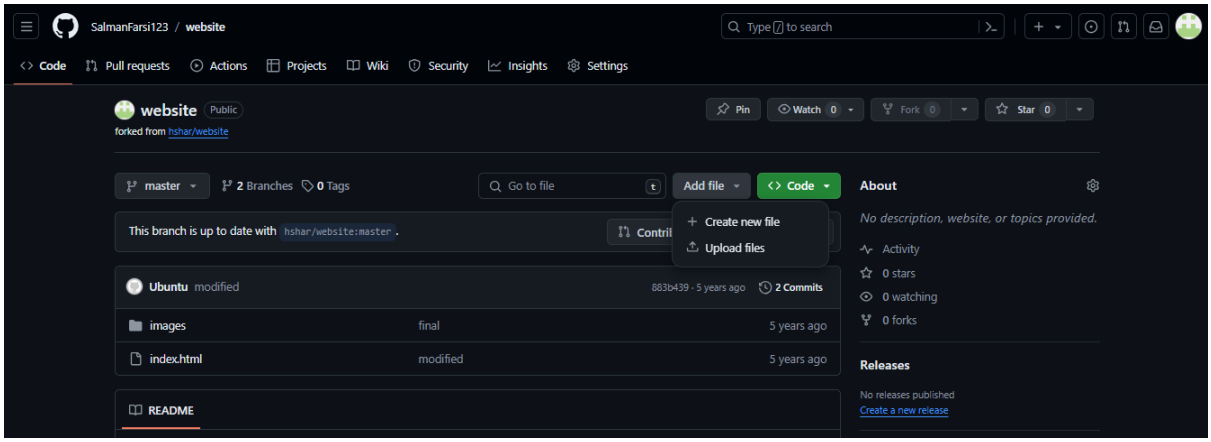
98. Build Now its executed and stored successfully.



99. we can see the files

```
ubuntu@Test-Server:~$ ls
jenkins
ubuntu@Test-Server:~$ cd jenkins
ubuntu@Test-Server:~/jenkins$ ls
remoting  remoting.jar  workspace
ubuntu@Test-Server:~/jenkins$ cd workspace
ubuntu@Test-Server:~/jenkins/workspace$ ls
Prod-Job  Sample-Job
ubuntu@Test-Server:~/jenkins/workspace$ cd Prod-Job
ubuntu@Test-Server:~/jenkins/workspace/Prod-Job$ ls
images  index.html
ubuntu@Test-Server:~/jenkins/workspace/Prod-Job$
```

100. Now we are going to create Dockerfile Click > Add file > Create new file



101. type the commands

FROM ubuntu

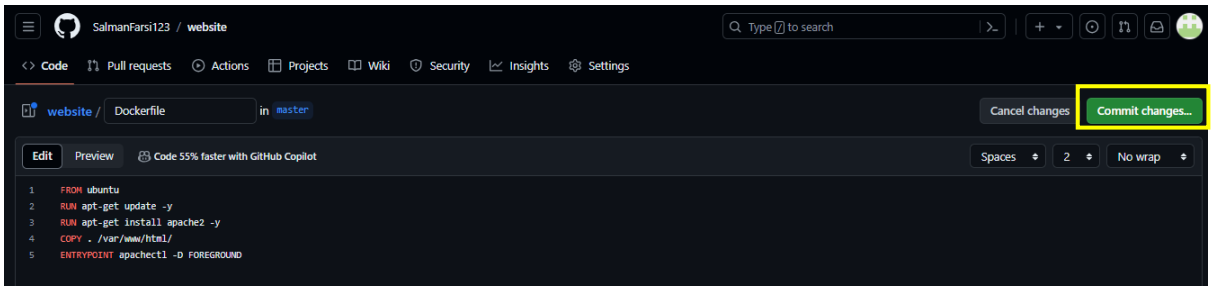
RUN apt-get update -y

RUN apt-get install apache2 -y

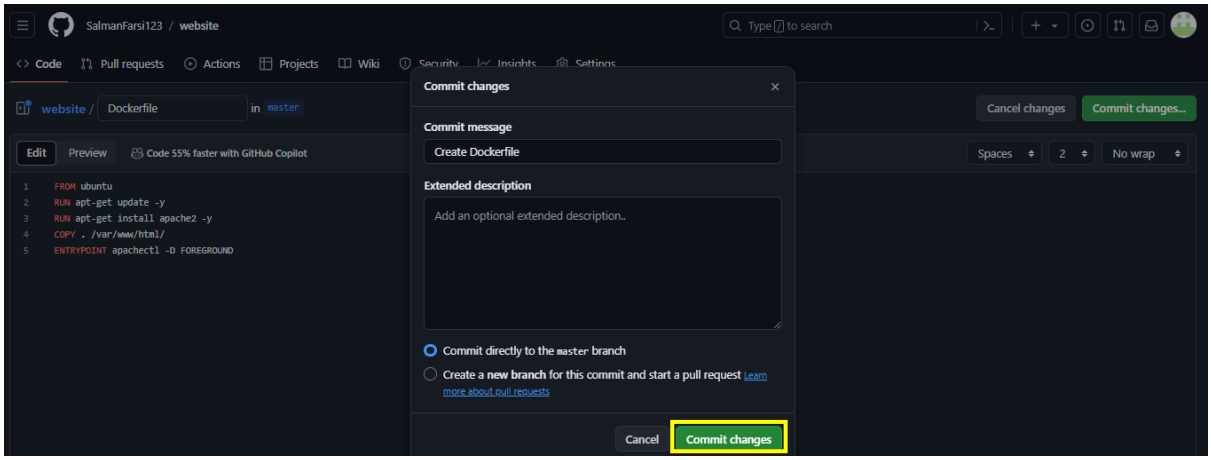
COPY . /var/www/html/

ENTRYPOINT apche2ctl -D FOREGROUND

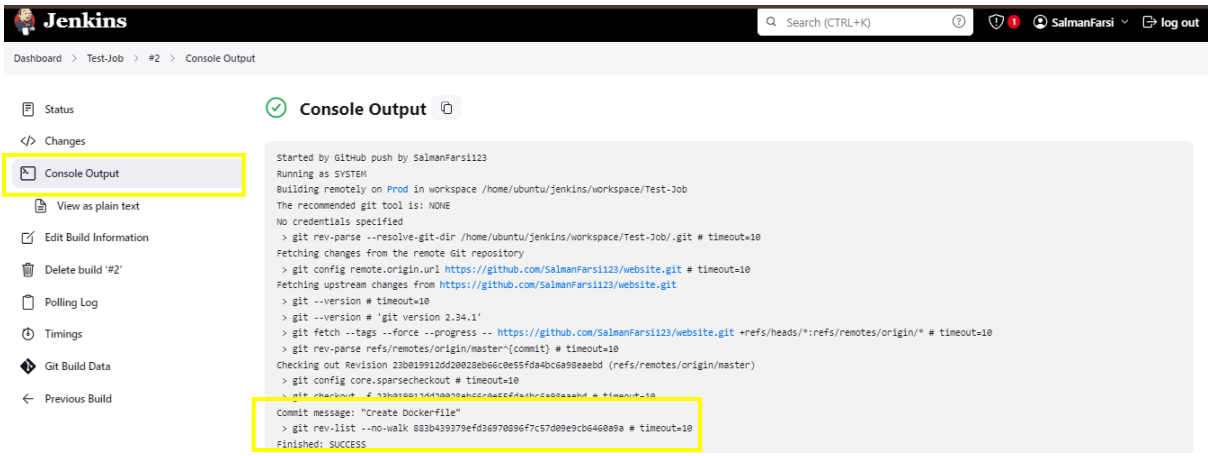
This Dockerfile sets up an Ubuntu image, installs Apache, copies current directory content to Apache's web directory, and starts Apache server as the entrypoint.



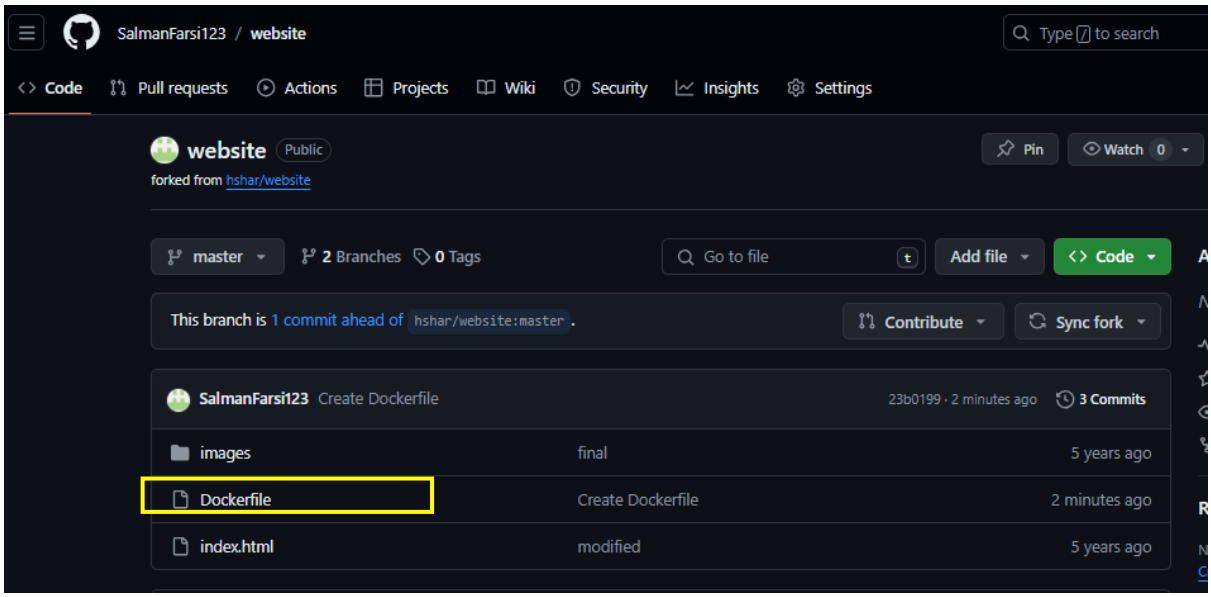
102. Commit changes



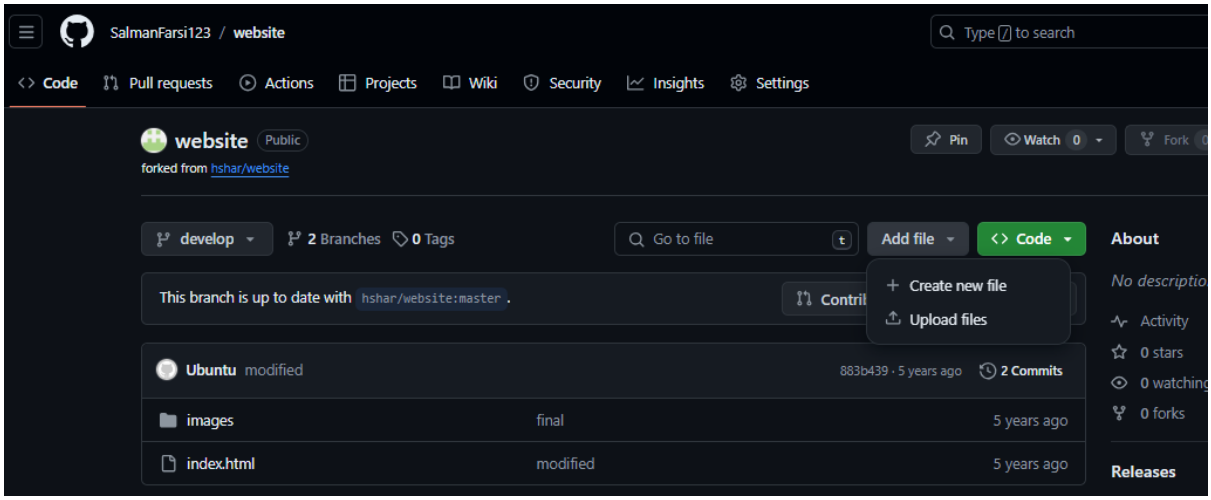
103. Automatically Building the job



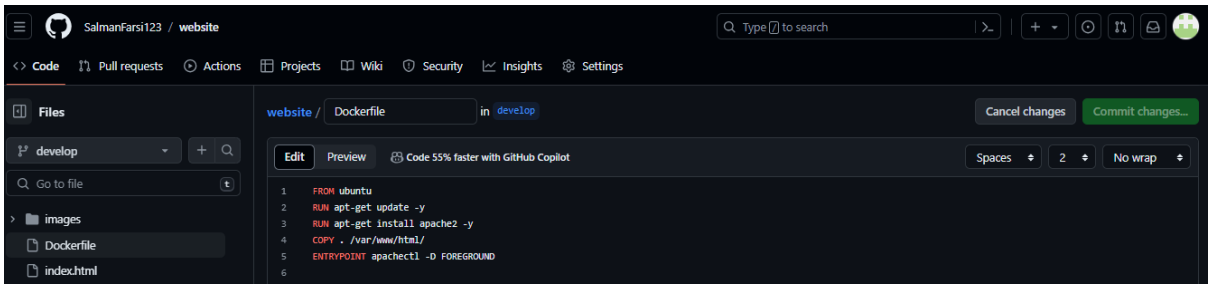
104. We can see the dockerfile in the master branch, we created in master branch



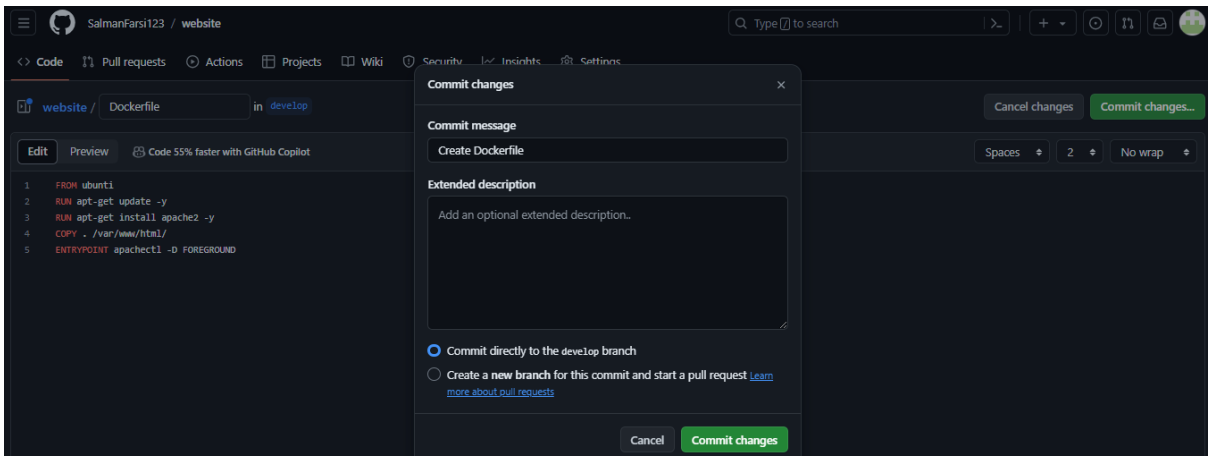
105. Same Creating dockerfile in develop branch



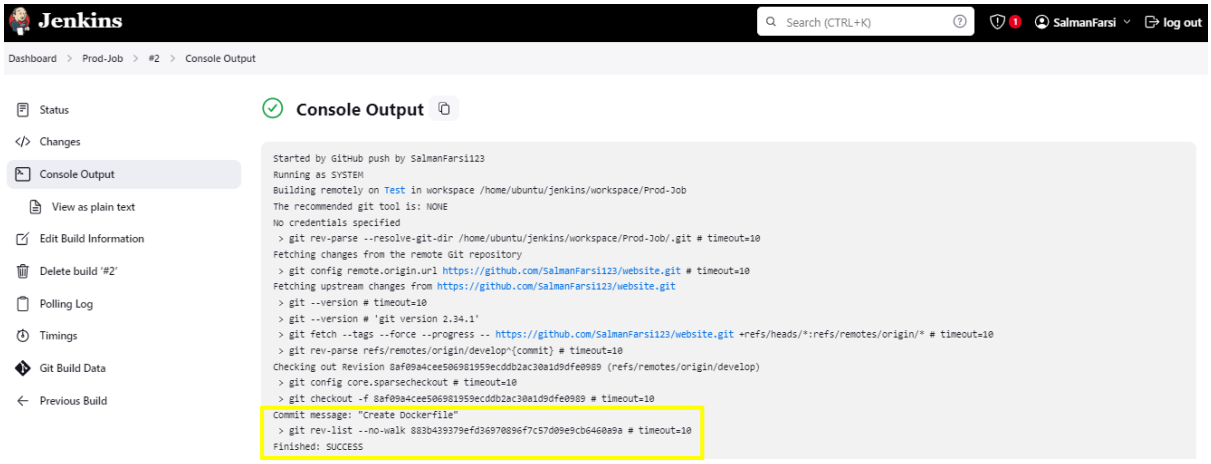
106. type the commands same as in stored in master branch



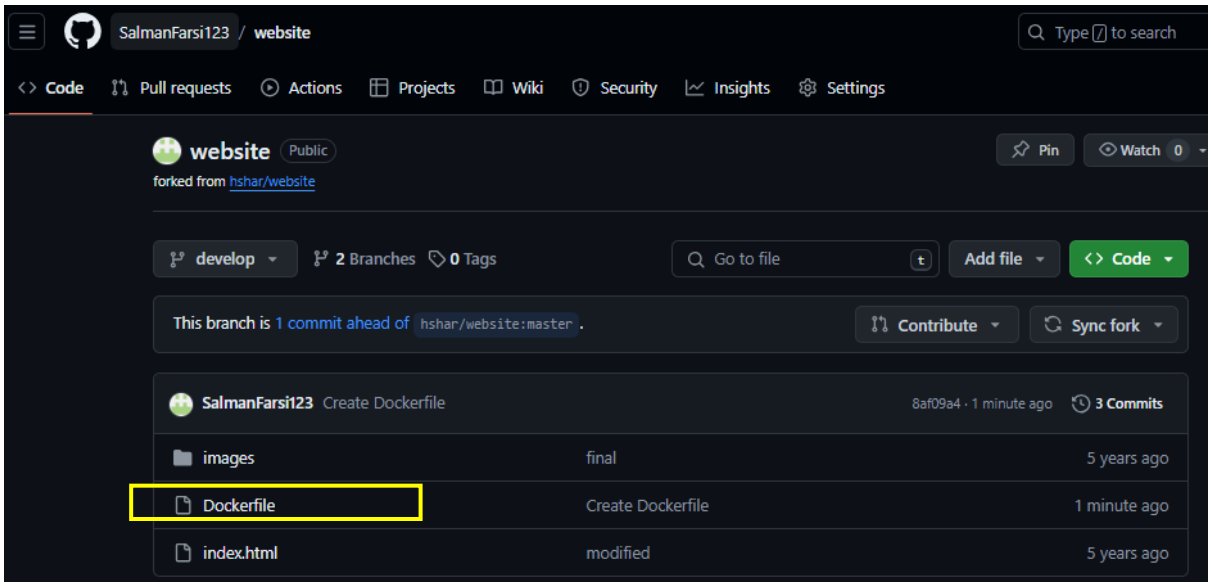
107. commit changes



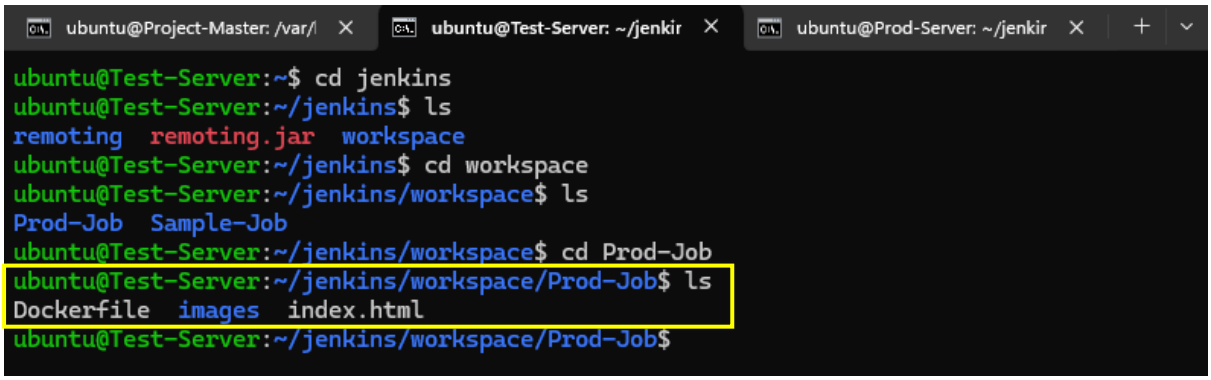
108. Automated and Job also successfully ran



109. We can see the dockerfile here also



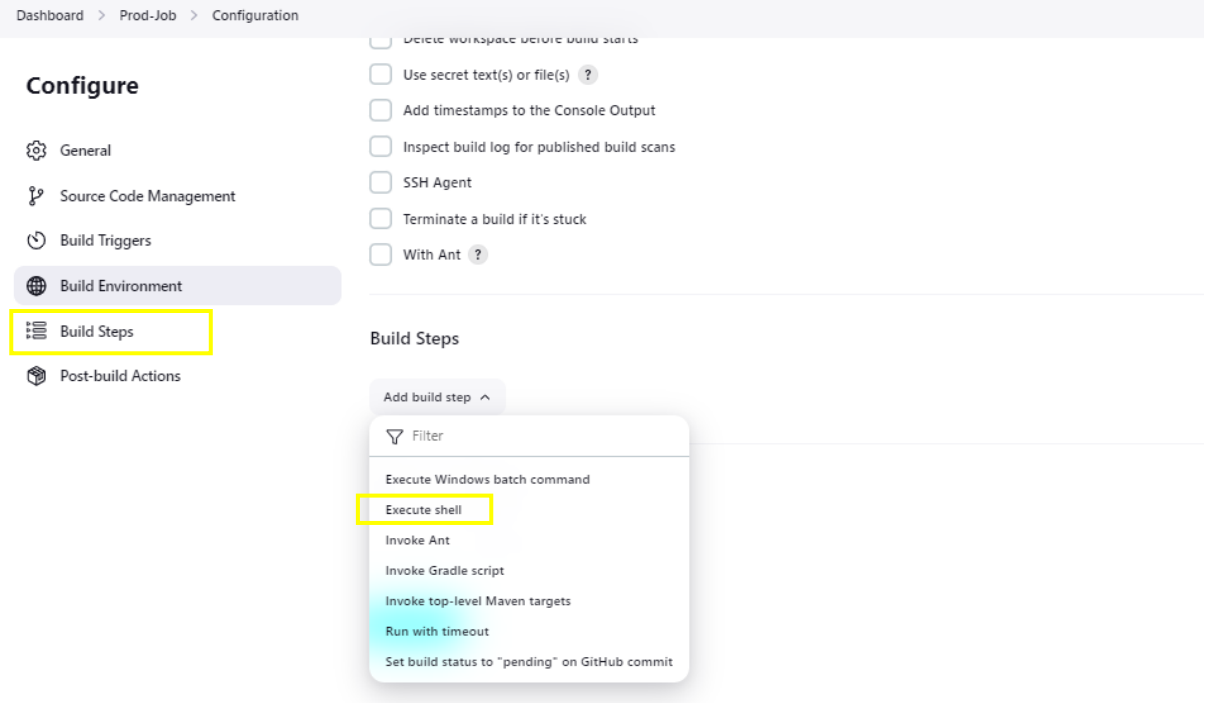
110. We can see the Dockerfile in Test-Server



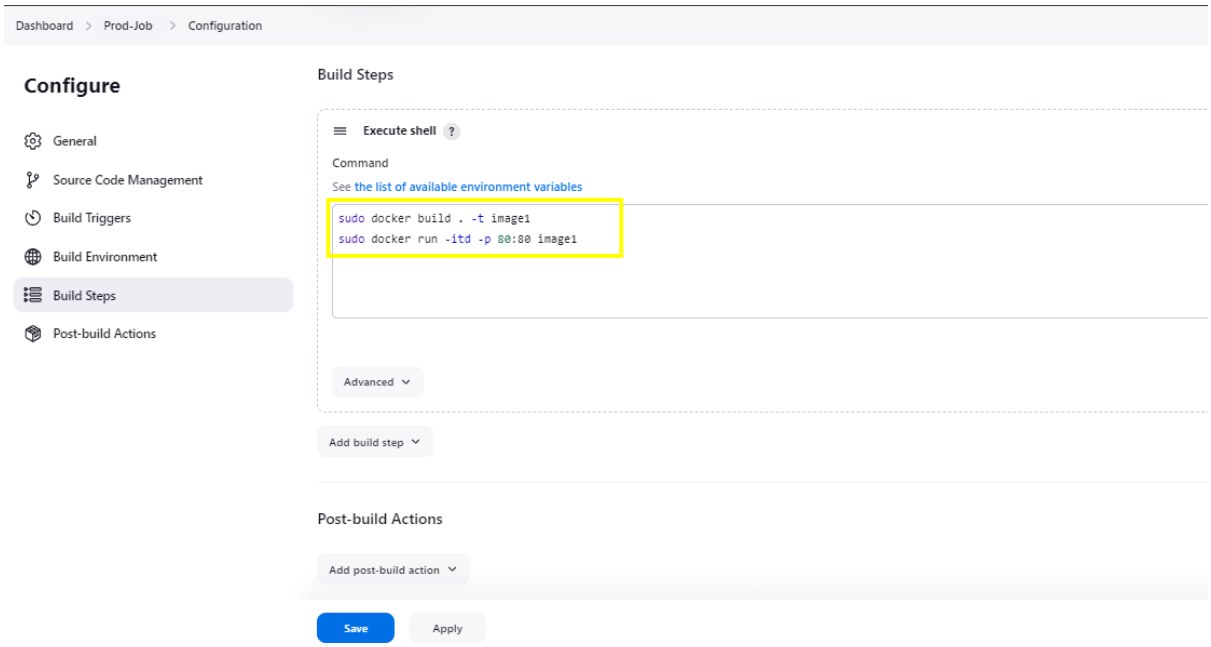
111. We can see the Dockerfile in Prod-Server

```
ubuntu@Prod-Server:~$ ls
jenkins
ubuntu@Prod-Server:~$ cd jenkins
ubuntu@Prod-Server:~/jenkins$ ls
remoting  remoting.jar  workspace
ubuntu@Prod-Server:~/jenkins$ cd workspace
ubuntu@Prod-Server:~/jenkins/workspace$ ls
Test-Job
ubuntu@Prod-Server:~/jenkins/workspace$ cd Test-Job
ubuntu@Prod-Server:~/jenkins/workspace/Test-Job$ ls
Dockerfile  images  index.html
ubuntu@Prod-Server:~/jenkins/workspace/Test-Job$
```

112. Go to Build Steps > Execute Shell



113. user was ubuntu and data and containers are storing and creating inside the ubuntu machines and building the dockerfile and named as image1



114. its ran successfully, 5 steps

This Dockerfile sets up an

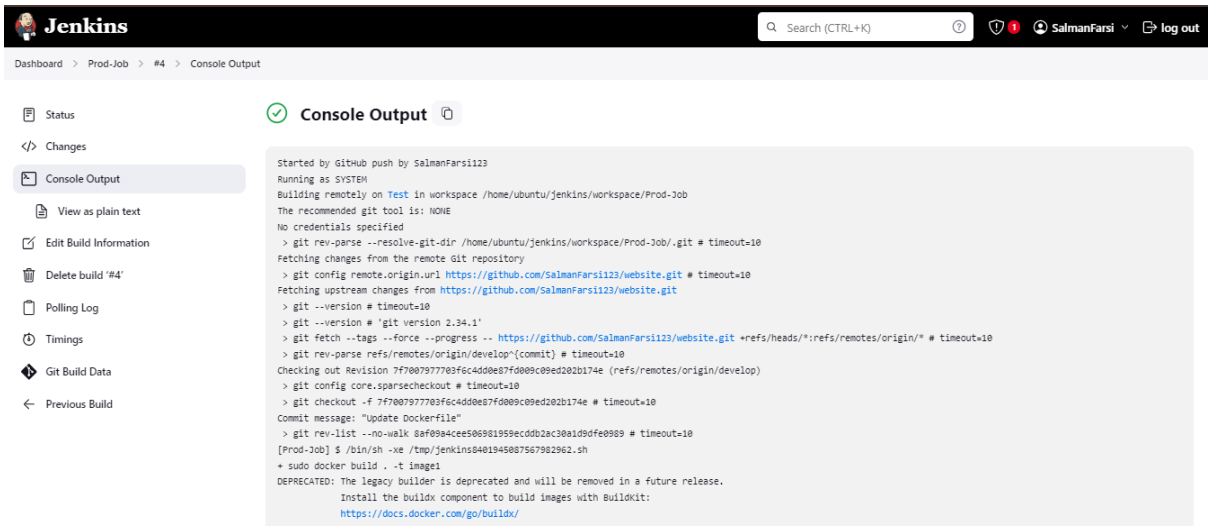
Step1: Ubuntu image,

Step2: Updating packages

Step3: installs Apache,

Step4: copies current directory content to Apache's web directory,

Step5: And starts Apache server as the entrypoint.



Dashboard > Prod-Job > #4 > Console Output

```
Sending build context to Docker daemon  274.4kB

Step 1/5 : FROM ubuntu
latest: Pulling from library/ubuntu
00d679a470c4: Pulling fs layer
00d679a470c4: Verifying Checksum
00d679a470c4: Download complete
00d679a470c4: Pull complete
Digest: sha256:e3f92abc0967a6c19d0dfa2d55838833e947b9d74edbcb0113e48535ad4be12a
Status: Downloaded newer image for ubuntu:latest
 ---> 17c0145030df
Step 2/5 : RUN apt-get update -y
 ---> Running in 9c1a1b486ec2
Get:1 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:2 http://archive.ubuntu.com/ubuntu noble InRelease [256 kB]
Get:3 http://archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [76.5 kB]
Get:5 http://archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:6 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [49.6 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [174 kB]
Get:8 http://archive.ubuntu.com/ubuntu noble/universe amd64 Packages [19.3 MB]
Get:9 http://archive.ubuntu.com/ubuntu noble/main amd64 Packages [1808 kB]
Get:10 http://archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [331 kB]
Get:11 http://archive.ubuntu.com/ubuntu noble/restricted amd64 Packages [117 kB]
Get:12 http://archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [84.8 kB]
Get:13 http://archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packages [3126 B]
Get:14 http://archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [75.0 kB]
Get:15 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [202 kB]
Get:16 http://archive.ubuntu.com/ubuntu noble-backports/universe amd64 Packages [7519 B]
Fetched 22.9 MB in 2s (10.7 MB/s)
Reading package lists...
Removing intermediate container 9c1a1b486ec2
 ---> 1ca8c01e2077
Step 3/5 : RUN apt-get install apache2 -y
 ---> Running in ddd08017a6bd
Reading package lists...
Building dependency tree...
Reading state information...
```

Dashboard > Prod-Job > #4 > Console Output

```
Reading state information...
The following additional packages will be installed:
  adduser apache2-bin apache2-data apache2-utils ca-certificates krb5-locales
  libapr1t64 libaprutil1-dbd-sqlite3 libaprutil1-ldap libaprutil1t64
  libbrotli1 libcurl4t64 libdb5.3t64 libexpat1 libgdbm-compat4t64 libgdbm6t64
  libgssapi-krb5-2 libicu74 libjansson4 libk5crypto3 libkeyutils1 libkrb5-3
  libkrb5support0 libldap-common libldap2 liblua5.4-0 libnghttp2-14
  libperl5.38t64 libpsl5t64 librtmp1 libsasl2-2 libsasl2-modules
  libsasl2-modules-db libsqlite3-0 libssh-4 libxml2 media-types netbase
  openssl perl perl-modules-5.38 publicsuffix ssl-cert
Suggested packages:
  liblocale-gettext-perl cron quota ecryptfs-utils apache2-doc
  apache2-suexec-pristine | apache2-suexec-custom www-browser ufw gdbm-l10n
  krb5-doc krb5-user libsasl2-modules-gssapi-mit
  | libsasl2-modules-gssapi-heimdal libsasl2-modules-ldap libsasl2-modules-otp
  libsasl2-modules-sql perl-doc libterm-readline-gnu-perl
  | libterm-readline-perl-perl make libtap-harness-archive-perl
The following NEW packages will be installed:
  adduser apache2 apache2-bin apache2-data apache2-utils ca-certificates
  krb5-locales libapr1t64 libaprutil1-dbd-sqlite3 libaprutil1-ldap
  libaprutil1t64 libbrotli1 libcurl4t64 libdb5.3t64 libexpat1
  libgdbm-compat4t64 libgdbm6t64 libgssapi-krb5-2 libicu74 libjansson4
  libk5crypto3 libkeyutils1 libkrb5-3 libkrb5support0 libldap-common libldap2
  liblua5.4-0 libnghttp2-14 libperl5.38t64 libpsl5t64 librtmp1 libsasl2-2
  libsasl2-modules libsasl2-modules-db libsqlite3-0 libssh-4 libxml2
  media-types netbase openssl perl perl-modules-5.38 publicsuffix ssl-cert
0 upgraded, 44 newly installed, 0 to remove and 6 not upgraded.
Need to get 27.0 MB of archives.
After this operation, 111 MB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu noble/main amd64 perl-modules-5.38 all 5.38.2-3.2build2 [3110 kB]
Get:2 http://archive.ubuntu.com/ubuntu noble/main amd64 libdb5.3t64 amd64 5.3.28+dfsg2-7 [734 kB]
Get:3 http://archive.ubuntu.com/ubuntu noble/main amd64 libgdbm6t64 amd64 1.23-5.1build1 [34.4 kB]
Get:4 http://archive.ubuntu.com/ubuntu noble/main amd64 libgdbm-compat4t64 amd64 1.23-5.1build1 [6710 B]
Get:5 http://archive.ubuntu.com/ubuntu noble/main amd64 libperl5.38t64 amd64 5.38.2-3.2build2 [4873 kB]
Get:6 http://archive.ubuntu.com/ubuntu noble/main amd64 perl amd64 5.38.2-3.2build2 [231 kB]
Get:7 http://archive.ubuntu.com/ubuntu noble/main amd64 libapr1t64 amd64 1.7.2-3.1build2 [107 kB]
Get:8 http://archive.ubuntu.com/ubuntu noble/main amd64 libexpat1 amd64 2.6.1-2build1 [87.0 kB]
Get:9 http://archive.ubuntu.com/ubuntu noble/main amd64 libaprutil1t64 amd64 1.6.3-1.1ubuntu7 [91.9 kB]
```

Dashboard > Prod-Job > #4 > Console Output

```
Enabling conf localized-error-pages.
Enabling conf other-vhosts-access-log.
Enabling conf security.
Enabling conf serve-cgi-bin.
Enabling site 000-default.
invoke-rc.d: could not determine current runlevel
invoke-rc.d: policy-rc.d denied execution of start.
Processing triggers for libc-bin (2.39-0ubuntu8.2) ...
Processing triggers for ca-certificates (20240203) ...
Updating certificates in /etc/ssl/certs...
0 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...
done.
Removing intermediate container ddd08017a6bd
 ---> ce952e024b76
Step 4/5 : COPY . /var/www/html/
 ---> 23619267b1be
Step 5/5 : ENTRYPOINT apachectl -D FOREGROUND
 ---> Running in c65f743a6fc6
Removing intermediate container c65f743a6fc6
 ---> ee31ca1775fc
Successfully built ee31ca1775fc
Successfully tagged image1:latest
+ sudo docker run -itd -p 80:80 image1
db0e0ac007c5553766d187880504 2b9af621f6552d9f9d28b5a25fe79045e354
Finished: SUCCESS
```

115. We can see the docker name image1 and its ported to 80 to 80

```
ubuntu@Project-Master: /var/l   ubuntu@Test-Server: ~/jenkir   ubuntu@Prod-Server: ~/jenkir
ubuntu@Test-Server:~$ ls
jenkins
ubuntu@Test-Server:~$ cd jenkins
ubuntu@Test-Server:~/jenkins$ ls
remoting  remoting.jar  workspace
ubuntu@Test-Server:~/jenkins$ cd workspace
ubuntu@Test-Server:~/jenkins/workspace$ ks
ks: command not found
ubuntu@Test-Server:~/jenkins/workspace$ ls
Prod-Job  Sample-Job
ubuntu@Test-Server:~/jenkins/workspace$ cd Prod-Job
ubuntu@Test-Server:~/jenkins/workspace/Prod-Job$ ls
Dockerfile  images  index.html
ubuntu@Test-Server:~/jenkins/workspace/Prod-Job$ sudo docker images
REPOSITORY   TAG       IMAGE ID       CREATED          SIZE
image1       latest    ee31ca1775fc   7 minutes ago    222MB
ubuntu       latest    17c0145030df   2 weeks ago      76.2MB
ubuntu@Test-Server:~/jenkins/workspace/Prod-Job$ sudo docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED         STATUS         PORTS                                   NAMES
db0e0ac007c5   image1    "/bin/sh -c 'apachec…"   7 minutes ago   Up 7 minutes   0.0.0.0:80->80/tcp, :::80->80/tcp   strange_newton
ubuntu@Test-Server:~/jenkins/workspace/Prod-Job$
```

116. Copy the Test-server public ip

Project-Master	i-0957f257a153efed2	Running	t2.micro	2/2 checks passed	View alarms +	us-east-2a
Test-Server	i-04b9d67f329c07539	Running	t2.micro	2/2 checks passed	View alarms +	us-east-2a
Prod-Server	i-0083d65942fb73468	Running	t2.micro	2/2 checks passed	View alarms +	us-east-2a

i-04b9d67f329c07539 (Test-Server)

Details | Status and alarms | Monitoring | Security | Networking | Storage | Tags

Instance summary Info

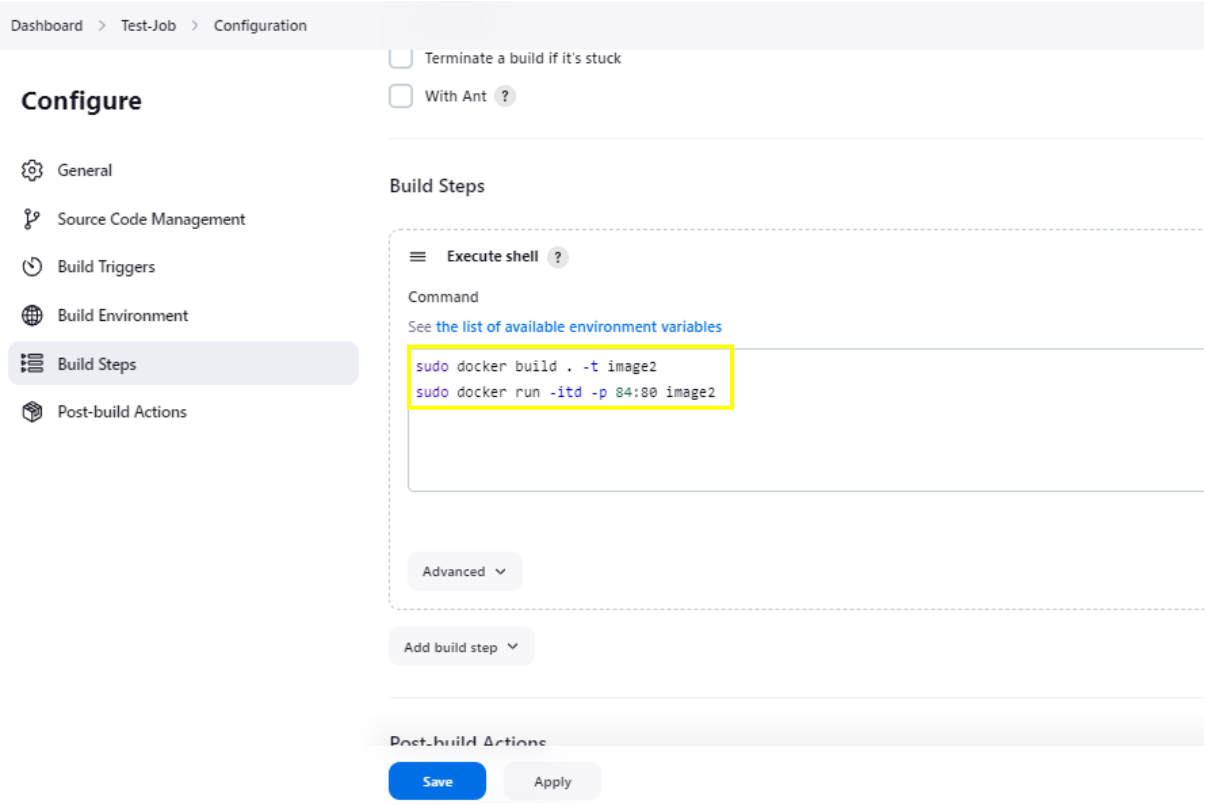
Instance ID
i-04b9d67f329c07539 (Test-Server)

Public IPv4 address
3.15.239.112 | open address

Private IPv4 addresses
172.31.11.191

117. The data of index.html file

118. Now same thing we are configuring in Test-Job and it will be run and store in Prod-server



119. we can see named as image2

```
ubuntu@Prod-Server:~/jenkins/workspace/Test-Job$ sudo docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
ubuntu@Prod-Server:~/jenkins/workspace/Test-Job$ sudo docker images
REPOSITORY   TAG       IMAGE ID       CREATED              SIZE
<none>       <none>    4db7a60aa206   About a minute ago   113MB
ubuntu       latest    17c0145030df   2 weeks ago          76.2MB
ubuntu@Prod-Server:~/jenkins/workspace/Test-Job$ sudo docker images
REPOSITORY   TAG       IMAGE ID       CREATED          SIZE
image2       latest    d52b4f34dc9e   15 seconds ago   222MB
ubuntu       latest    17c0145030df   2 weeks ago      76.2MB
ubuntu@Prod-Server:~/jenkins/workspace/Test-Job$ sudo docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS                                   NAMES
2b7b2bd88257   image2    "/bin/sh -c 'apachec…"   43 seconds ago   Up 42 seconds   0.0.0.0:84->80/tcp, :::84->80/tcp   blissful_darwin
ubuntu@Prod-Server:~/jenkins/workspace/Test-Job$
```

120. Copy the Prod Public-Ip

Name	Instance ID	Instance state	Instance type	Status check	Alarm status
Project-Master	i-0957f257a153efed2	Running	t2.micro	2/2 checks passed	View alarms +
Test-Server	i-04b9d67f329c07539	Running	t2.micro	2/2 checks passed	View alarms +
Prod-Server	i-0083d65942fb73468	Running	t2.micro	2/2 checks passed	View alarms +

i-0083d65942fb73468 (Prod-Server)

Instance ID
i-0083d65942fb73468 (Prod-Server)

Public IPv4 address
3.147.46.43 | open address

Private IPv4 addresses
172.31.5.221

121. We can See the page was run successfully

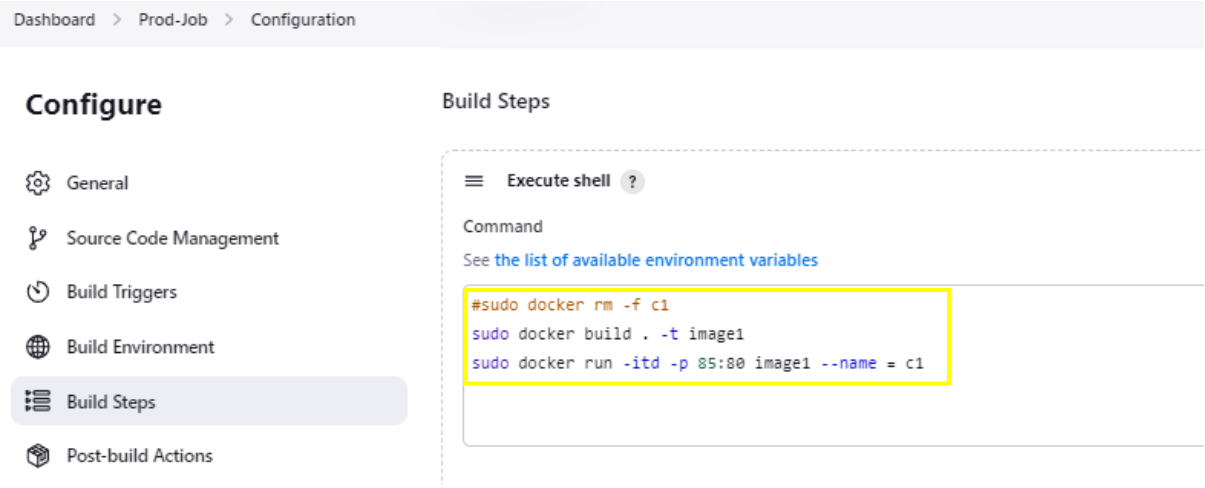
Not secure 3.147.46.43:84

Interfacing SIM900... Downloads New Tab Python Exercises: C... MEGA Women's Health: 25... Imported

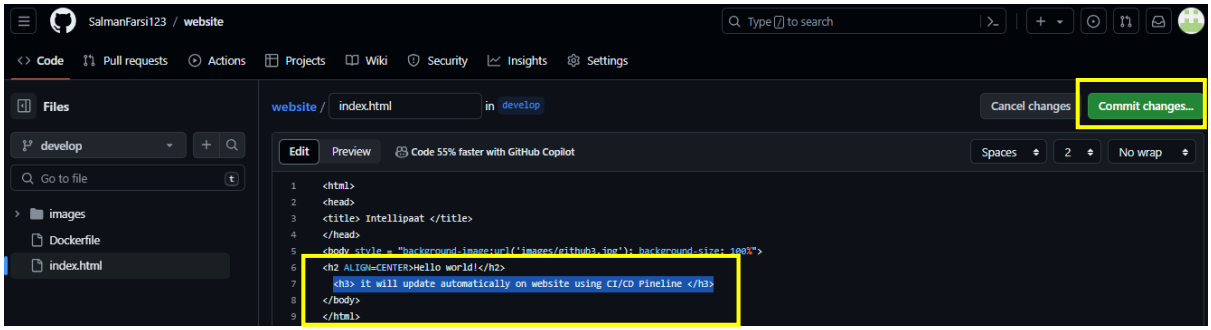
Hello world!

GitHub

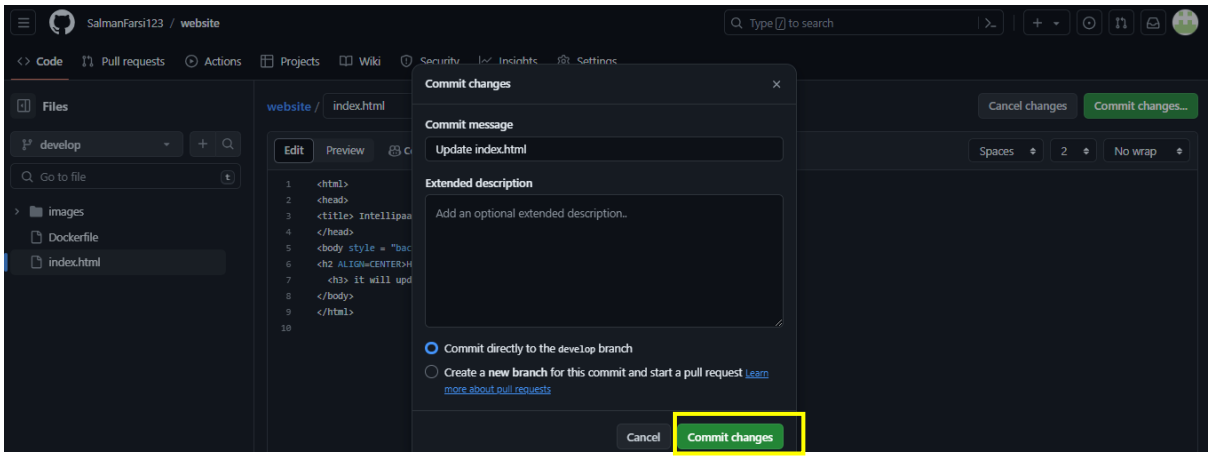
122. Now Once Again we are creating a image and named as c1 and porting to 85



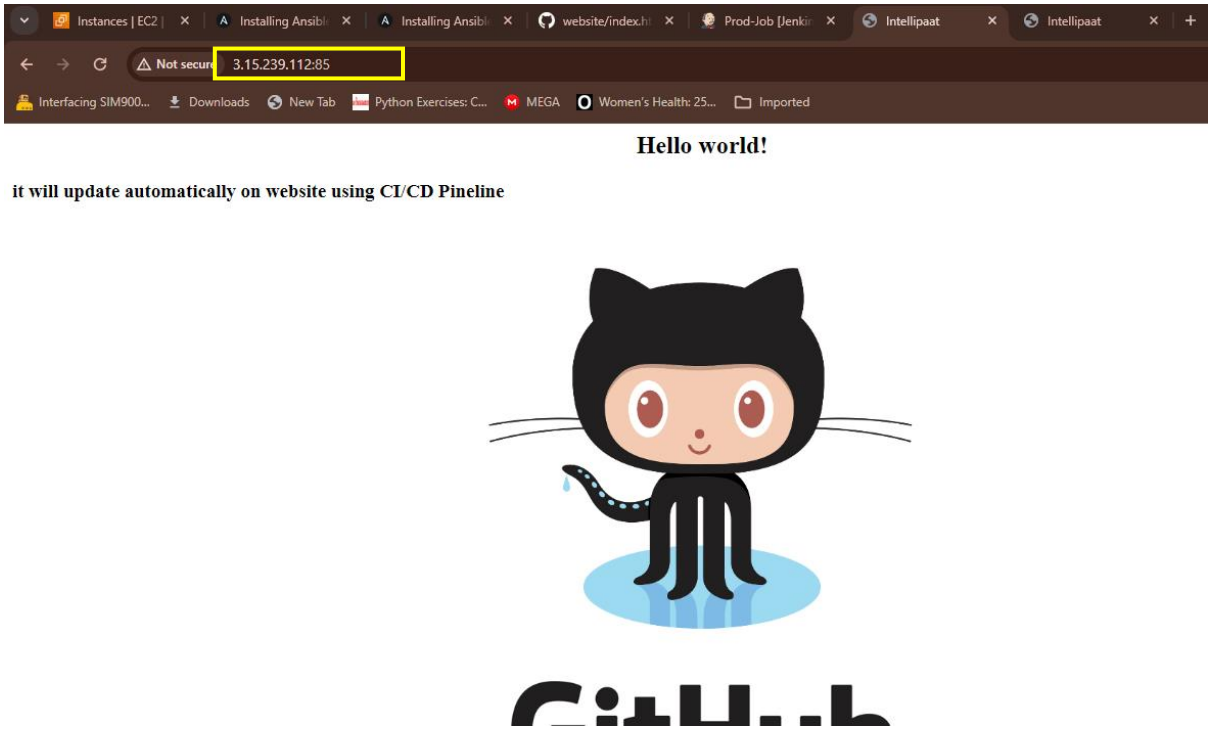
123. we are Configuring the dockerfile at develop branch



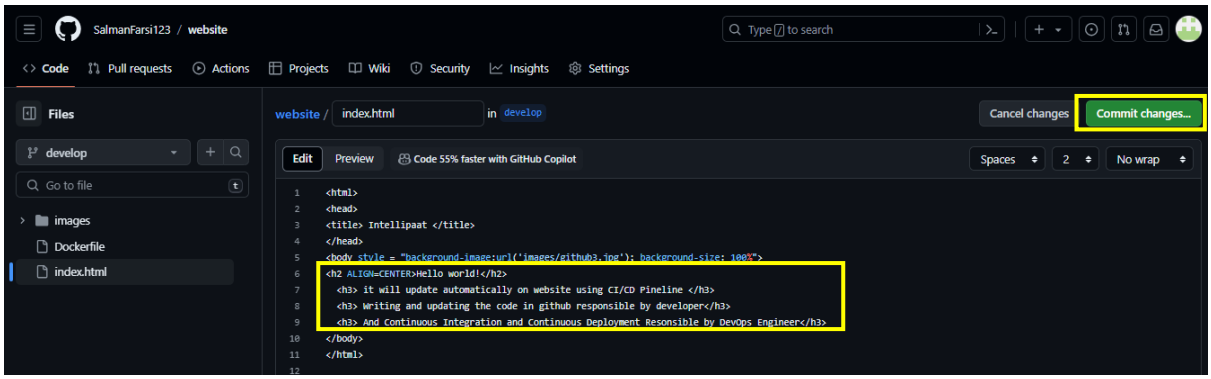
124. commit changes



125. Its working Successfully at port 85 > image.

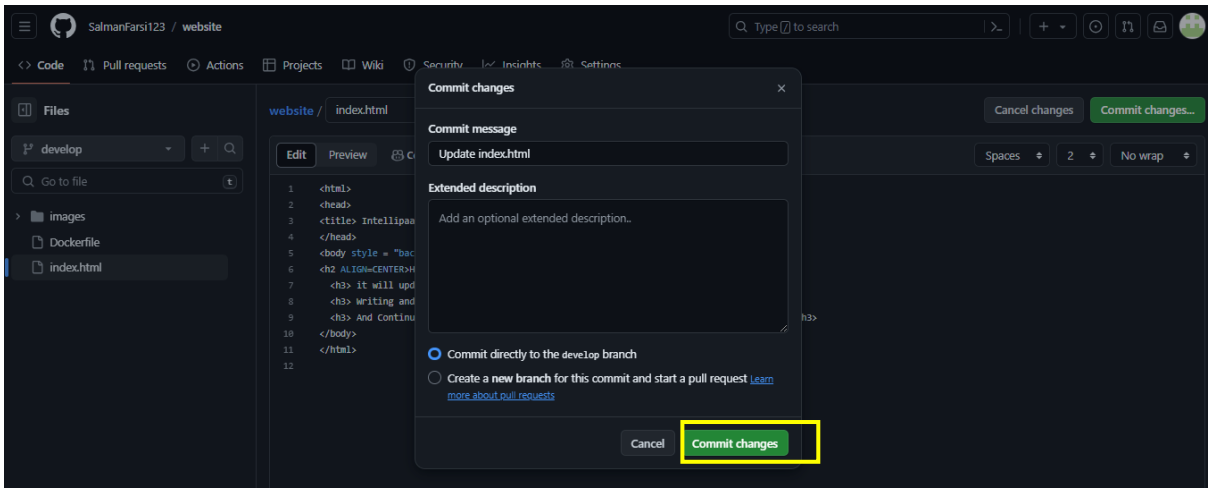


126. Now Again Changing in dockerfile



127. According to trainer, Now it will remove c1 and replace updated data in c1, and port I changed 85

129. Commit changes

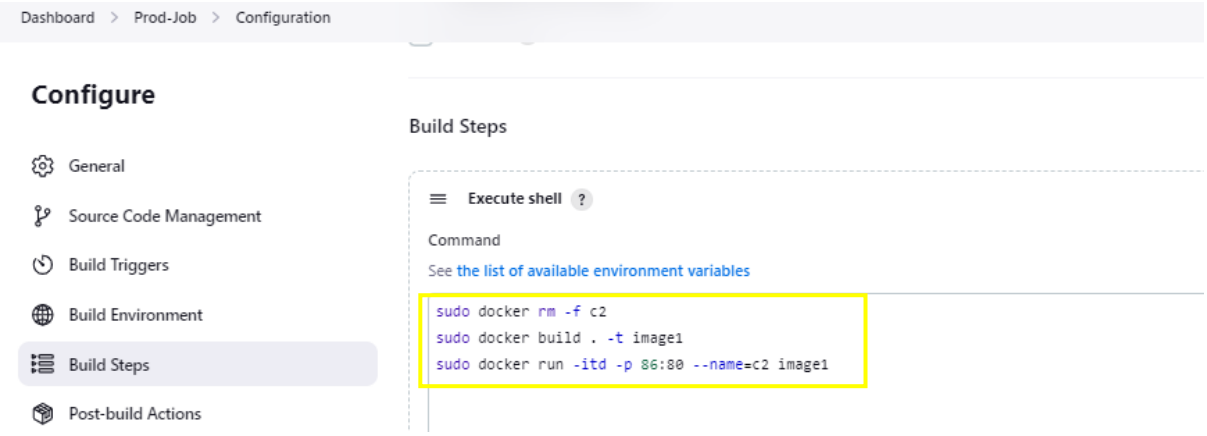
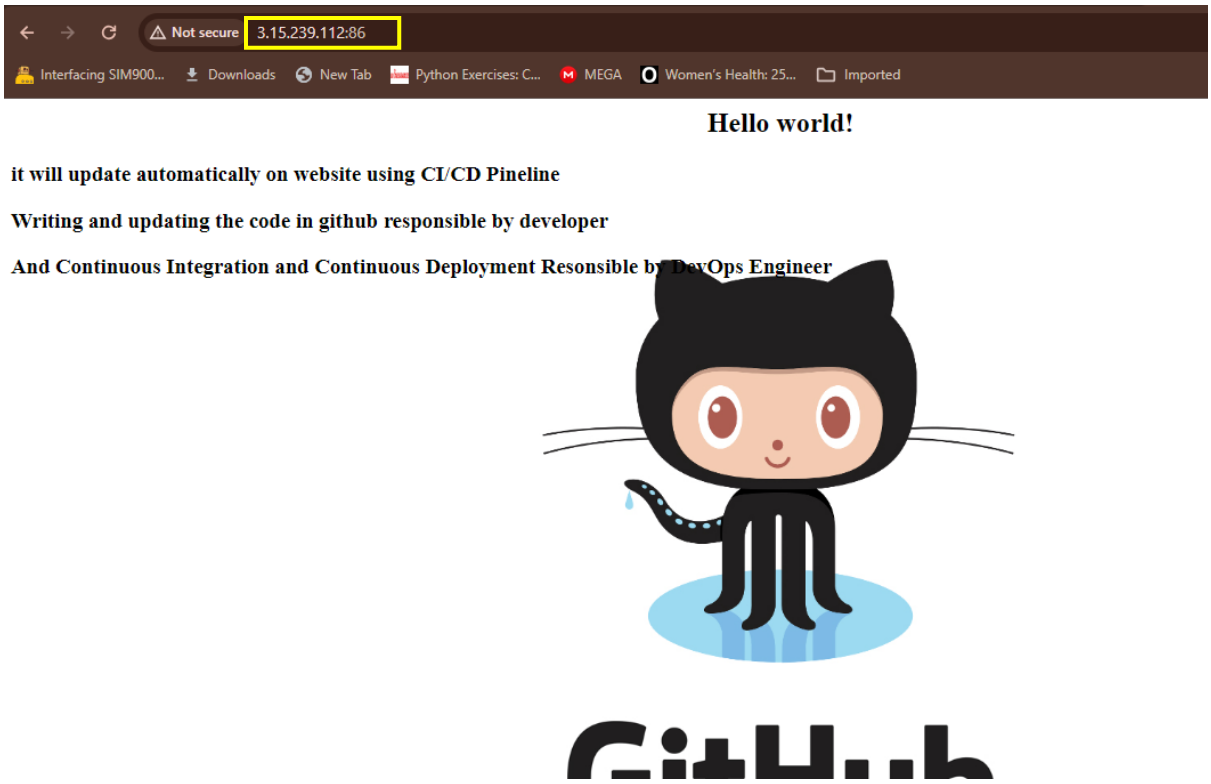


130. it gave an error.

```
docker: Error response from daemon: driver failed programming external connectivi
for 0.0.0.0:85 failed: port is already allocated.
Build step 'Execute shell' marked build as failure
Finished: FAILURE
```

131. Now changing the port number and it will create a another image.

But the thing is, it will Build new docker image it contains(More data)



132. We can see the changes in browser.