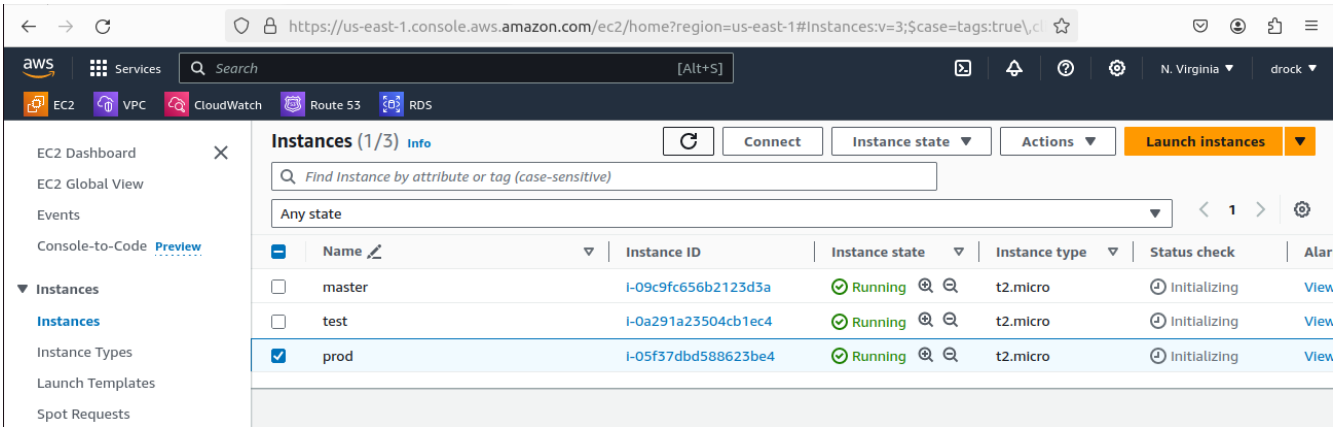
STRUCTURE OF THE JOB

	To be Performed on:	Branch
Job1 :	slave1	Develop
Job2 :	slave1	master
Job3 :	slave2	master

Launch 3 instance:

1 instance as master
2 instance as test
3 instance as prod



Run the command on master
sudo apt install software-properties-common
sudo add-apt-repository --yes --update ppa:ansible/ansible
sudo apt install ansible

```
ubuntu@ip-172-31-25-185:~$ ansible --version
ansible [core 2.15.9]
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/home/ubuntu/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  ansible collection location = /home/ubuntu/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.10.12 (main, Nov 20 2023, 15:14:05) [GCC 11.4.0] (/usr/bin/python3)
  jinja version = 3.0.3
  libyaml = True
ubuntu@ip-172-31-25-185:~$
```

SSH key authentication
On master,

generate ssh-keygen

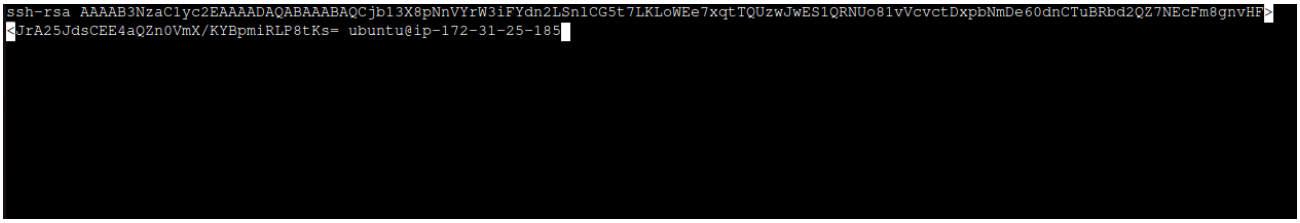
sudo cat ...‘the public key path’… /home/ubuntu/.ssh/id_rsa.pub

sudo cat /home/ubuntu/.ssh/id_rsa.pub

copy the pubic-key to paste in the slave machines

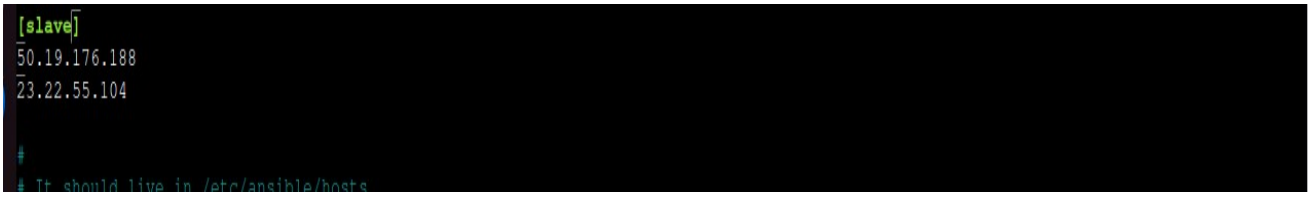
On slave machines

cd .ssh
ls
sudo nano authorized_keys



On master machine

cd /etc/ansible
ls
sudo nano hosts



ansible -m ping all

```
ubuntu@ip-172-31-25-185:/etc/ansible$ ansible -m ping all
50.19.176.188 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
23.22.55.104 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
```

sudo nano playbook.yml

```
}
ubuntu@ip-172-31-25-185:/etc/ansible$ sudo nano playbook.yml
ubuntu@ip-172-31-25-185:/etc/ansible$
```

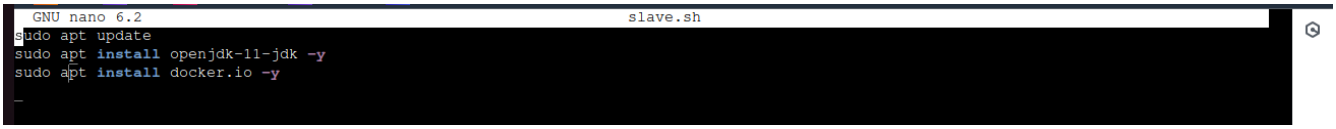
```
---
- name: Task for master
  hosts: localhost
  become: true
  tasks:
    - name: Executing script on master
      script: master.sh

- name: Task for slave
  hosts: all
  become: true
  tasks:
    - name: Executing script on slaves
      script: slave.sh
```

From yaml file above:
sudo nano master.sh
Install Java and Jenkins in master

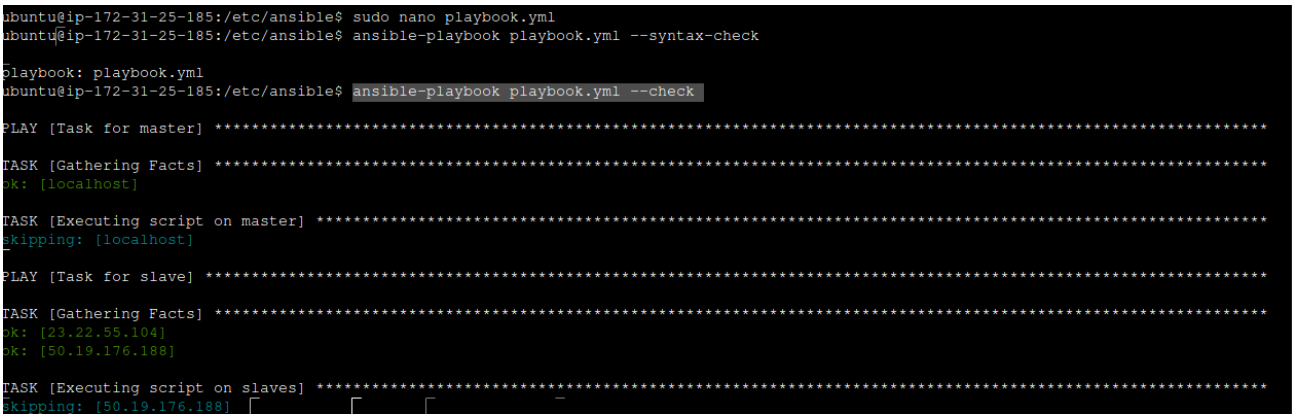
sudo apt updatesudo apt install openjdk-11-jdk -y
sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
https://pkg.jenkins.io/debian/jenkins.io-2023.key
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
https://pkg.jenkins.io/debian binary/ | sudo tee \
/etc/apt/sources.list.d/jenkins.list > /dev/null
sudo apt-get update
sudo apt-get install jenkins -y

sudo nano slave.sh
Install Java and docker in slave machines

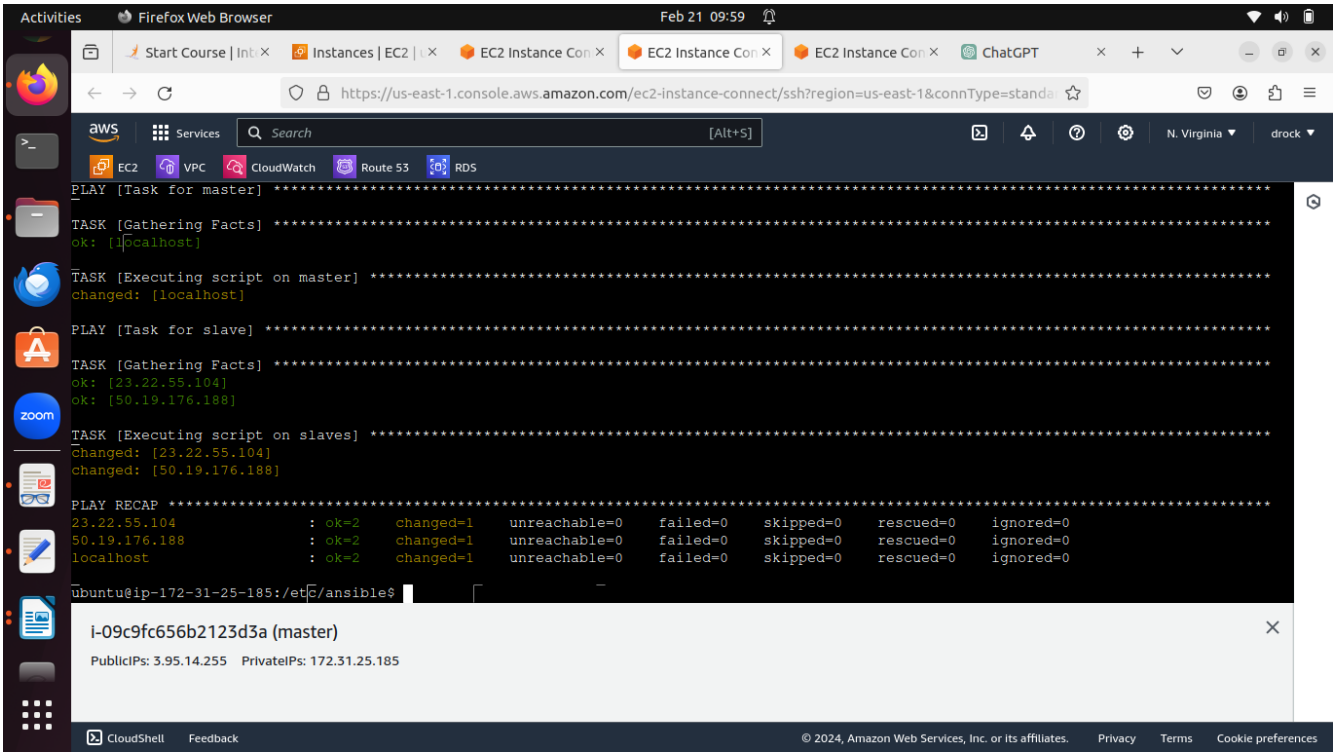
sudo apt update
sudo apt install openjdk-11-jdk -y
sudo apt install docker.io -y



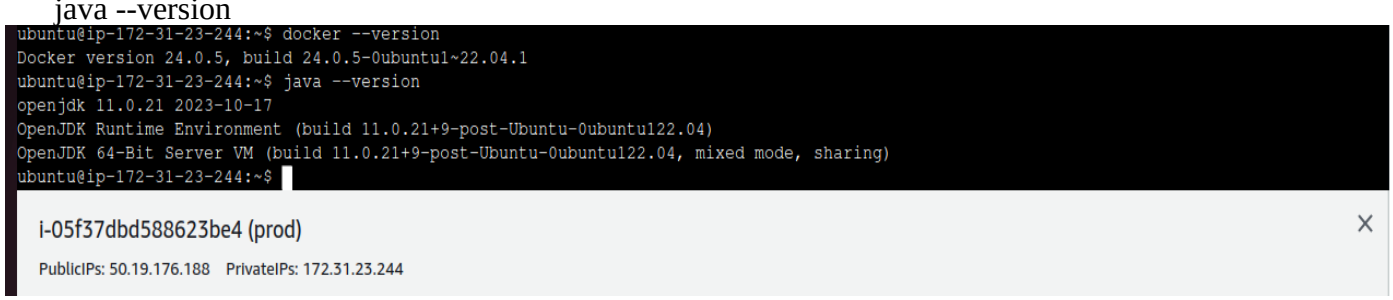
ansible-playbook playbook.yml –syntax-check
ansible-playbook playbook.yml –check



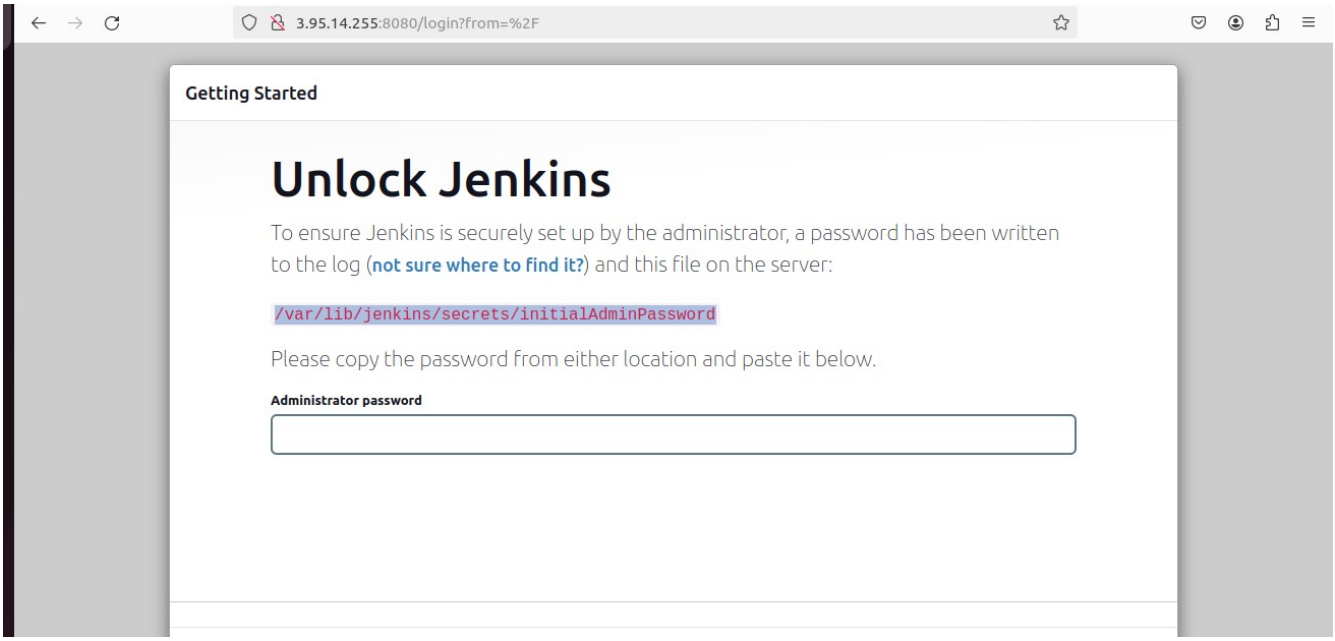
ansible-playbook playbook.yml



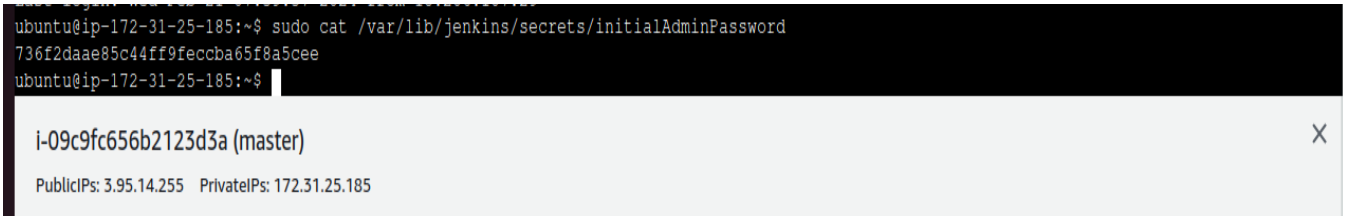
On Slaves:
docker –version
java --version



JENKINS
PUBLIC IP :8080

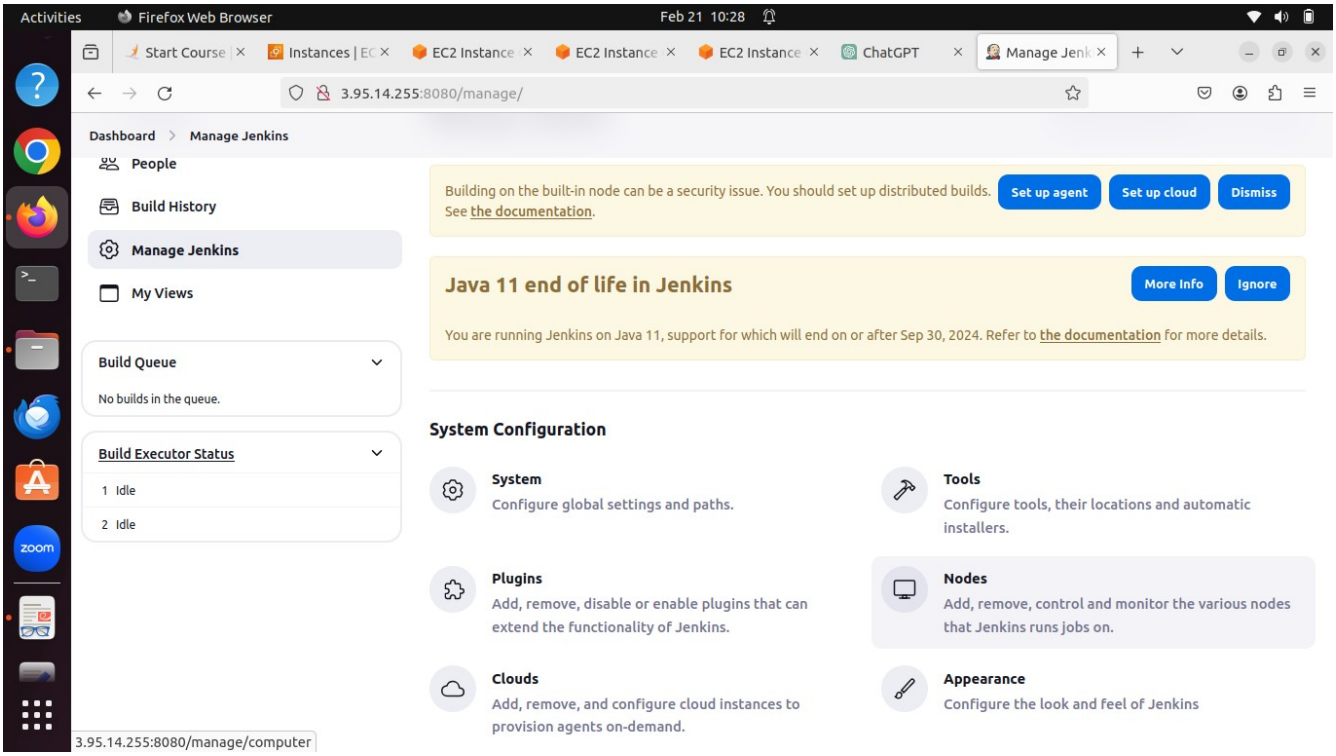


sudo cat `/var/lib/jenkins/secrets/initialAdminPassword`

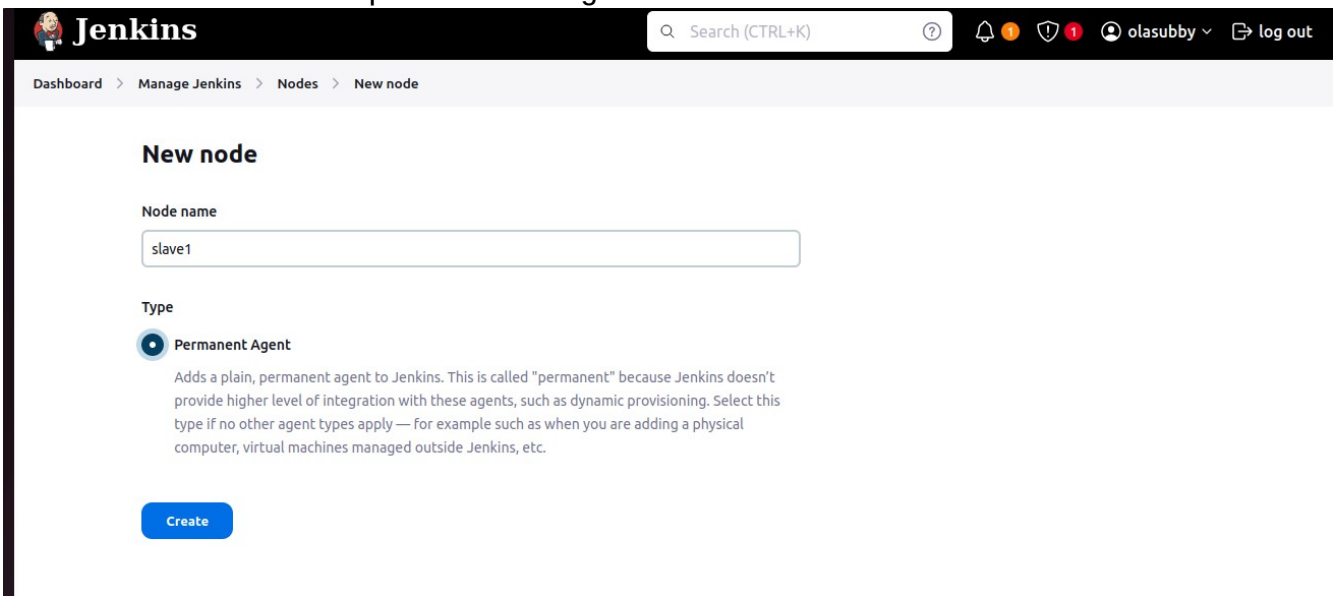


copy password from here to jenkins page

Manage Jenkins – Syem Configure -- Nodes

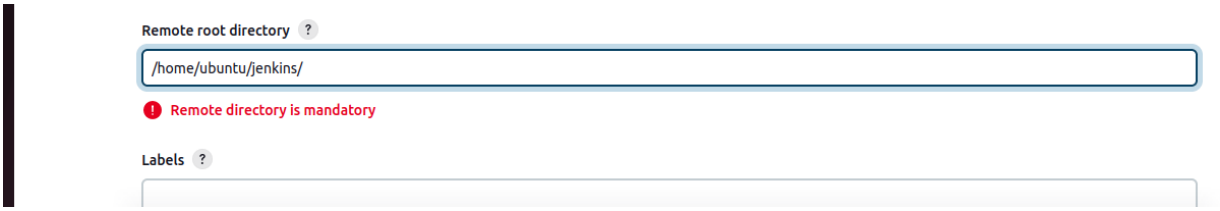


Node Name: slave1> permanent agent ----create



Down scroll to:

Remote root directory:
*/home/*ubuntu/jenkins

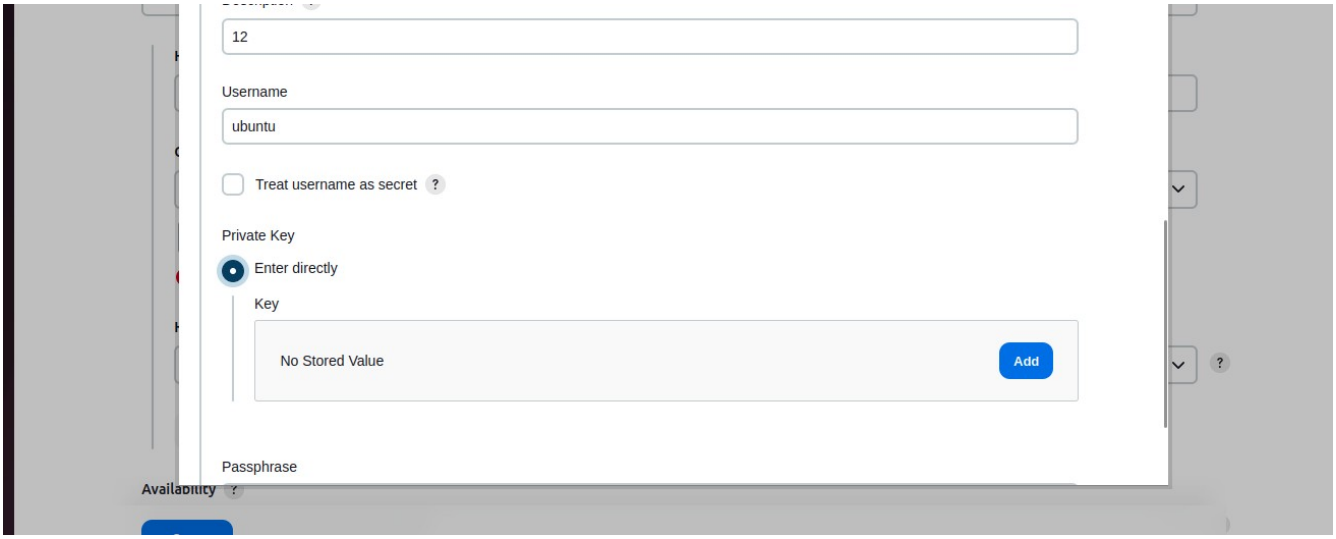


.
.

Launch Method
  Launch Agent via ssh
  Private ip of slave1
Credentials:
            Add
           Jenkins

username: ubuntu

click: Enter directory

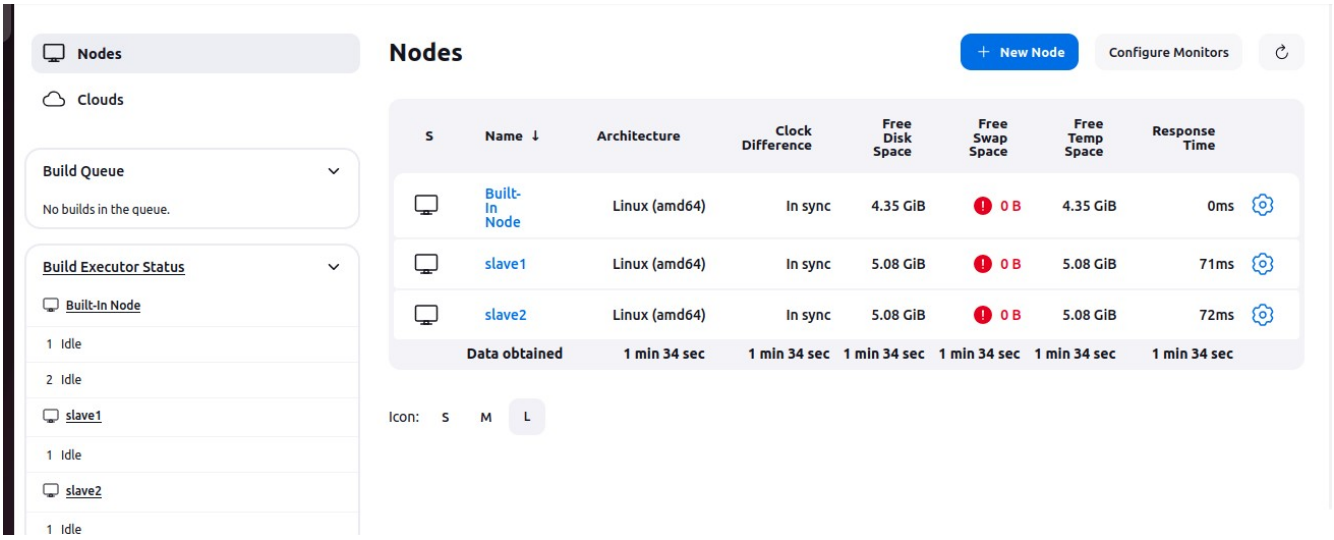


put in the private key from the slave server and save

Private key>eneter directly:paste the pem file>add

Host key Verification: Non verifying strategy

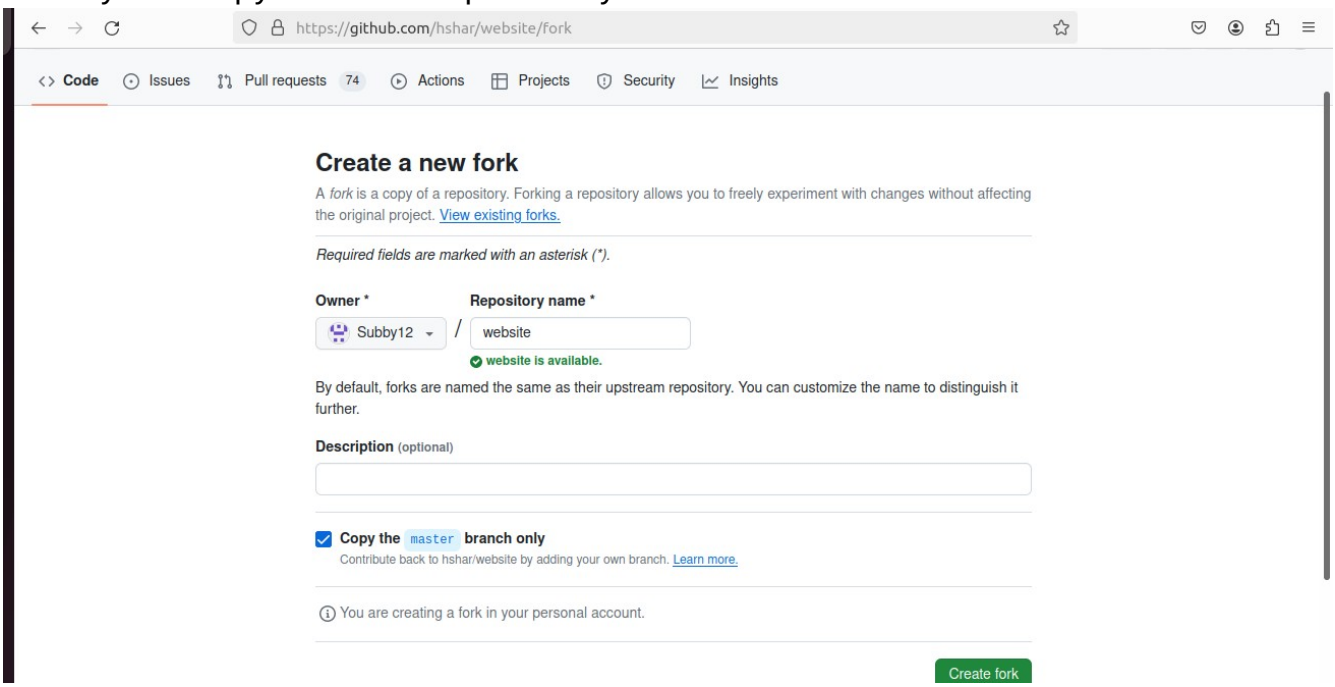
Save



Create a Dockerfile

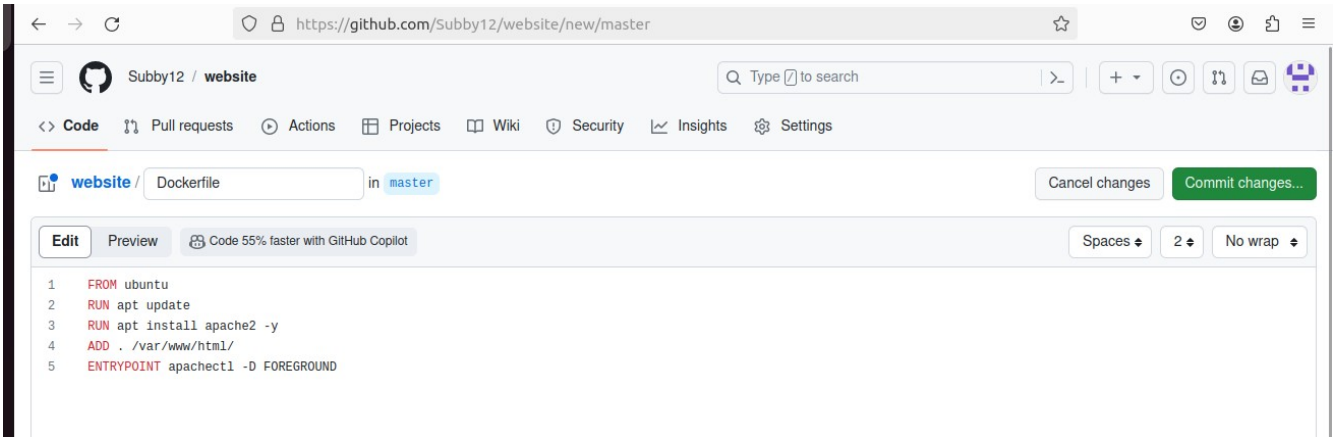
From the github link: https://github.com/hshar/website.git

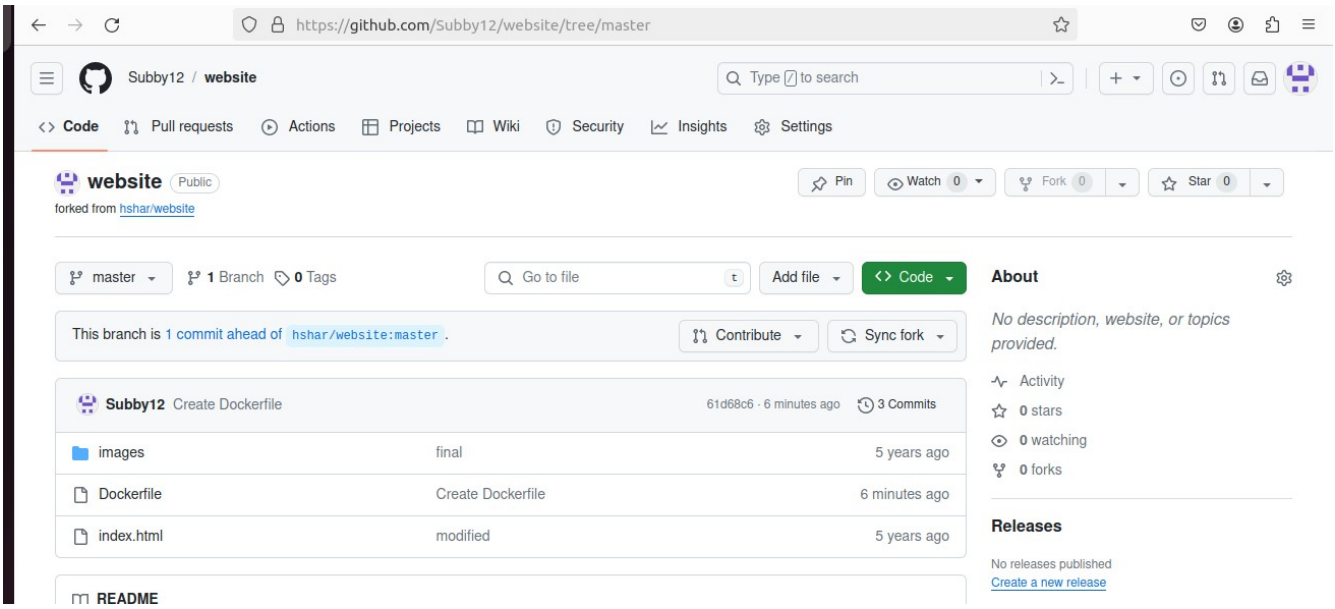
fork your copy of the repository



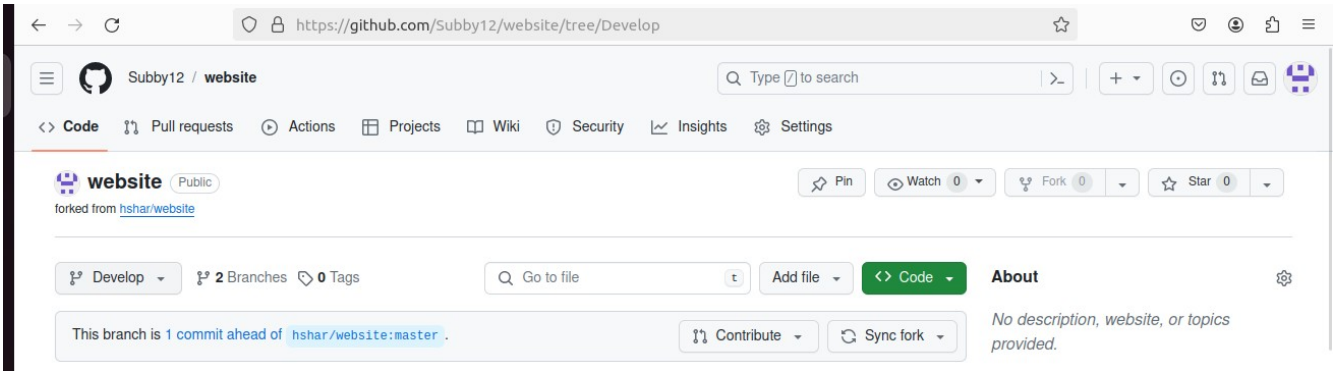
content:

```
FROM ubuntu
RUN apt update
RUN apt install apache2 -y
ADD . /var/www/html/
ENTRYPOINT apachectl -D FOREGROUND
```

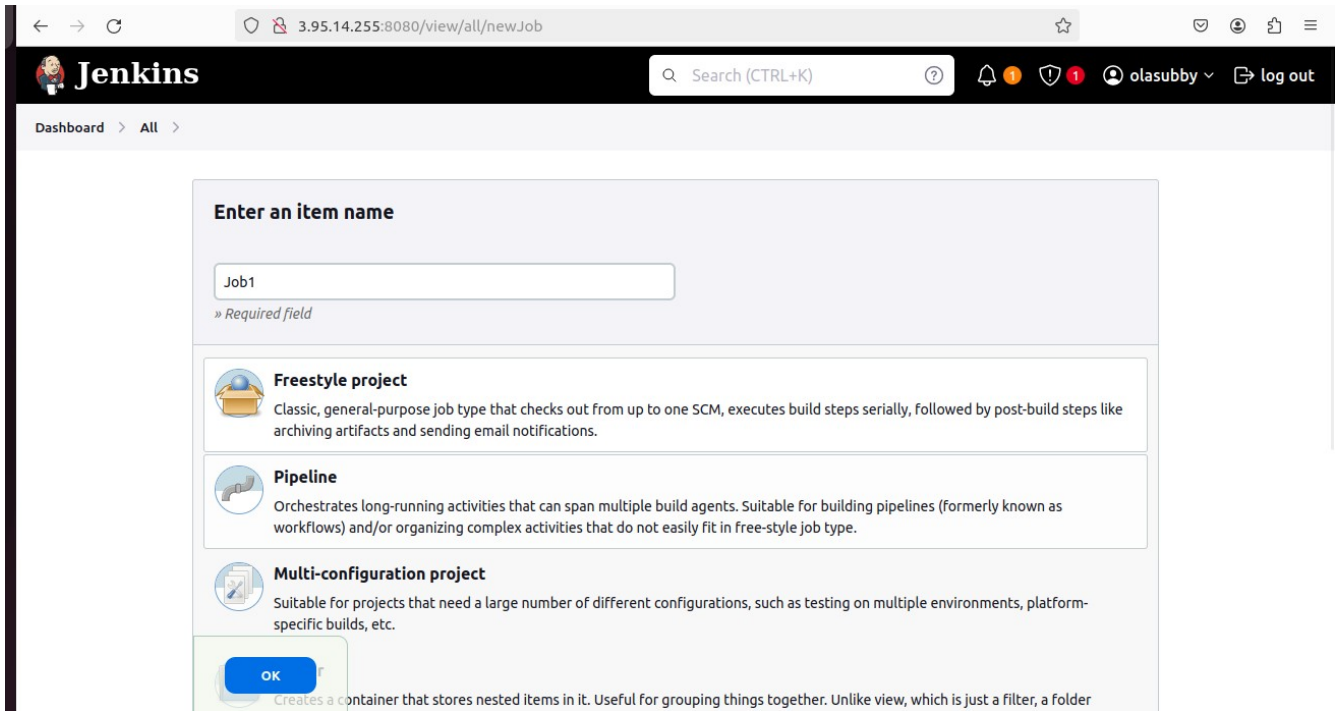




Create branch name =Develop IN THE GITHUB

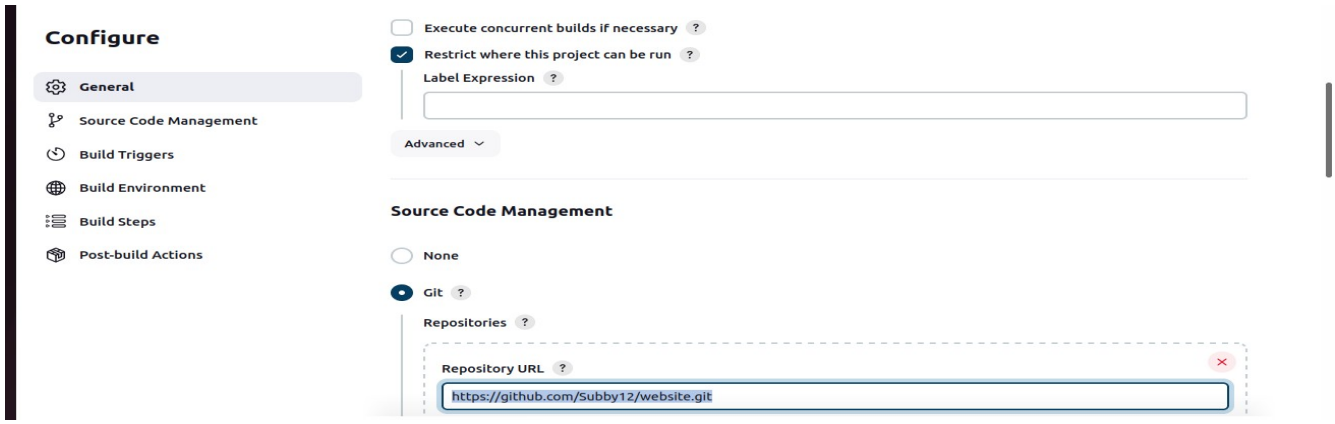


Go ahead and create the jobs in Jenkins Dashboard…. New Items
Give the job name: Job1
Pick: Freestyle option

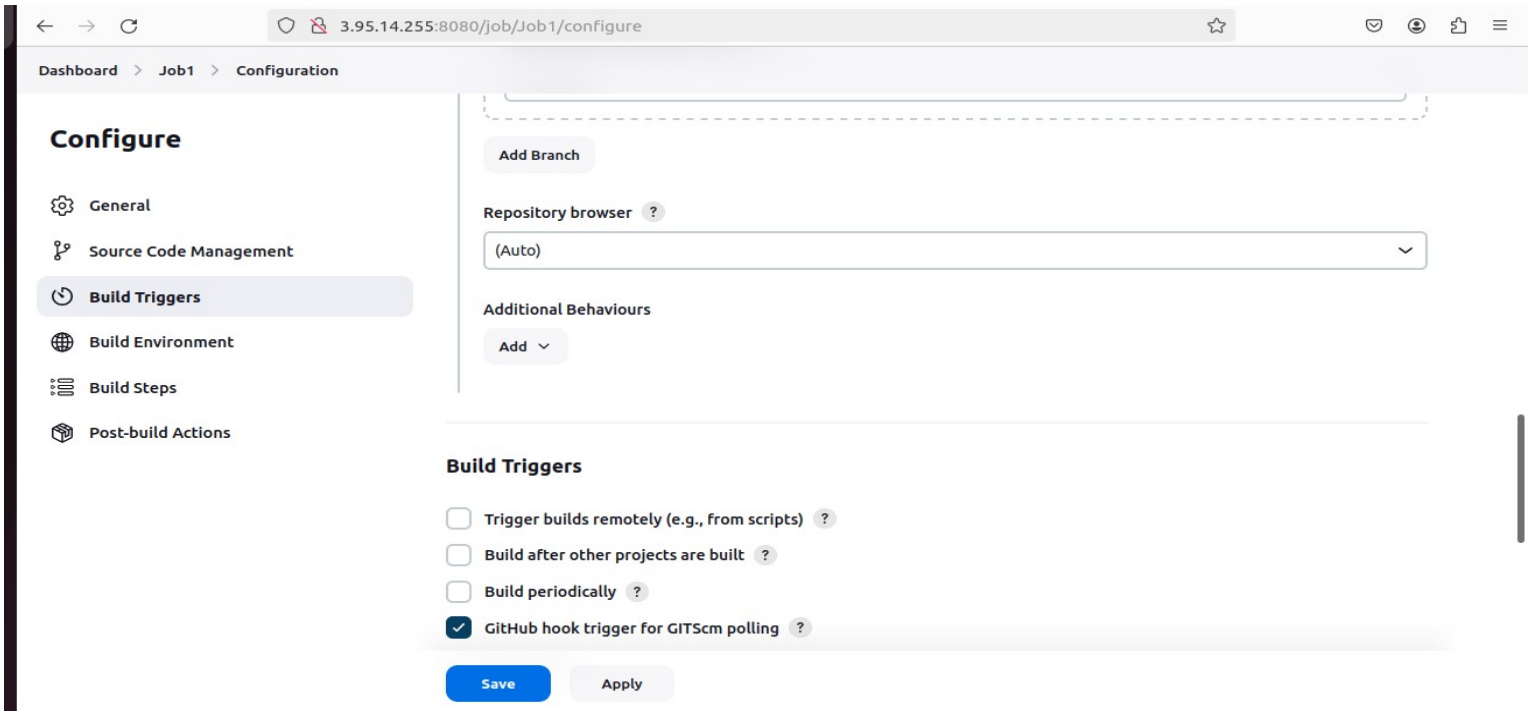


GIt hub project: add repo

Click on Restrict where the project run: Slave1 Source Code Management: Git> paste URL

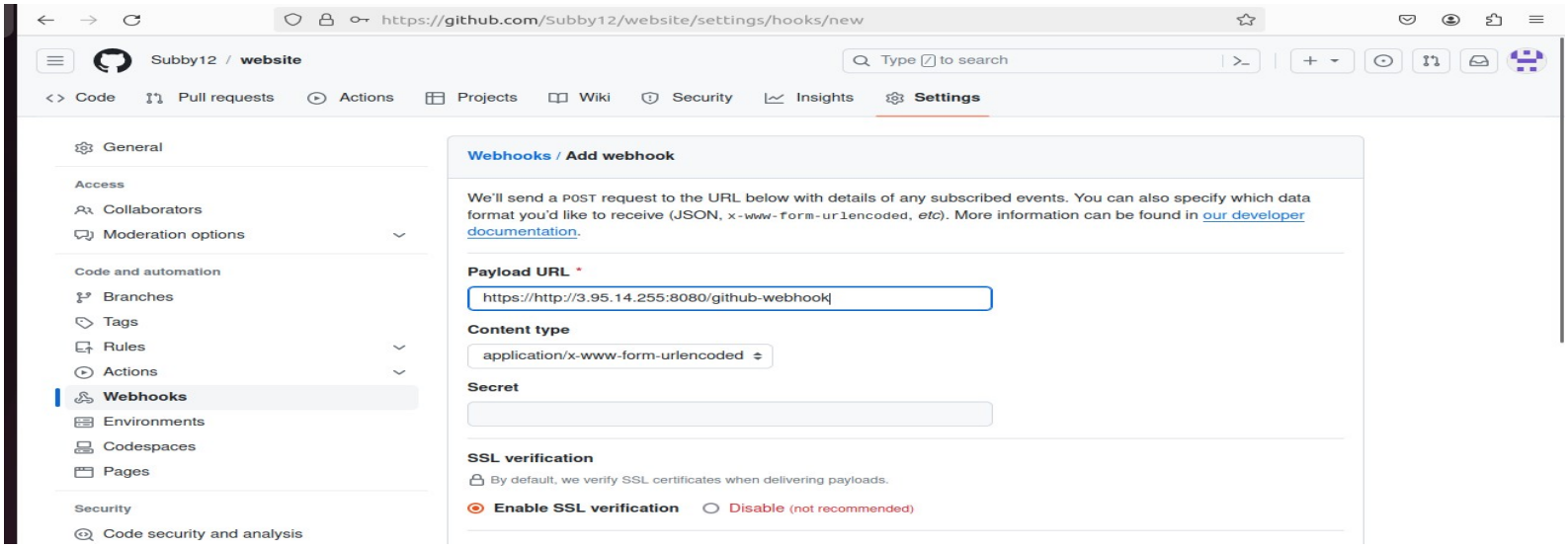


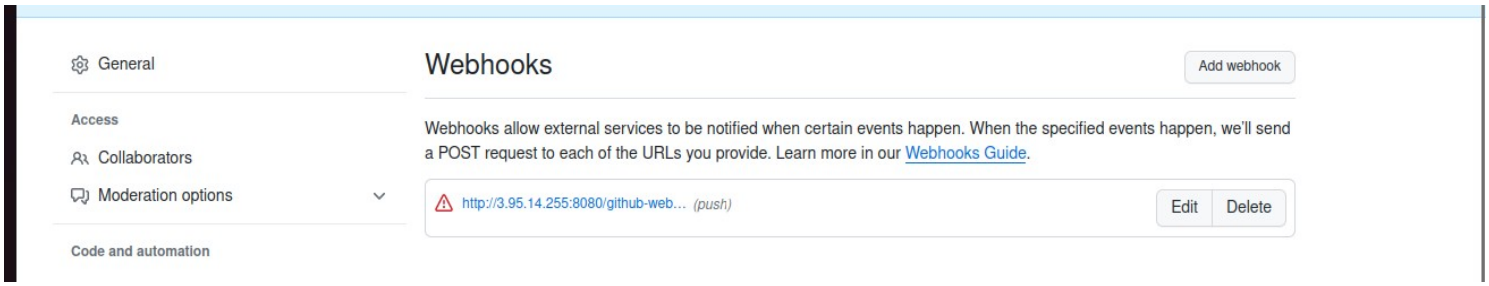
Specify branches: Develop



pick Git scm polling > Copy the jenkins Url i.e http://2.426.23.42:8080>

go to github>settings>webhook>add webhook> paste the url(url/github-webhook/) > add > click the link> recent deliveries





Come back to dashboard>
Save >Build job1

Slave1 Server
Dockerfile created can be seen here

```
ubuntu@ip-172-31-26-16:~$ ls
jenkins
ubuntu@ip-172-31-26-16:~$ cd jenkins
ubuntu@ip-172-31-26-16:~/jenkins$ ls
remoting  remoting.jar  workspace
ubuntu@ip-172-31-26-16:~/jenkins$ cd workspace
ubuntu@ip-172-31-26-16:~/jenkins/workspace$ ls
Job1
ubuntu@ip-172-31-26-16:~/jenkins/workspace$ cd job1
-bash: cd: job1: No such file or directory
ubuntu@ip-172-31-26-16:~/jenkins/workspace$ cd job1
-bash: cd: job1: No such file or directory
ubuntu@ip-172-31-26-16:~/jenkins/workspace$ cd Job1
ubuntu@ip-172-31-26-16:~/jenkins/workspace/Job1$ ls
Dockerfile  images  index.html
ubuntu@ip-172-31-26-16:~/jenkins/workspace/Job1$
```

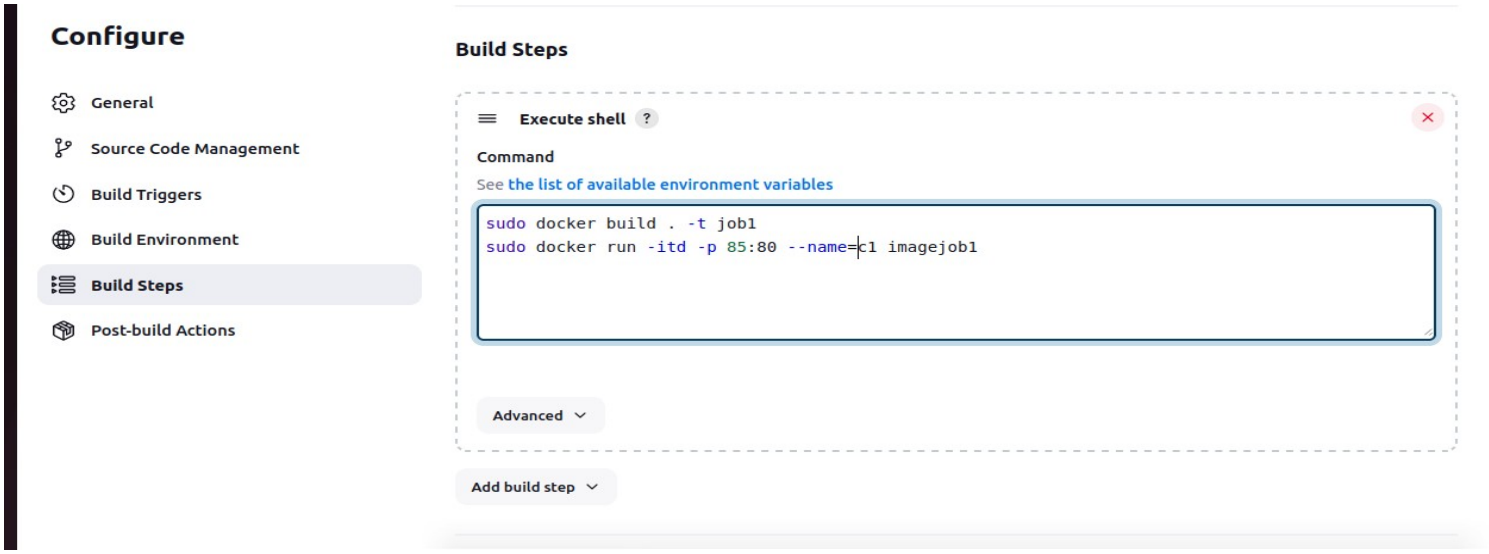
i-0a291a23504cb1ec4 (test)

PublicIPs: 23.22.55.104 PrivateIPs: 172.31.26.16

We need to : CREATE AN IMAGE from the Dockerfile

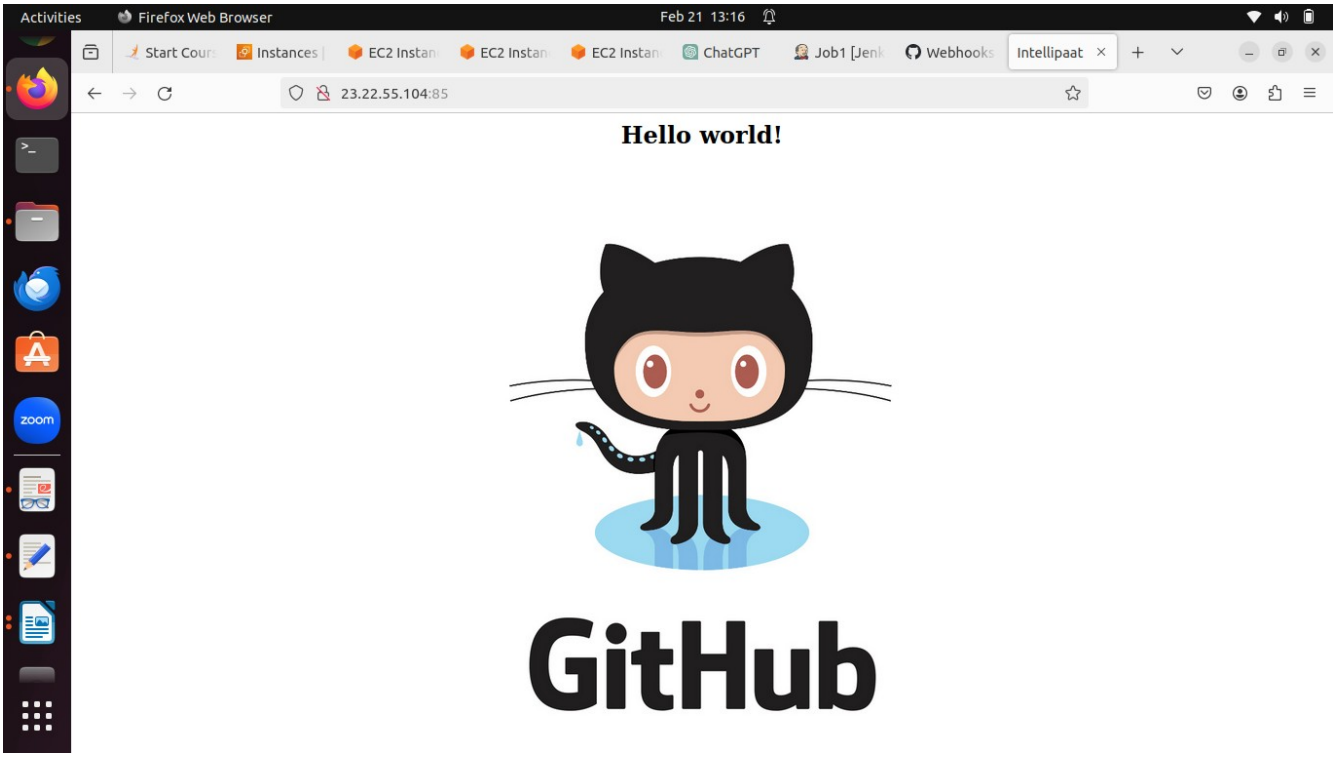
Go back to dashboard
Job1> Configure> Build Step>Execute shell

```
sudo docker build /home/ubuntu/jenkins/workspace/Job1/ -t imagejob1
sudo docker run -itd -p 85:80 –name c1 imagejob1
```



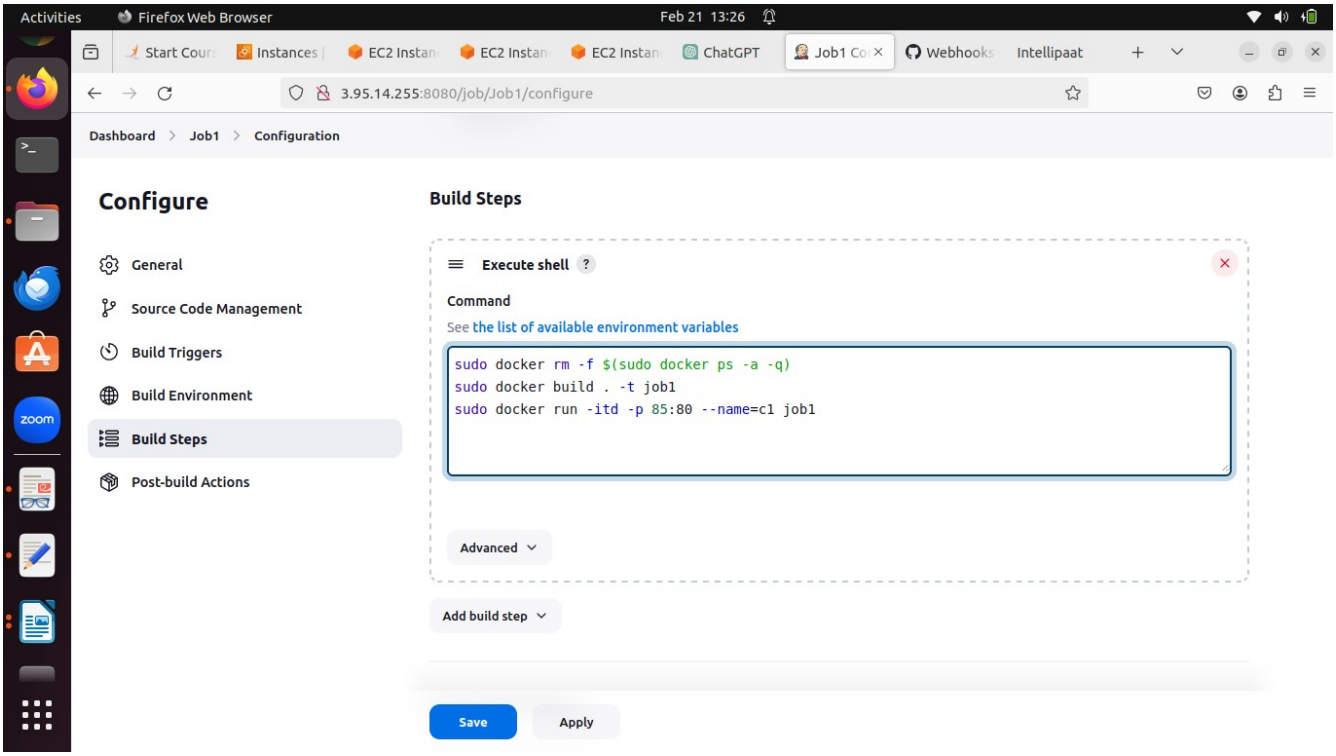
Add >BUild now

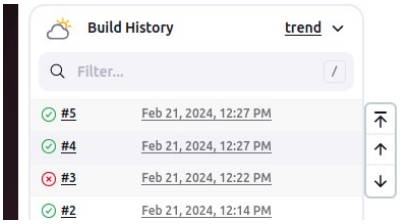
Copy the public ip of test and run it with port 85



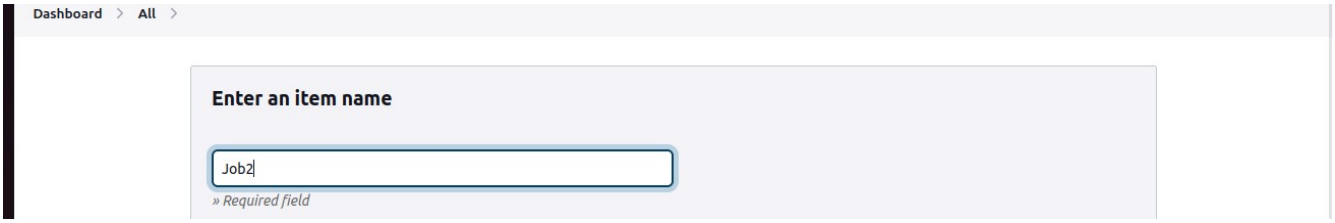
To build Again without fail: change the command to include:

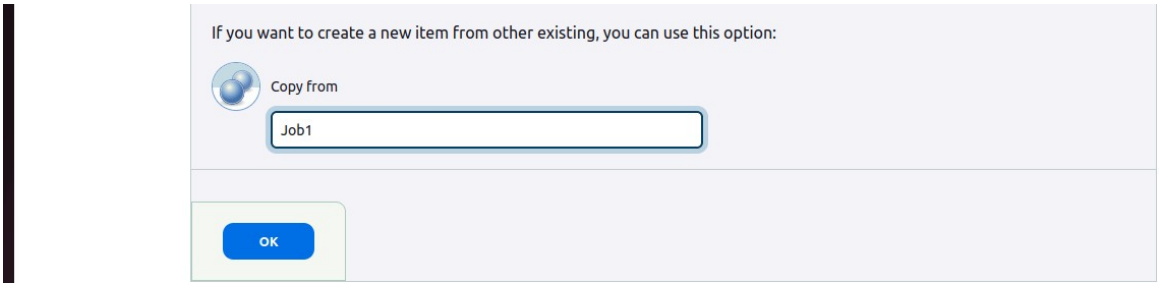
```
sudo docker rm -f $(sudo docker ps -a -q)
sudo docker build . -t job1
sudo docker run -itd -p 85:80 --name=c1 job1
```


Activities
Firefox Web Browser
Feb 21 13:26
Start Cours
Instances
EC2 Instan
EC2 Instan
EC2 Instan
ChatGPT
Job1 Co
Webhooks
Intellipaat
3.95.14.255:8080/job/Job1/configure
Dashboard
Job1
Configuration
Configure
General
Source Code Management
Build Triggers
Build Environment
Build Steps
Post-build Actions
Build Steps
Execute shell
Command
See the list of available environment variables
sudo docker rm -f $(sudo docker ps -a -q)
sudo docker build . -t job1
sudo docker run -itd -p 85:80 --name=c1 job1
Advanced
Add build step
Save
Apply

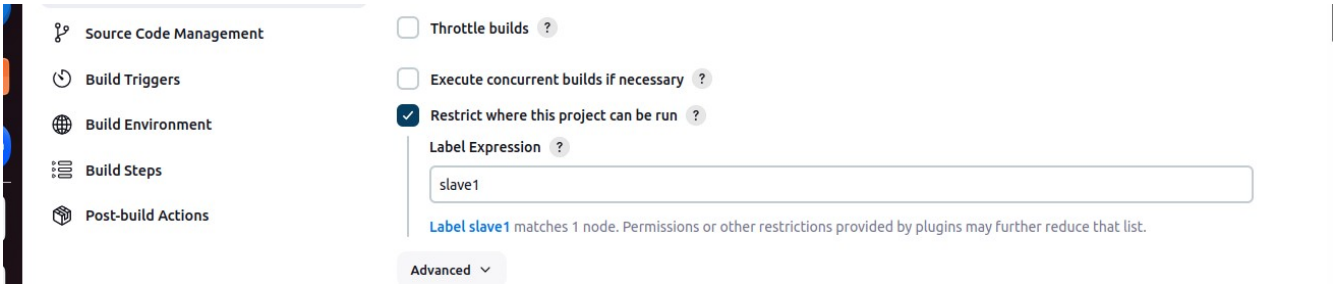

Build History
trend
Filter...
#5 Feb 21, 2024, 12:27 PM
#4 Feb 21, 2024, 12:27 PM
#3 Feb 21, 2024, 12:22 PM
#2 Feb 21, 2024, 12:14 PM

Create Job2
Dashboard>New Item

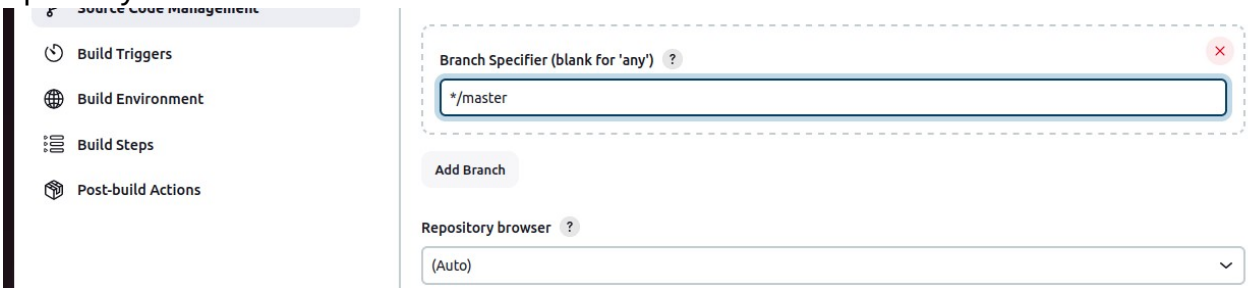




Pick slave1



specify: */master



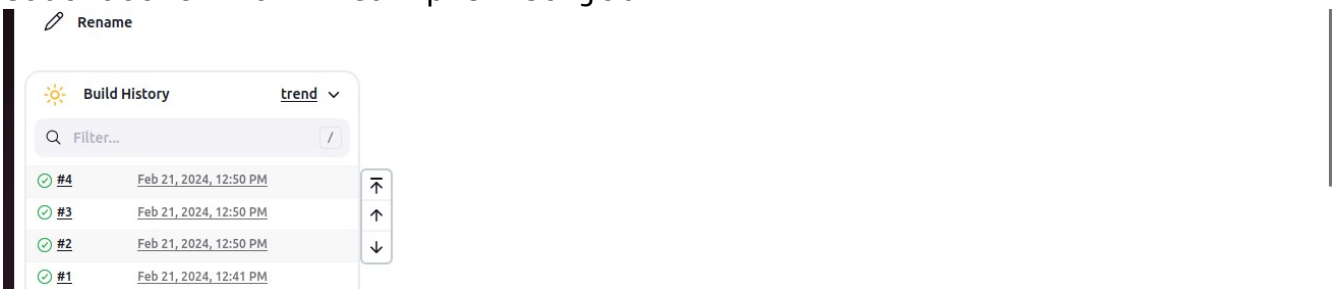
Command:
sudo docker build . -t job2
sudo docker run -itd -p 80:80 job2

apply & Save

Build Now

To rebuild after 1st build:
use command:(change port….)

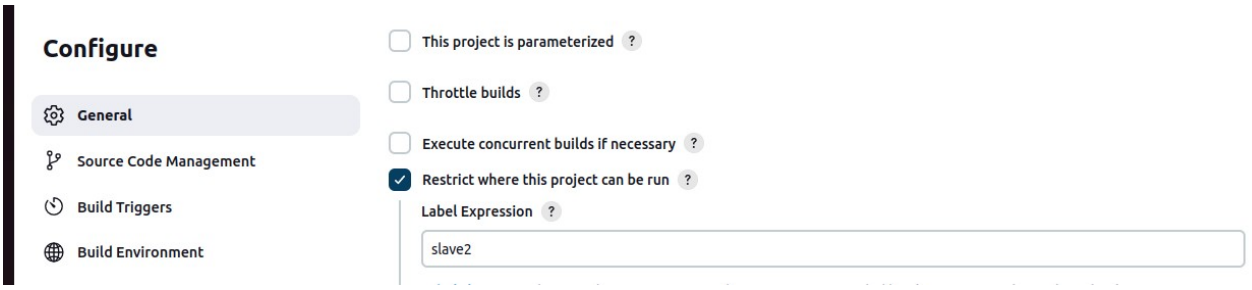
```
sudo docker rm -f $(sudo docker ps -a -q)
sudo docker build . -t job2
sudo docker run -itd -p 82:80 job2
```



Public ip of slave1:port82

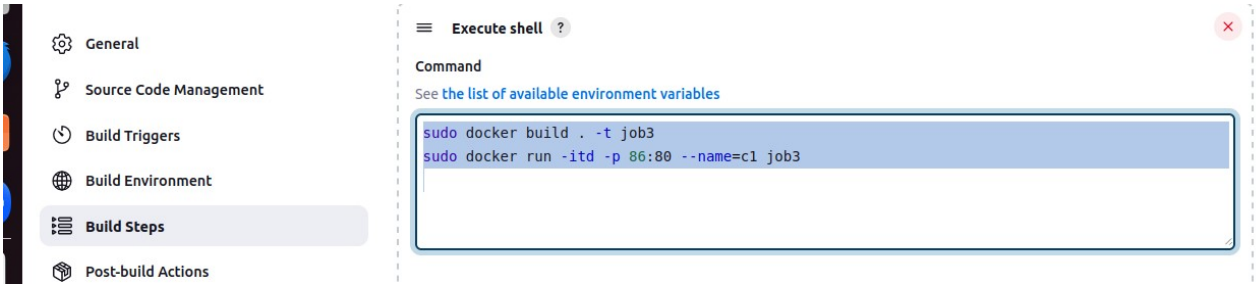
Job 3:
Dashboard>New Item

Choose slave2

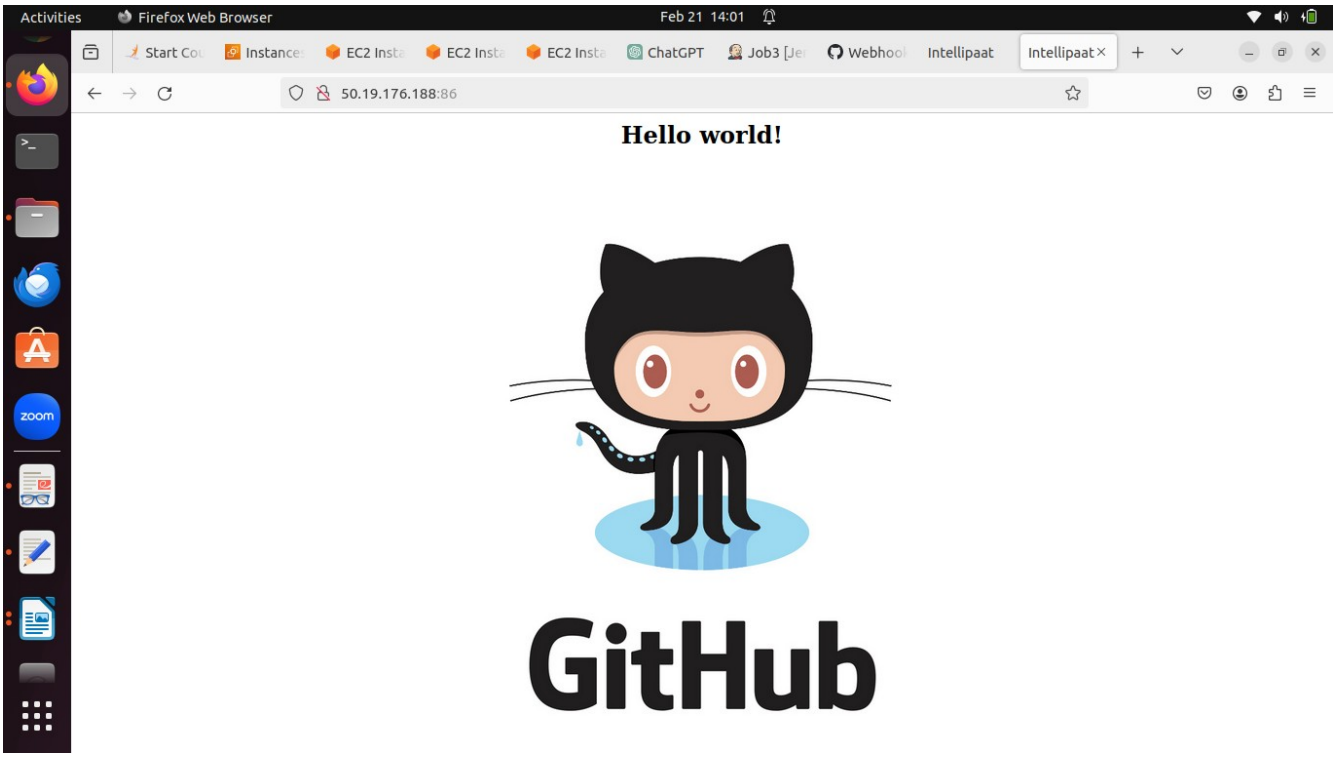


Branch: Master

Command:
sudo docker build . -t job3
sudo docker run -itd -p 86:80 --name=c1 job3



Public ip of slave2:86



If I proceed to make changes to html file in the github and commit it: