## Create a Complete CI/CD from Development to Testing to Production using DevOps Tools

You have been Hired Sr. DevOps Engineer in Abode Software. They want to implement DevOpsLifecycle 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 bebuilt 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**

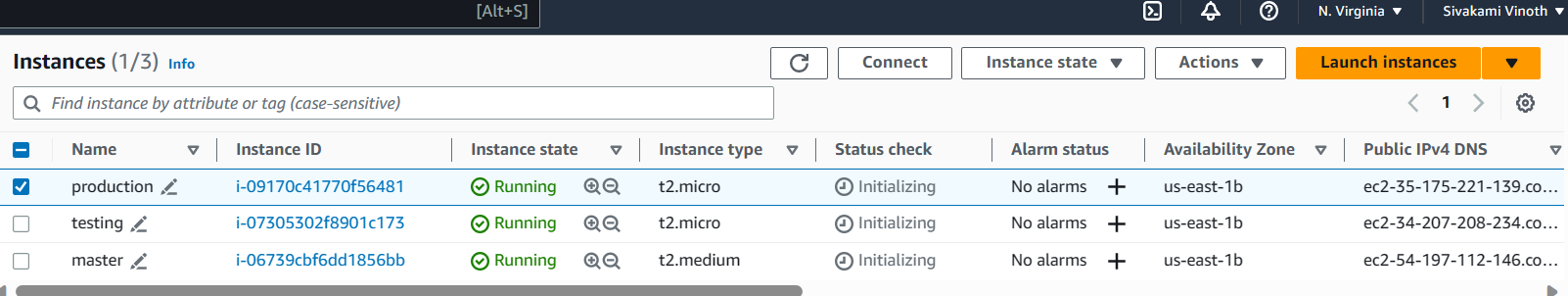
**Job2: test**

**Job3 : prod**

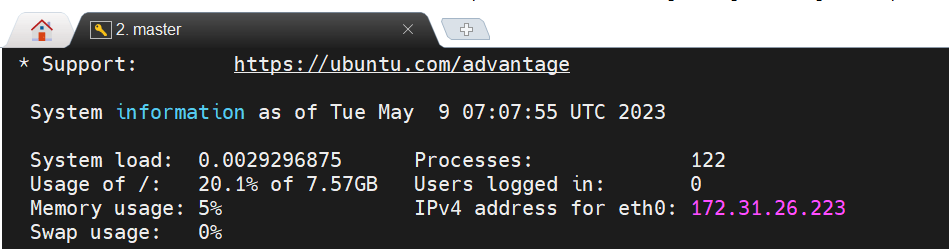
**Step:1 Create 3 EC2 Instances (master, testing and production)**

**Master: OS-> ubuntu, Instance type-> t2.medium**

**Testing & production: OS-> ubuntu, Instance type-> t2.micro**

****

**Step:2 Connect to the master machine via mobaxterm**

****

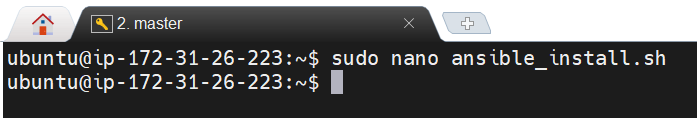
**Step: 3 Install Ansible -> configuration management tool**

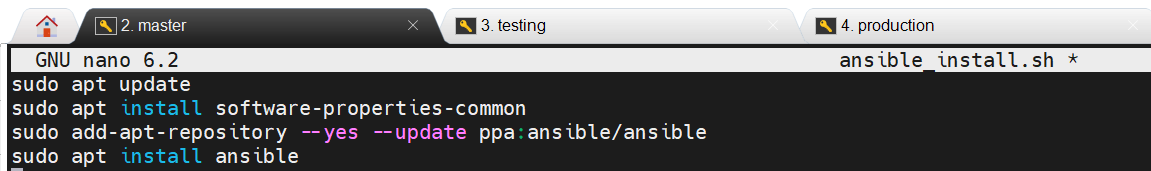
**Open the link->** [Installing Ansible on specific operating systems — Ansible Documentation](https://docs.ansible.com/ansible/latest/installation_guide/installation_distros.html#installing-ansible-on-ubuntu)



**Copy the code and create a bash file**

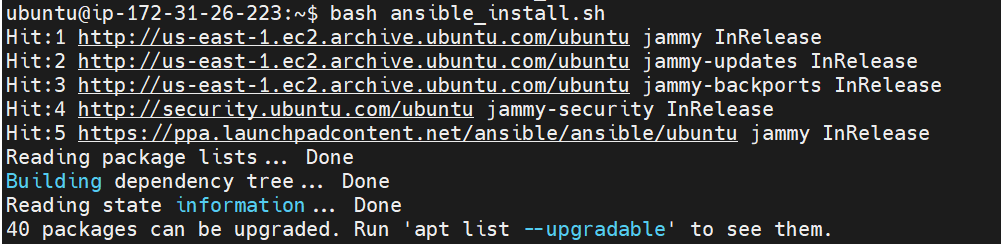
* **sudo nano ansible\_install.sh**

****

****

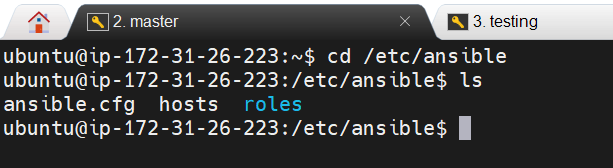
**Run the bash file**

* **bash ansible\_install.sh**

****

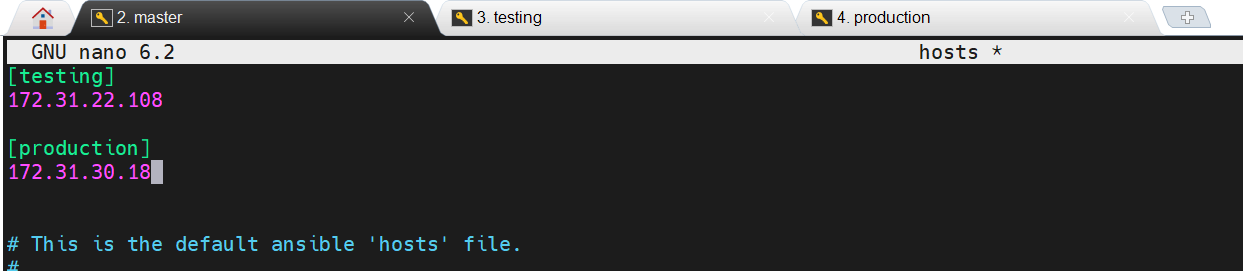
**Ansible installed successfully**

* **cd /etc/ansible**
* **ls**

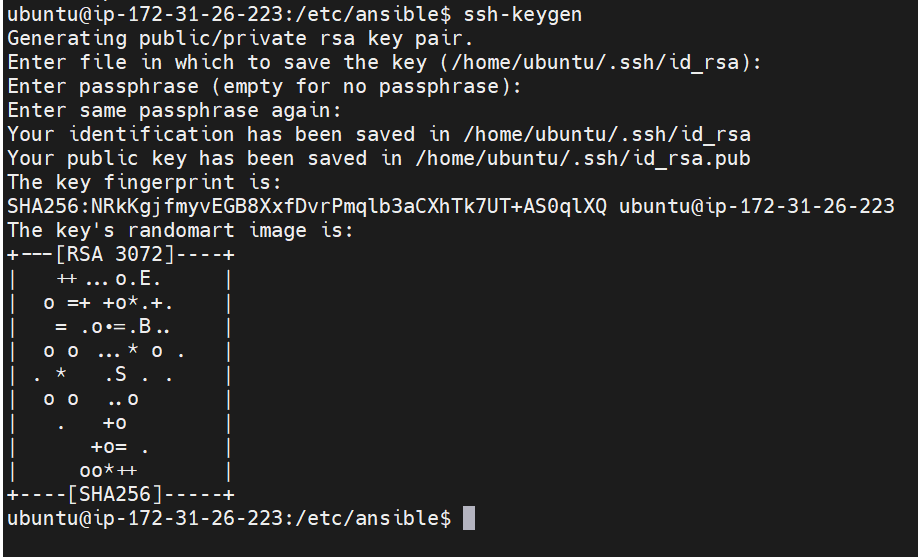
****

**Step:4 open the hosts file and add testing and production machines**

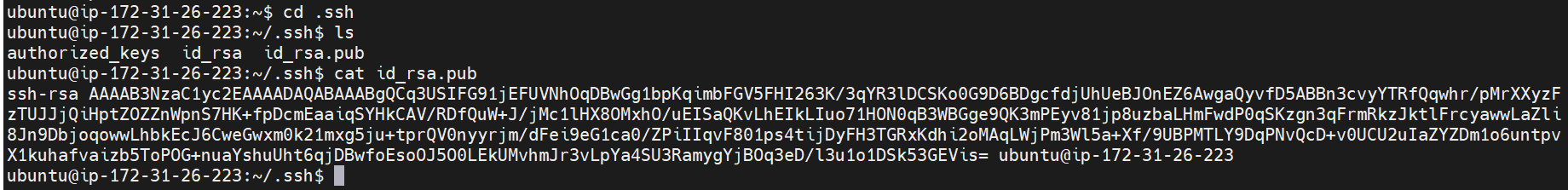
**Copy the private ip of testing and production machine and make the changes to hosts file as follow**

****

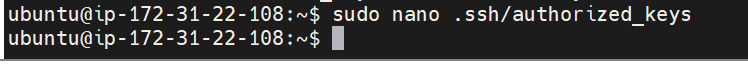
**Step:5 Generate a keypair to establish connection between master, testing and production machine**

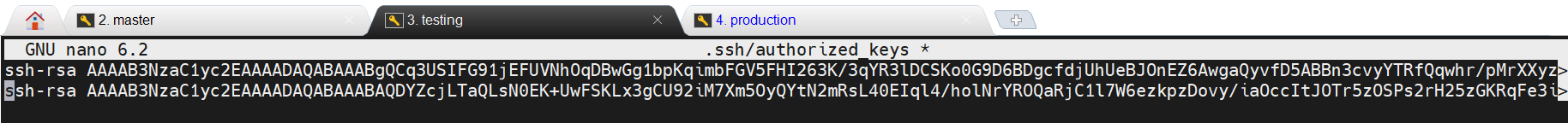
****

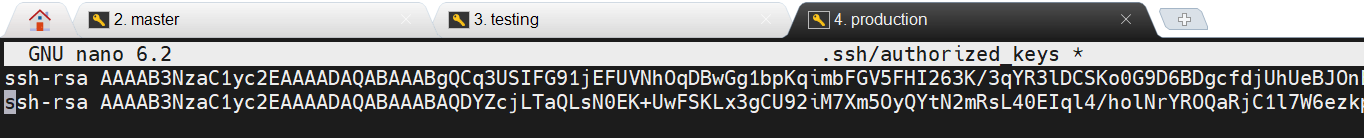
**Copy the public key generated**

****

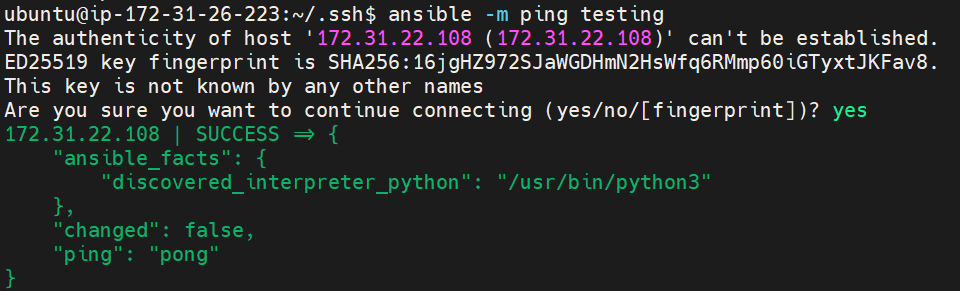
**Paste the key in .ssh/authorized\_keys file from testing and production machine**

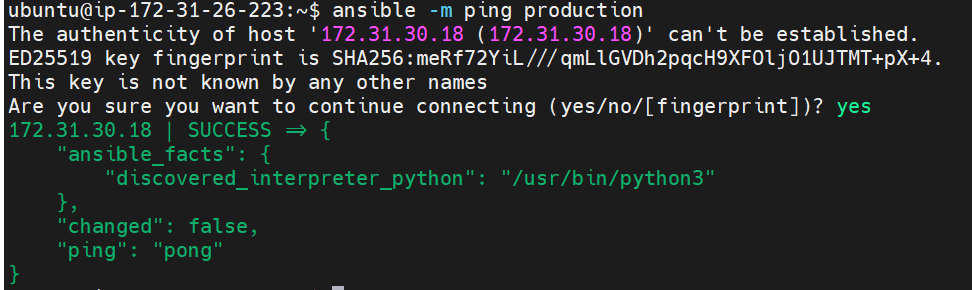
****

****

****

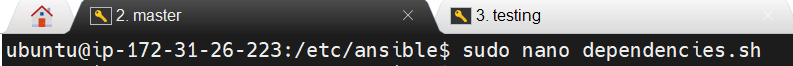
**Now we can able to ping testing and production machine from master**

****

****

**Step: 6 Install java, docker and jenkins**

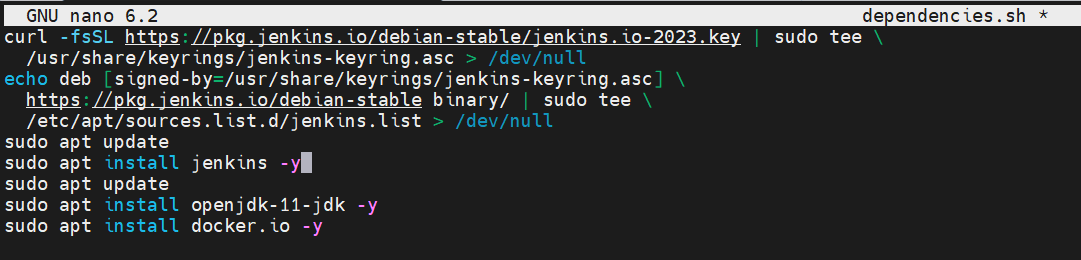
**Create two bash files dependencies.sh (to install jenkins, java and docker in master machine) and dependencies1.sh(to install java and docker in testing and production machines)**

****

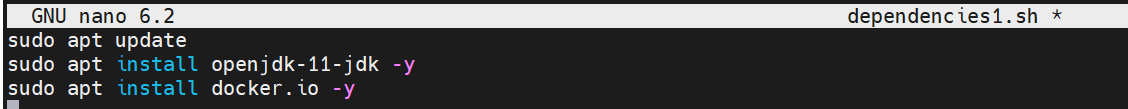
**To install jenkins open the URL** [Linux (jenkins.io)](https://www.jenkins.io/doc/book/installing/linux/) **from browser**

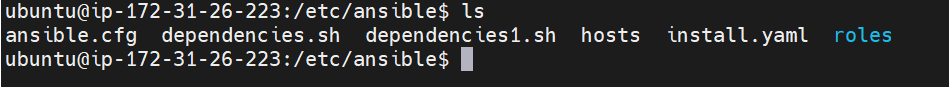
**Copy the commands and make the bash file**



****

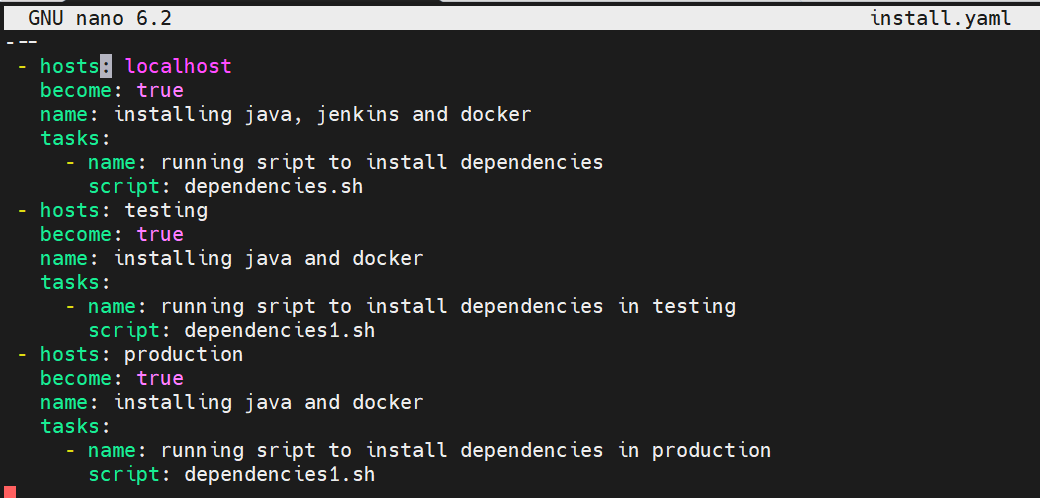
****

****

****

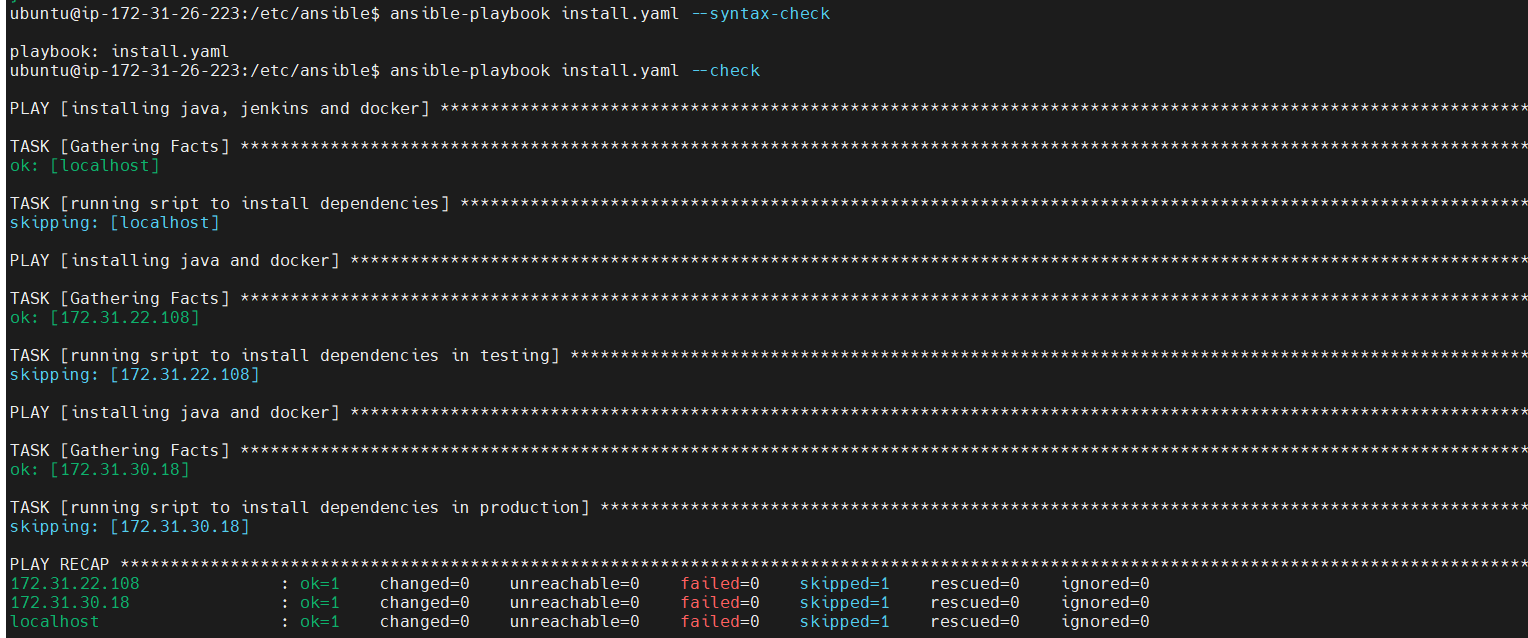
**Create a playbook install.yaml to run the bash files created**

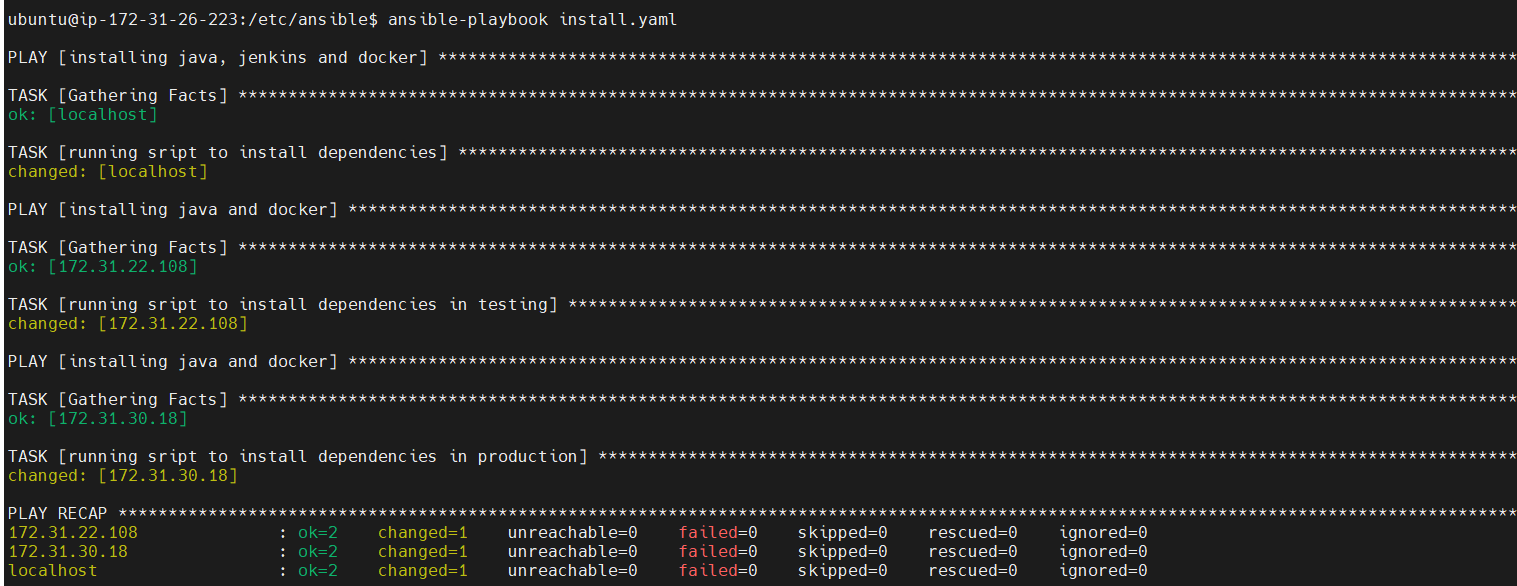
****

****

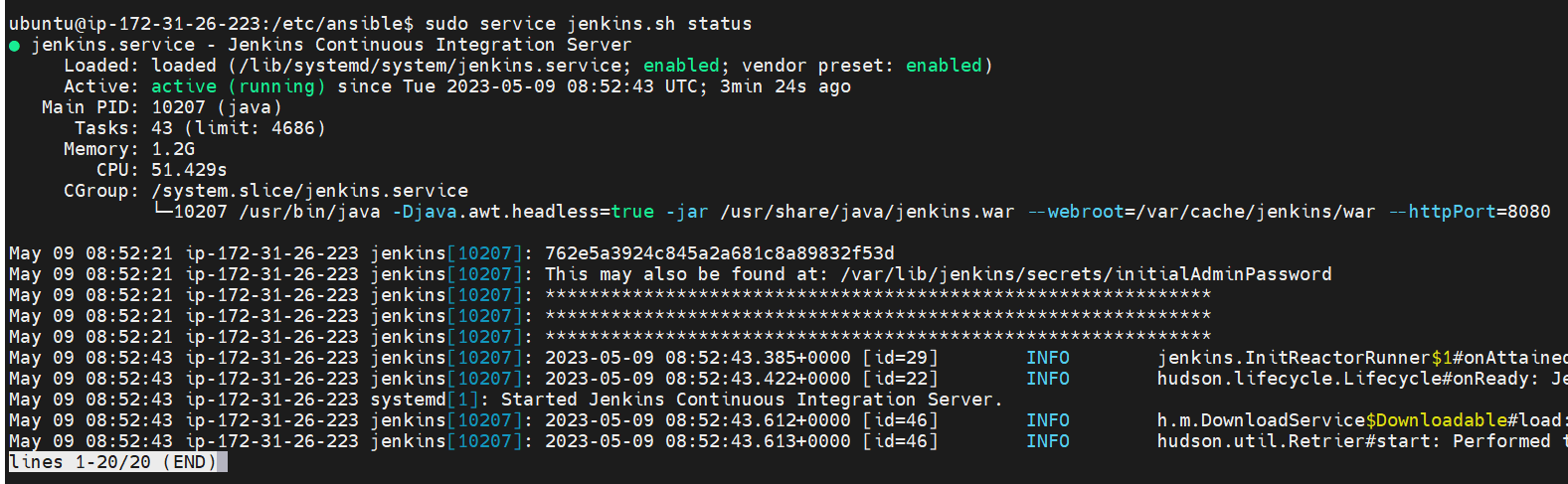
**Execute the playbook using following commands**

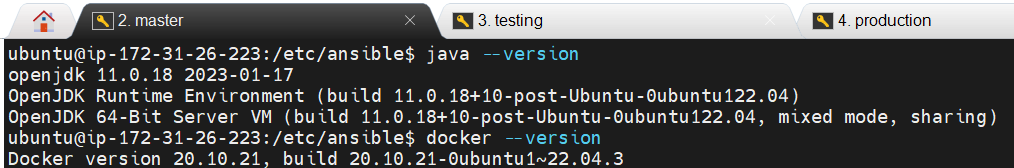
* **ansible-playbook install.yaml - -syntax-check (checking for syntax error)**
* **ansible-playbook install.yaml - -check (trail run)**
* **ansible-playbook install.yaml (execute)**

****

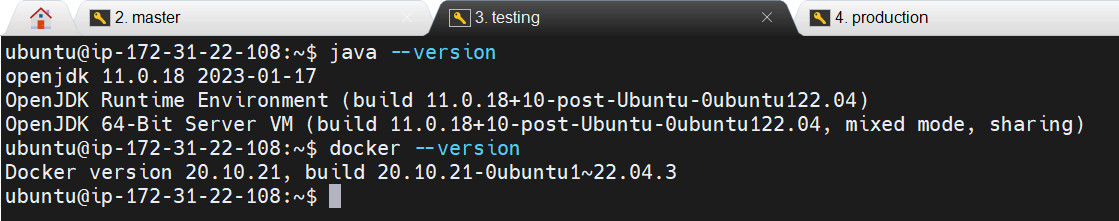
****

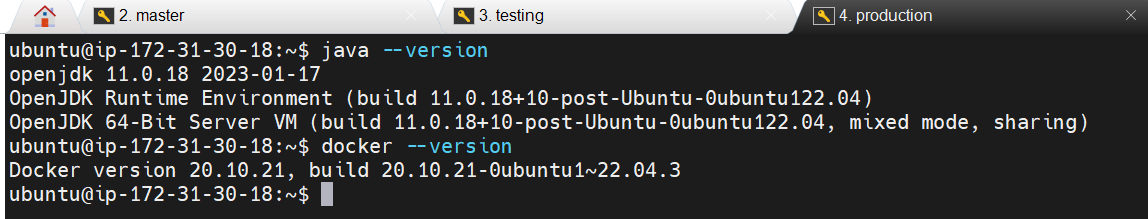
**Jenkins, java and docker are installed and running Successfully in master machine**

****

****

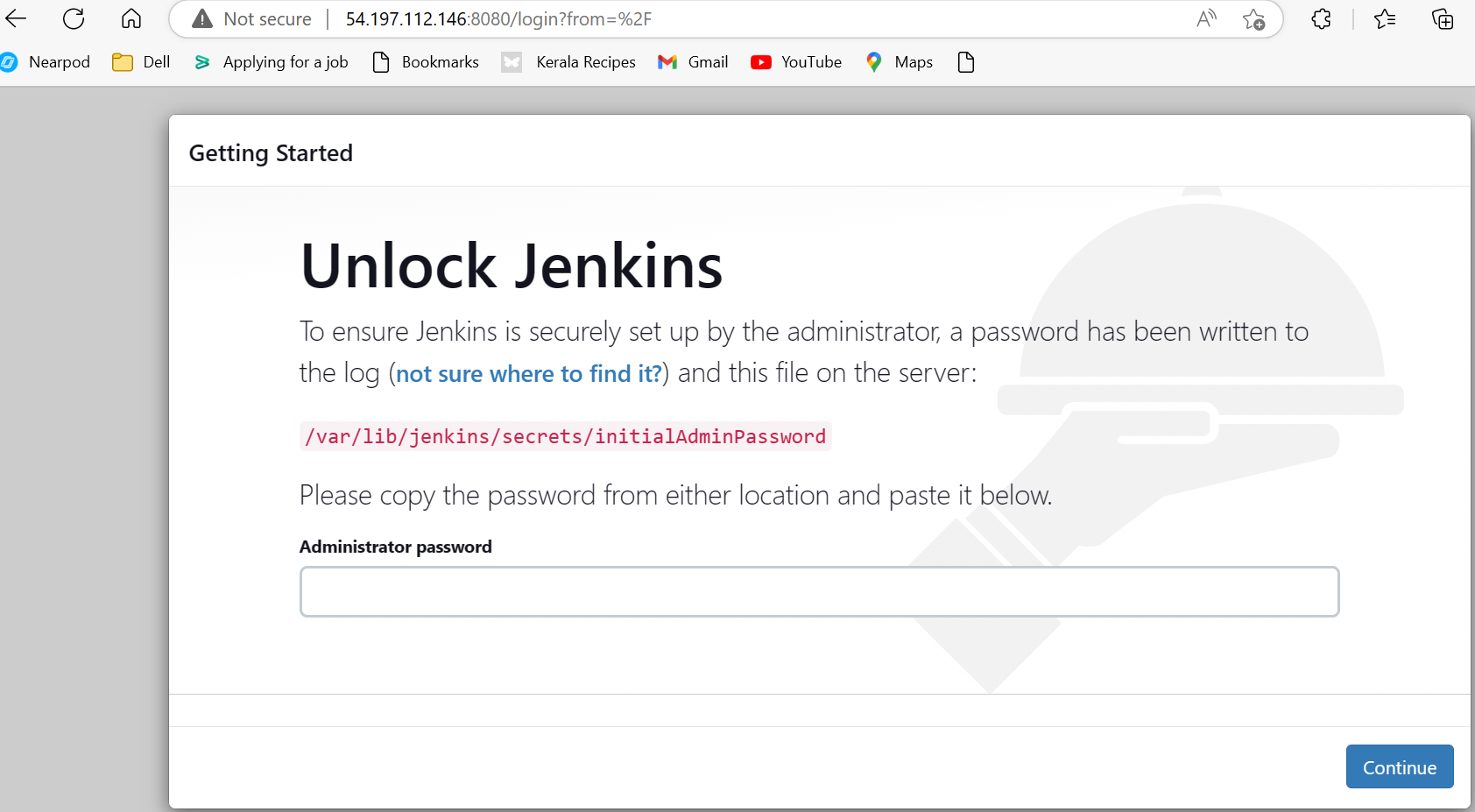
**java and docker are installed and running Successfully in testing and production machine**

****

****

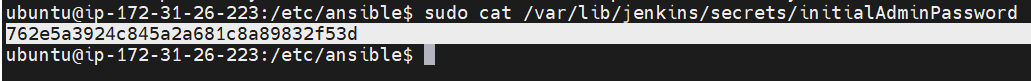
**Step: 7 open Jenkins dashboard and configure jenkins master**

**Copy the master machine public ip and open a web browser and paste it over there with :8080**

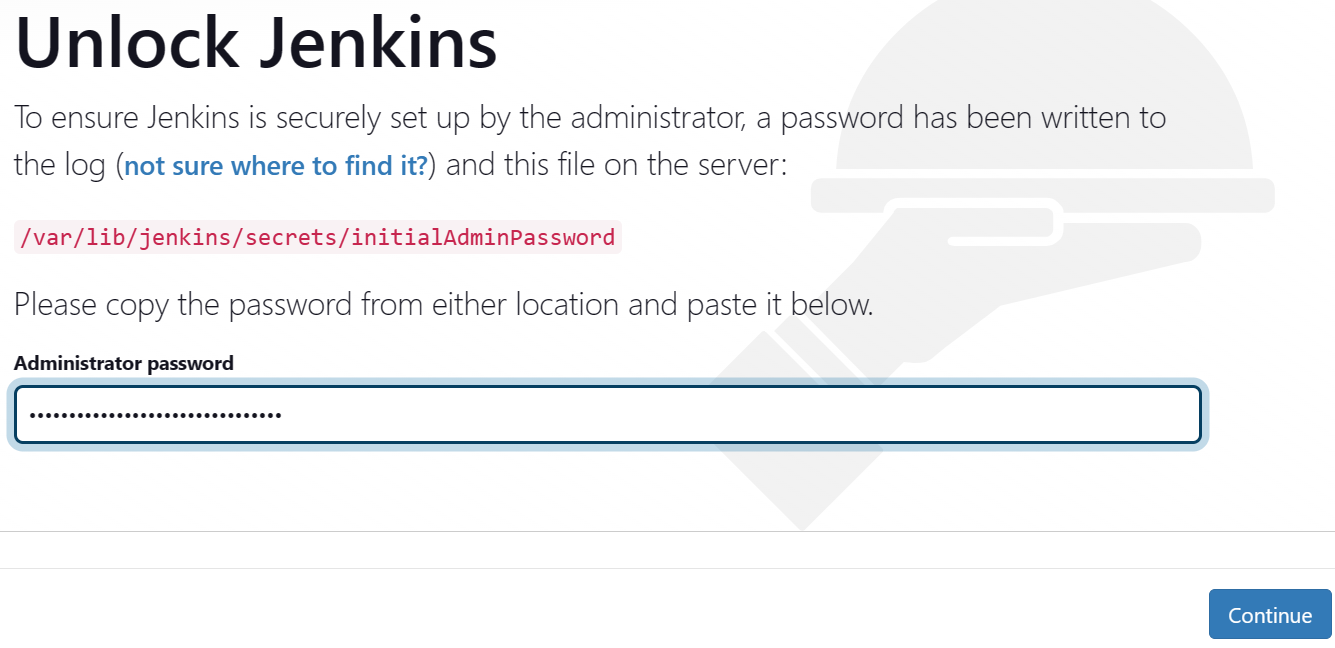
****

**Copy the path ->** /var/lib/jenkins/secrets/initialAdminPassword

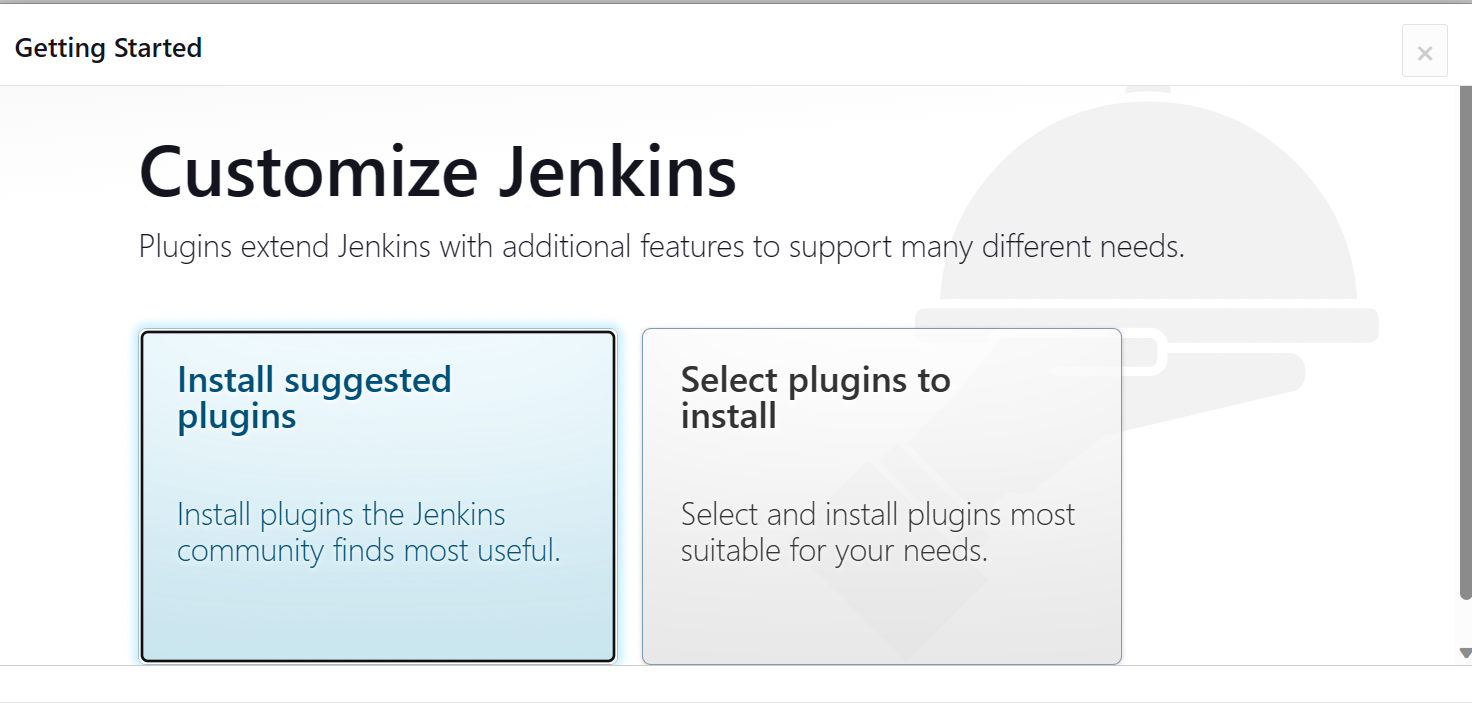
**Open the path from the master machine and copy the password**

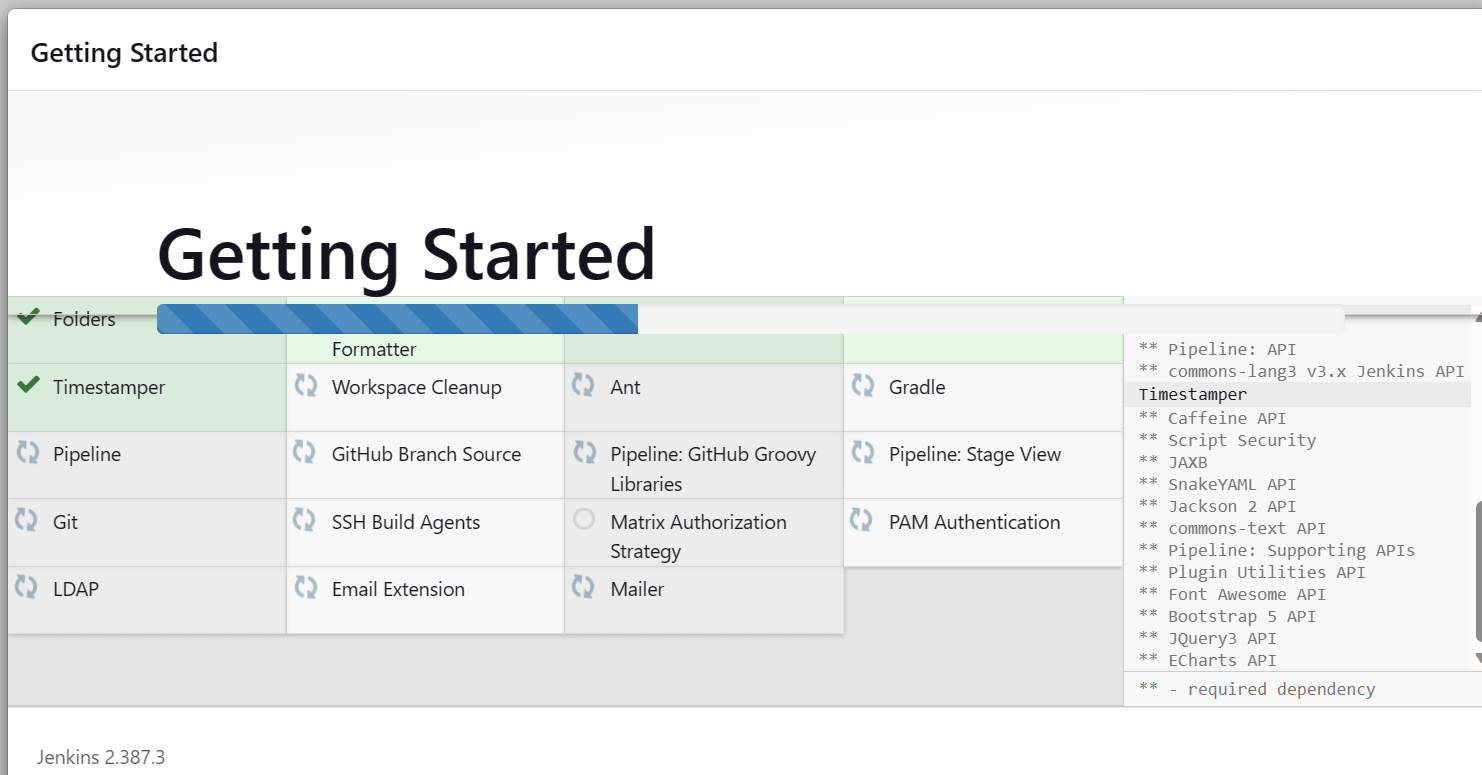
****

**Paste the password copied in the jenkins dashboard and click continue**

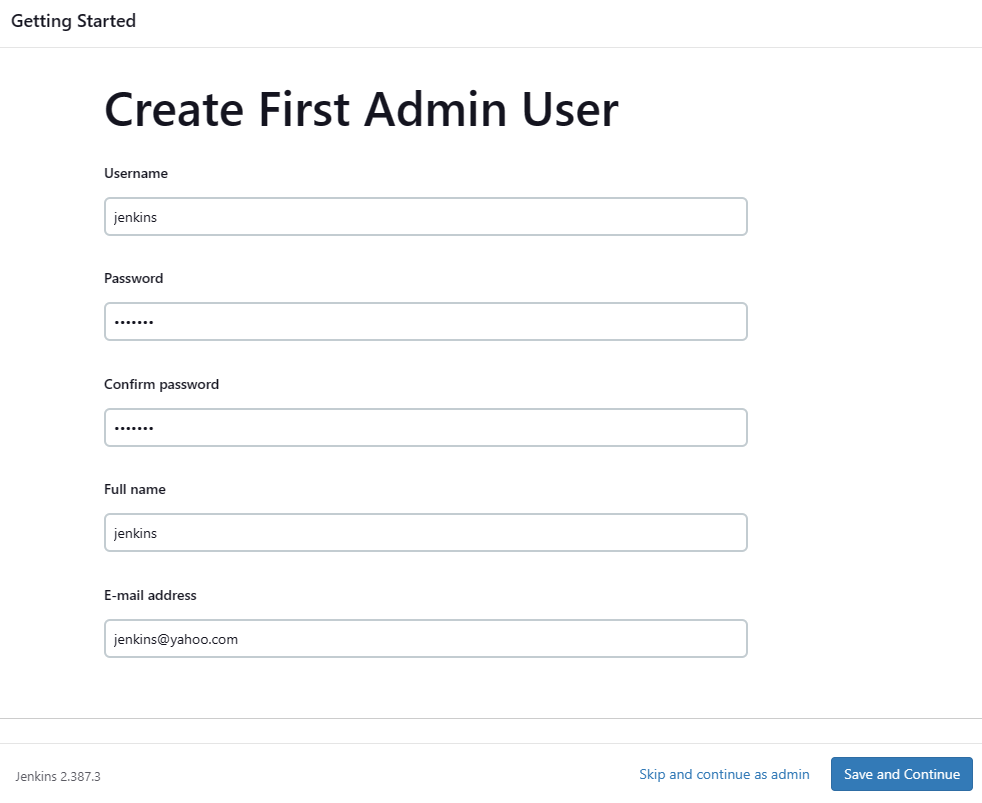
****

**Click on Install suggested plugins**

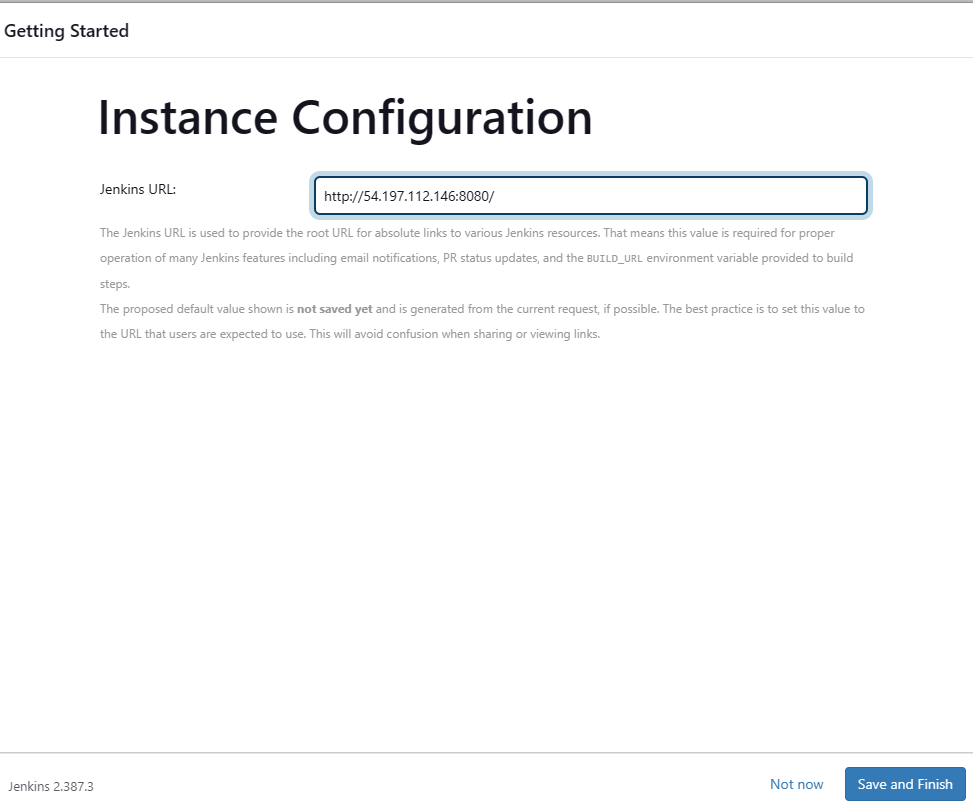
****

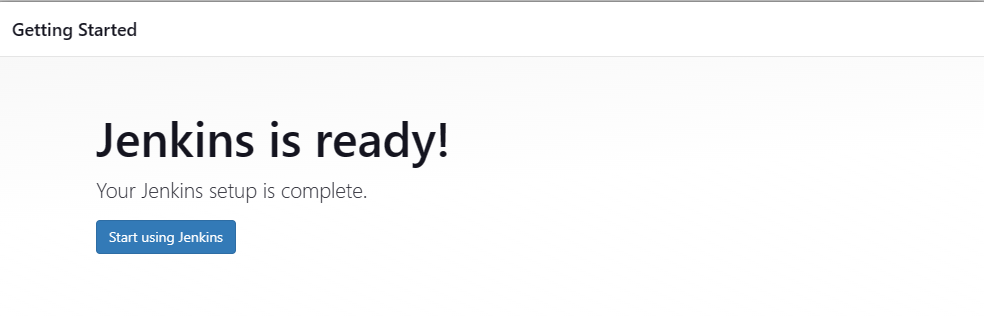
****

**Enter the required details and click save and continue**

****

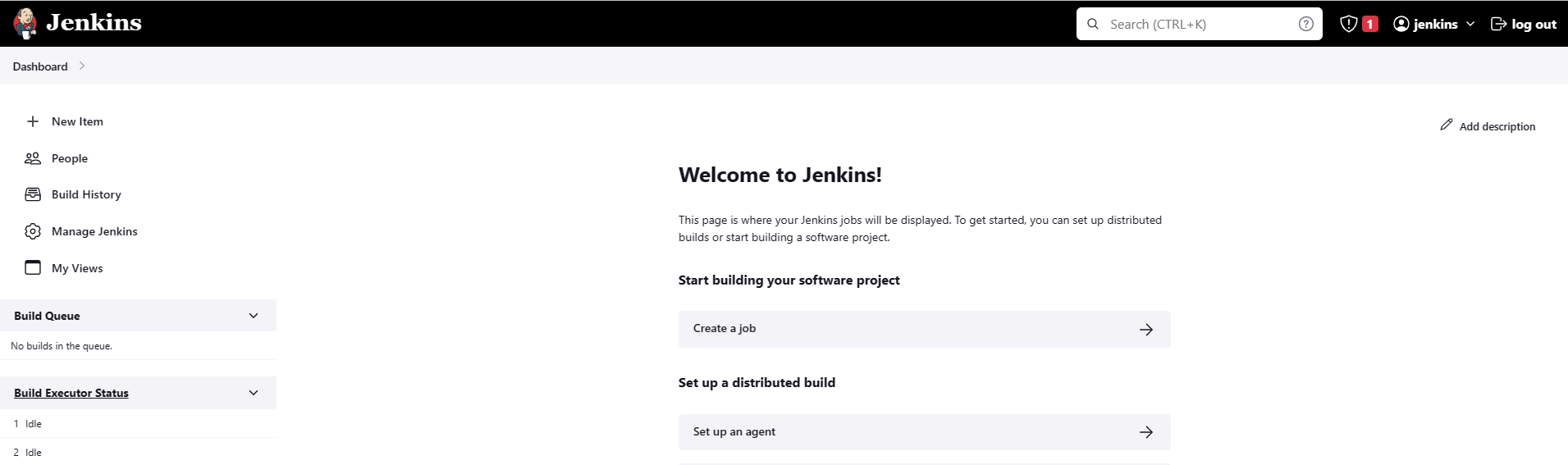
**Click save and finish**

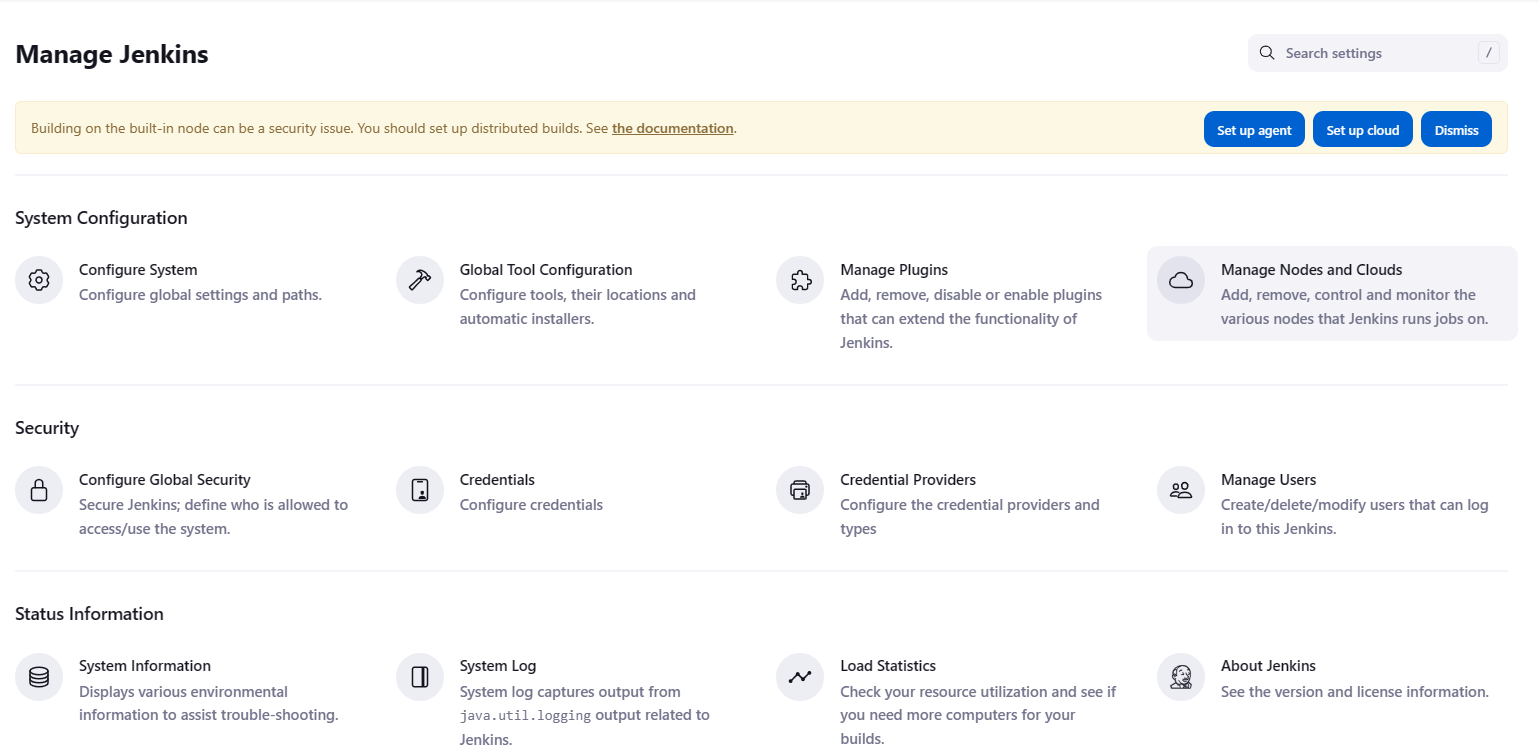
****

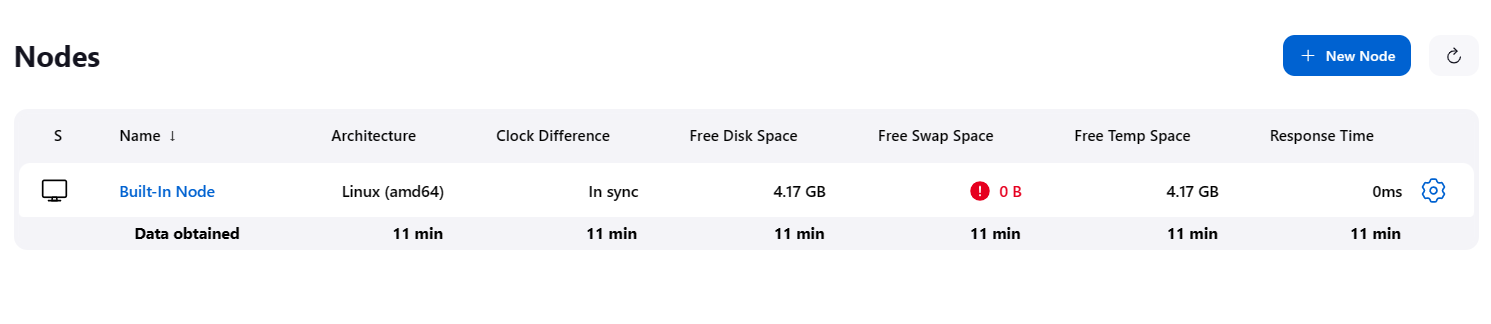
****

**Step: 8 Add nodes testing and production into Jenkins dashboard**

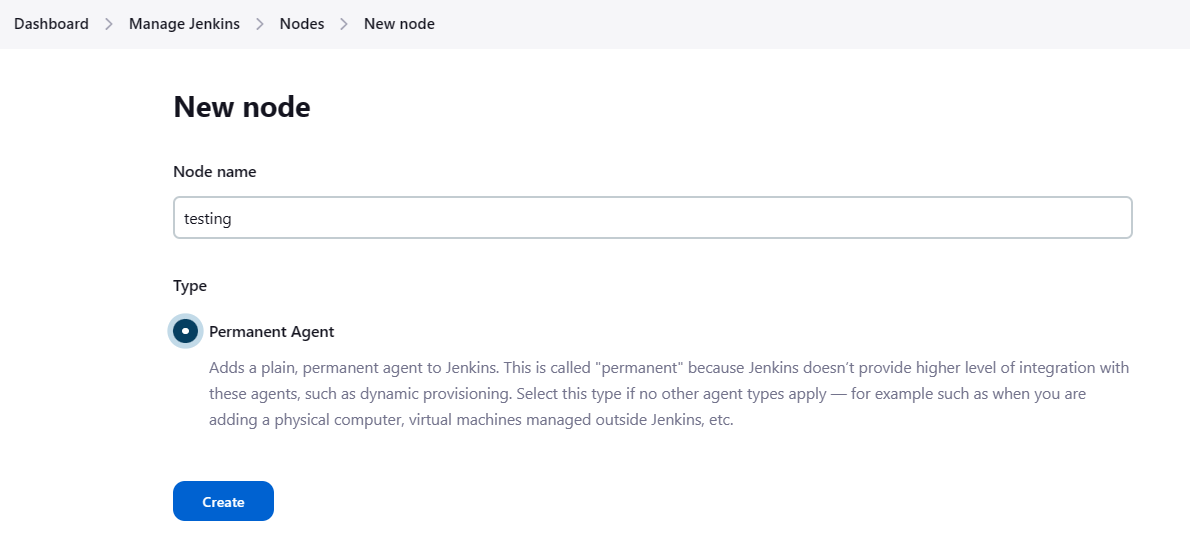
* **Open jenkins dashboard and click on manage jenkins**
* **Click on Manage Nodes and clouds**
* **Click on New node**

****

****

****

**Enter Node name, Select Permanent Agent and Click Create**

****

**Enter name, description and Remote root directory**

****

**Launch method-> Lanuch agents via SSH**

**Click on Add to add a credential**

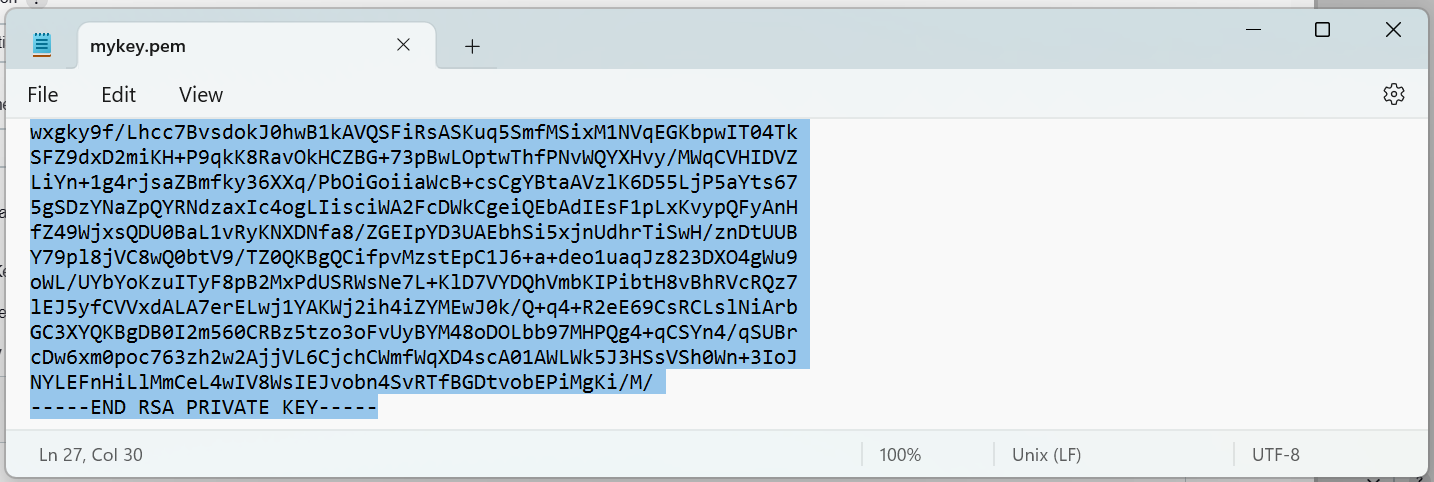
****

**Select kind-> SSH Username with private key**

**Enter ID and Description**

****

**Open the pem file used while creating the EC2 instance and copy the key**

****

**Enter the username: ubuntu**

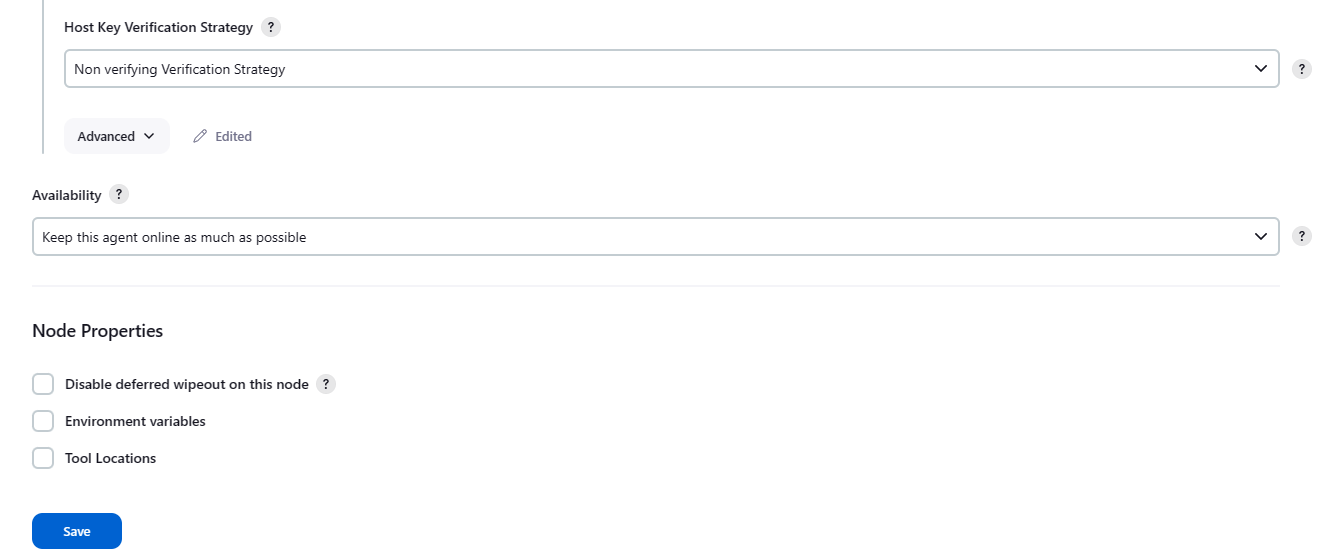
**Enter directly the copied key and click on Add**

****

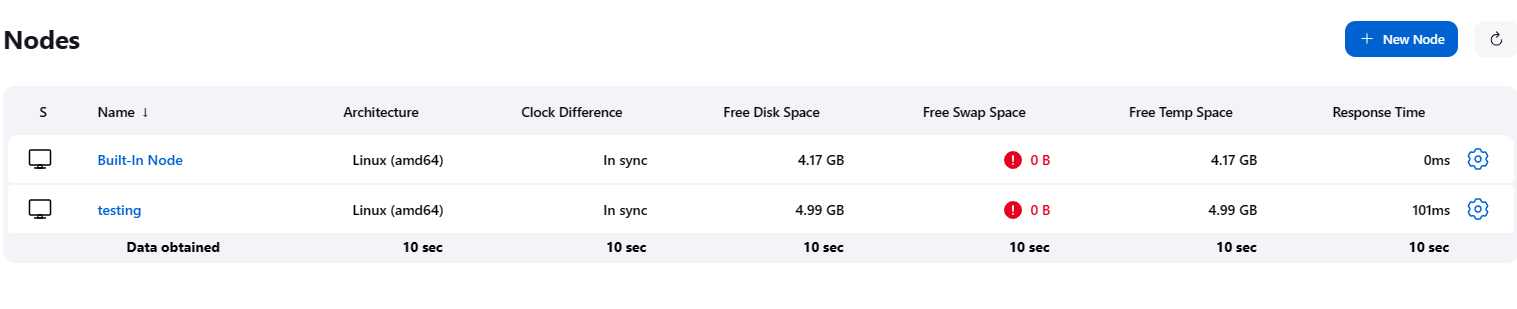
**Select the credential created**

**Select Non verfying Verification Strategy**

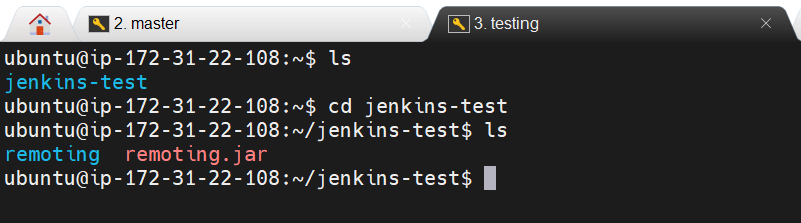
**Click Save**

****

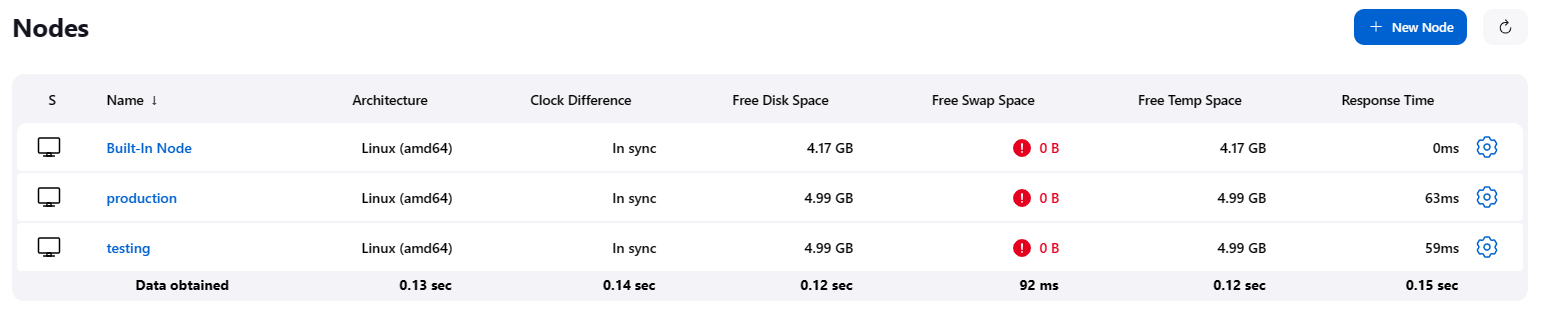
**Testing node created successfully**

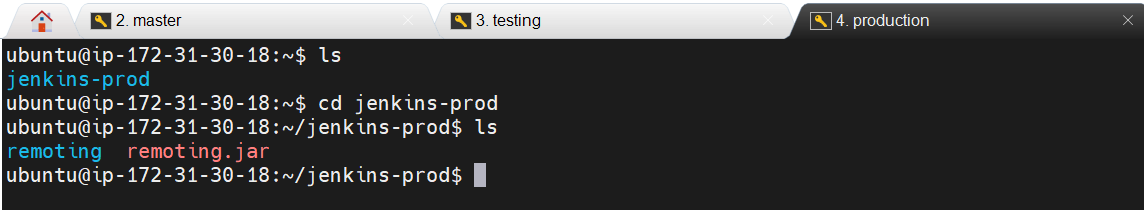
****

**Check from testing machine**

****

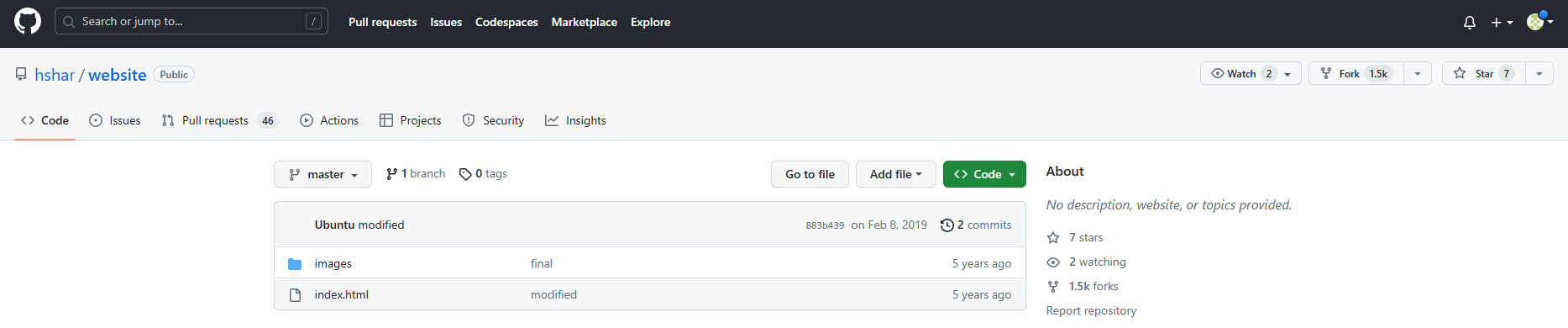
**Similarly add production node as well**

****

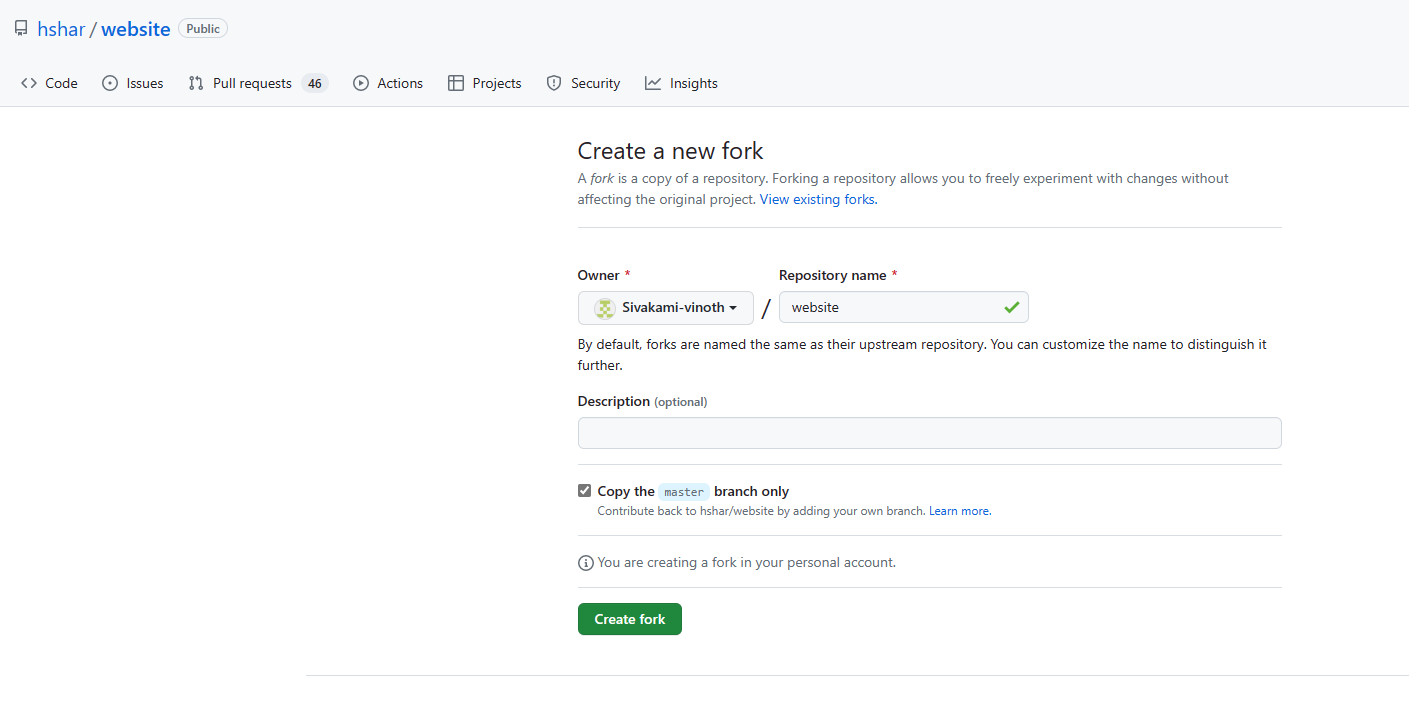
****

**Step: 8 Fork the data from** [**https://github.com/hshar/website.git**](https://github.com/hshar/website.git) **to your github account**

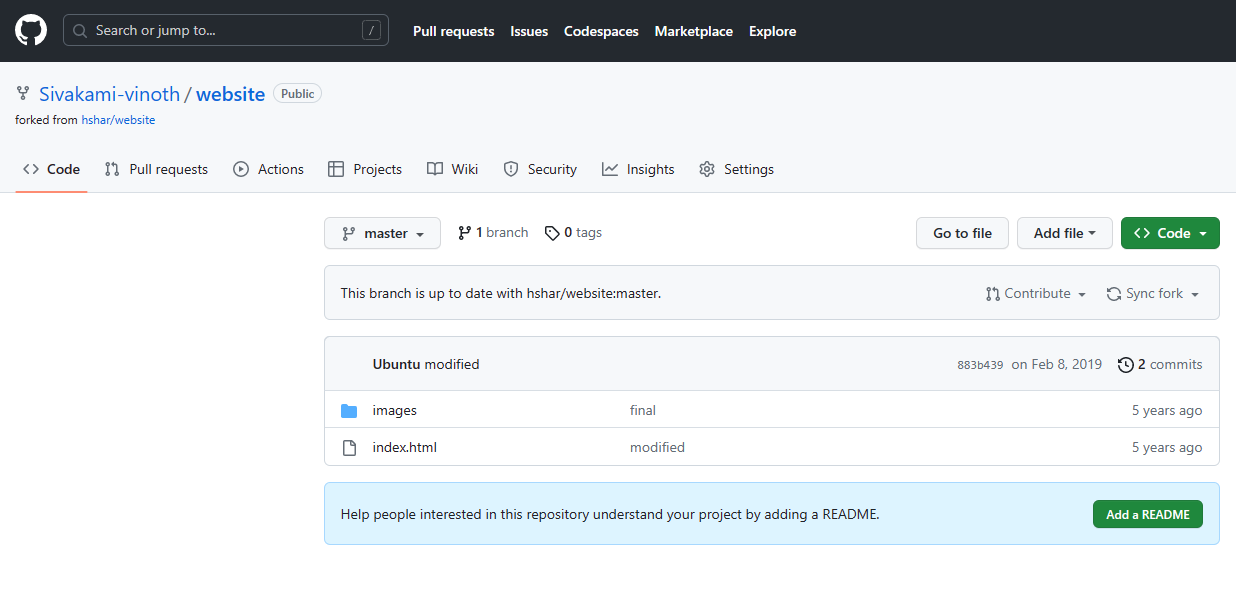
**Open the link** [**https://github.com/hshar/website.git**](https://github.com/hshar/website.git) **and click on Fork**

****

**Click on Create fork**

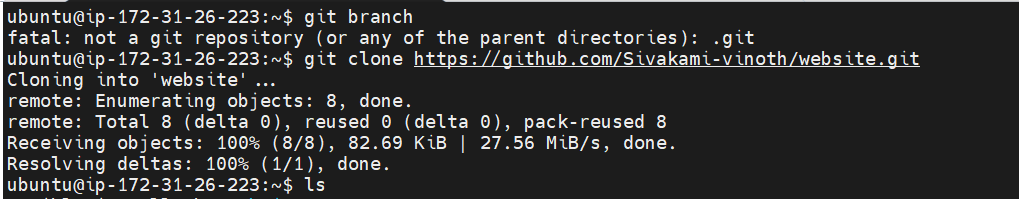
****

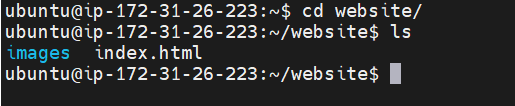
**Repository forked to my github account**

****

**Step: 9 Clone the date from the remote repository(https://github.com/Sivakami-vinoth/website.git) to local machine**

**Run command -> git clone https://github.com/Sivakami-vinoth/website.git**

****

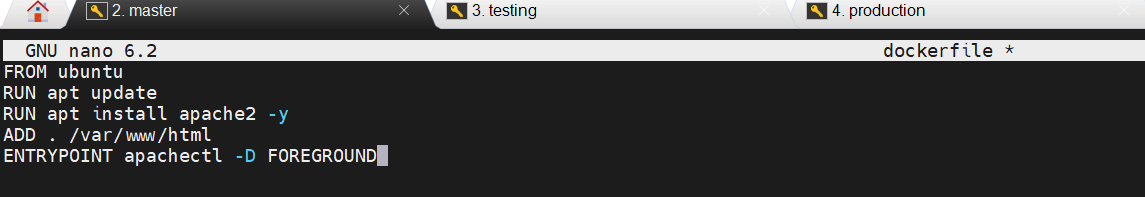
****

**Step: 9 Create a docker file and add dockerfile to git and commit the same**

**Create a dockerfile-> sudo nano dockerfile**

****

**Dockerfile (to install apache2)**

****

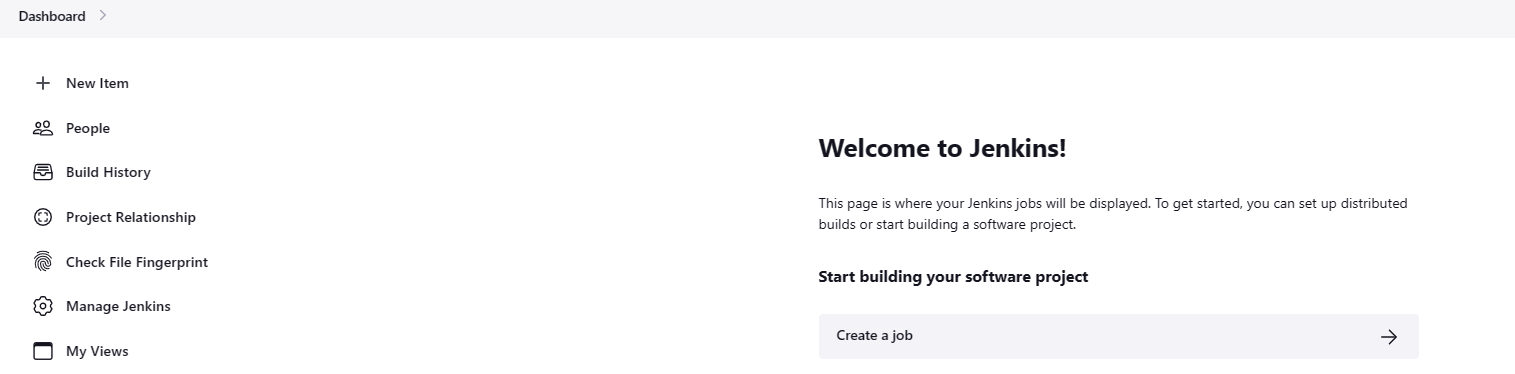
**Add the dockerfile to git and commit the same-> git commit –m “adding dockerfile”**

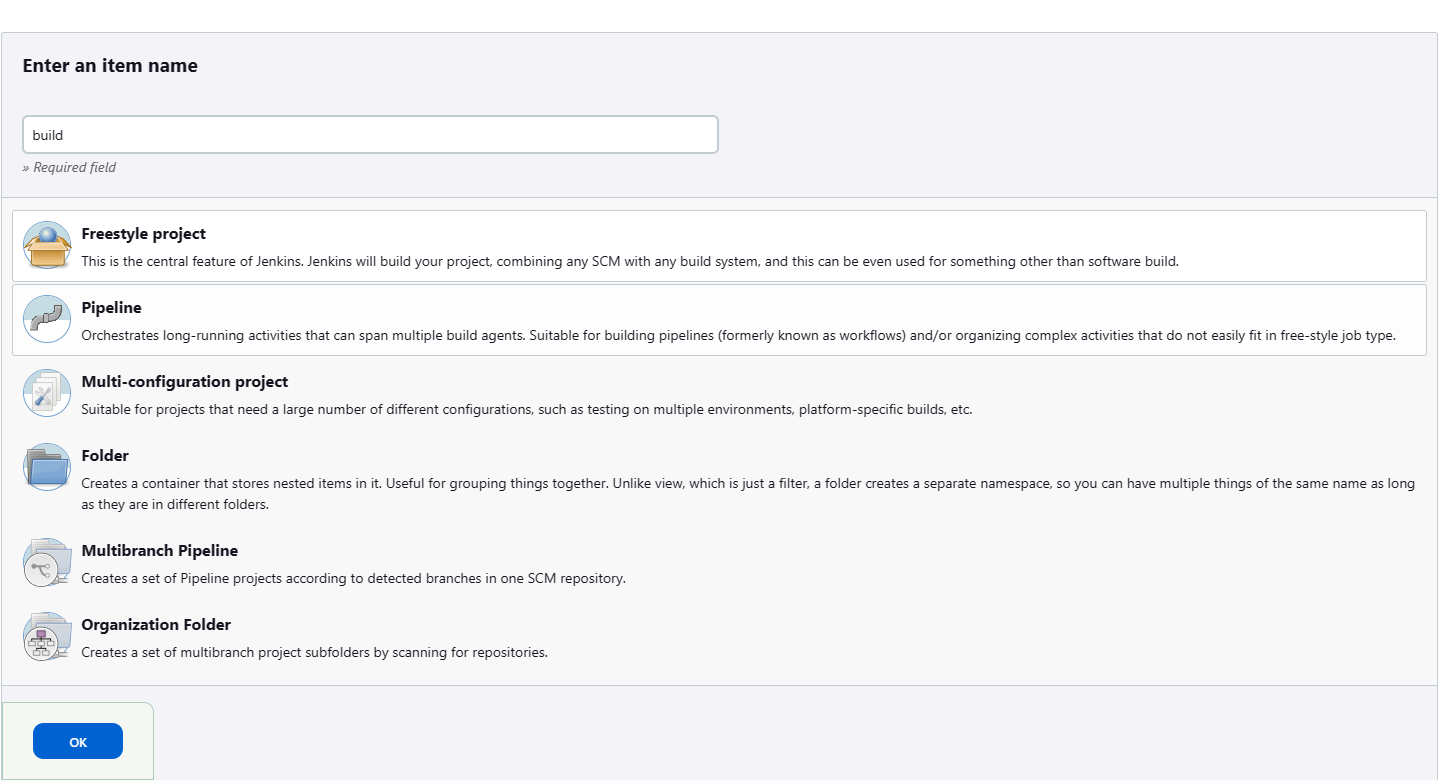
****

**Step: 10 Create a job build, will be triggered once commit is made to master branch**

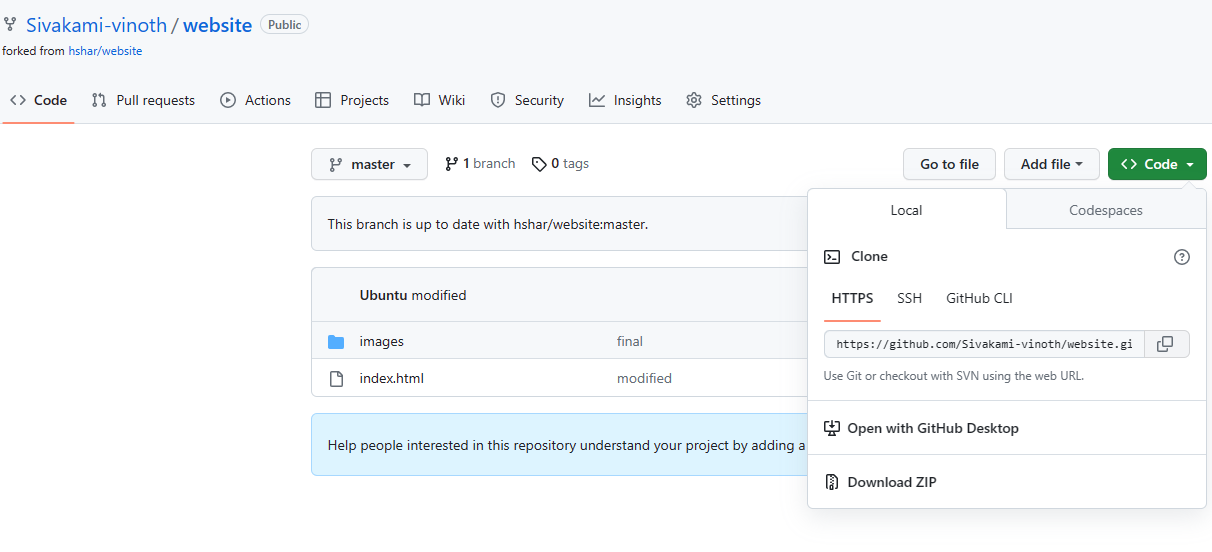
**Open jenkins dashboard -> Click on New Item**

**Enter the item name, Freestyle project and click ok**

****

****

**Open the github and copy the repository url link**

****

**Enter the Description**

**Paste the url link in the github project url**

**Restrict where this project can be run ->testing**

****

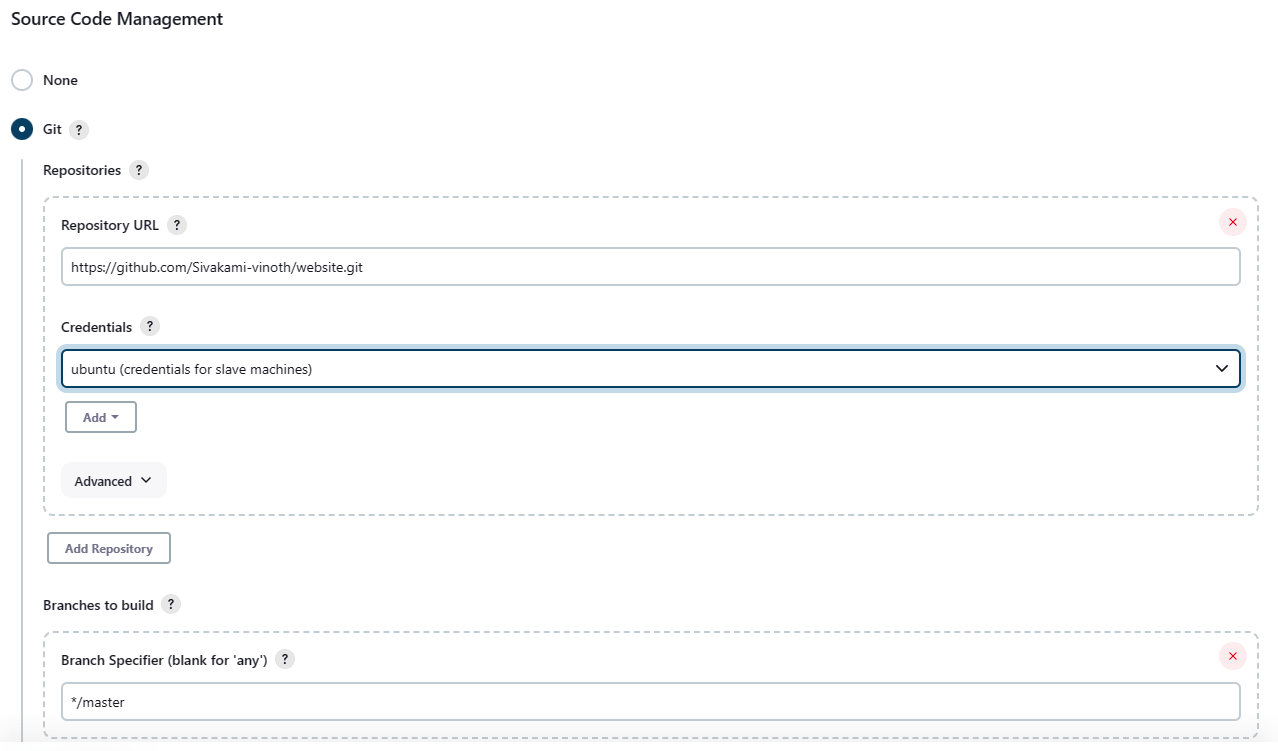
**Source code management**

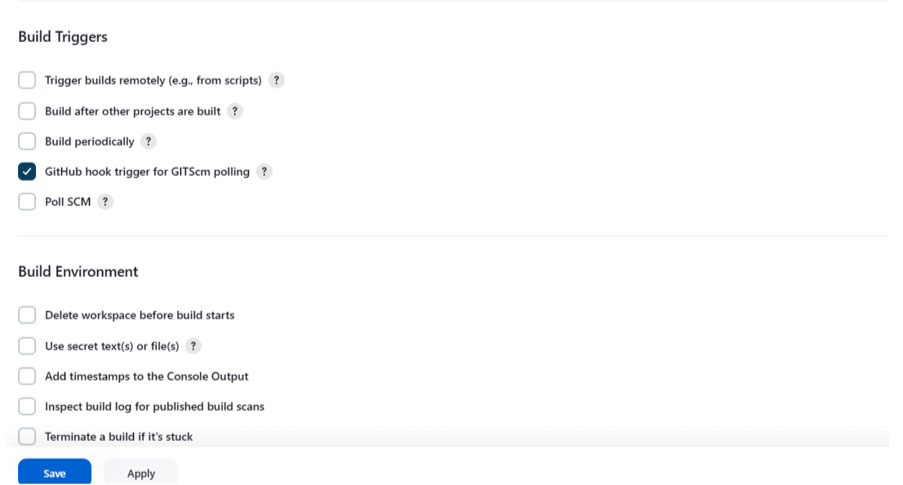
**Select->git, paste the copied url, select the credential**

**Select branch ->/master**

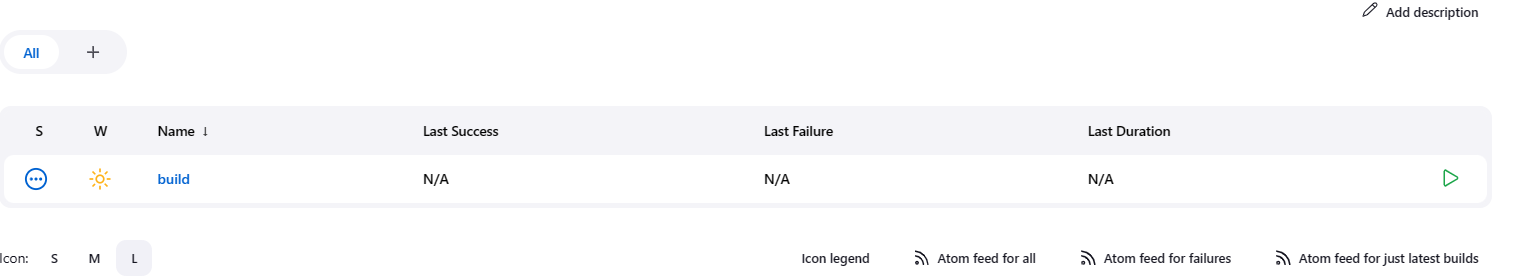
**Build trigger->Github hook trigger for GITScm polling**

**Click on Apply and Save**

****

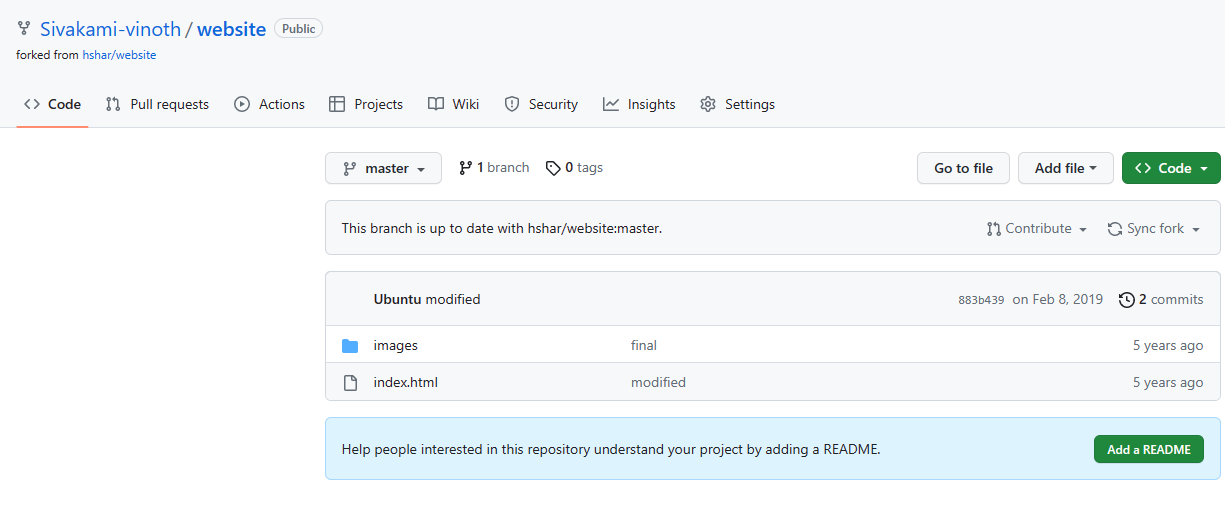
****

**Build Job created**

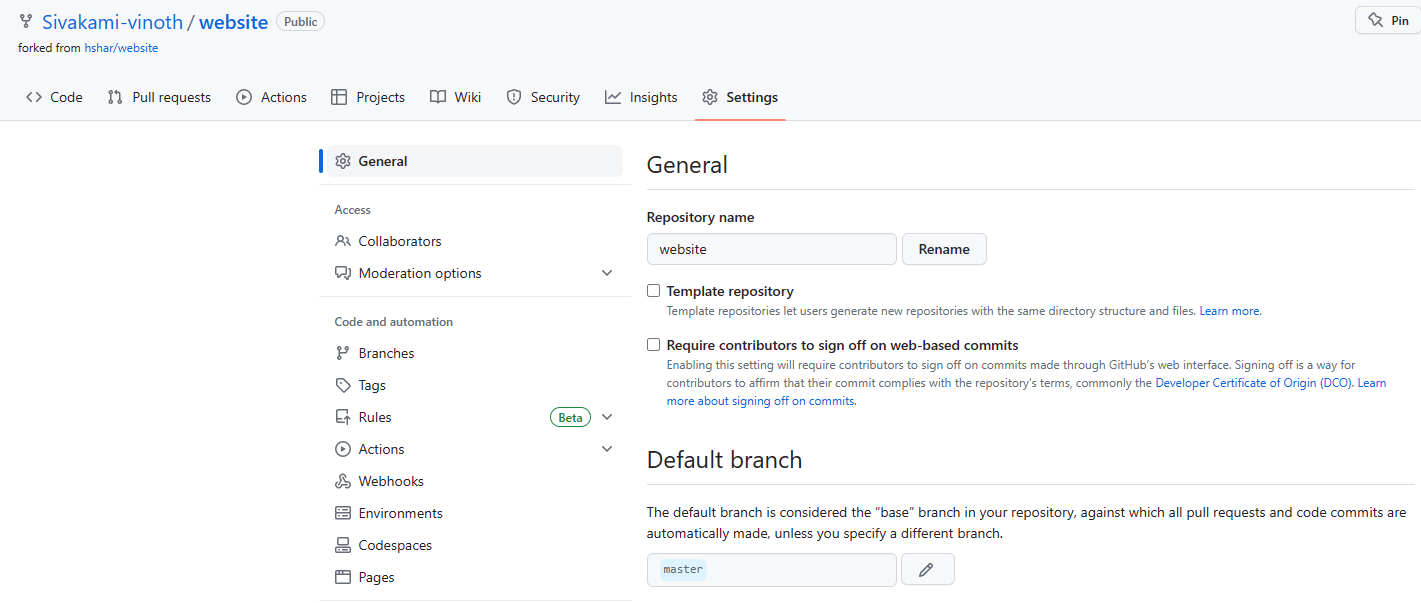
****

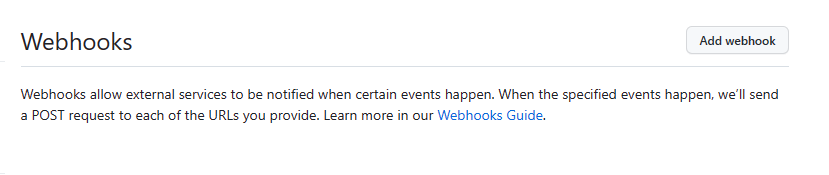
**Create a webhook to create connection between github and jenkins master**

**Open the github repository-> open settings**

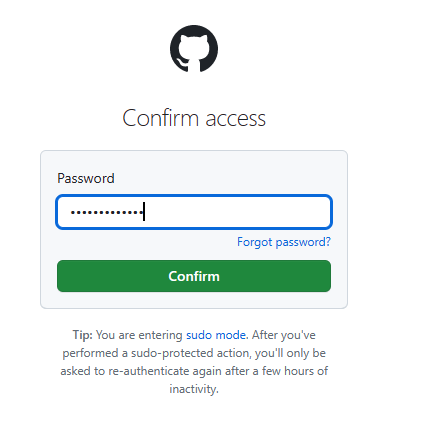
****

**Open Webhooks -> Add webhook**

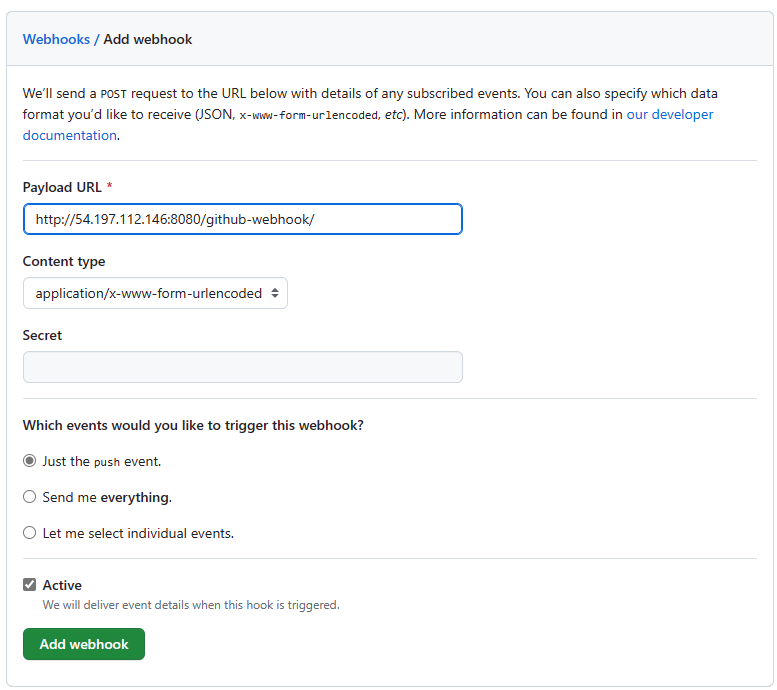
****

****

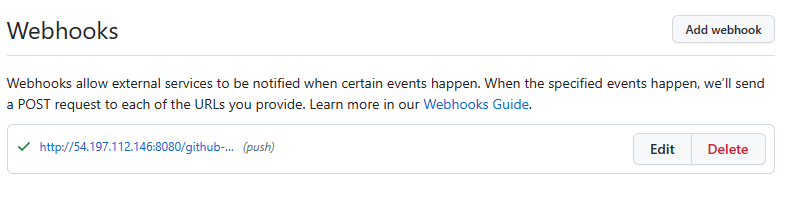
**Enter the github password**

****

**Enter the jenkins master url and click Add Webhook**

****

**Connection estabished successfully**

****

**Create a github Token to push the data from local repository to remote**

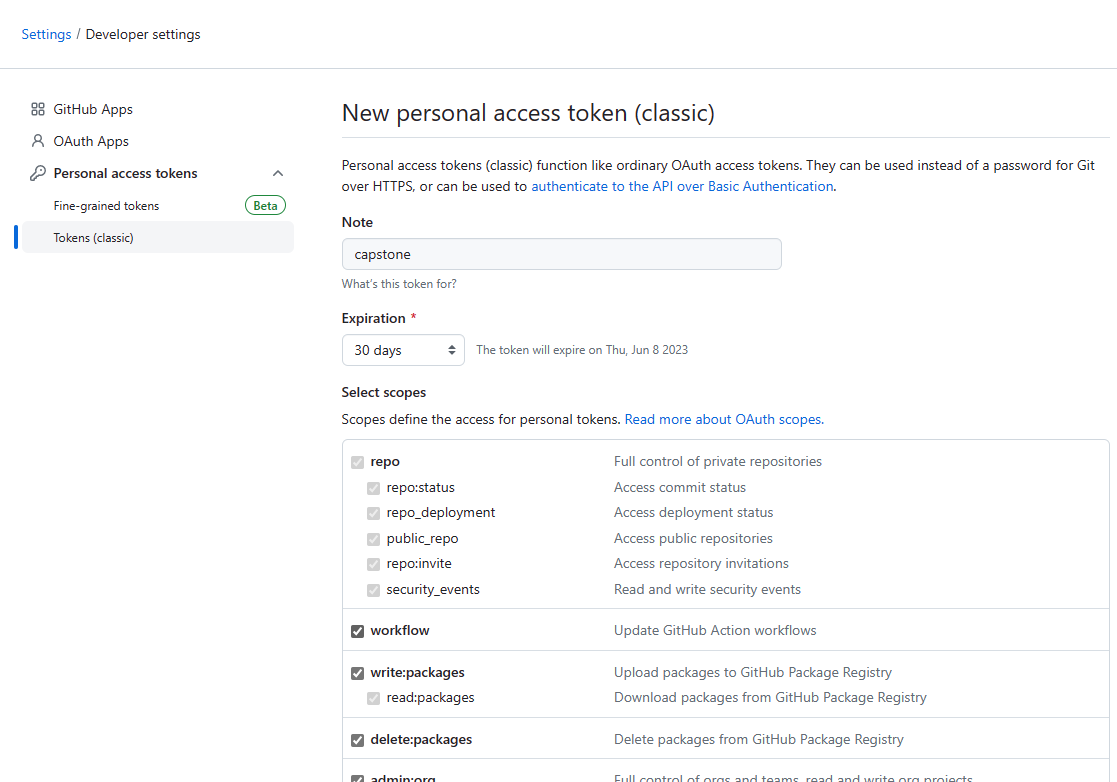
**Goto Github accout->settings**

**Open Developer settings**

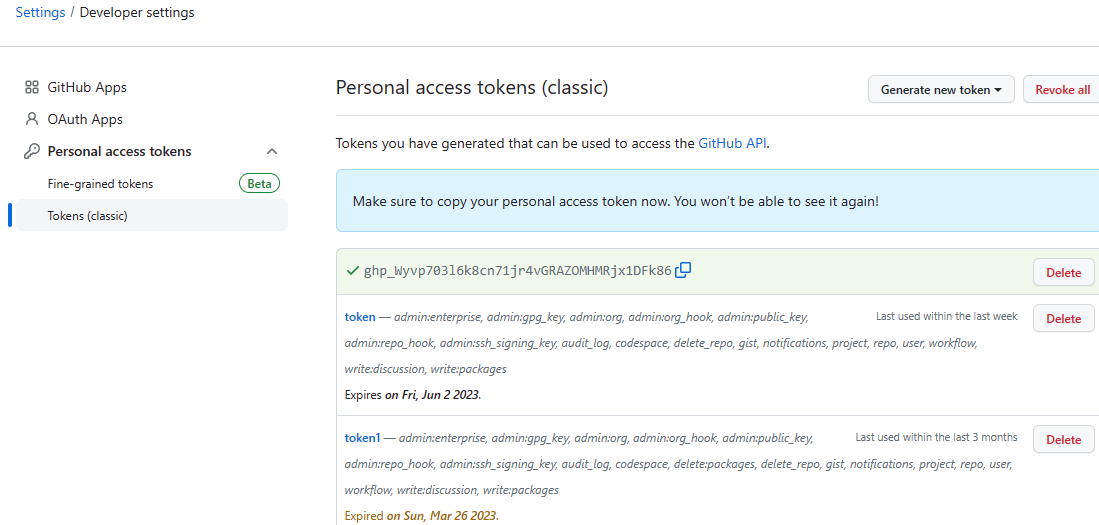
**Personal access tokens ->tokens(classic)**

**Enter Note and select all the scopes**

**Click on Generate Token**

****

**Copy the token and save it**

****

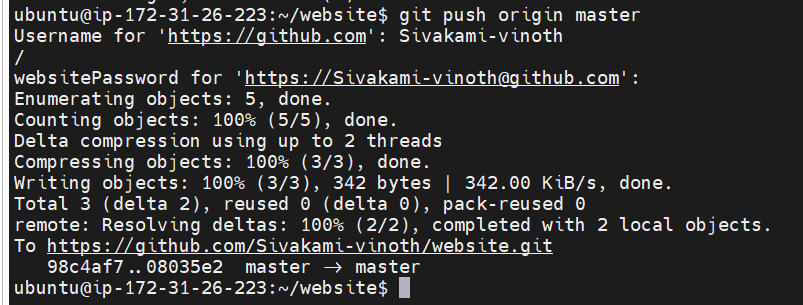
**Step: 11 Make a commit to master branch and push the same to github origin**

**Edit the index.html file as follows**

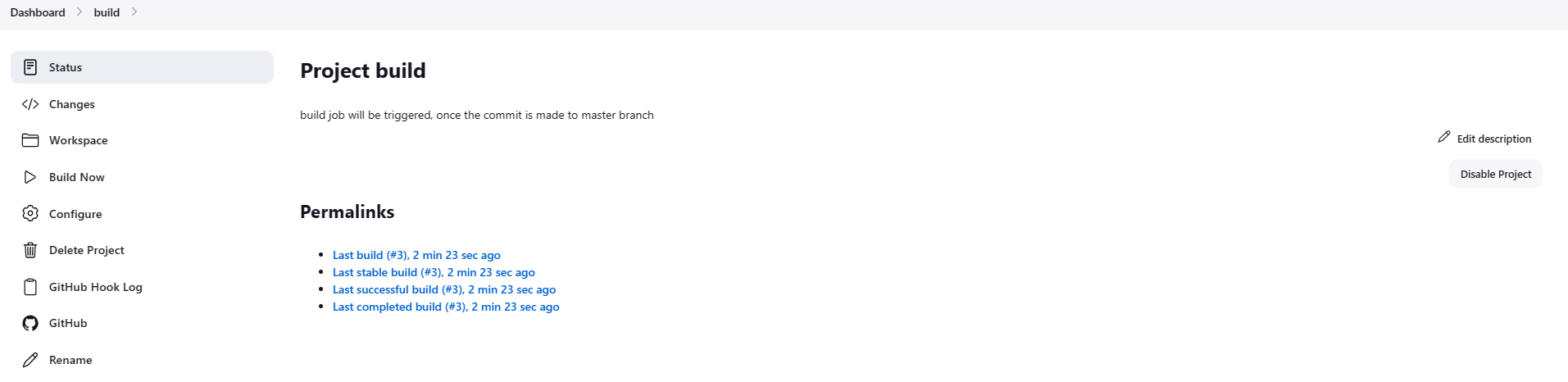
**Add and commit the index.html file**

****

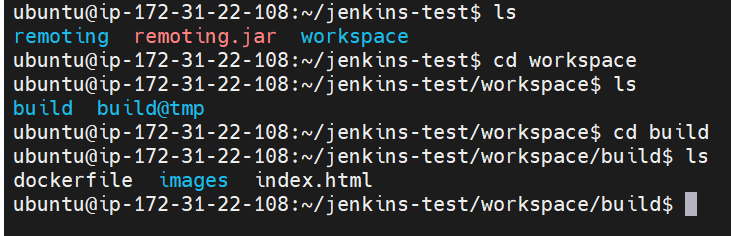
**Push the master branch to the origin**

****

**Job build got triggered**

****

**Check the testing machine (files from github master branch copied to jenkins-test folder)**

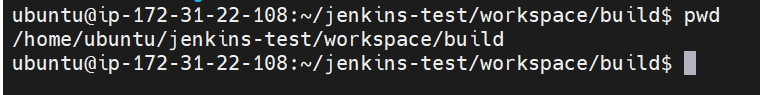
****

**Open the Configuration of build job**

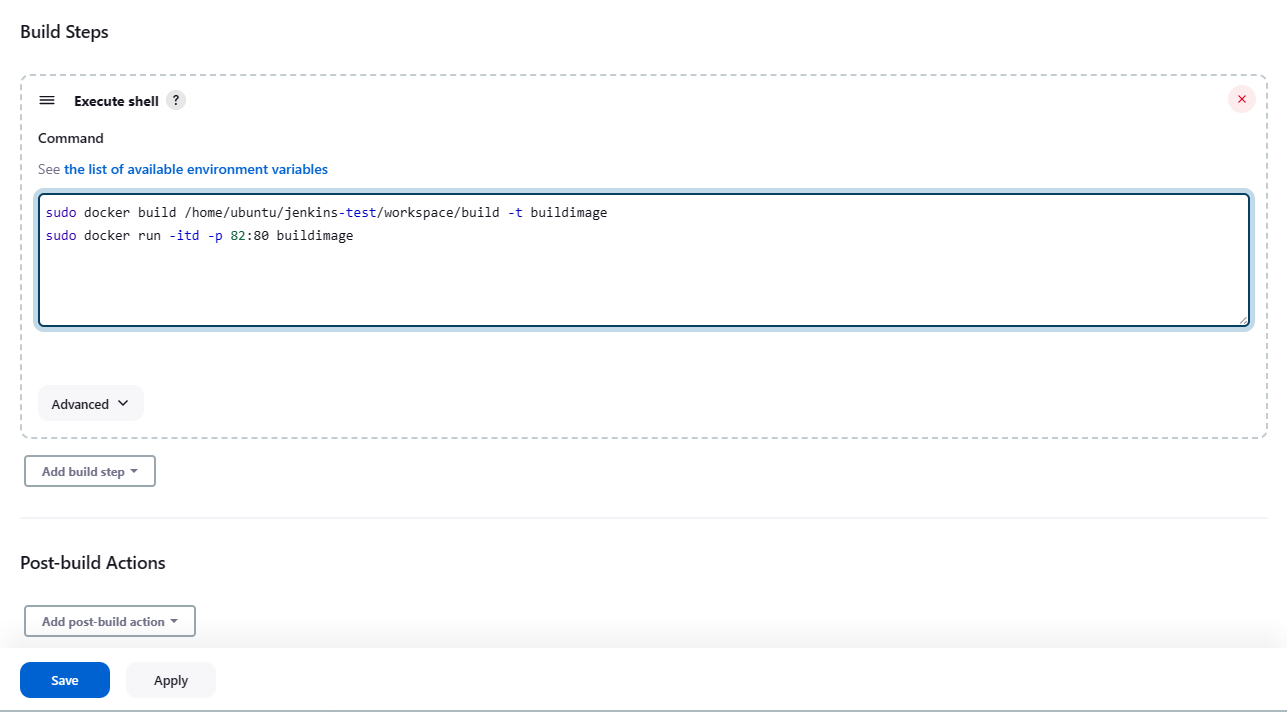
**Add build setup-> Execute SHELL**

****

**Copy the path (testing machine)**

****

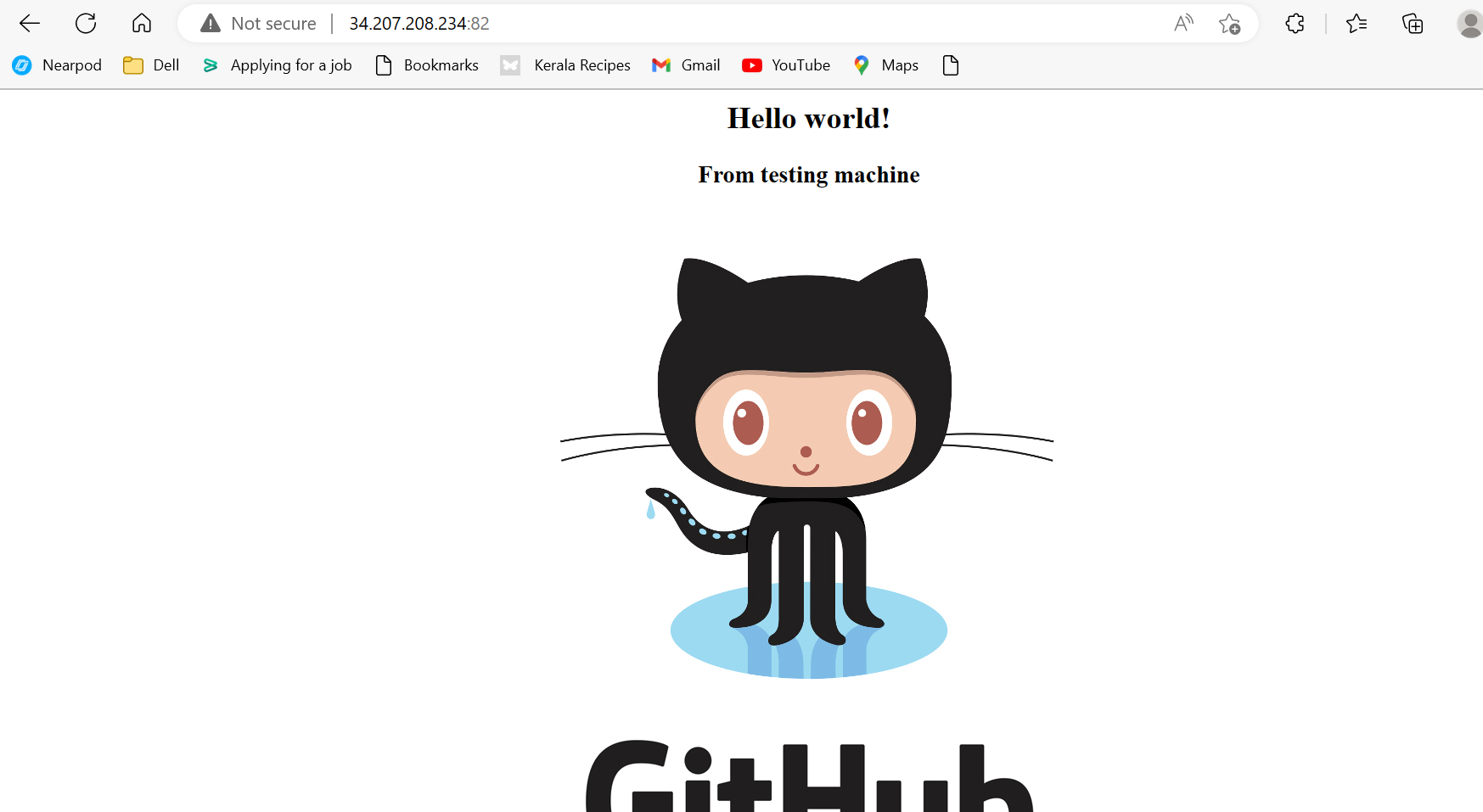
**Update the script as follows (build the docker image buildimage and run the same in port 82)**

****

**Trigger the build job**

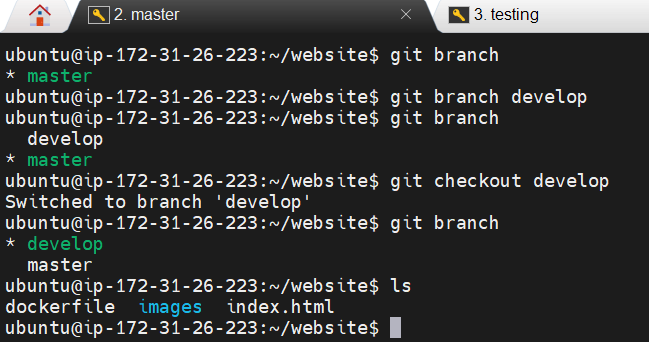
****

**Copy testing machine public ip and open from browser in port 82**

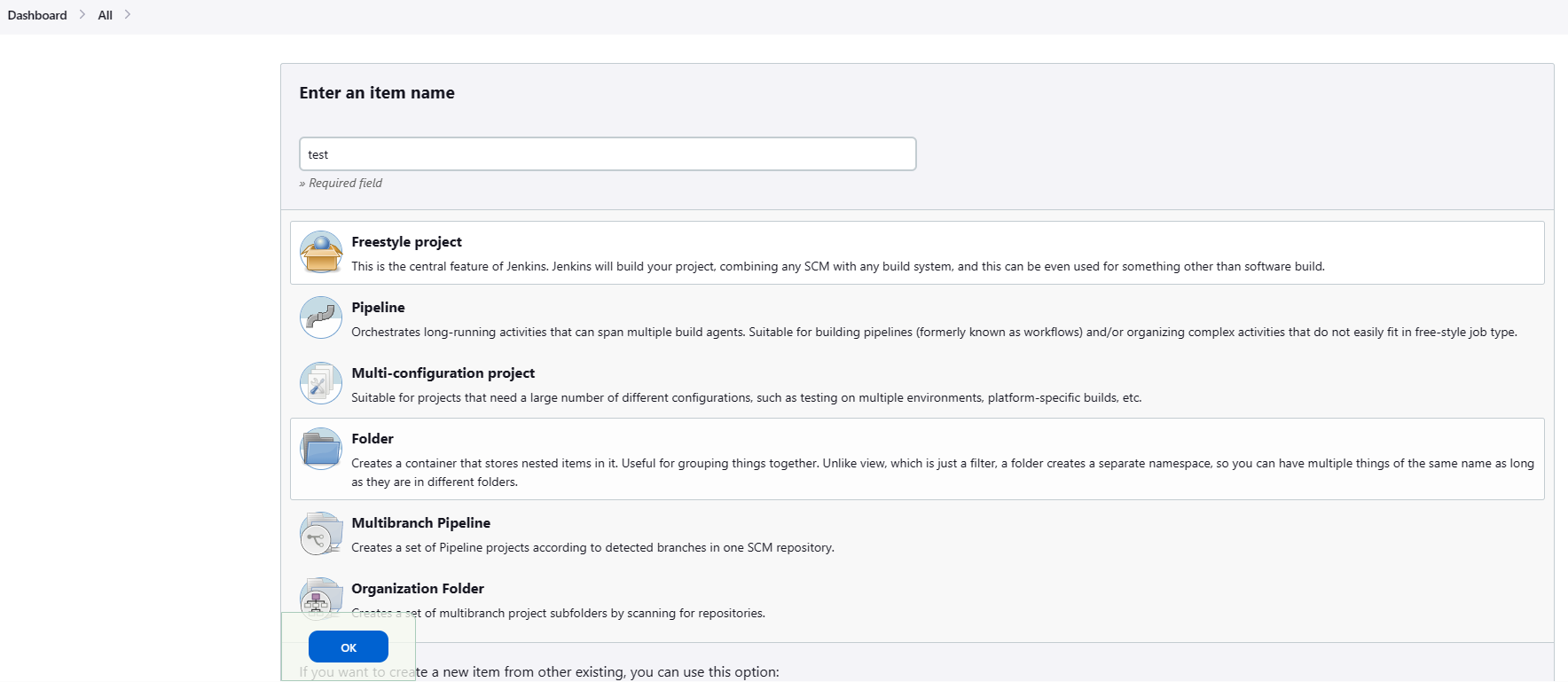
****

**Step:11 Create job test, will be triggered once commit is made to develop branch**

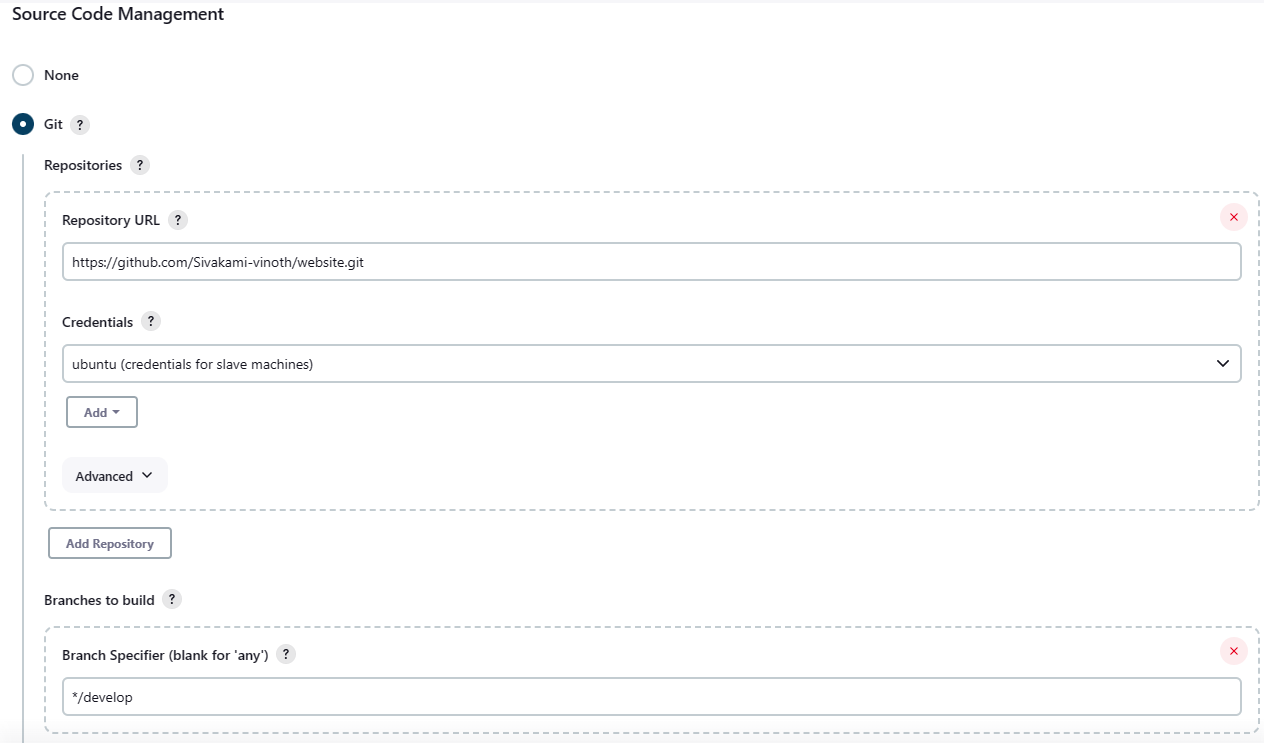
**Create a develop branch**

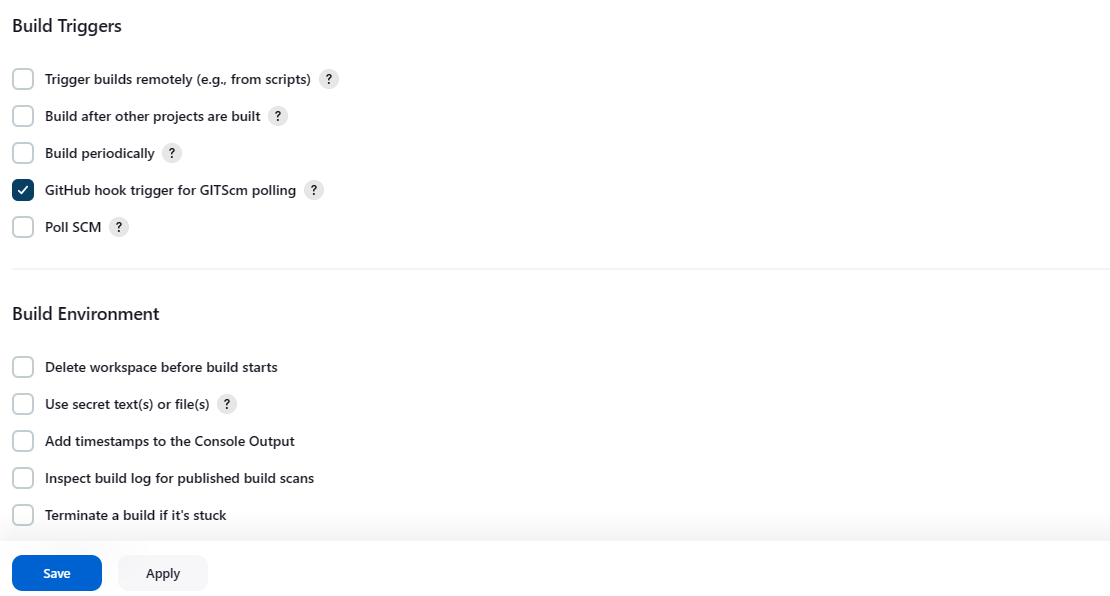
****

**Create job test following the stepls below**

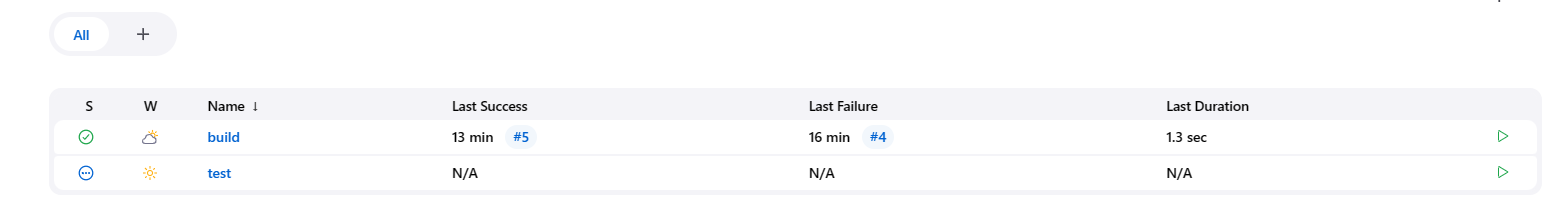
****

****

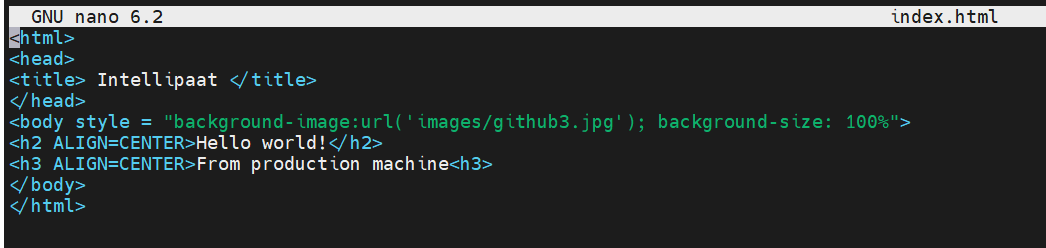
****

****

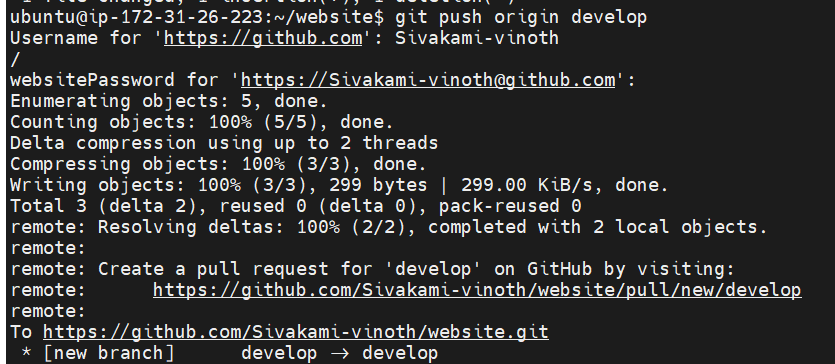
**Job test created successfully**

****

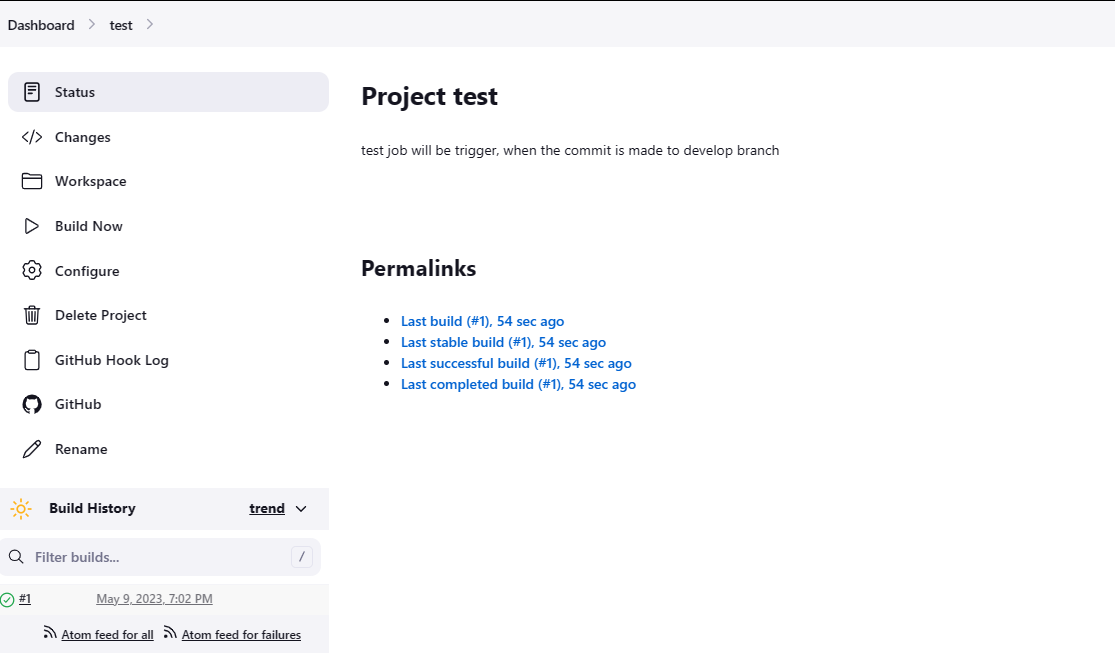
**Edit the index.html file**

****

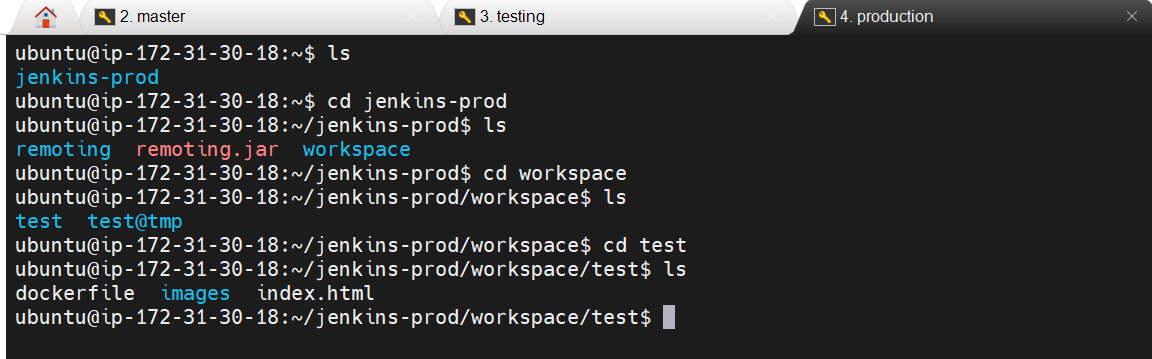
**Push the develop branch to the origin**

****

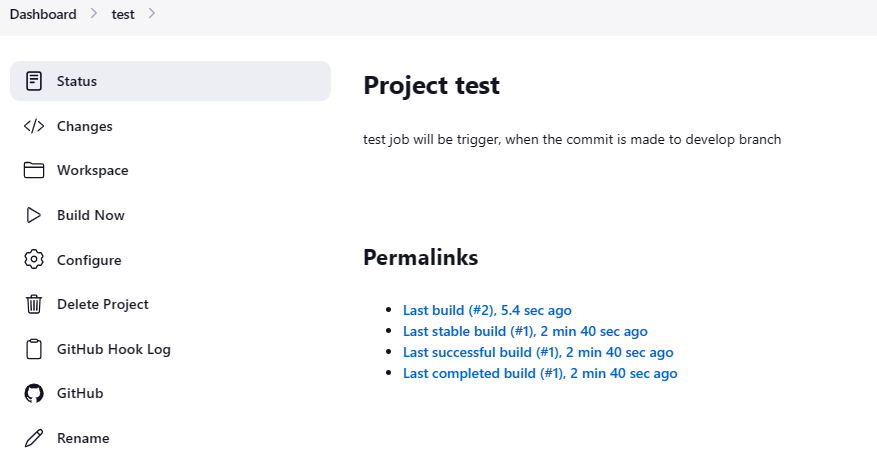
**Job test got triggered**

****

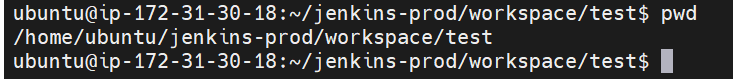
**Check the production machine (files from github develop branch copied to jenkis-prod folder)**

****

**Open the configuration of test job**

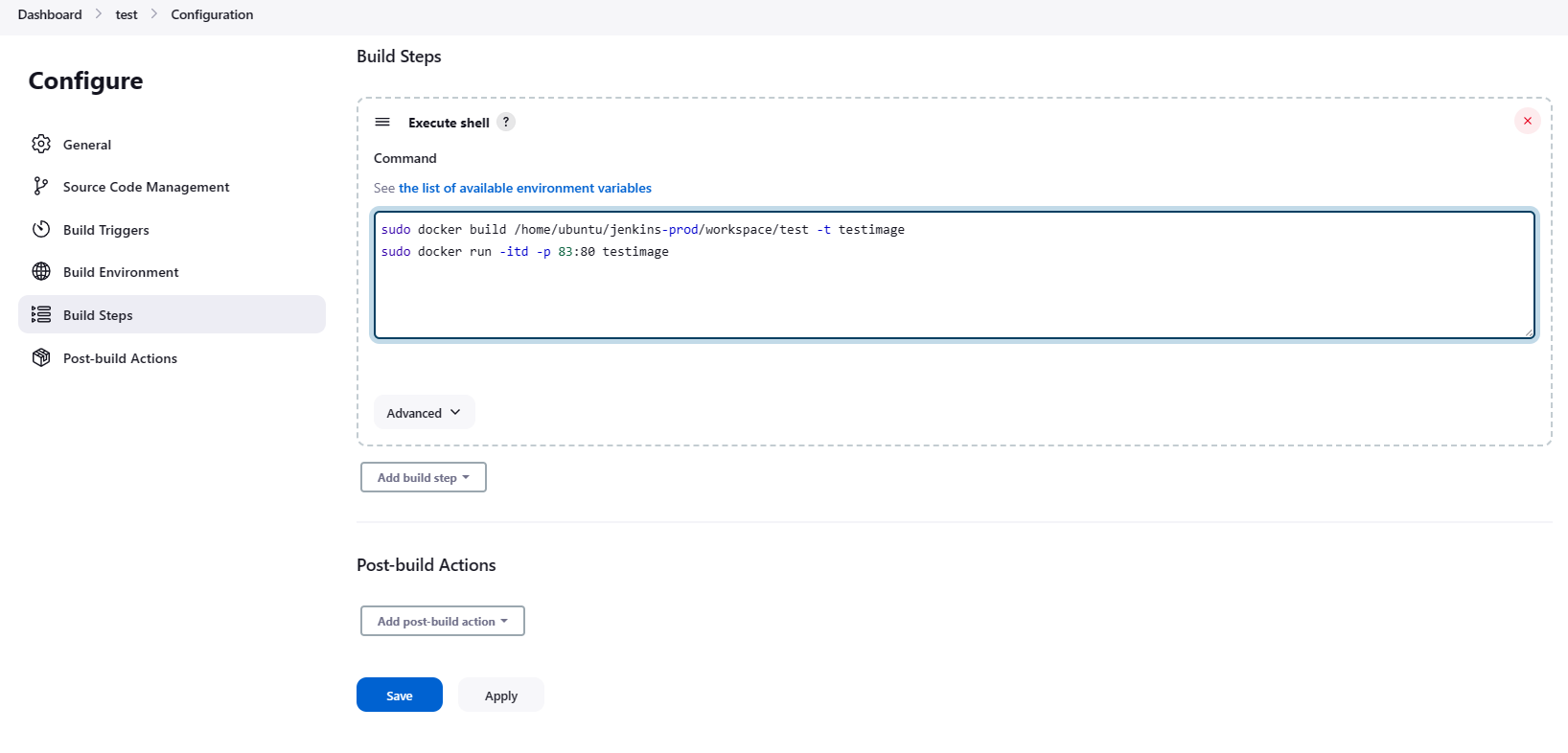
****

**Copy the path (production machine)**

****

**Add build steps-> Execute shell**

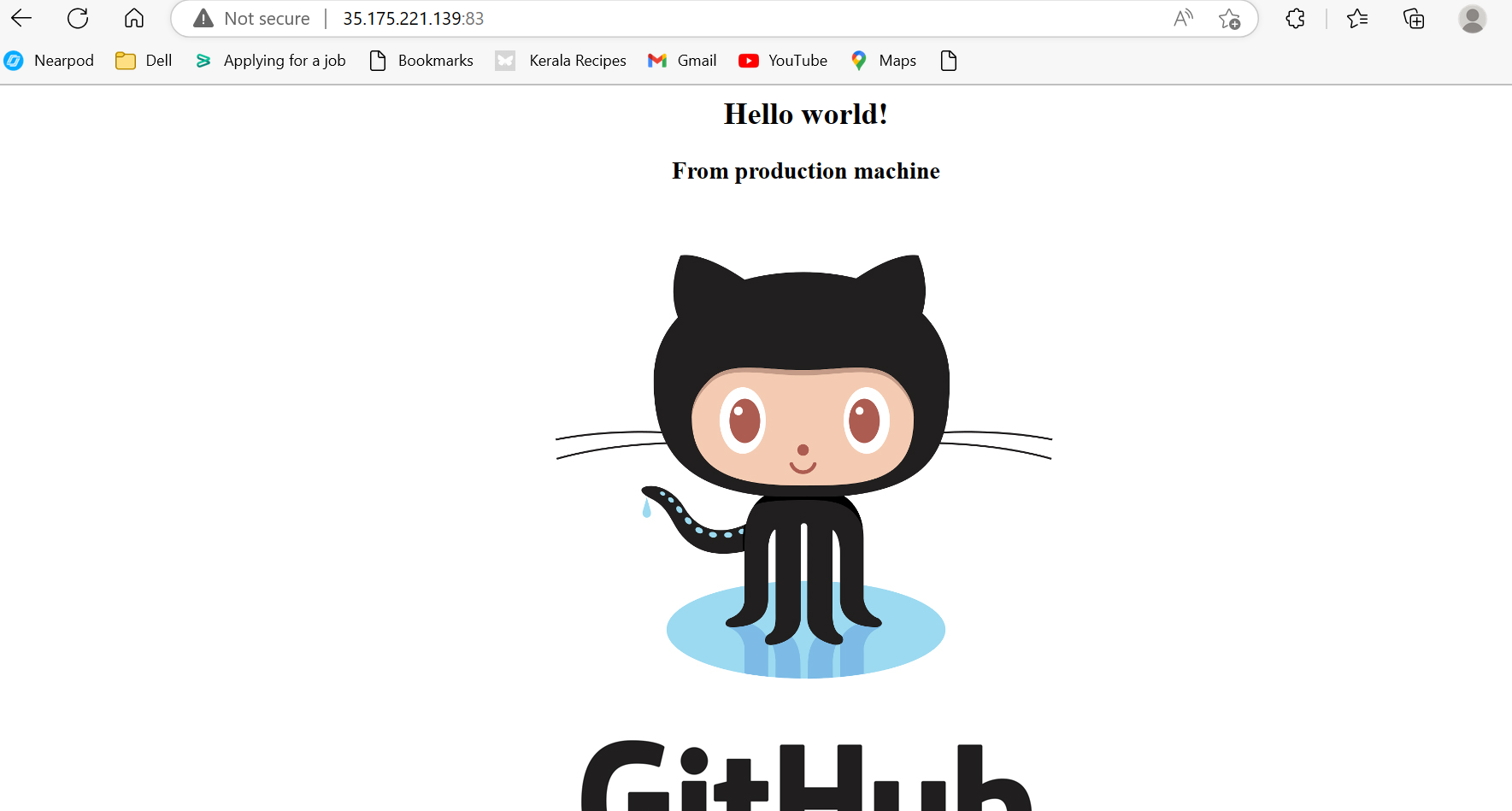
**Create a docker image testimage and run the same in port 83**

****

**Job test triggered**

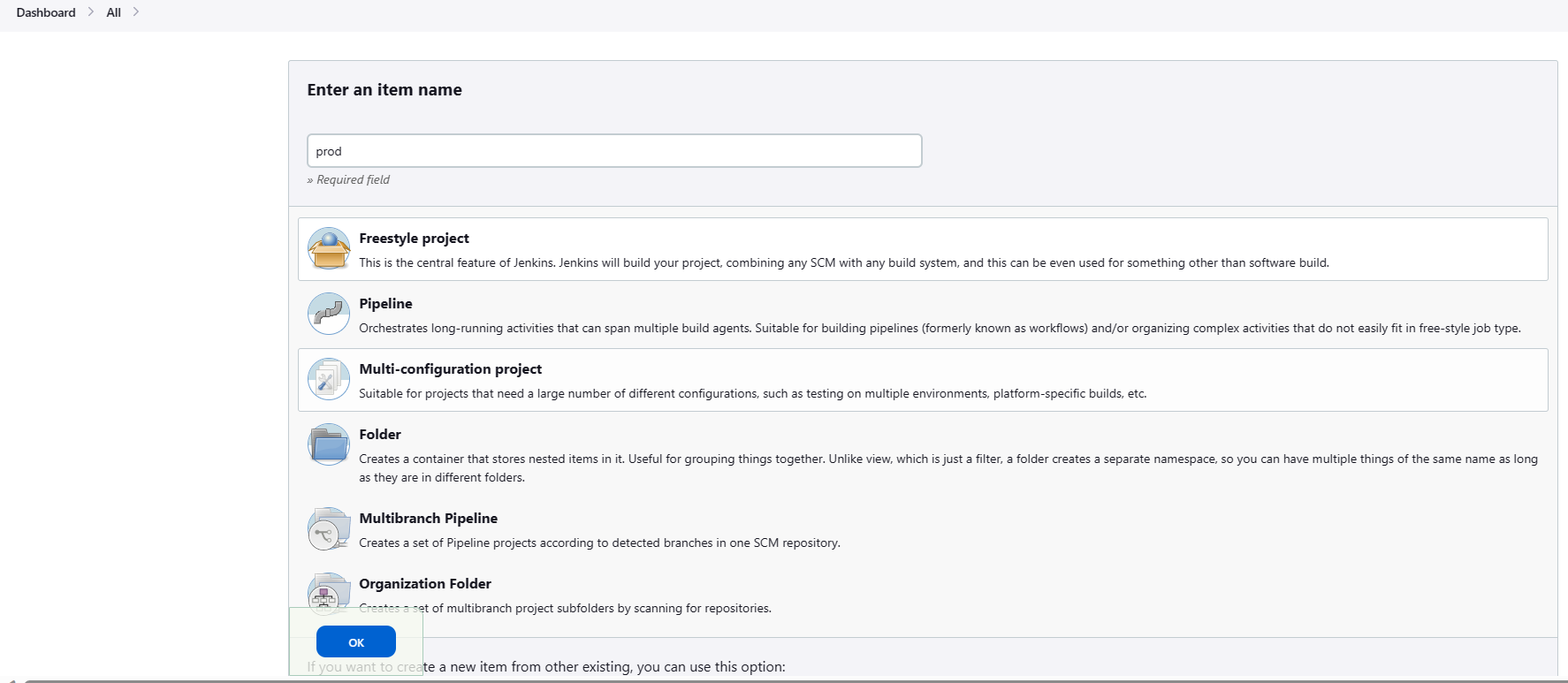
****

**Copy production machine public ip and open from browser in port 83**

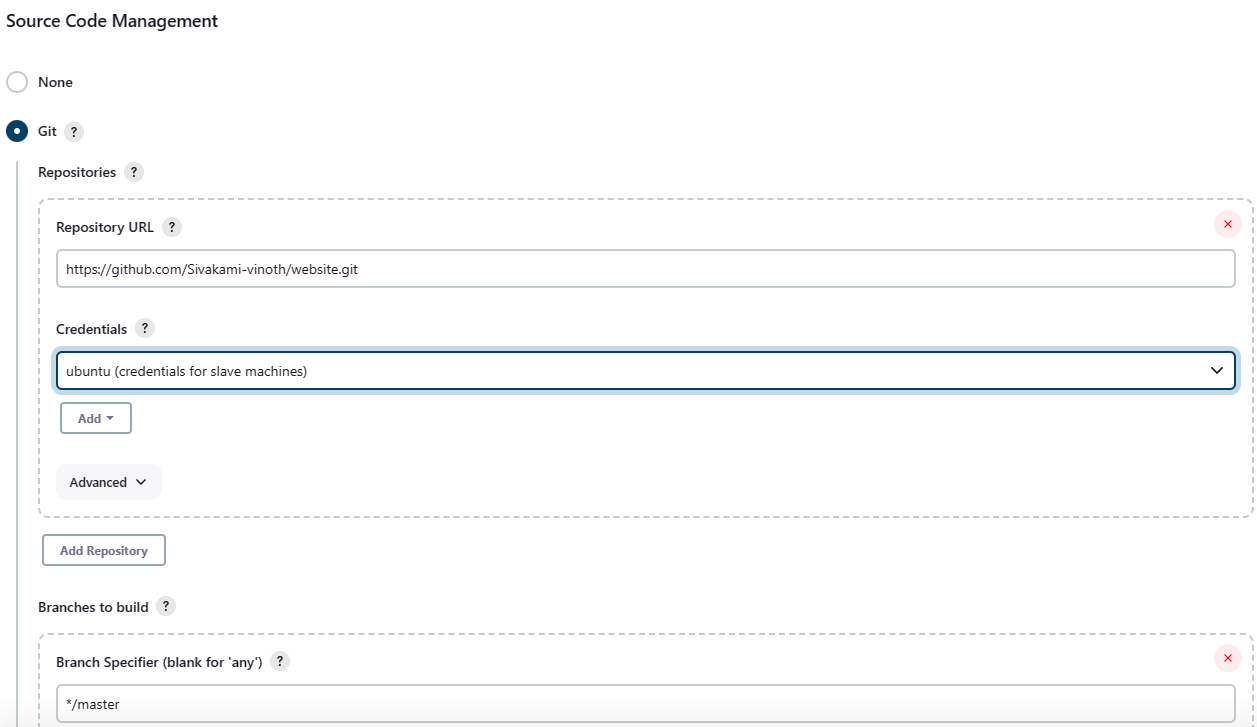
****

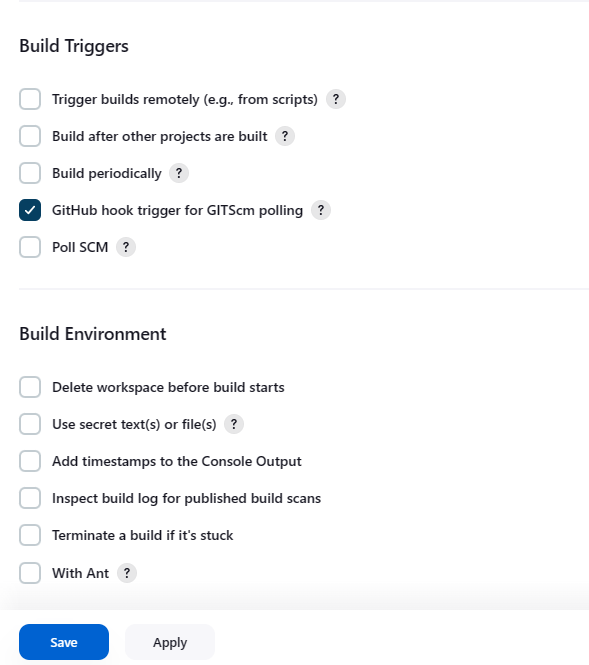
**Step:12 Create a job prod, will be triggered once commit is made to master branch and build job is triggered.**

**Create a job prod**

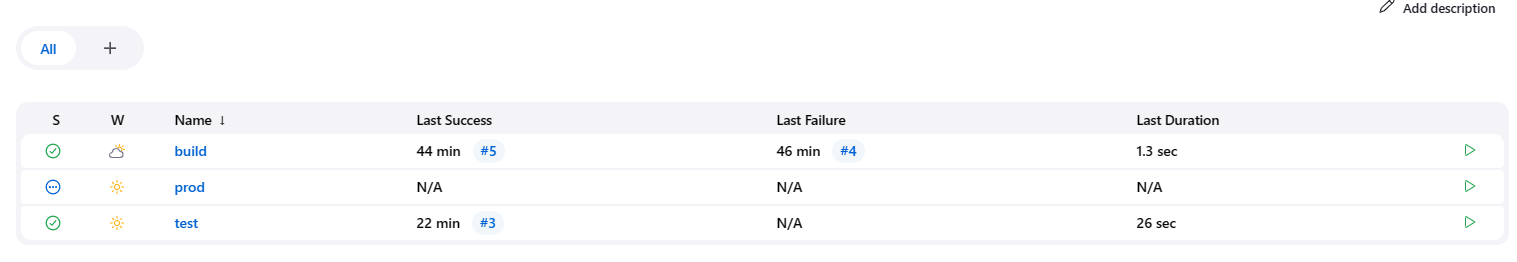
****

****

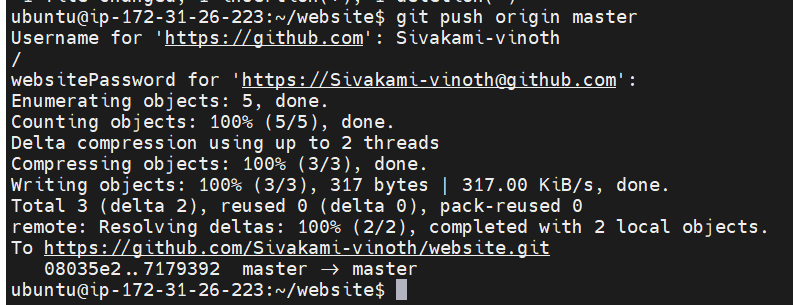
****

****

**Job prod created successfully**

****

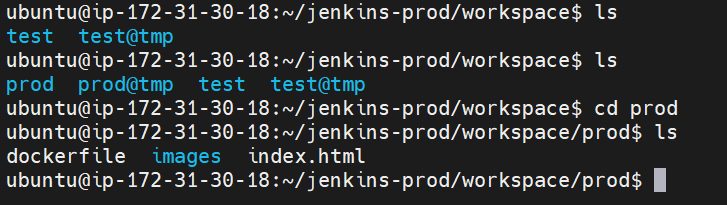
**Push master branch to origin**

****

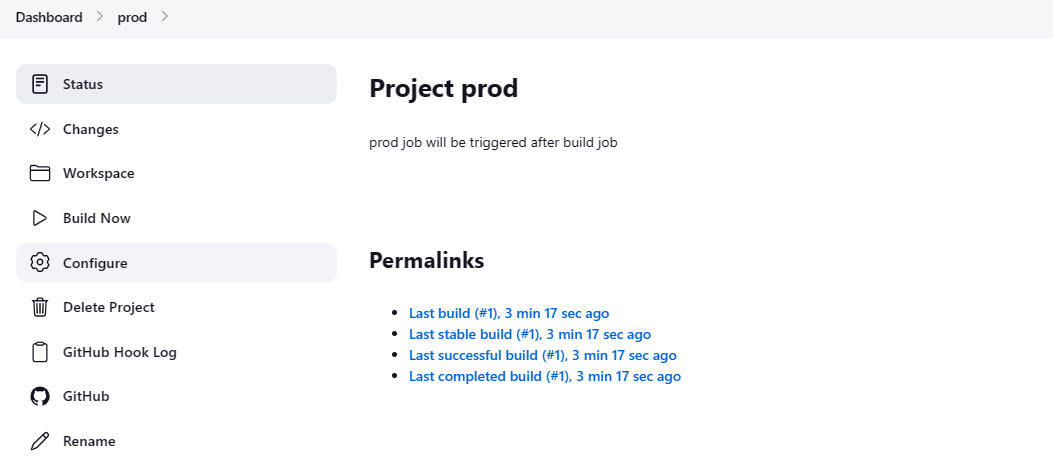
**Job prod got triggered**

****

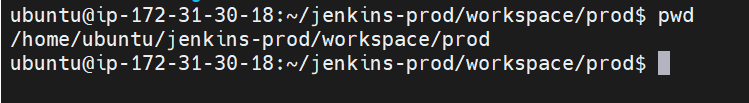
**Check the same from production machine**

****

**Open the configuration of prod job**

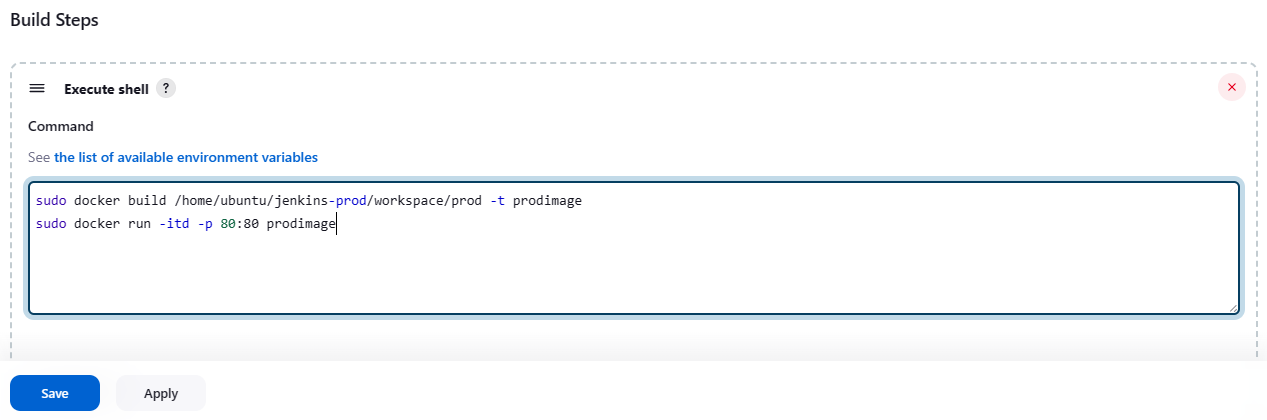
****

**Copy the path**

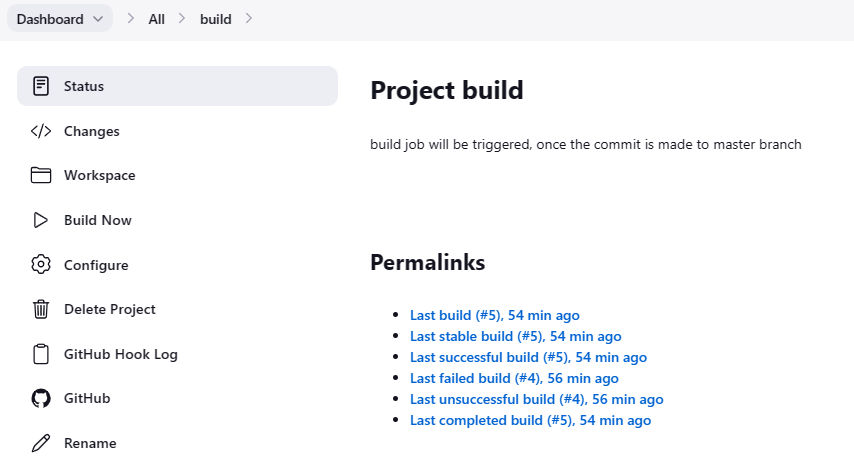
****

**Add build steps-> execute shell**

**Create docker image prodimage and run that in port 80**

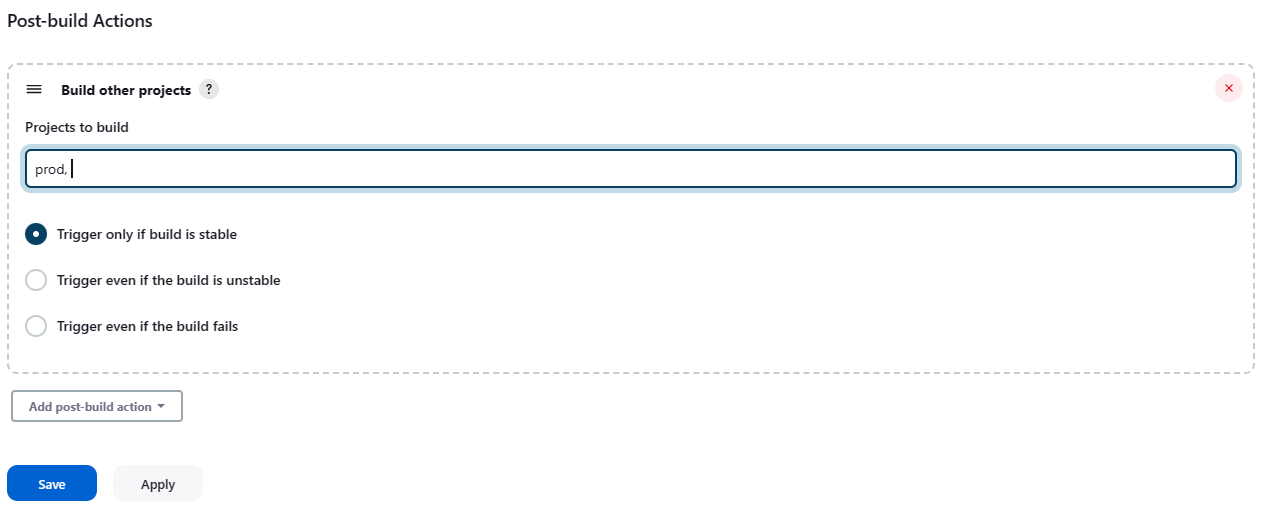
****

**Open the configuration of build job**

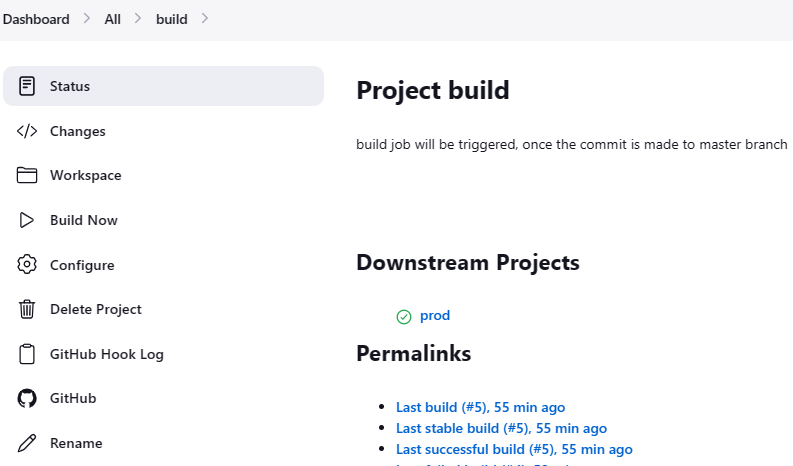
****

**Add post-build Actions**

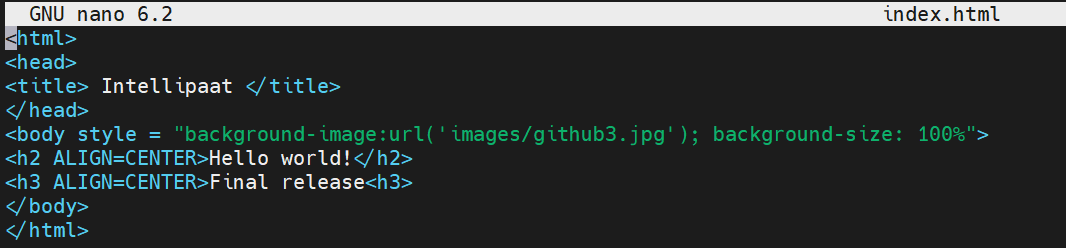
**Select job prod**

****

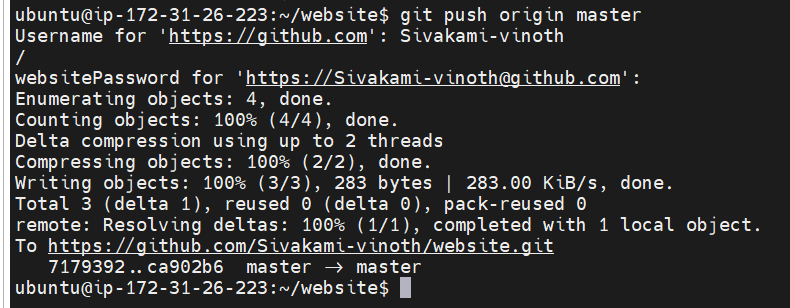
**Configured successfully**

****

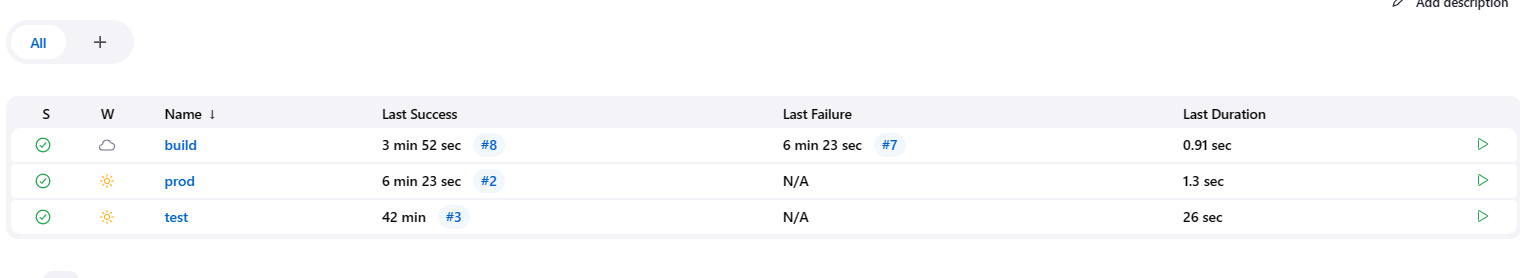
**Edit the index.html file**

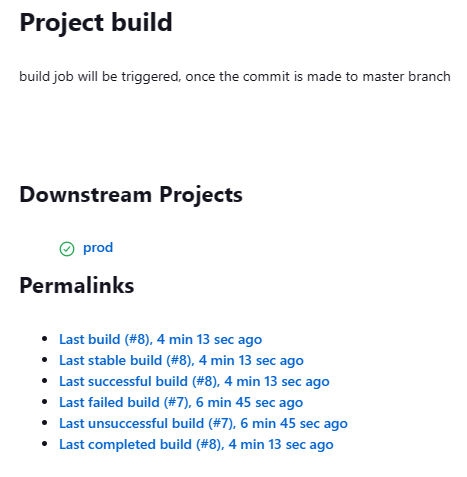
****

**Add and commit the index.html file and push the master to origin**



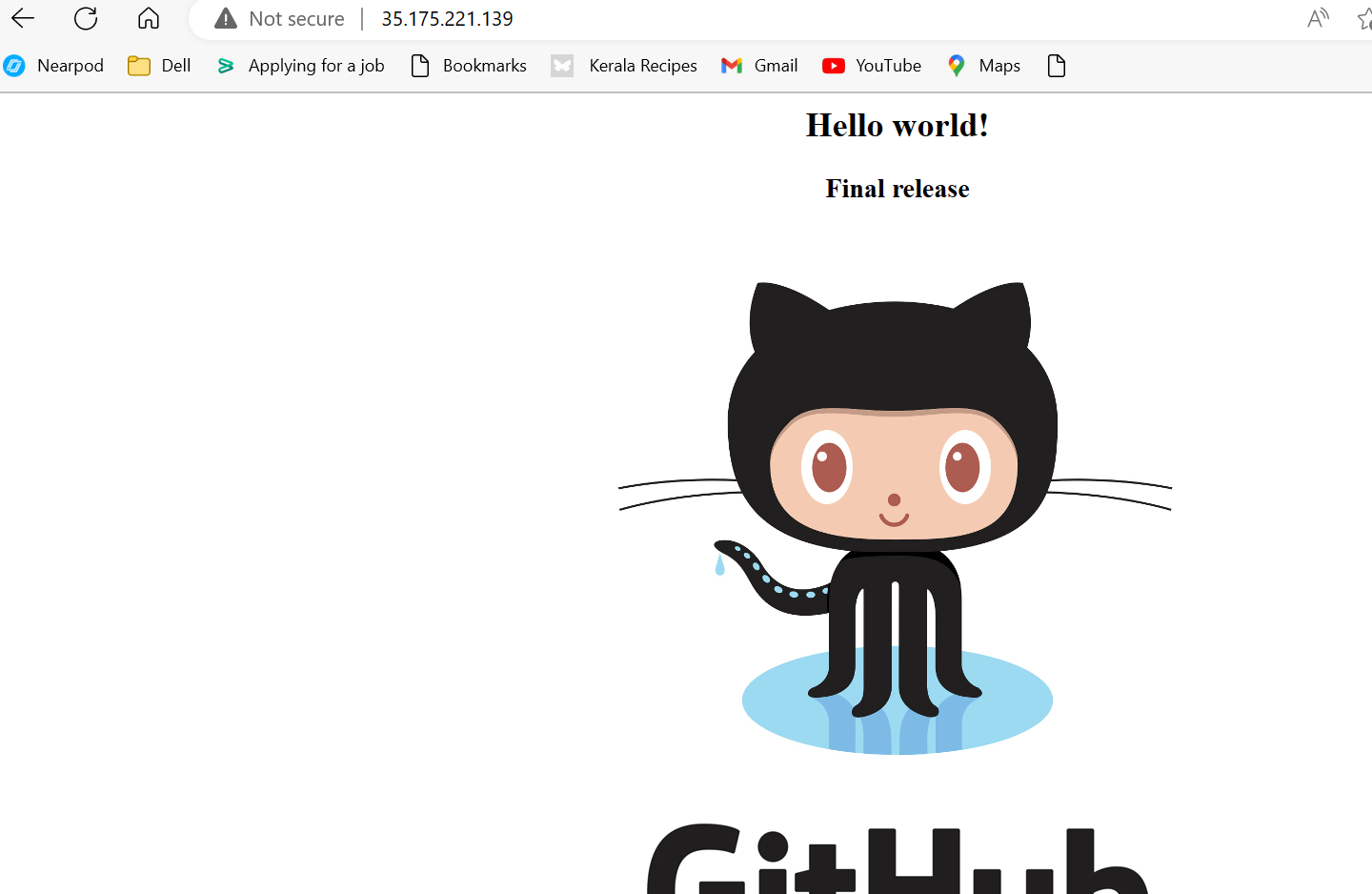
Job build and prod triggered successfully







**Open the public ip of production machine in port 80**

****