DevOps Capstone

Project-1

You have been hired as a 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 and their product is available on this GitHub link.

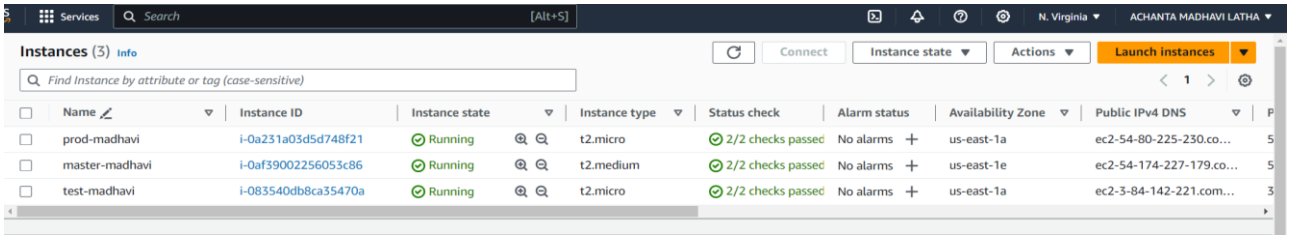
https://github.com/hshar/website.git

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. CodeBuild should automatically be triggered once a commit is made to master branch or develop branch.
    a. If a commit is made to master branch, test and push to prod
    b. If a 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 GitHub. 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:
    a. Job1 : build
    b. Job2 : test
    c. Job3 : prod

To start with this tasks we need to deploy 3 EC2 instances

1. Master instance
2. Test instance
3. Prod instance



Connect to each instance and update the machine.

Now in the master instance we will install **Ansible** using g following commands.

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

```
ubuntu@ip-172-31-61-175:~$ ansible --version
ansible [core 2.12.10]
  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 = 3.8.10 (default, May 26 2023, 14:05:08) [GCC 9.4.0]
  jinja version = 2.10.1
  libyaml = True
```

ansible is installed
now key generation
use command ssh-keygen and click on enter till your public key occurs

```
ubuntu@ip-172-31-61-175:~$ 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:zrIiV+mVGqwSv1/REitWLqyTnLqBR7NTqapoiiGc12c ubuntu@ip-172-31-61-175
The key's randomart image is:
+---[RSA 3072]----+
|                 |
|                 |
|        o        |
|     o o +       |
|  o o.=.S..      |
|.oo*.===oo       |
|+o*+*=.E+        |
|+=++=.=+         |
|X o=.+o          |
+----[SHA256]-----+
```

Now our private key is stored
To open this private key go to the location using cat command

```
ubuntu@ip-172-31-61-175:~$ sudo cat  /home/ubuntu/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDV/LWDSDhkP9CcTOnIZAzpPFZjBD/WCTdgDdHOq3yQuk1rxy1KjOaMzBMS/0f6xSJGCjqmw4tJMVaIPoCo5z4BgtLUaZIdxd2RiFF+CnRrJ7J3NjoWQrycuProqCNY1cJFUi
g1uUPM2pkOI20yBZs/aWTAec/w7BIIUrAZTVs2zm+1yWHDcCOCgalV7tZSY1HbUs36QgBlAB+rIH+C6+N1wwBlgv3XLF+hDt2S8iQgSBlX9mdKIx0qY0UHXgn51T175OVULGAgEczCVXU0lLdKDeLdecXU6I9/zvnuLDk2G3zc
0RfLMWtP50XqXz9xbDv2RUW3wvzloxyaTADgCLWWXwh95yKkVuv4vuRchcfF8VmSRy+Hw/EYfsCVhttcJP8dlP+gQK3SL3quvqduhe8sqTkzSncAT76h/61/TNLiNVAoWgP8KCsLhgqnDxGjYgZ2iyV/1m0ZCgR/CgoEiHbLF8
j1JC4/S81AjG65Upyy6F435U9h5R6KP5OZP49p+zM= ubuntu@ip-172-31-61-175
```

Now copy this private keys and paste in test and prod instances
Test instance

```
ubuntu@ip-172-31-47-213:~$ cd .ssh
ubuntu@ip-172-31-47-213:~/.ssh$ ls
authorized_keys
ubuntu@ip-172-31-47-213:~/.ssh$ sudo nano authorized_keys
```

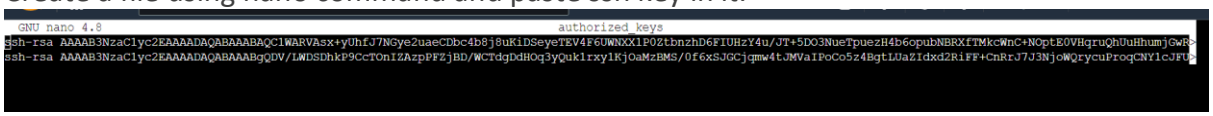
create a file using nano command and paste ssh key in it

```
  GNU nano 4.8                                  authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQC1WARVAsx+yUhfJ7NGye2uaeCDbc4b8j8uKiDSeyeTEV4F6UWNXX1PQZtbnzhD6FIUHzY4u/JT+5DO3NueTpuezH4b6opubNBRXfTMkcWnC+NOptE0VHqruQhUuHhumjGWH
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDV/LWDSDhkP9CcTOnIZAzpPFZjBD/WCTdgDdHOq3yQuk1rxy1KjOaMzBMS/0f6xSJGCjqmw4tJMVaIPoCo5z4BgtLUaZIdxd2RiFF+CnRrJ7J3NjoWQrycuProqCNY1cJFUi
```

Prod instance

```
ubuntu@ip-172-31-39-123:~$ cd .ssh
ubuntu@ip-172-31-39-123:~/.ssh$ ls
authorized_keys
ubuntu@ip-172-31-39-123:~/.ssh$ sudo nano authorized_keys
```

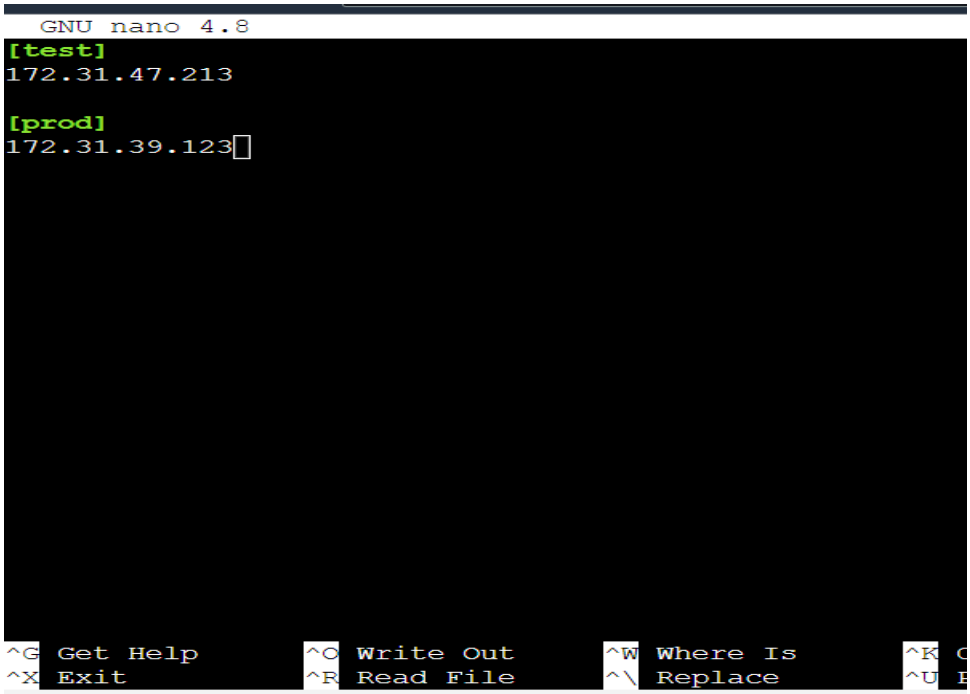
Create a file using nano command and paste ssh key in it.



Now go to master instance

```
ubuntu@ip-172-31-61-175:~$ cd /etc/ansible
ubuntu@ip-172-31-61-175:/etc/ansible$ ls
ansible.cfg  hosts  roles
ubuntu@ip-172-31-61-175:/etc/ansible$ sudo nano hosts
```

In this nano file delete all the text and type as shown to create a cluster between master to test and prod.



i-0af39002256053c86 (master-madhavi)

PublicIPs: 54.174.227.179 PrivateIPs: 172.31.61.175

Now we can ping whether the cluster is created between the test and prod

```
ubuntu@ip-172-31-61-175:/etc/ansible$ ansible -m ping all
The authenticity of host '172.31.39.123 (172.31.39.123)' can't be established.
ECDSA key fingerprint is SHA256:qPRHm6dk1LujyTdHT/z7qlWGxYhLEPeMHu/13d22P7s.
The authenticity of host '172.31.47.213 (172.31.47.213)' can't be established.
ECDSA key fingerprint is SHA256:1L79hKCEpjBrCagGOU6C6UNivOHR0xD91mOo6ysjgyI.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
172.31.39.123 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
yes
172.31.47.213 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
ubuntu@ip-172-31-61-175:/etc/ansible$
```

i-0af39002256053c86 (master-madhavi)

PublicIPs: 54.174.227.179 PrivateIPs: 172.31.61.175

```
ubuntu@ip-172-31-61-175:/etc/ansible$ sudo nano install.yaml
ubuntu@ip-172-31-61-175:/etc/ansible$ ls
ansible.cfg  hosts  install.yaml  roles
ubuntu@ip-172-31-61-175:/etc/ansible$ sudo cat install.yaml
---
 - hosts: localhost
   become: true
   name: install Jenkins, Java and Docker
   tasks:
    - name: master task
      script: jenkins.sh

 - hosts: test
   become: true
   name: install Java and Docker
   tasks:
   - name: test task
     script: docker.sh

 - hosts: prod
   become: true
   name: install Java and Docker
   tasks:
     - name: prod task
       script: docker.sh
```

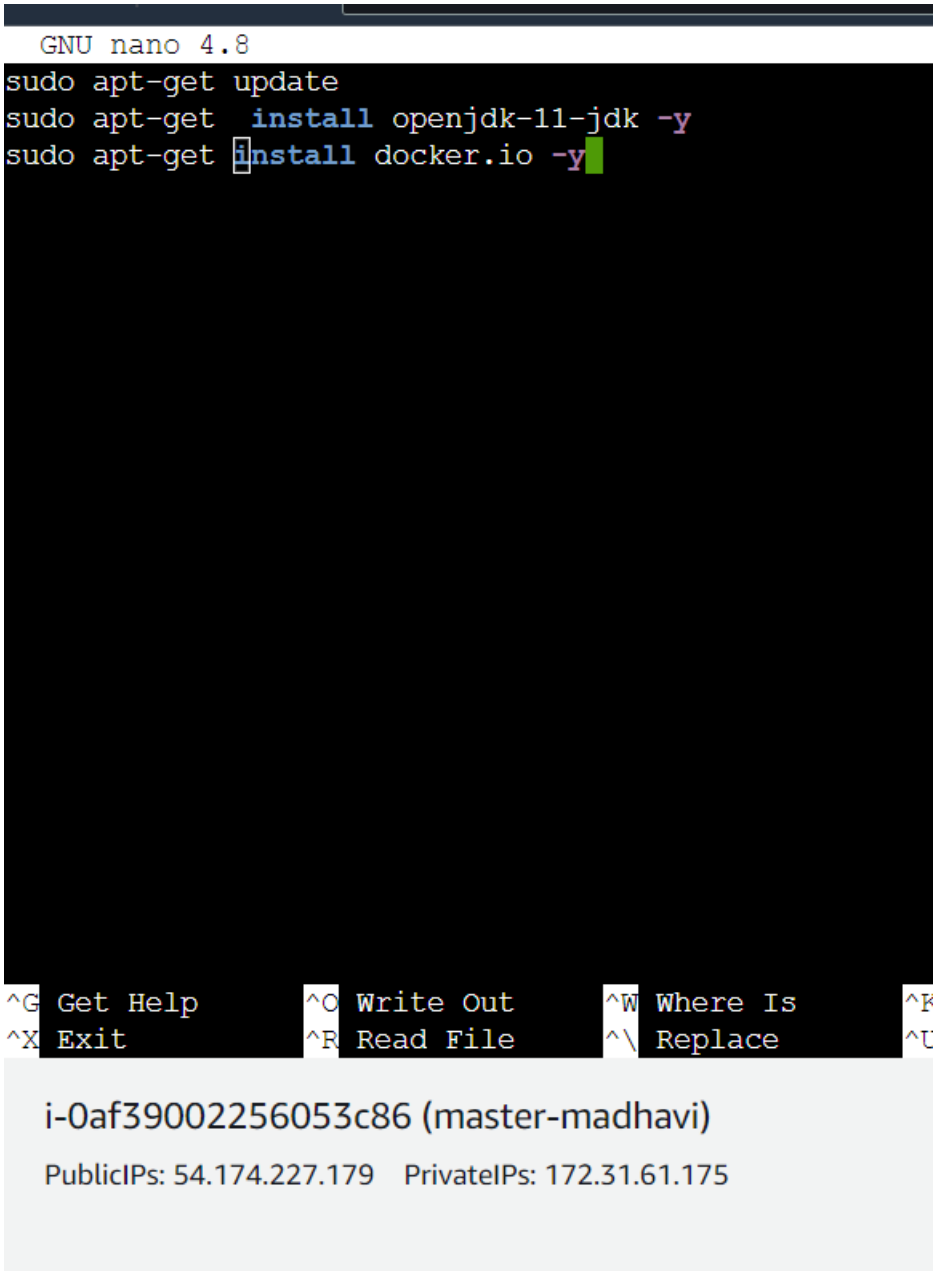
Now we will create a Jenkins.sh file to install Jenkins.

```
  GNU nano 4.8
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 openjdk-11-jdk -y
sudo apt-get install docker.io -y
sudo apt-get install jenkins -y
^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text
^X Exit        ^R Read File   ^\ Replace     ^U Paste Text
```

i-0af39002256053c86 (master-madhavi)

PublicIPs: 54.174.227.179 PrivateIPs: 172.31.61.175

Noe we will create adocker.sh file so that docker and java will be installed.




```
ubuntu@ip-172-31-61-175:/etc/ansible$ sudo nano jenkins.sh
ubuntu@ip-172-31-61-175:/etc/ansible$ sudo nano docker.sh
ubuntu@ip-172-31-61-175:/etc/ansible$ ls
ansible.cfg  docker.sh  hosts  install.yaml  jenkins.sh  roles
```

We are checking whether the commands we have given are correct or not

```
ubuntu@ip-172-31-61-175:/etc/ansible$ ansible-playbook install.yaml --syntax-check

playbook: install.yaml
ubuntu@ip-172-31-61-175:/etc/ansible$ ansible-playbook install.yaml --check

PLAY [install Jenkins, Java and Docker] ****************************************************************************

TASK [Gathering Facts] *********************************************************************************************
ok: [localhost]

TASK [master task] *************************************************************************************************
skipping: [localhost]

PLAY [install Java and Docker] *************************************************************************************

TASK [Gathering Facts] *********************************************************************************************
ok: [172.31.47.213]

TASK [test task] ***************************************************************************************************
skipping: [172.31.47.213]

PLAY [install Java and Docker] *************************************************************************************

TASK [Gathering Facts] *********************************************************************************************
ok: [172.31.39.123]

TASK [prod task] ***************************************************************************************************
```

i-0af39002256053c86 (master-madhavi)

PublicIPs: 54.174.227.179 PrivateIPs: 172.31.61.175

```
TASK [Gathering Facts] *********************************************************************************************
ok: [172.31.39.123]

TASK [prod task] ***************************************************************************************************
skipping: [172.31.39.123]

PLAY RECAP *********************************************************************************************************
172.31.39.123              : ok=1    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
172.31.47.213              : 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@ip-172-31-61-175:/etc/ansible$ ansible-playbook install.yaml
```

Now we are read to install Jenkins and java and docker

Jenkins, Java and Docker are installed successfully in master instance

```
ubuntu@ip-172-31-61-175:/etc/ansible$ jenkins --version
2.432
ubuntu@ip-172-31-61-175:/etc/ansible$ java --version
openjdk 11.0.20.1 2023-08-24
OpenJDK Runtime Environment (build 11.0.20.1+1-post-Ubuntu-0ubuntu120.04)
OpenJDK 64-Bit Server VM (build 11.0.20.1+1-post-Ubuntu-0ubuntu120.04, mixed mode, sharing)
ubuntu@ip-172-31-61-175:/etc/ansible$ docker --version
Docker version 24.0.5, build 24.0.5-0ubuntu1~20.04.1
ubuntu@ip-172-31-61-175:/etc/ansible$
```

i-0af39002256053c86 (master-madhavi)

PublicIPs: 54.174.227.179 PrivateIPs: 172.31.61.175

Java and Docker are installed successfully in test instance

```
ubuntu@ip-172-31-47-213:~/.ssh$ sudo nano authorized_keys
ubuntu@ip-172-31-47-213:~/.ssh$ sudo nano authorized_keys
ubuntu@ip-172-31-47-213:~/.ssh$ java --version
openjdk 11.0.20.1 2023-08-24
OpenJDK Runtime Environment (build 11.0.20.1+1-post-Ubuntu-0ubuntu120.04)
OpenJDK 64-Bit Server VM (build 11.0.20.1+1-post-Ubuntu-0ubuntu120.04, mixed mode, sharing)
ubuntu@ip-172-31-47-213:~/.ssh$ docker --version
Docker version 24.0.5, build 24.0.5-0ubuntu1~20.04.1
ubuntu@ip-172-31-47-213:~/.ssh$
```

i-083540db8ca35470a (test-madhavi)

PublicIPs: 3.84.142.221 PrivateIPs: 172.31.47.213

Java and docker are installed successfully in prod instance

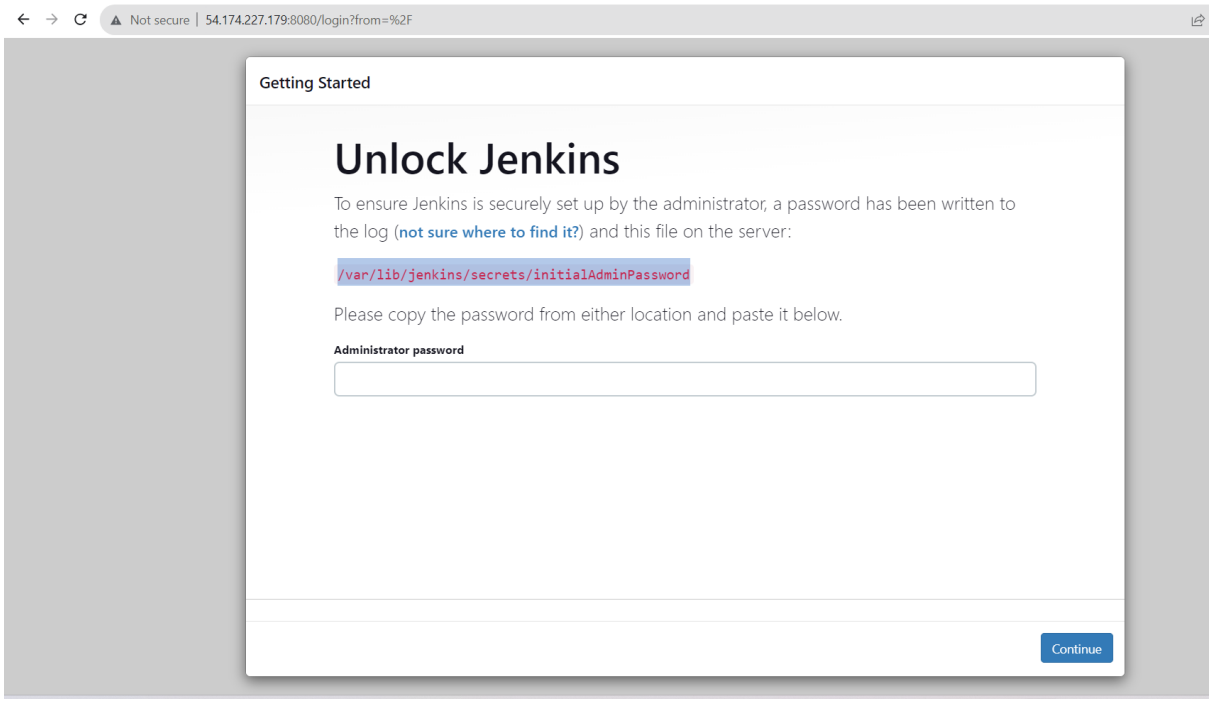
```
ubuntu@ip-172-31-39-123:~/.ssh$ java --version
openjdk 11.0.20.1 2023-08-24
OpenJDK Runtime Environment (build 11.0.20.1+1-post-Ubuntu-0ubuntu120.04)
OpenJDK 64-Bit Server VM (build 11.0.20.1+1-post-Ubuntu-0ubuntu120.04, mixed mode, sharing)
ubuntu@ip-172-31-39-123:~/.ssh$ docker --version
Docker version 24.0.5, build 24.0.5-0ubuntu1~20.04.1
ubuntu@ip-172-31-39-123:~/.ssh$
```

i-0a231a03d5d748f21 (prod-madhavi)

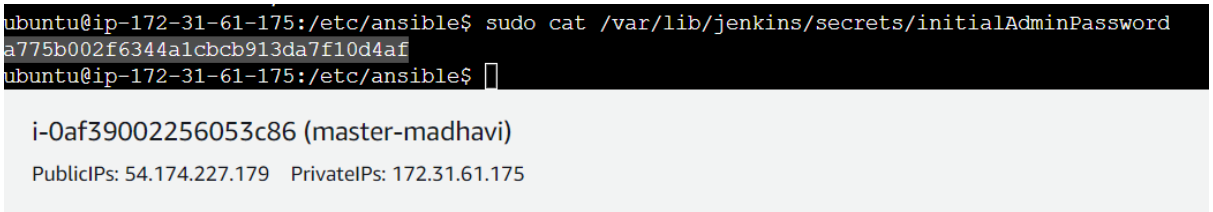
PublicIPs: 54.80.225.230 PrivateIPs: 172.31.39.123

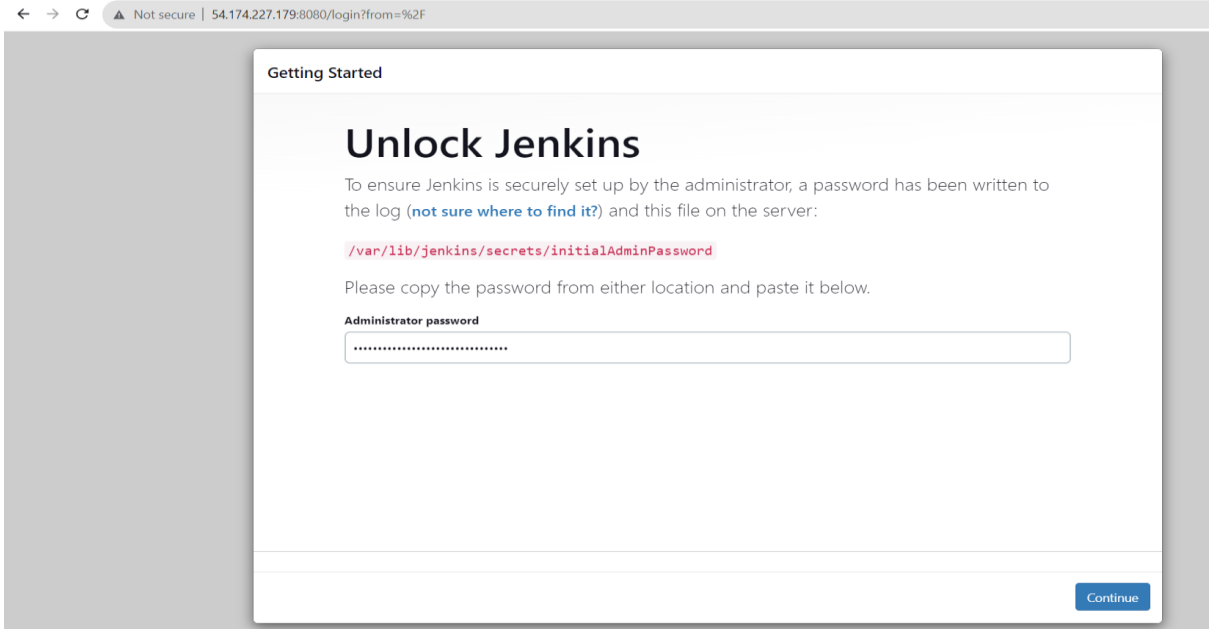
Open the public Ip of master instance with 8080 port

we will find a Jenkins browser

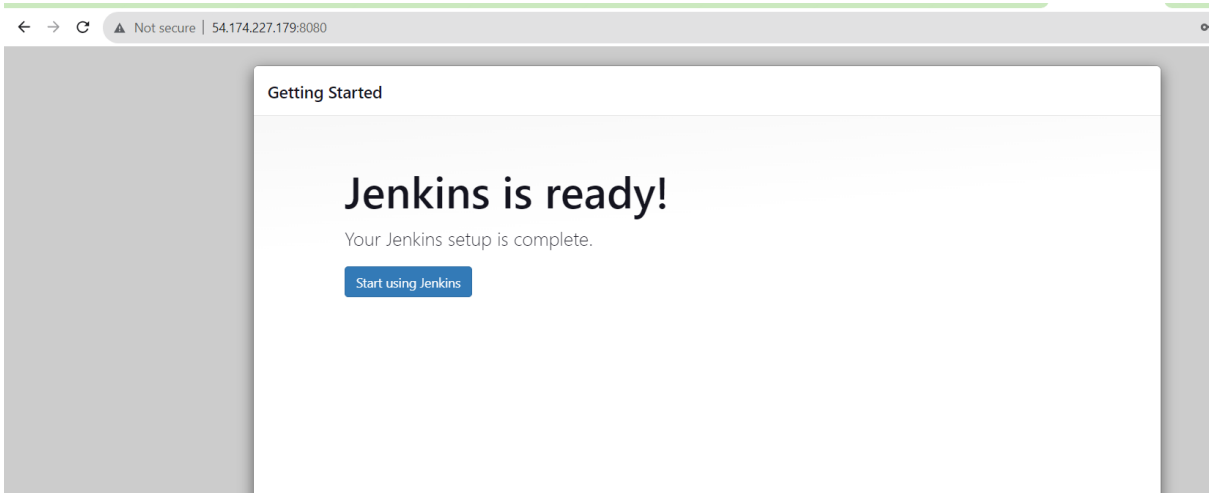


Copy the path and paste in instance we will get the password and paste this password in Jenkins browser



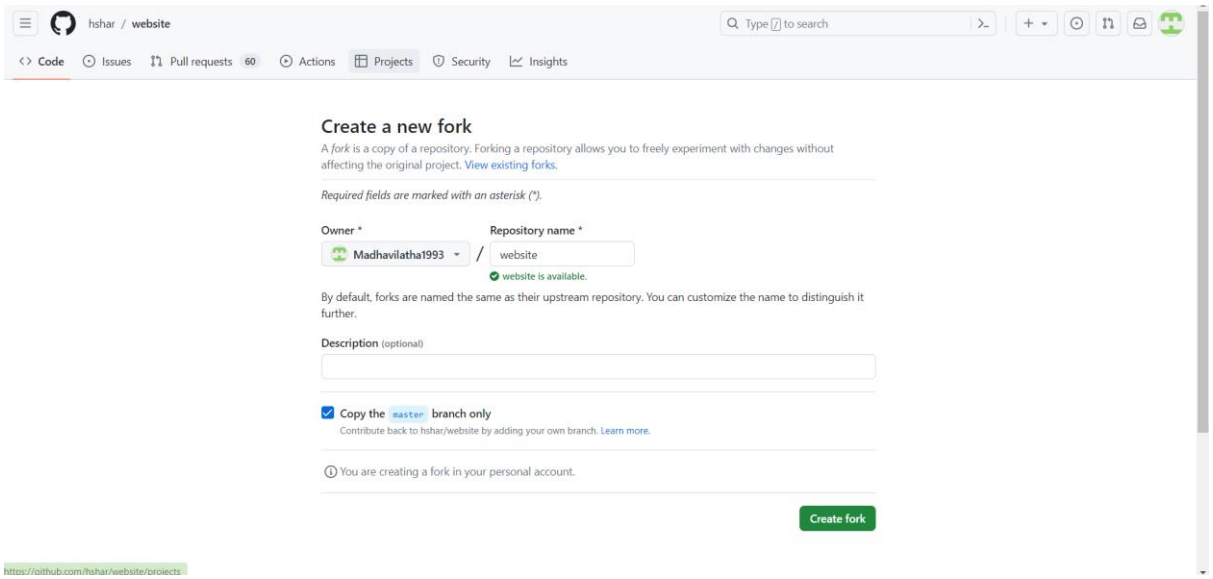


Give the required credentials like user name , password, email. By clicking on save and finish our Jenkins is ready



Now we have asked to clone the data from github repository
https://github.com/hshar/website

We will fork and get sign into our git hub and create a new fork



Copy the code of repository and clone in the master instance

```
ubuntu@ip-172-31-61-175:/etc/ansible$ cd
ubuntu@ip-172-31-61-175:~$ git clone https://github.com/Madhavilatha1993/website.git
Cloning into 'website'...
remote: Enumerating objects: 8, done.
remote: Total 8 (delta 0), reused 0 (delta 0), pack-reused 8
Unpacking objects: 100% (8/8), 82.67 KiB | 10.33 MiB/s, done.
ubuntu@ip-172-31-61-175:~$ ls
website
ubuntu@ip-172-31-61-175:~$ cd website
ubuntu@ip-172-31-61-175:~/website$ ls
images  index.html
ubuntu@ip-172-31-61-175:~/website$
```

i-0af39002256053c86 (master-madhavi)

PublicIPs: 54.174.227.179 PrivateIPs: 172.31.61.175

We are creating a docker file

```
ubuntu@ip-172-31-61-175:~/website$ sudo nano Dockerfile
ubuntu@ip-172-31-61-175:~/website$ sudo cat Dockerfile
FROM ubuntu
RUN apt update
RUN apt install apache2 -y
ADD . /var/www/html
ENTRYPOINT apachectl -D FOREGROUND
ubuntu@ip-172-31-61-175:~/website$ git status
On branch master
Your branch is up to date with 'origin/master'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Dockerfile
```

i-0af39002256053c86 (master-madhavi)

PublicIPs: 54.174.227.179 PrivateIPs: 172.31.61.175

```
ubuntu@ip-172-31-61-175:~/website$ git add .
ubuntu@ip-172-31-61-175:~/website$ git commit -m "Hello"
[master b0303d2] Hello
 Committer: Ubuntu <ubuntu@ip-172-31-61-175.ec2.internal>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 1 file changed, 5 insertions(+)
 create mode 100644 Dockerfile
ubuntu@ip-172-31-61-175:~/website$ git branch
* master
ubuntu@ip-172-31-61-175:~/website$ git branch develop
ubuntu@ip-172-31-61-175:~/website$ git branch
  develop
* master
ubuntu@ip-172-31-61-175:~/website$
```

i-0af39002256053c86 (master-madhavi)

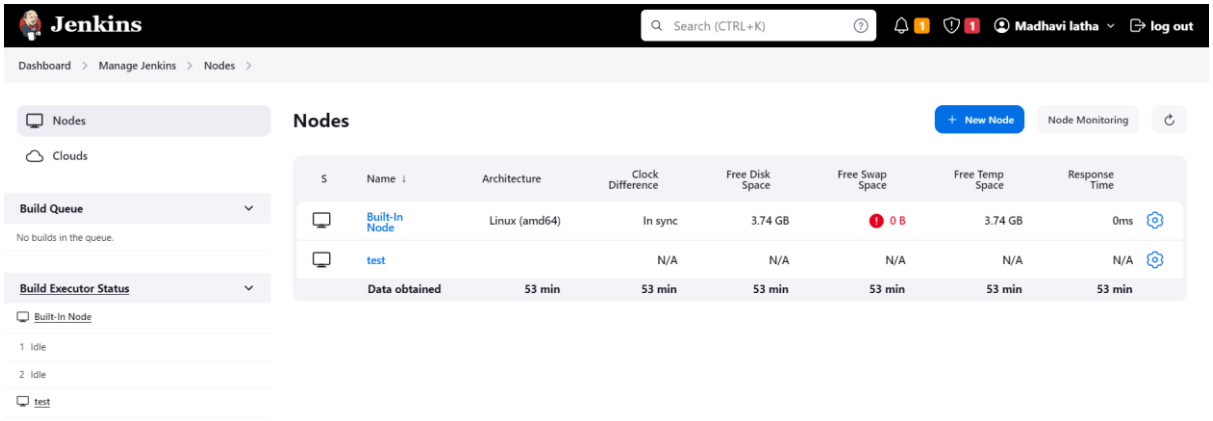
PublicIPs: 54.174.227.179 PrivateIPs: 172.31.61.175

Let us create master-slave architecture

In Jenkins dashboard→Manage Jenkins→nodes→new node→name, permanent agent→remote root directory(/home/ubuntu/Jenkins)→launch agents via ssh→host name(private IP address of test instance)→credentials→non verifying verification strategy→save

[In credentials→ssh username with private key→id as required→username ubuntu→enter key directly(paste the keypair of test instance→add]

Now test node is ready



Do the same process for the prod node.

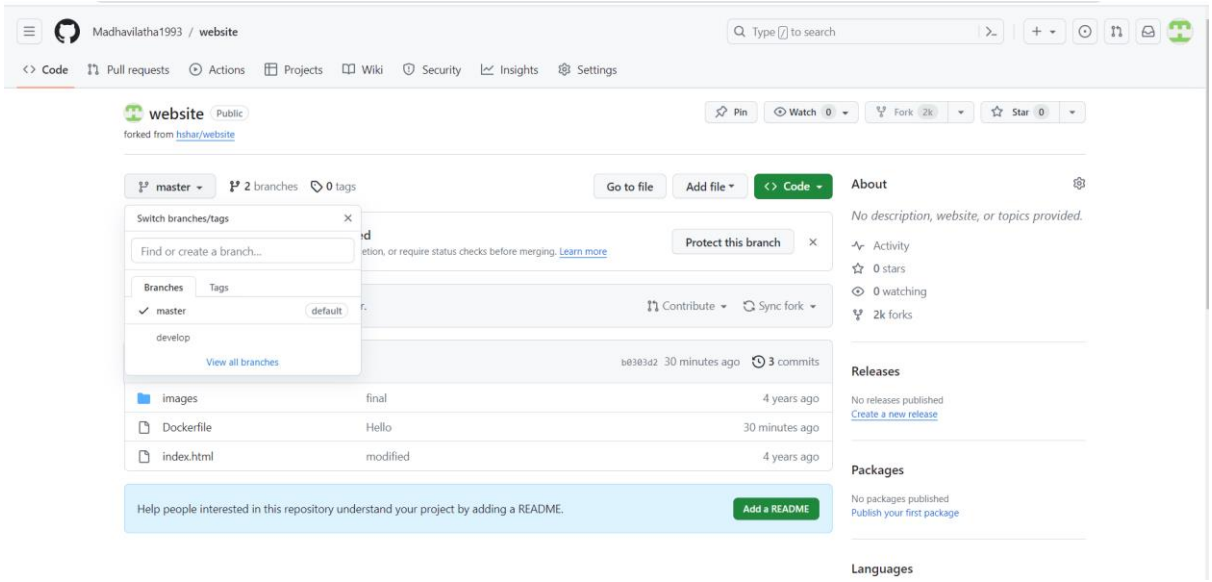
Give the private ip of prod instance in hostname

Now we are pushing the data to GitHub repository

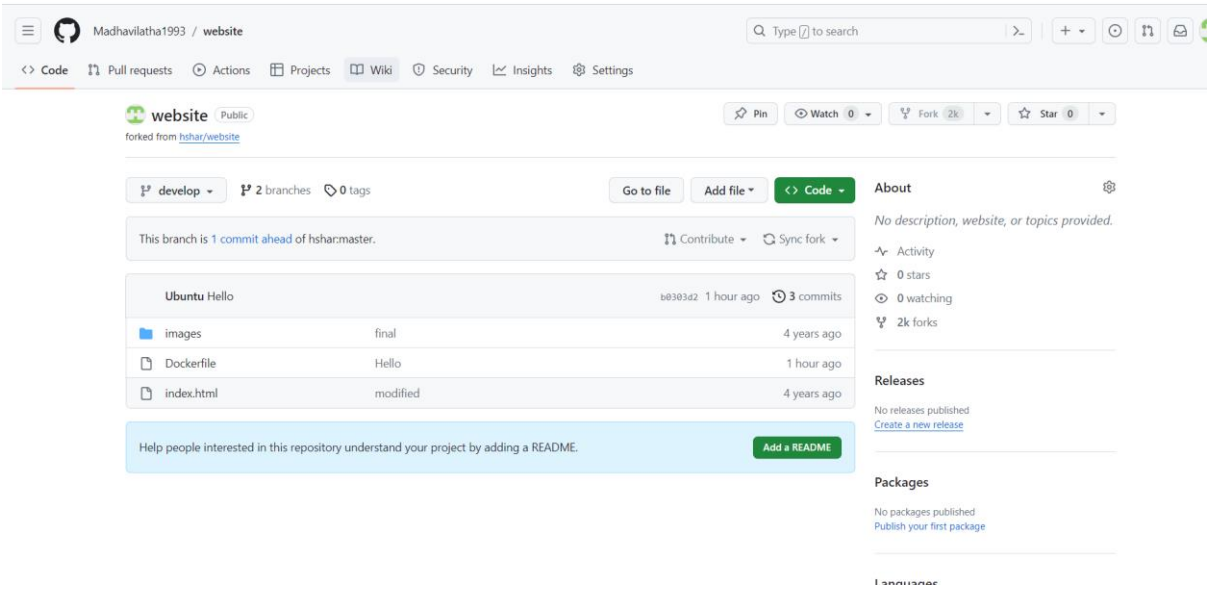
```
ubuntu@ip-172-31-61-175:~/website$ git push origin --all
Username for 'https://github.com': Madhavilatha1993
Password for 'https://Madhavilatha1993@github.com':
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 2 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 412 bytes | 412.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/Madhavilatha1993/website.git
   883b439..b0303d2  master -> master
 * [new branch]      develop -> develop
ubuntu@ip-172-31-61-175:~/website$
```

i-0af39002256053c86 (master-madhavi)

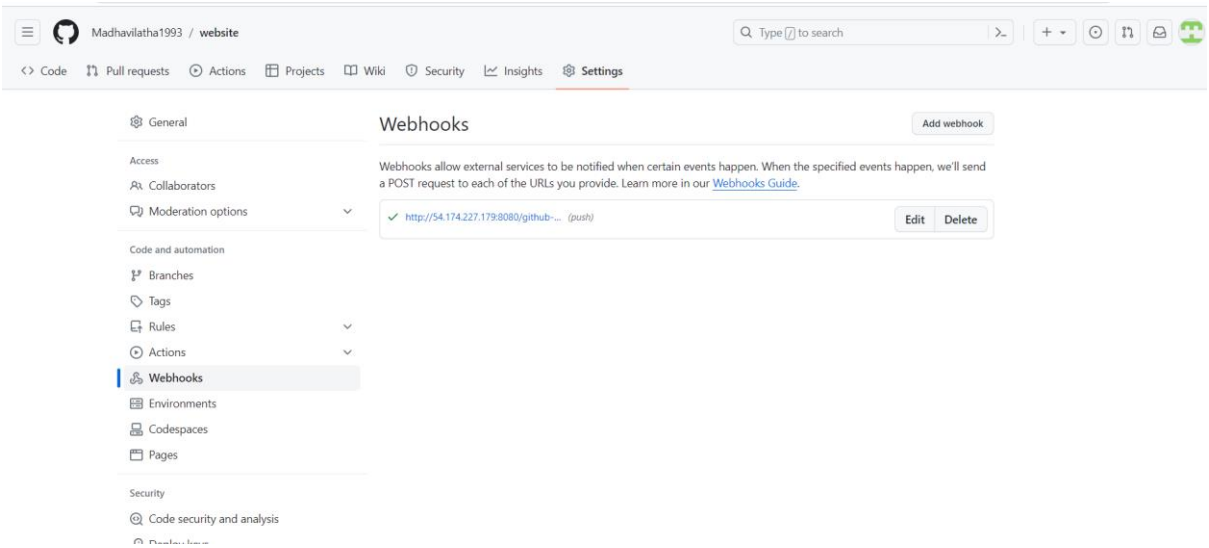
PublicIPs: 54.174.227.179 PrivateIPs: 172.31.61.175



Now we are in develop branch



Created a webhook



Now goto Jenkins dashboard→new item→job1→freestyle project→description as anything→select github project and copy code url→restrict where this project run to be label test → git →provide the url→branch develop→select github hook trigger→apply and save

Click on **Bulid now** to know the pipeline is running successfully

Status
Changes
Workspace
Build Now
Configure
Delete Project
GitHub Hook Log
GitHub
Rename

job1

Testing on develop branch with Dockerfile

Permalinks

Build History trend

Filter builds... /

#1
Nov 17, 2023, 8:39 AM

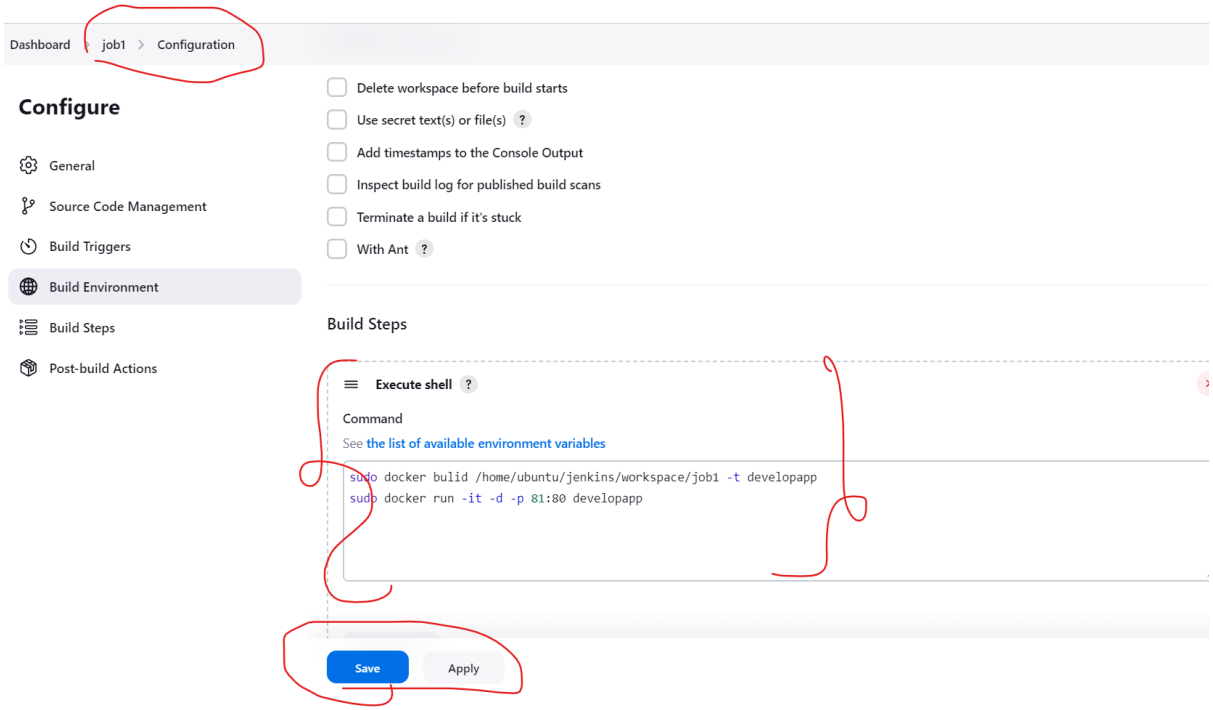
Atom feed for all Atom feed for failures

```
ubuntu@ip-172-31-47-213:~$ ls
jenkins
ubuntu@ip-172-31-47-213:~$ cd jenkins
ubuntu@ip-172-31-47-213:~/jenkins$ ls
remoting  remoting.jar  workspace
ubuntu@ip-172-31-47-213:~/jenkins$ cd workspace
ubuntu@ip-172-31-47-213:~/jenkins/workspace$ ls
job1
ubuntu@ip-172-31-47-213:~/jenkins/workspace$ cd job1
ubuntu@ip-172-31-47-213:~/jenkins/workspace/job1$ ls
Dockerfile  images  index.html
ubuntu@ip-172-31-47-213:~/jenkins/workspace/job1$
```

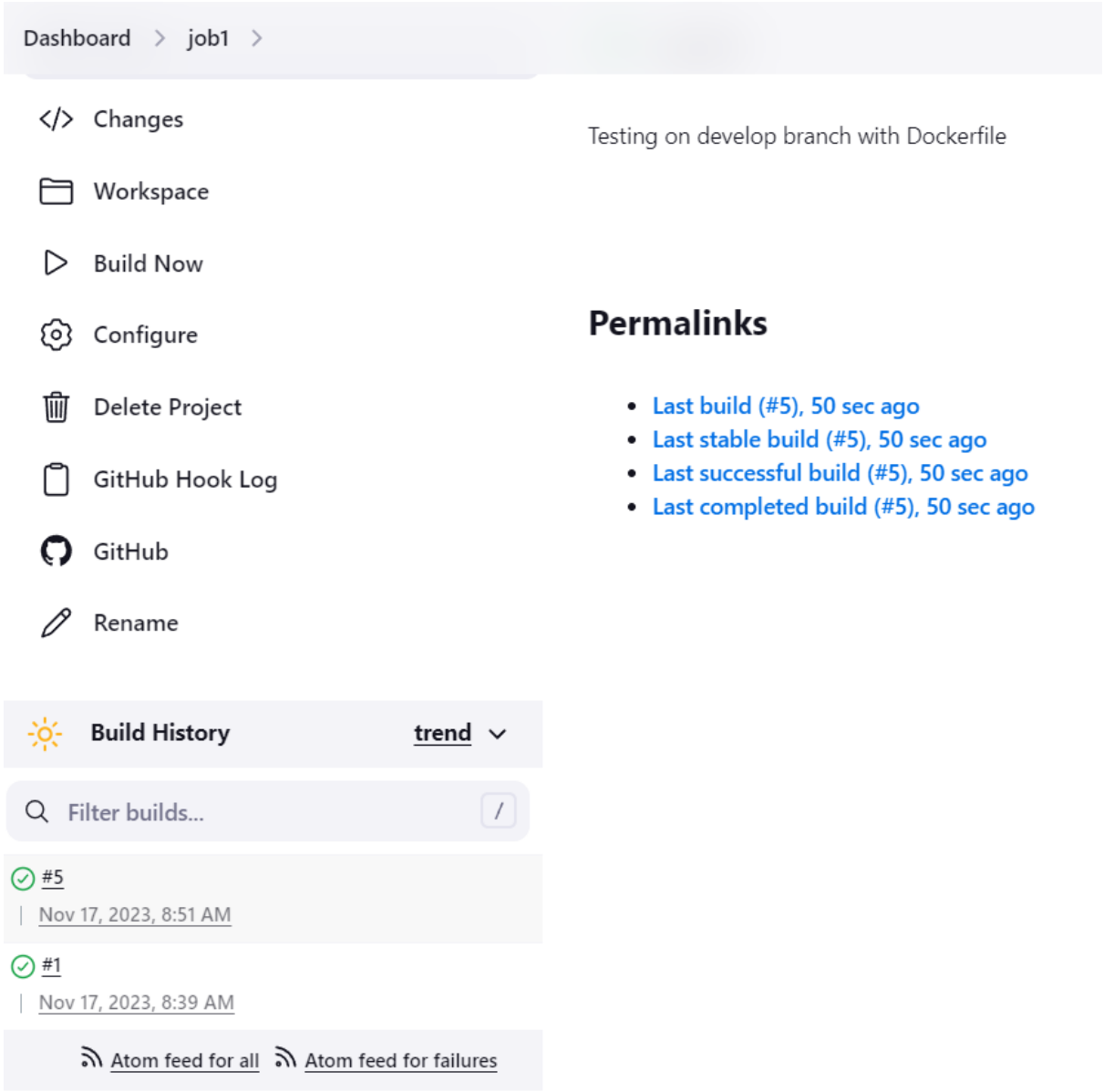
i-083540db8ca35470a (test-madhavi)

PublicIPs: 3.84.142.221 PrivateIPs: 172.31.47.213

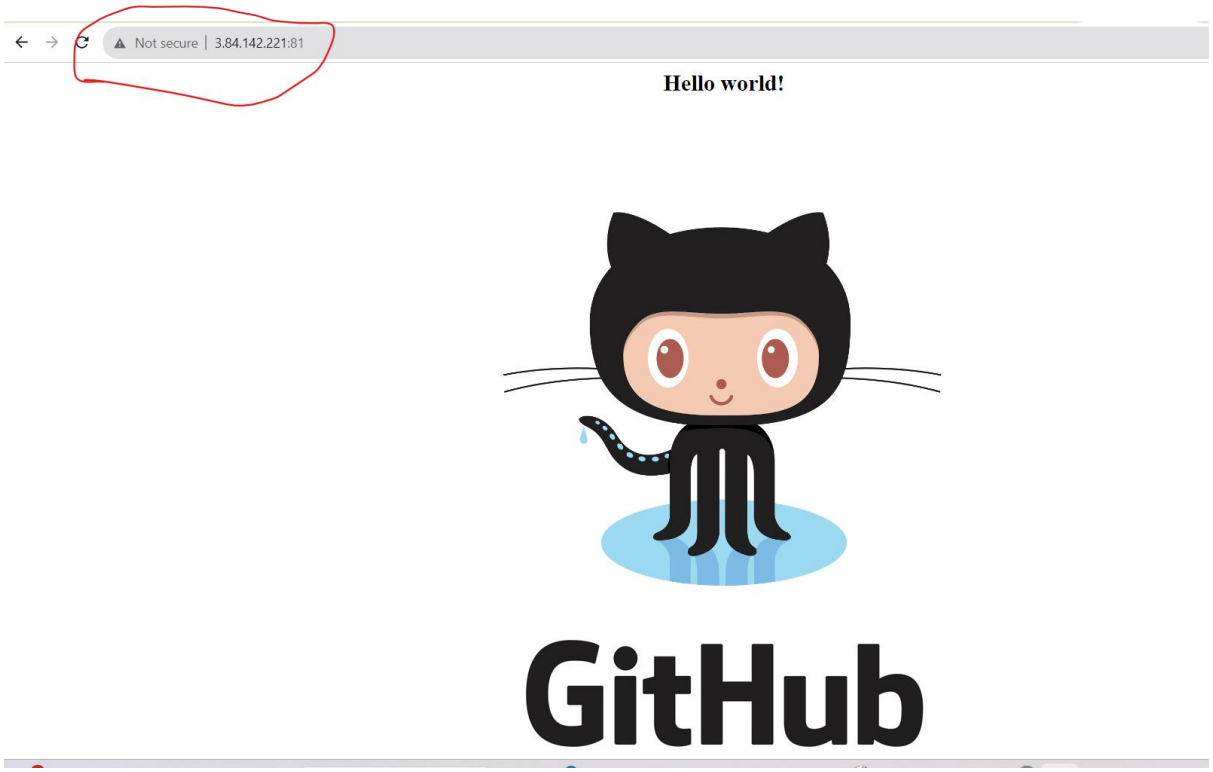
Now we will run the dockerfile



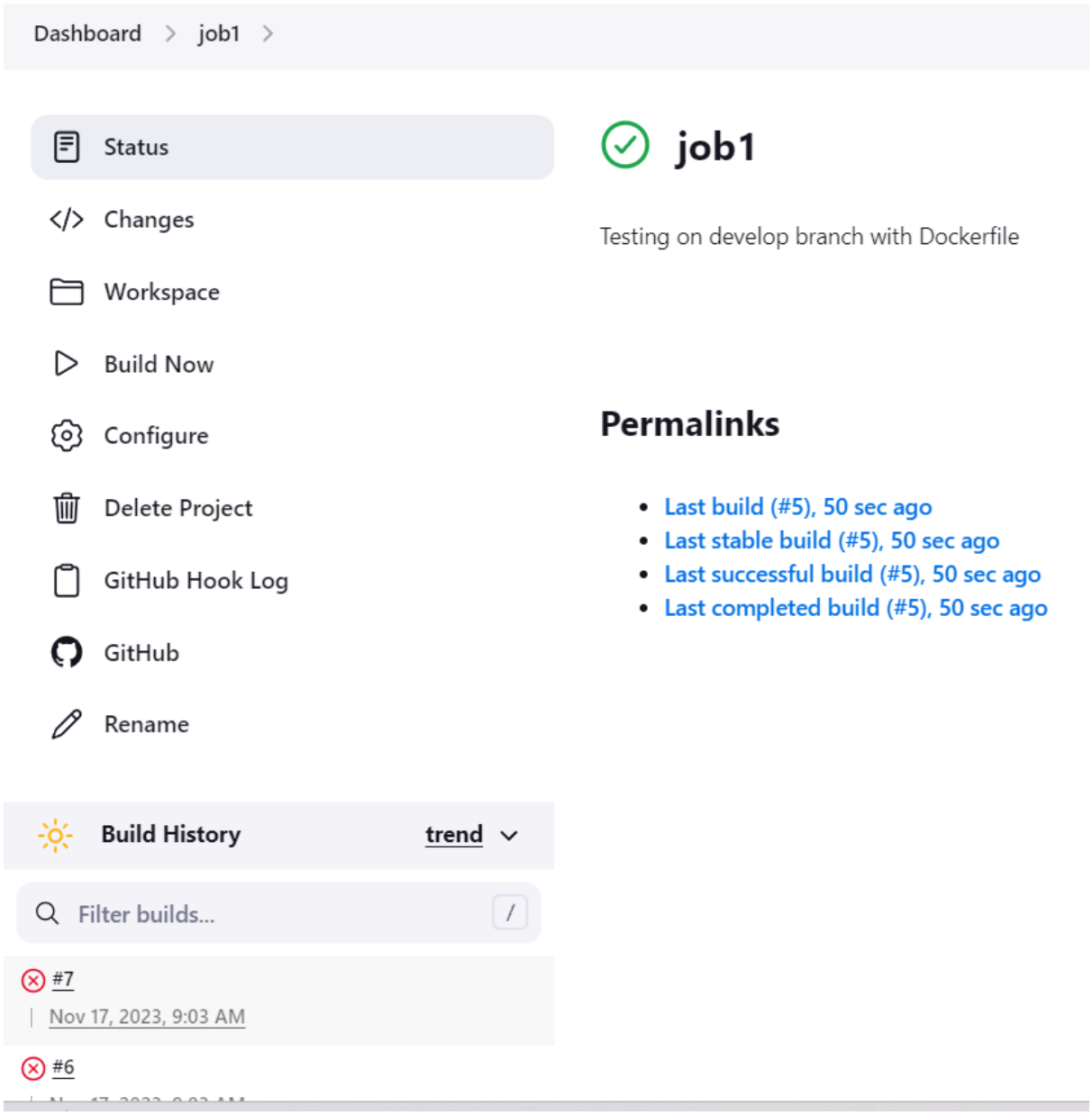
Now it was successfully executed



If we browse the public IP of test instance with port 81 the output will be as shown

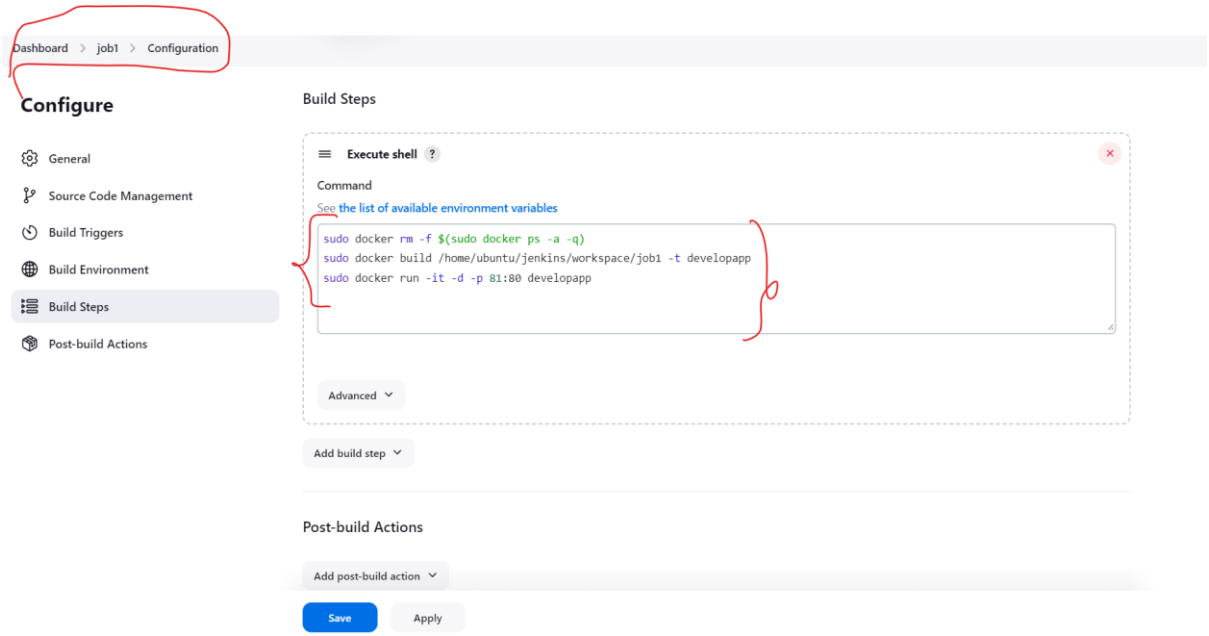


If we click on build now again and again we will get an error

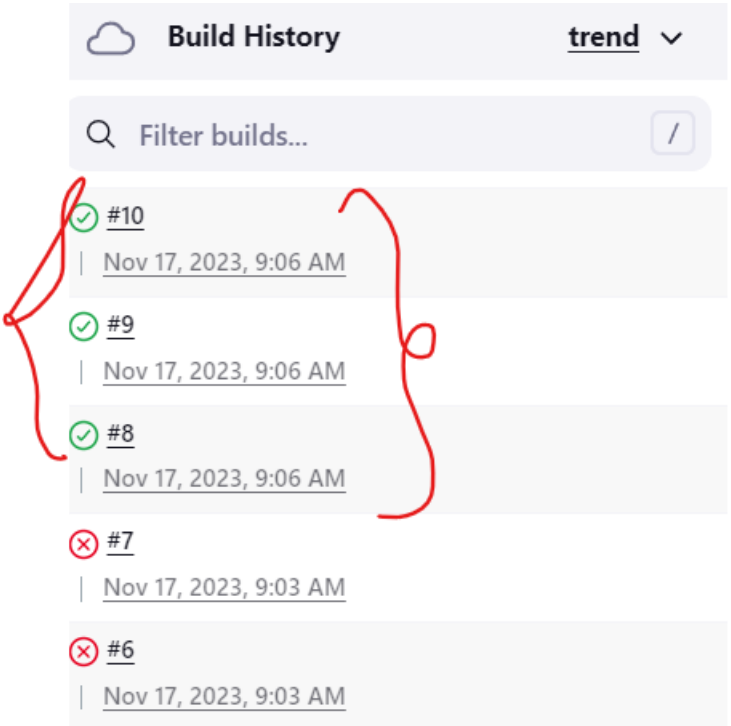

Dashboard
job1
Status
Changes
Workspace
Build Now
Configure
Delete Project
GitHub Hook Log
GitHub
Rename
job1
Testing on develop branch with Dockerfile
Permalinks
Last build (#5), 50 sec ago
Last stable build (#5), 50 sec ago
Last successful build (#5), 50 sec ago
Last completed build (#5), 50 sec ago
Build History
trend
Filter builds...
/
#7
Nov 17, 2023, 9:03 AM
#6

To resolve this error

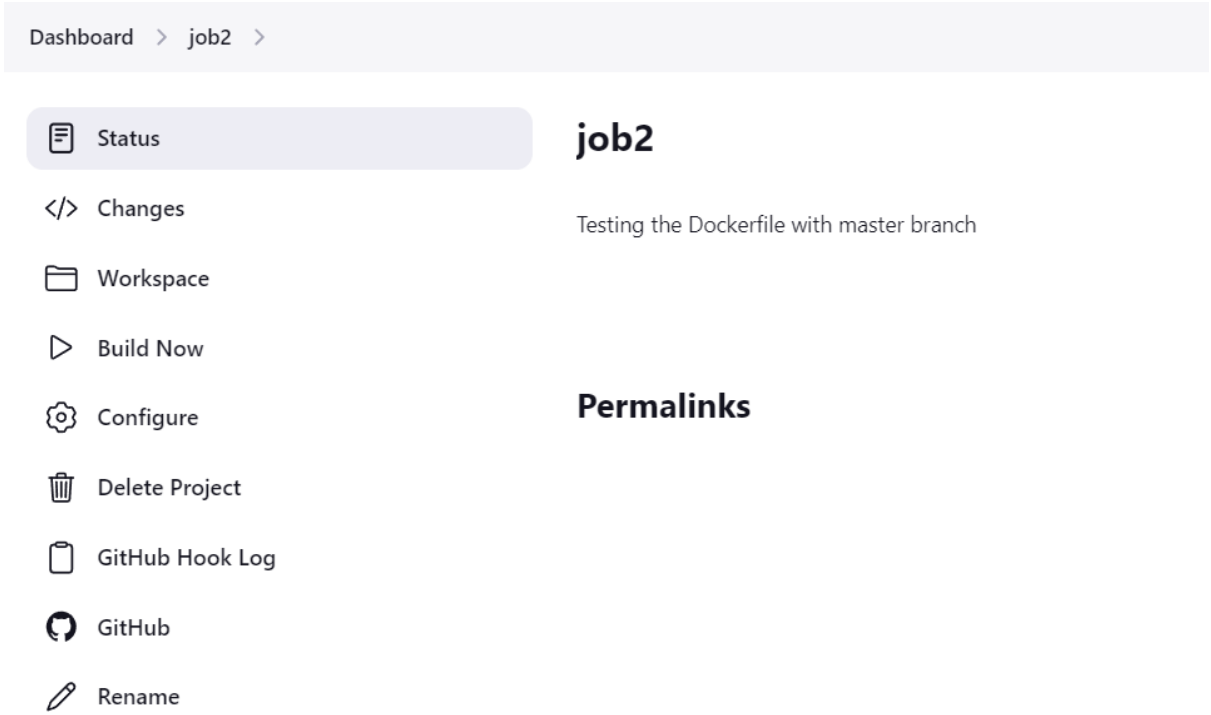
We have to give the docker container deletion command



Now it is successful



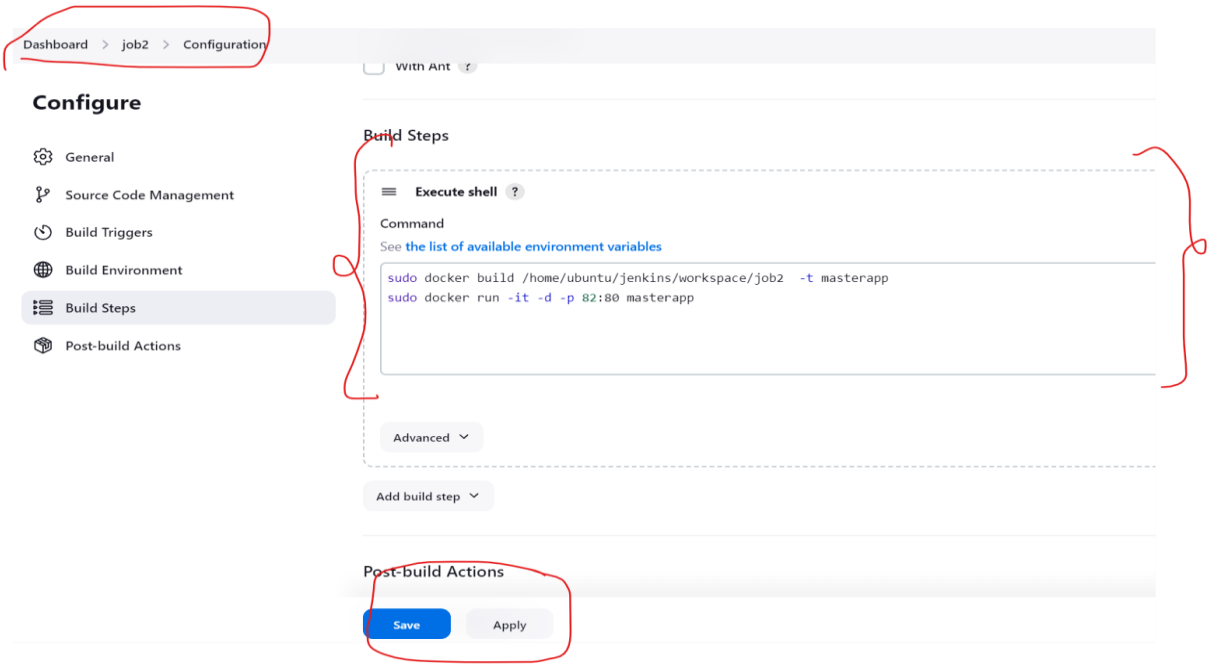
Now we are doing the same job executing the Dockerfile in master branch as job2.



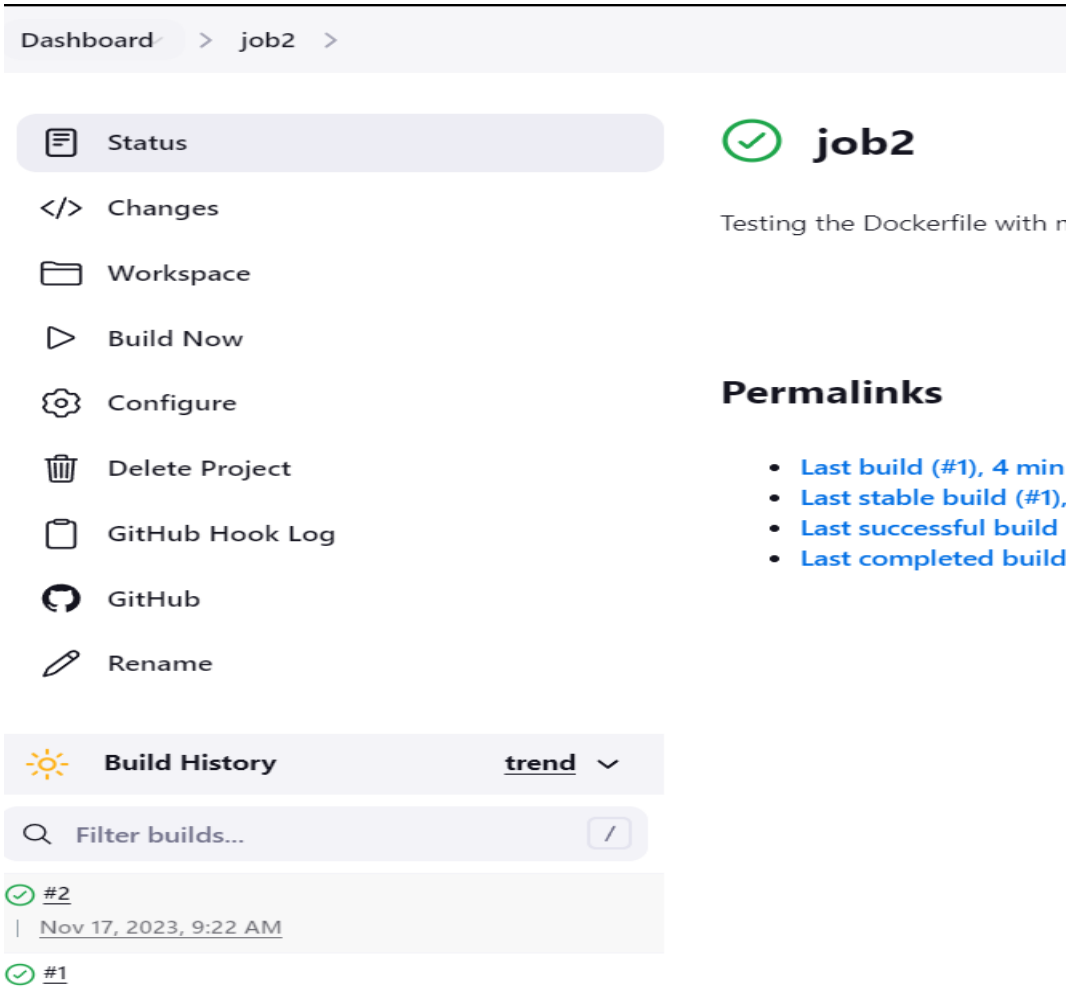

```
buntu@ip-172-31-47-213:~/jenkins/workspace/job1$ cd ..
buntu@ip-172-31-47-213:~/jenkins/workspace$ ls
ob1  job2
buntu@ip-172-31-47-213:~/jenkins/workspace$ cd job2
buntu@ip-172-31-47-213:~/jenkins/workspace/job2$ ls
ockerfile  images  index.html
buntu@ip-172-31-47-213:~/jenkins/workspace/job2$
```

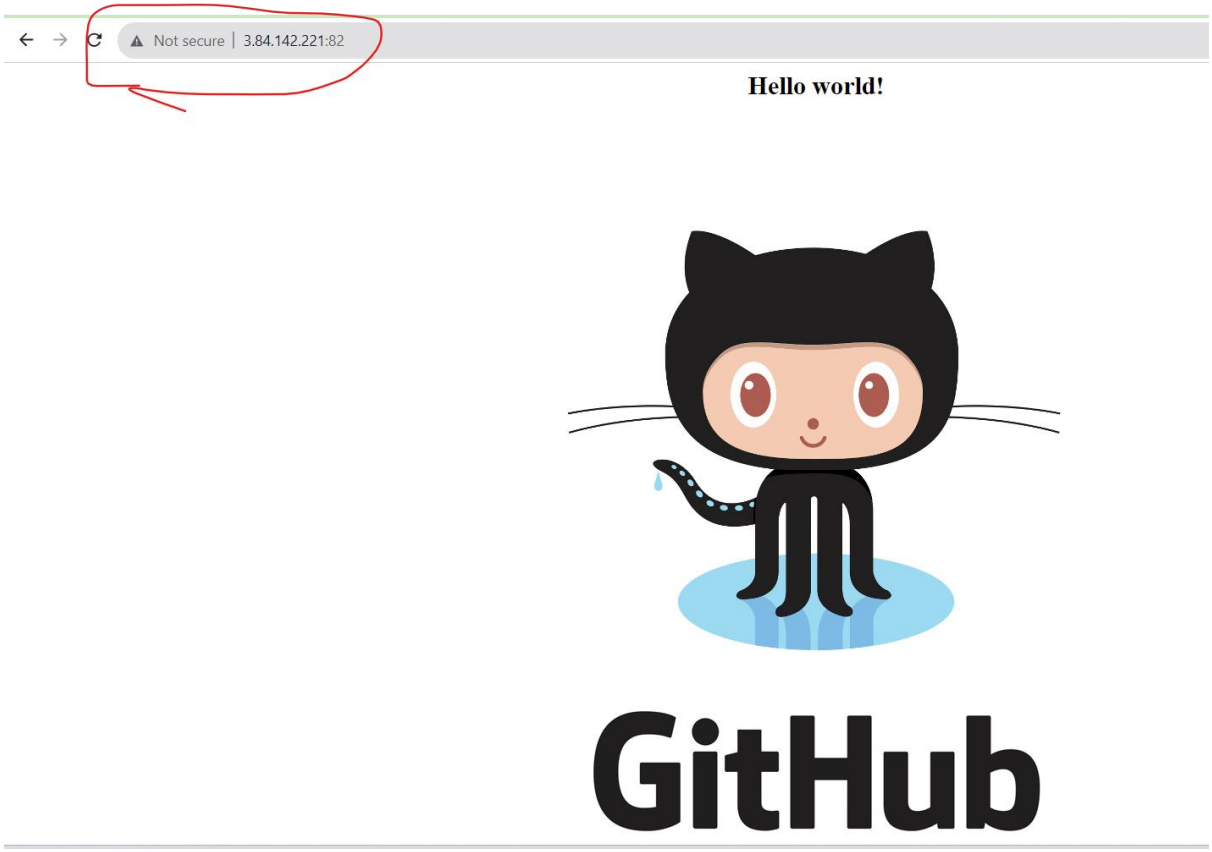
i-083540db8ca35470a (test-madhavi)

PublicIPs: 3.84.142.221 PrivateIPs: 172.31.47.213

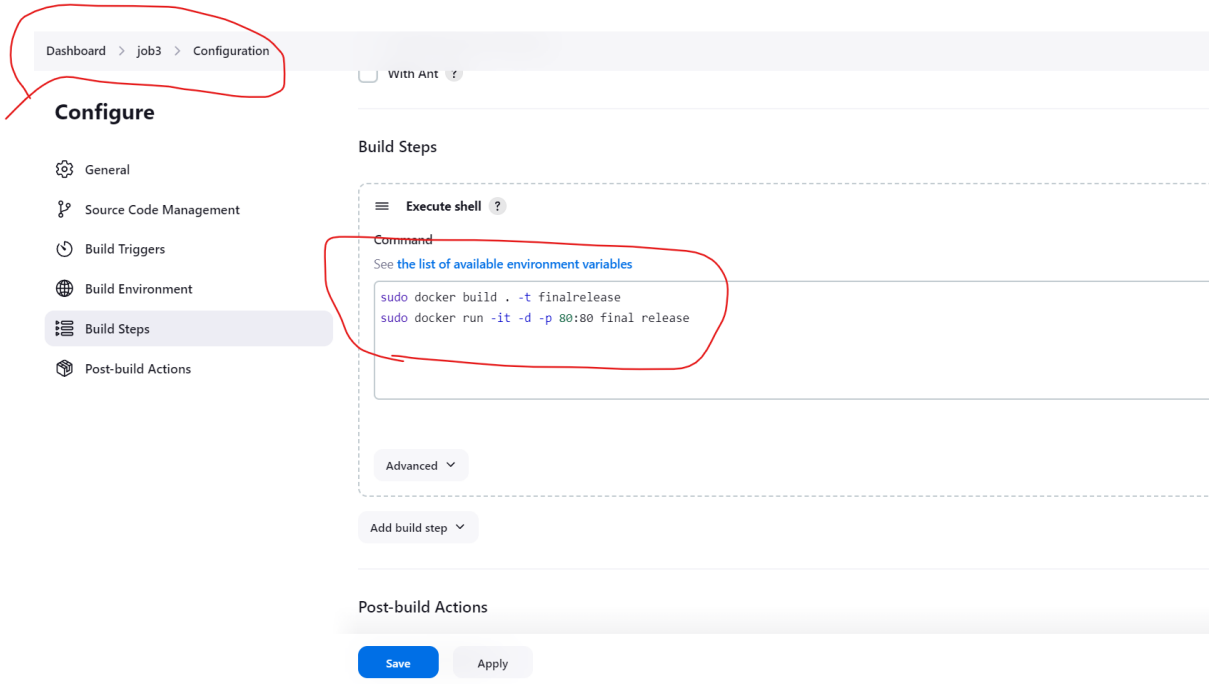


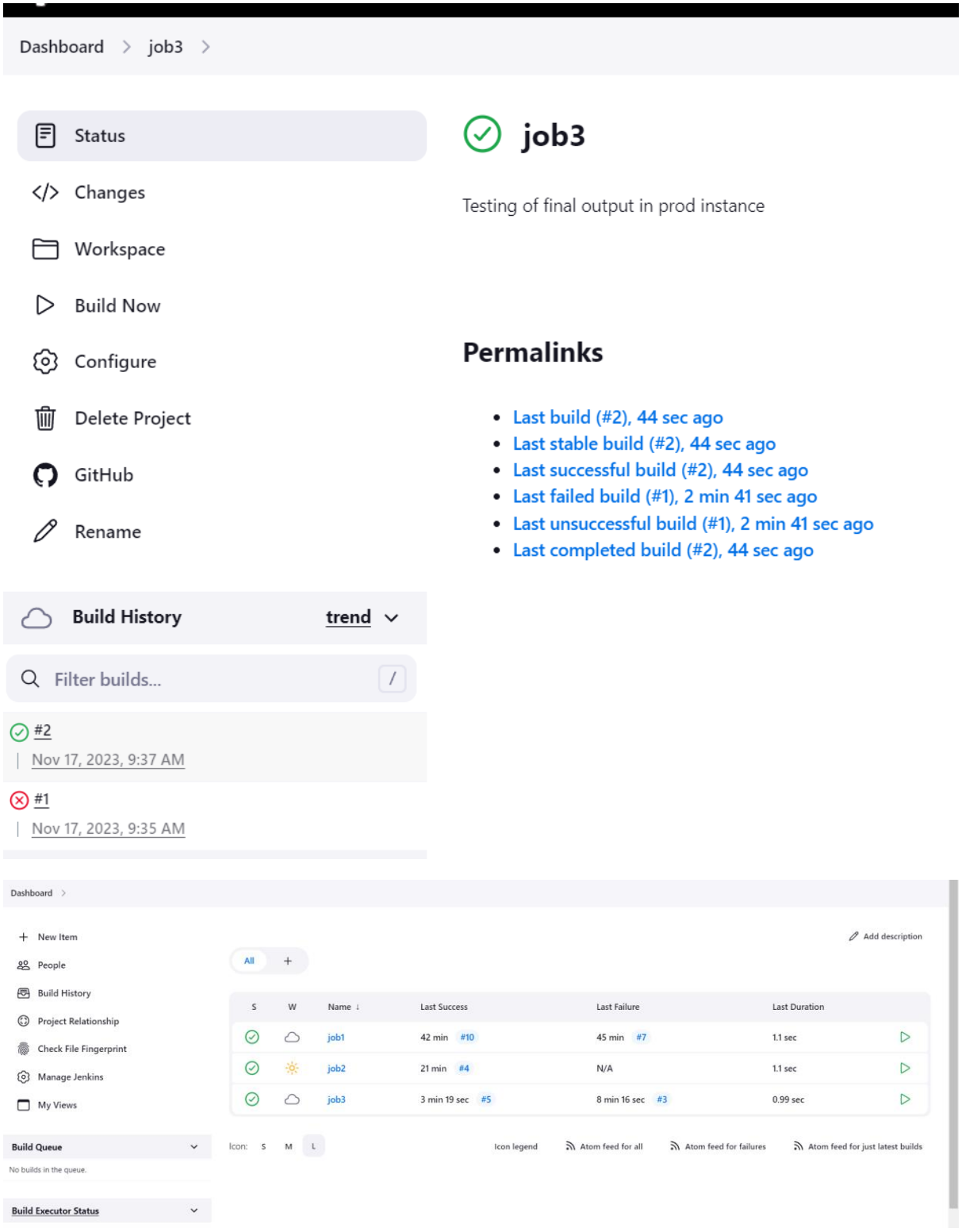
Now the file is executed successfully





Now for final output in prod instance we have created job3 in dashboard





Now as asked job3 must be executed only after job2 is executed

So change the post build actions in job2 with build other projects job3

After changes are made, in job3 click on Build now→ we will occur an error.

Again add the deletion container command to job3

Dashboard > job3 >

- Status
- Changes
- Workspace
- Build Now
- Configure
- Delete Project
- GitHub
- Rename

job3

Testing of final output in prod instance

Upstream Projects

job2

Permalinks

- Last build (#2), 3 min 43 sec ago
- Last stable build (#2), 3 min 43 sec ago
- Last successful build (#2), 3 min 43 sec ago
- Last failed build (#1), 5 min 39 sec ago
- Last unsuccessful build (#1), 5 min 39 sec ago
- Last completed build (#2), 3 min 43 sec ago

Build History trend

Filter builds... /

#3
Nov 17, 2023, 9:41 AM

#2
Nov 17, 2023, 9:37 AM

#1

Dashboard > job3 > Configuration

Configure

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

Build Steps

Execute shell ?

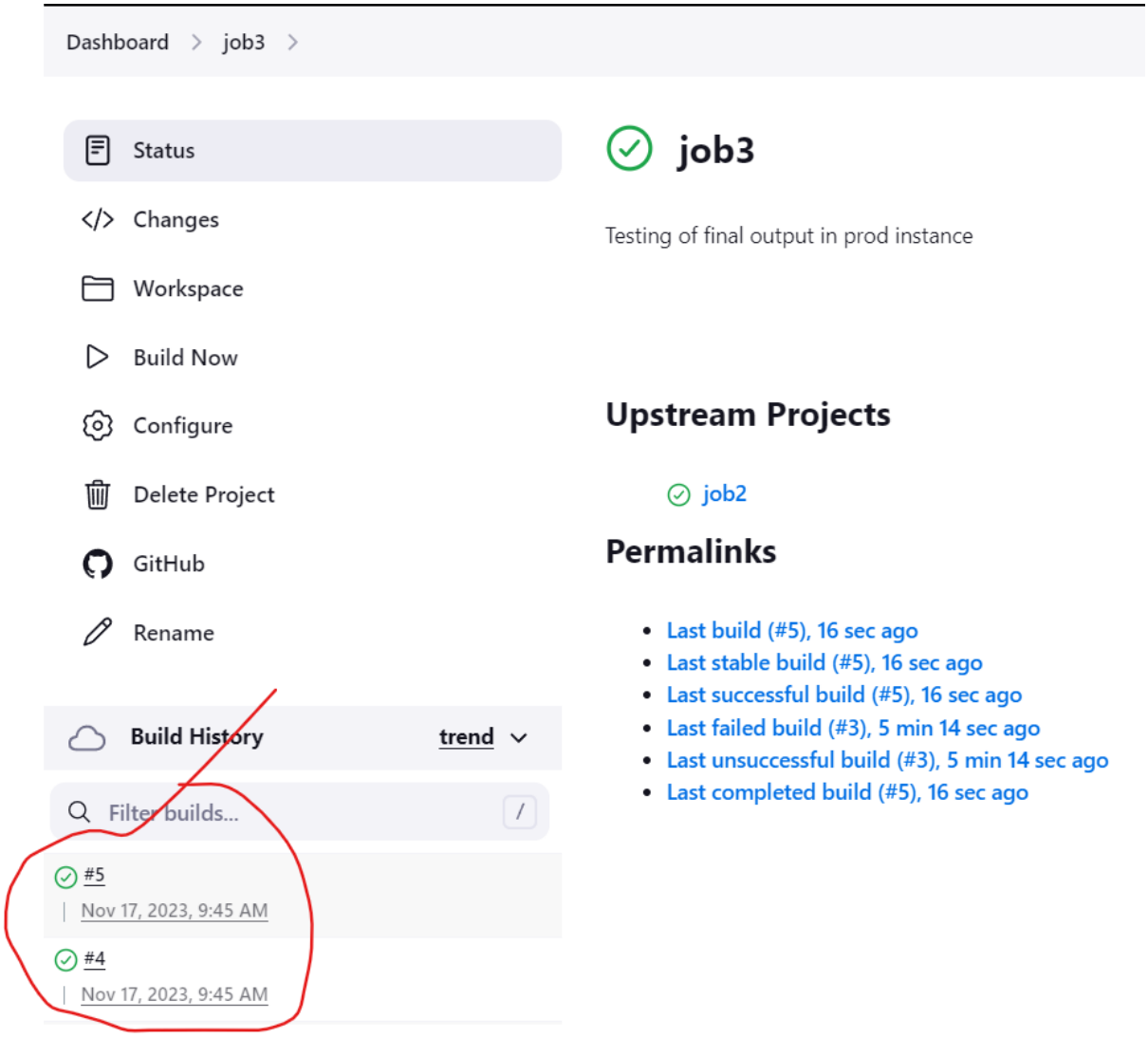
Command

See the list of available environment variables

```
sudo docker rm -f $(sudo docker ps -a -q)
sudo docker build . -t finalrelease
sudo docker run -it -d -p 80:80 finalrelease
```

Advanced

Now it was successful



Prod instance Public IP in web browser