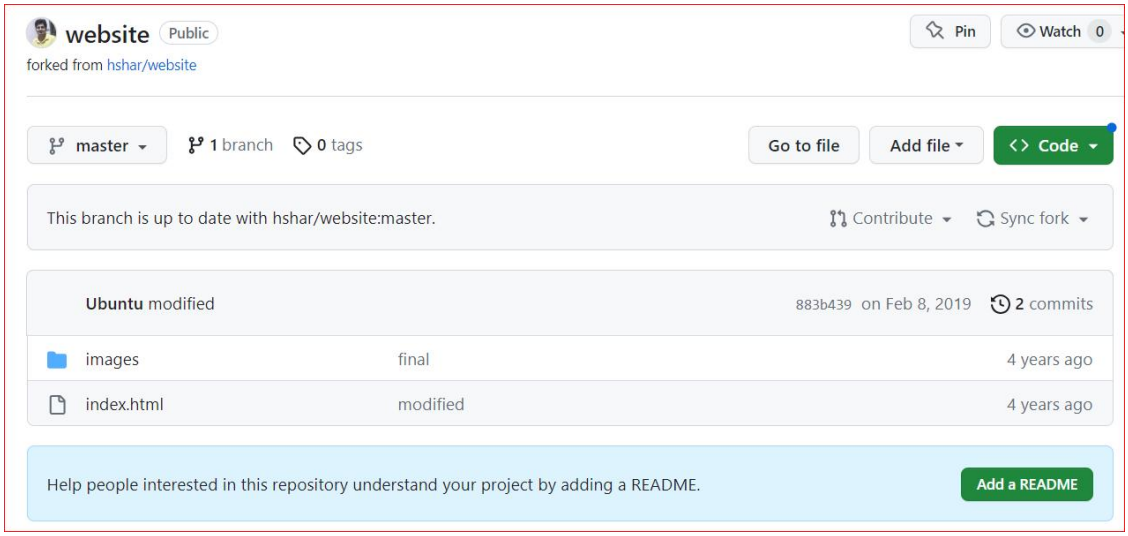
Module 6: Case Study - Integration of DevOps Tools with Jenkins

Problem Statement:

You have been hired as a DevOps Engineer in XYZ software company. They want to implement CI/CD pipeline in their company. You have been asked to implement this lifecycle as fast as possible. As this 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 continuous integration:
1. Git workflow has to be implemented
2. CodeBuild should automatically be triggered once a commit is made to master branch or develop branch. If a commit is made to master branch, build and publish a website on port 82. If a commit is made to develop a
branch, just build the product, do not publish.
3. Create a pipeline for the above tasks
4. Create a container with Ubuntu and Apache installed in it and use that container to build the code and the code should be on '/var/www/html'.

Solution:-

* Fork this repo to your git hub account:-
https://github.com/hshar/website.git




```
$ which git
$ mkdir Git
$ cd Git
$ sudo git init
$ ls -al
```

```
ubuntu@Jenkins-Master:~$ which git
/usr/bin/git
ubuntu@Jenkins-Master:~$ mkdir Git
ubuntu@Jenkins-Master:~$ cd Git/
ubuntu@Jenkins-Master:~/Git$ sudo git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /home/ubuntu/Git/.git/
ubuntu@Jenkins-Master:~/Git$ ls -al
total 12
drwxrwxr-x 3 ubuntu ubuntu 4096 Aug 27 07:12 .
drwxr-x--- 6 ubuntu ubuntu 4096 Aug 27 07:10 ..
drwxr-xr-x 7 root   root   4096 Aug 27 07:12 .git
ubuntu@Jenkins-Master:~/Git$
```

```
$ docker -v
```

```
root@Jenkins-Master:~# docker -v
Docker version 24.0.5, build ced0996
root@Jenkins-Master:~#
```

$ git clone https://github.com/github-amit-repository/website.git

```
ubuntu@Jenkins-Master:~/Git$ git clone https://github.com/github-amit-repository/website.git
Cloning into 'website'...
remote: Enumerating objects: 8, done.
remote: Total 8 (delta 0), reused 0 (delta 0), pack-reused 8
Receiving objects: 100% (8/8), 82.69 KiB | 9.19 MiB/s, done.
Resolving deltas: 100% (1/1), done.
```

$ cd website
$ vi dockerfile
$ cat dockerfile

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

```
ubuntu@Jenkins-Master:~/Git$ cd website
ubuntu@Jenkins-Master:~/Git/website$ sudo vi dockerfile
ubuntu@Jenkins-Master:~/Git/website$ cat dockerfile
FROM ubuntu
RUN apt get update -y
RUN apt get install apache2 -y
ADD . /var/www/html
ENTRYPOINT apachectl -D FOREGROUND
```

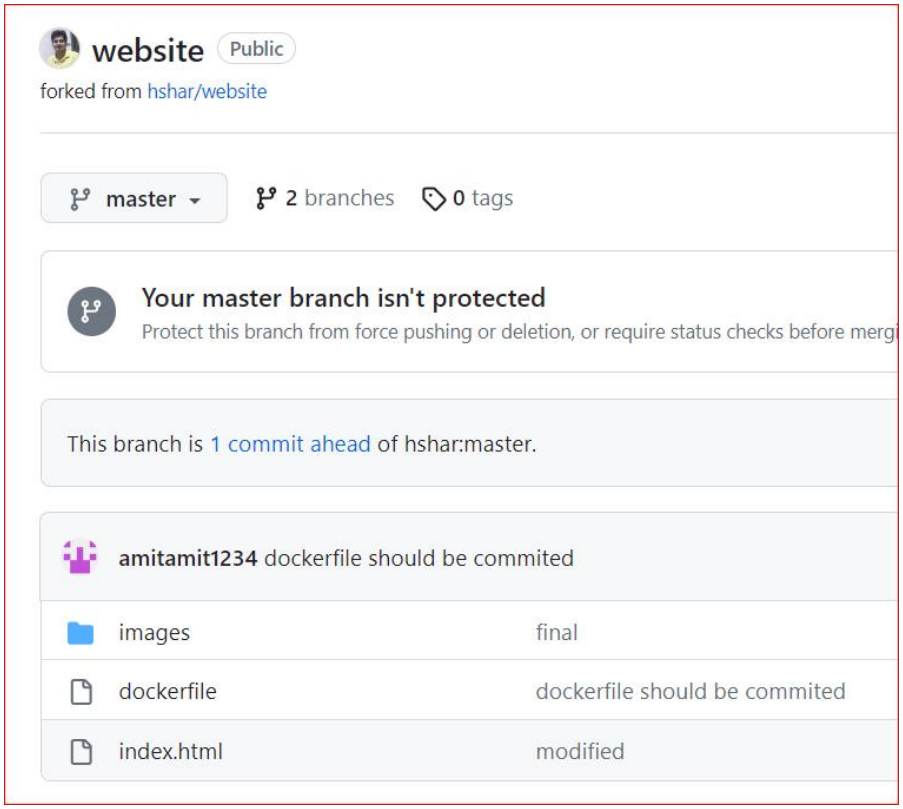
$ git add .
$ git commit -m “dockerfile should be commited”

```
ubuntu@Jenkins-Master:~/Git/website$ git add .
ubuntu@Jenkins-Master:~/Git/website$ git commit -m "dockerfile should be commited"
[master 87eb8d1] dockerfile should be commited
 1 file changed, 5 insertions(+)
 create mode 100644 dockerfile
```

$ git branch && git branch develop && git checkout develop && ls && git checkout master && git push --all

```
ubuntu@Jenkins-Master:~/Git/website$ git branch && git branch develop && git checkou
t develop &&  ls && git checkout master && git push --all
* master
Switched to branch 'develop'
dockerfile  images  index.html
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)
Username for 'https://github.com': github-amit-repository
Password for 'https://github-amit-repository@github.com':
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 420 bytes | 420.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/github-amit-repository/website.git
   883b439..87eb8d1  master -> master
 * [new branch]      develop -> develop
```

* Now in the github repo we can see files:-



Prod:-

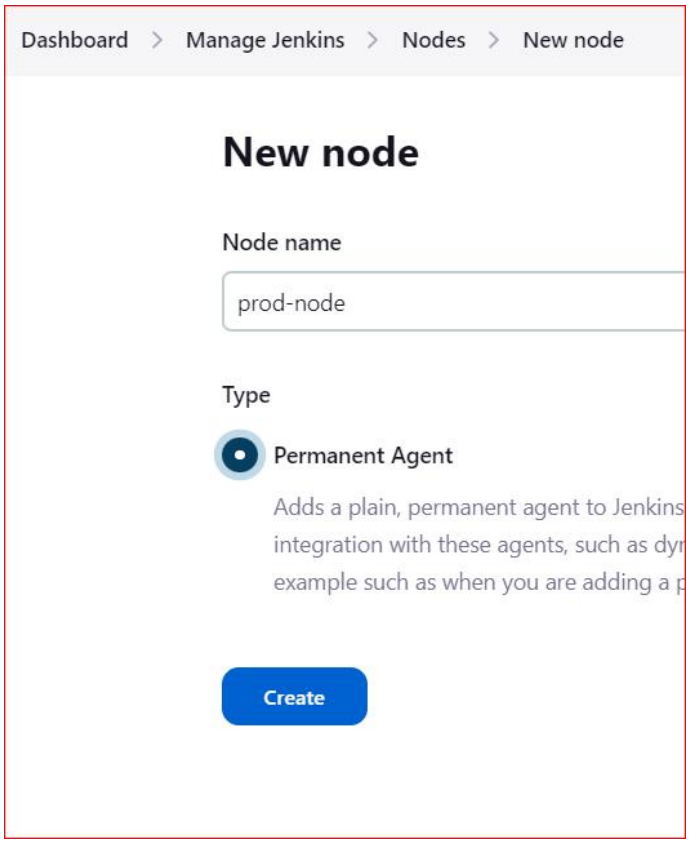
$ sudo apt install docker.io -y

```
ubuntu@prod-server:~$ sudo apt install docker.io -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
```

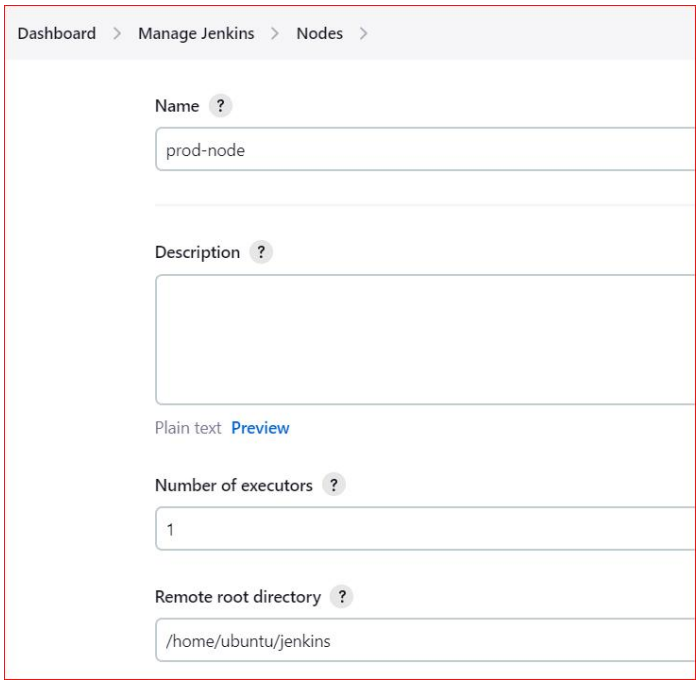
$ docker -v

```
ubuntu@prod-server:~$ docker -v
Docker version 20.10.25, build 20.10.25-0ubuntu1~22.04.2
ubuntu@prod-server:~$
```

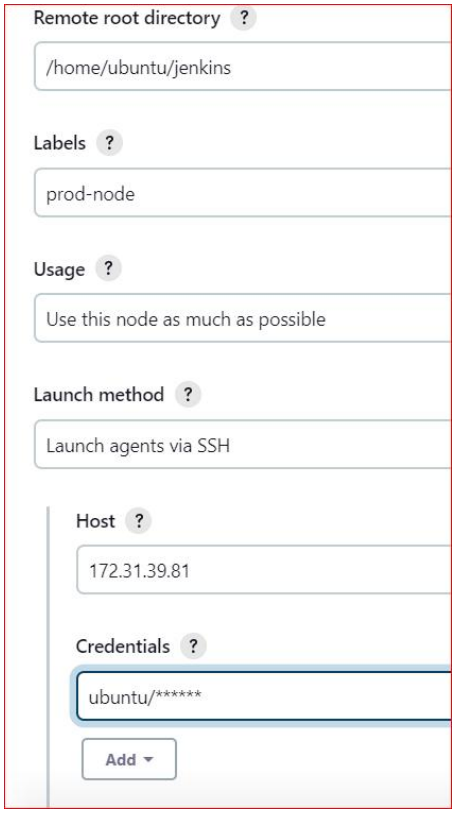
* Create new node:- dashboard->manage jenkins -> nodes -> new node -> prod-node-> select permanent agent->create



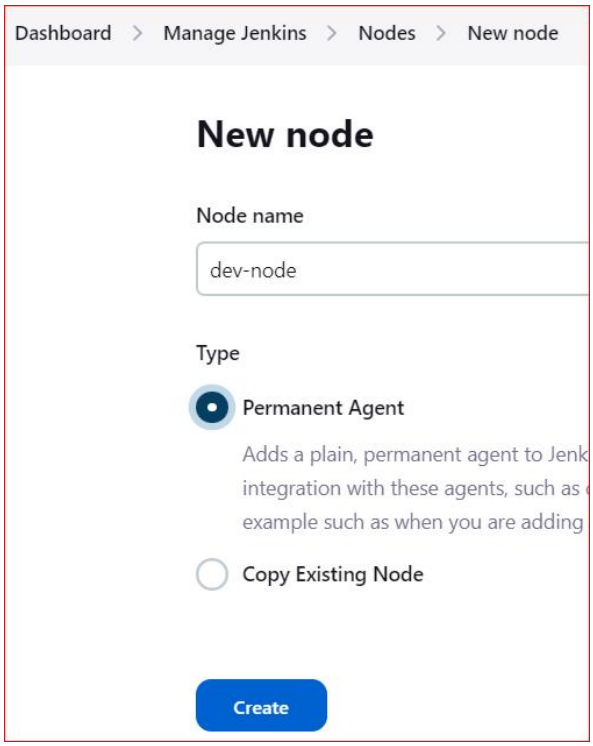
* Give details for node:- Remote Root Directory -> /home/ubuntu/jenkins



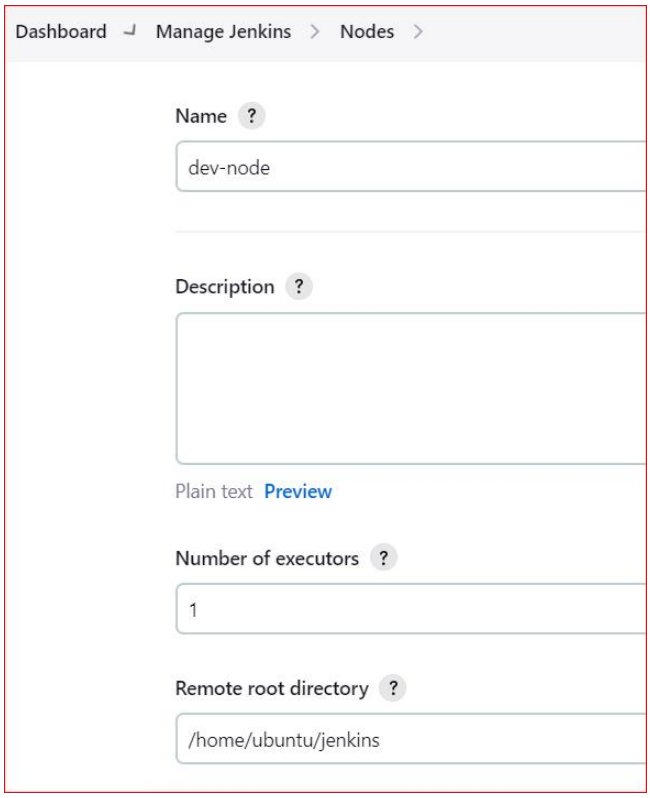
* Give:- Labels-> prod-node->launch method-> launch agent via SSH-> host-> give credentials



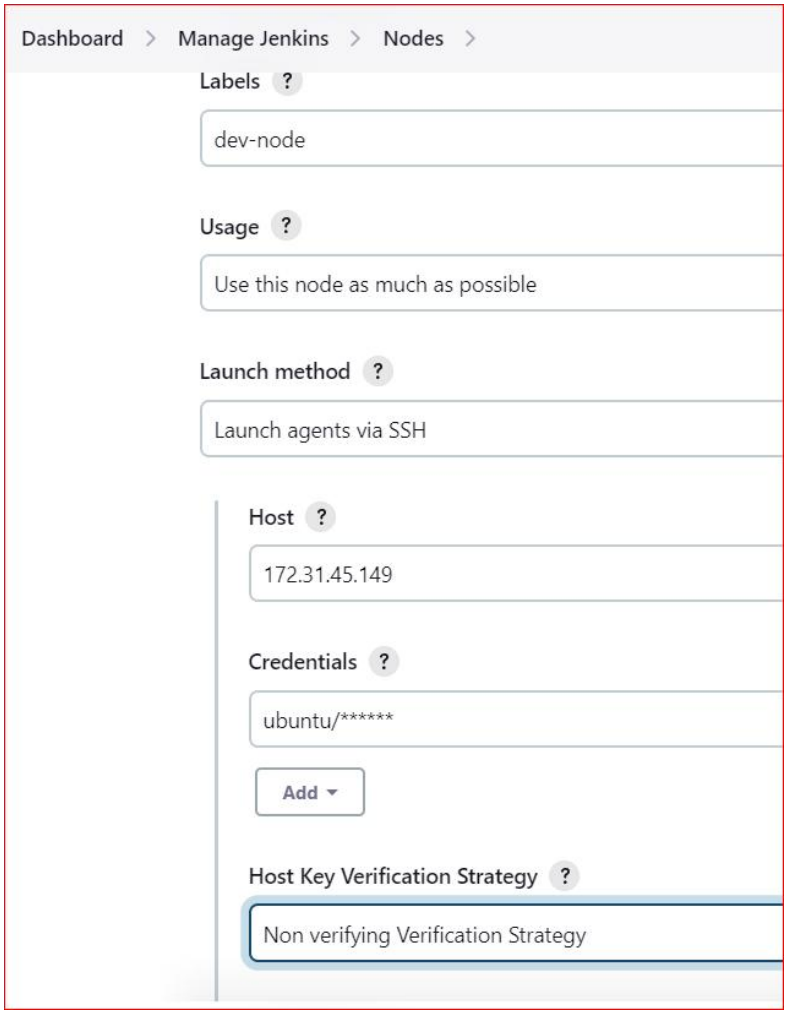
* Now create dev-node nodes:- dashboard->manage jenkins-> nodes->new nodes-> select permanent agent->create



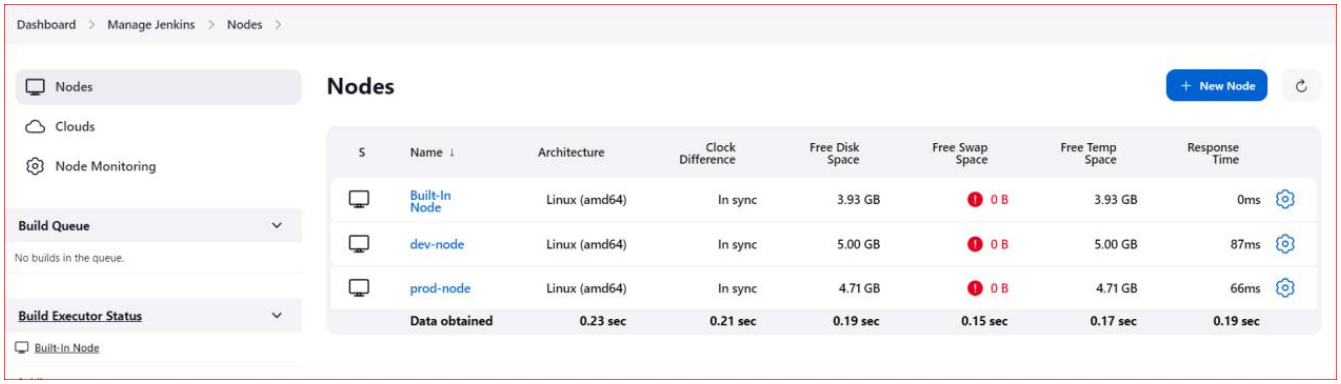
* Insert:- Remote root directory-> /home/ubuntu/jenkins



* Give:- Labels-> prod-node->launch method-> launch agent via SSH-> host-> give credentials
-> host key verification strategy->Non verifying verification strategy

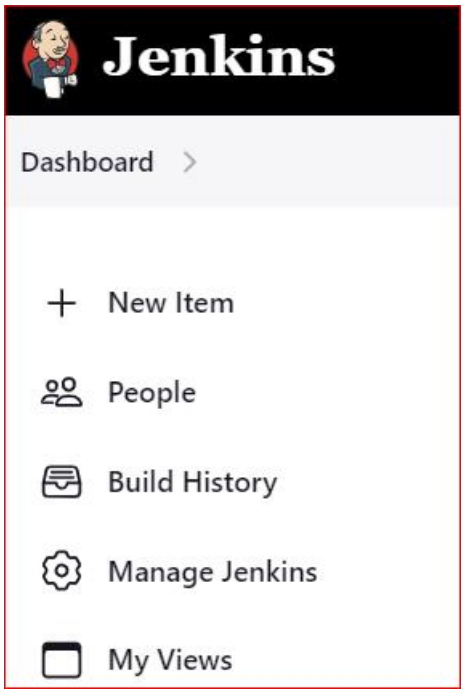


* Here we have nodes:-

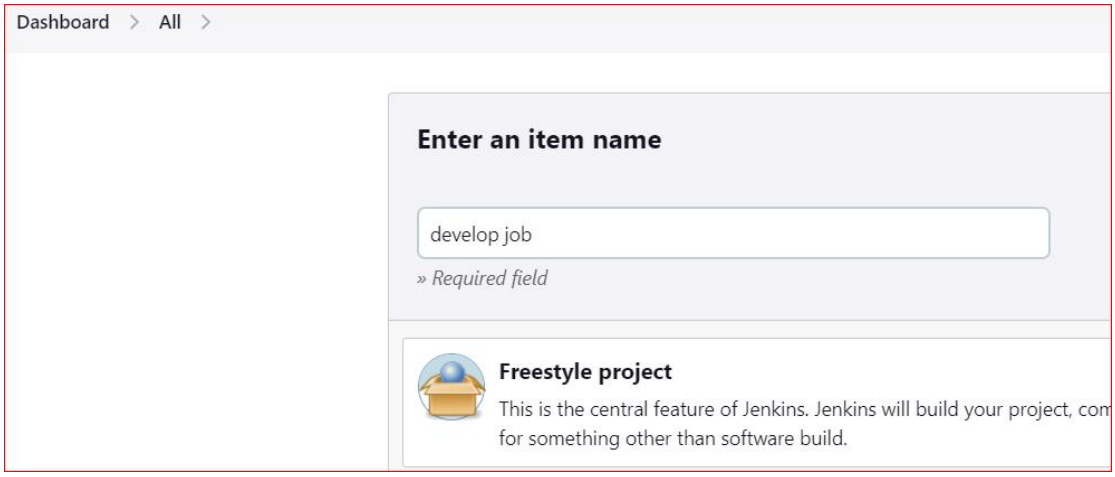


Jenkins:-

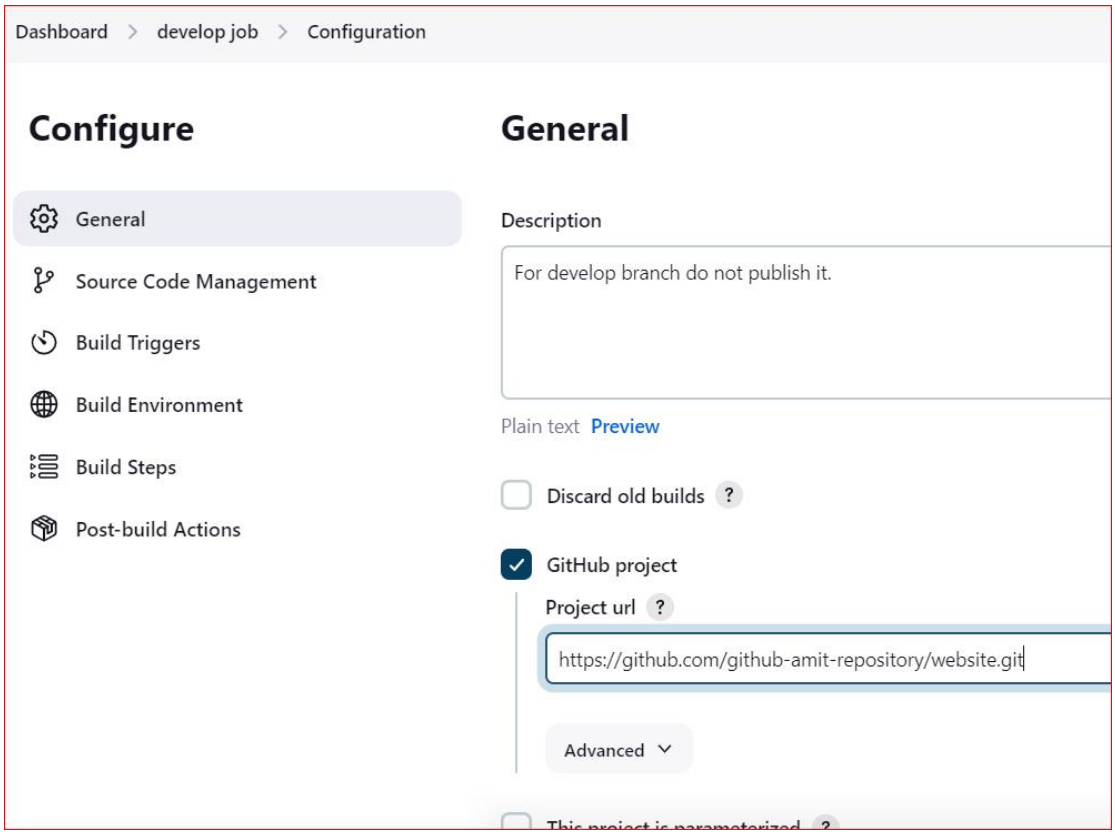
* Go to dashboard select New Item:-



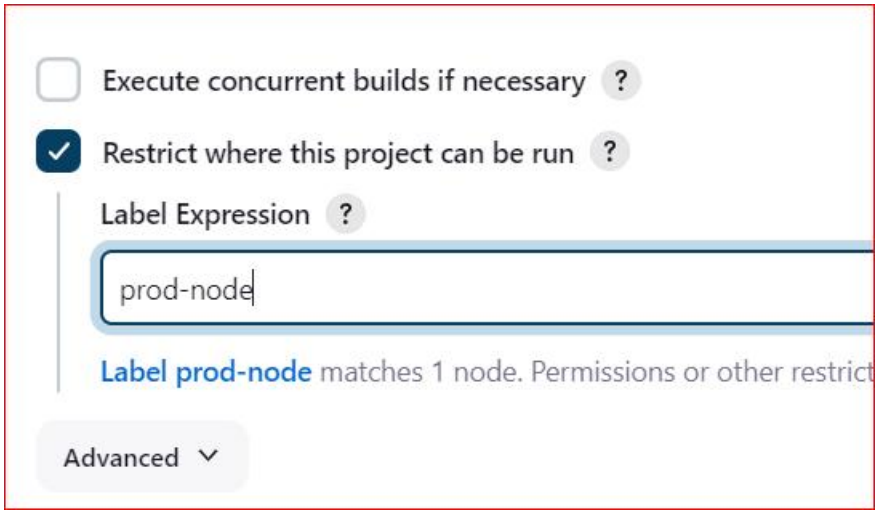
* Create job:- new items -> name (develop job) -> free style project -> ok



* Insert github url:- configure -> description-> For develop branch do not publish it. -> Github project -> project url (https://github.com/github-amit-repository/website.git)

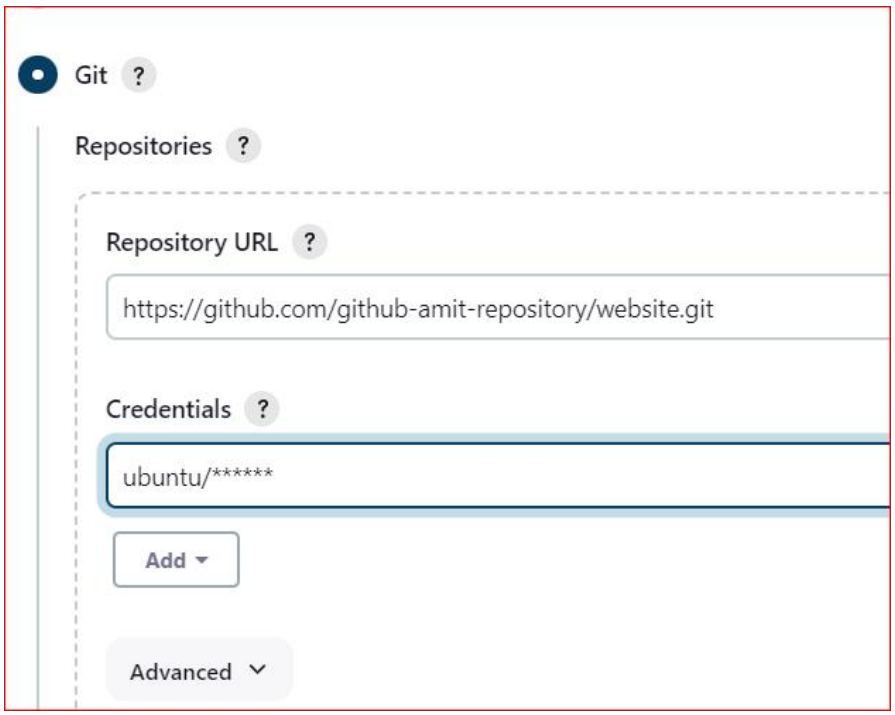


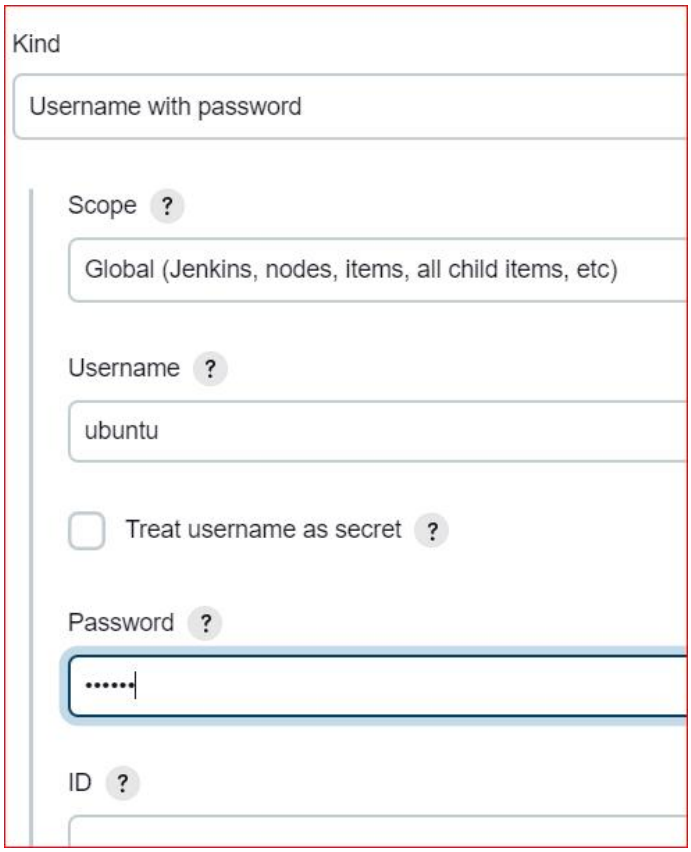
* Select:- restrict where this project can be run -> Label expression -> prod-node



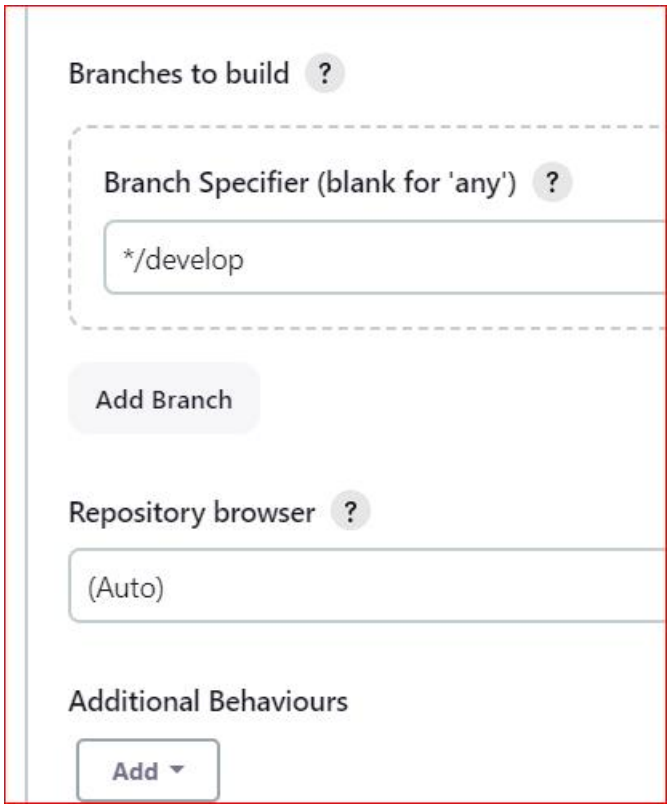
* Select and insert:-

source code management -> git -> repo url -> paste repo website url -> give credential means insert username and password

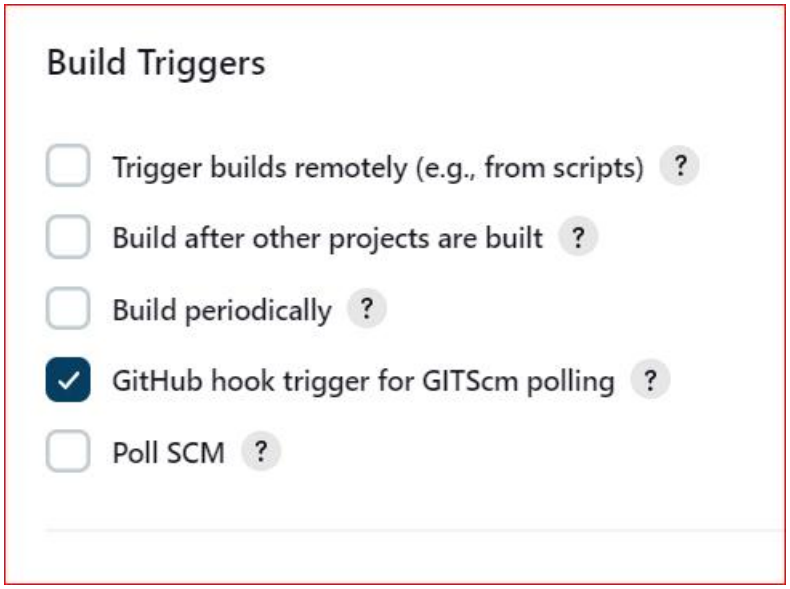




* Give branch:- */develop



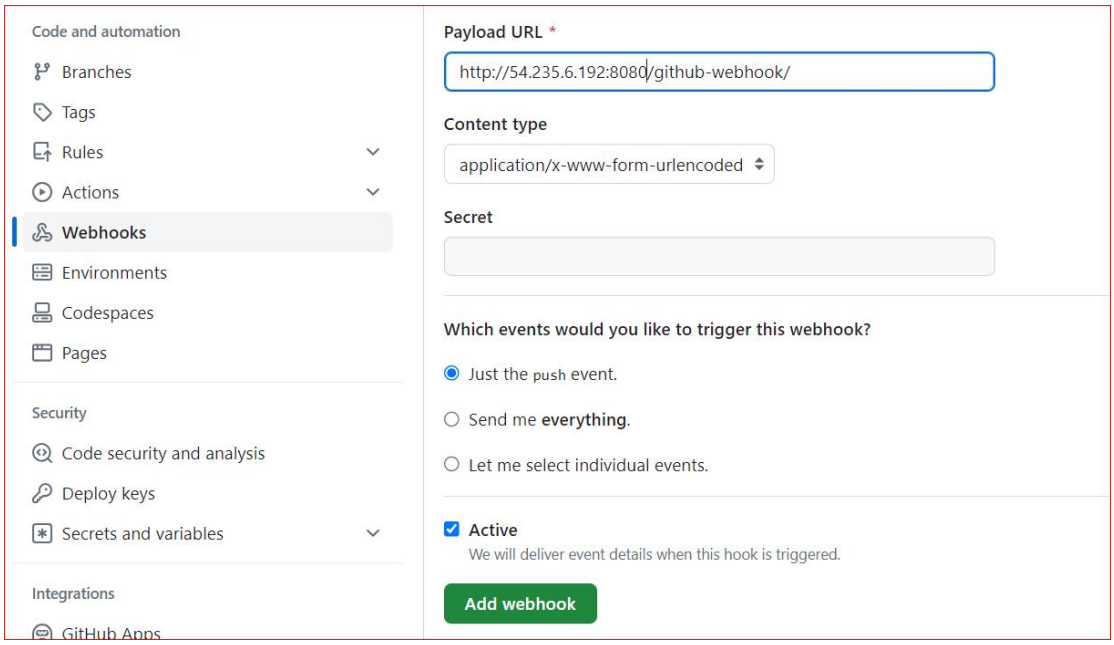
* Select gitscm:- build triggers -> github hook trigger for gitscm polling -> apply -> save



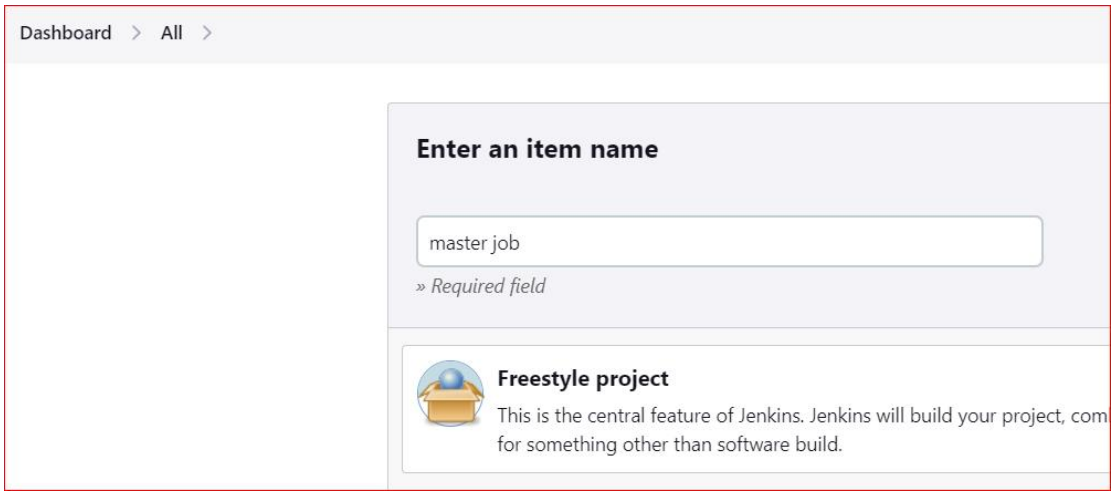
Project develop job

For develop branch do not publish it.

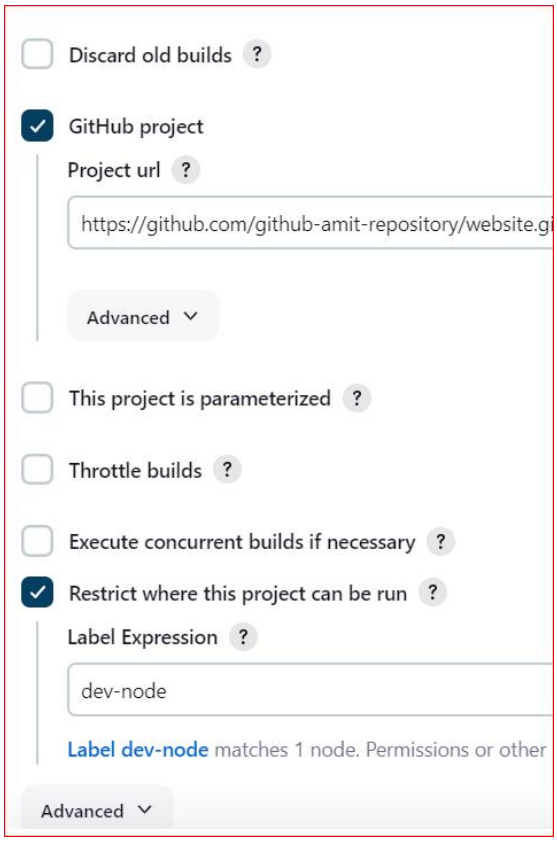
* Create webhook connection:- Github -> copy jenkins url and paste in github website repo -> github repo -> settings -> webhooks -> add webhook -> payload url -> http://jenkins url till port/github-webhook/ -> add webhook -> webhooks/manage webhook/ recent deliveries -> save



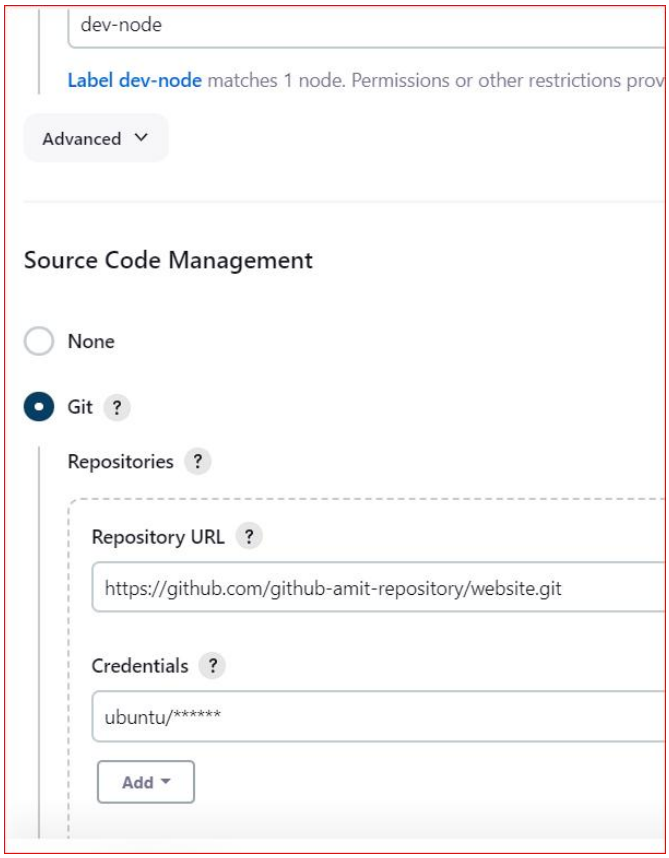
* Create job:- new items -> name (develop job) -> free style project -> ok



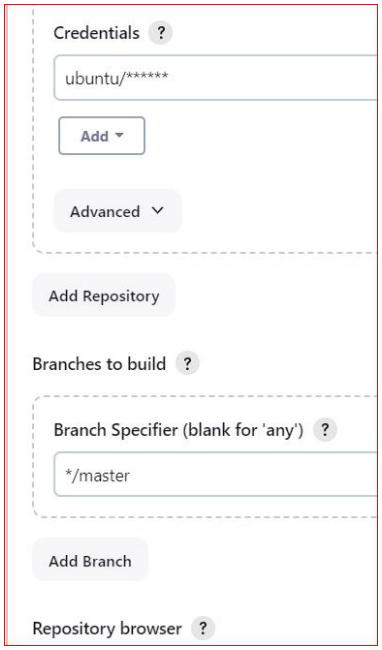
* Insert github url:- configure -> description-> For develop branch do not publish it. -> Github project -> project url (https://github.com/github-amit-repository/website.git) -> restrict where this project can be run -> Label expression -> dev-node



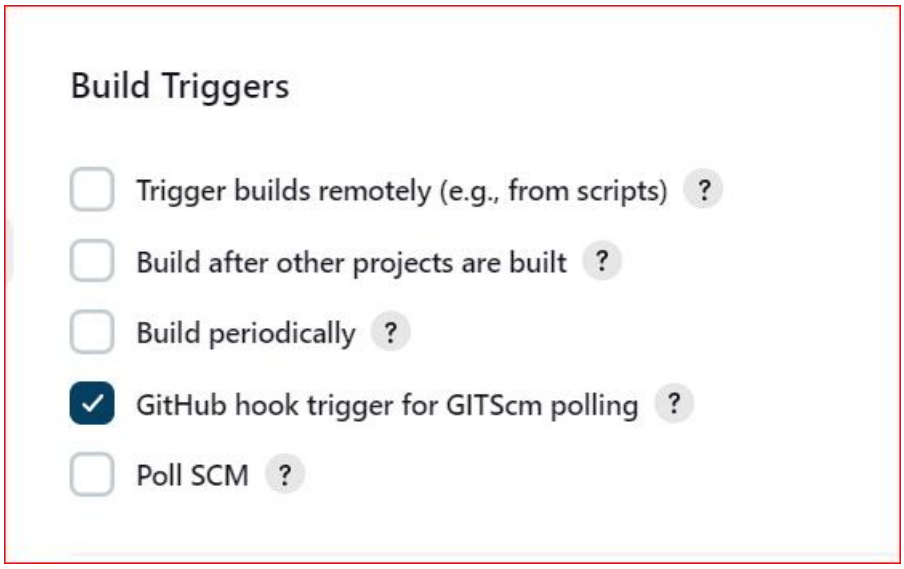
* Select and insert:- source code management -> git -> repo url -> paste repo website url -> give credential means insert username and password



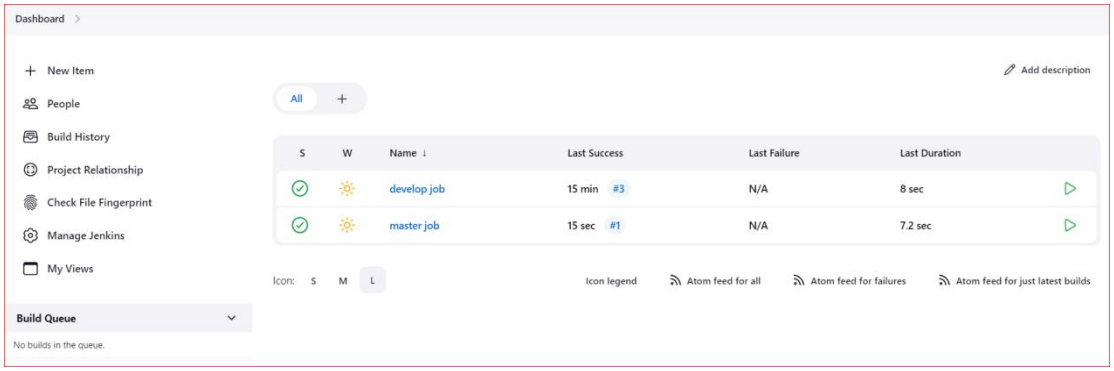
* Give branch:- */master



* Select gitscm:- build triggers -> github hook trigger for gitscm polling -> apply -> save



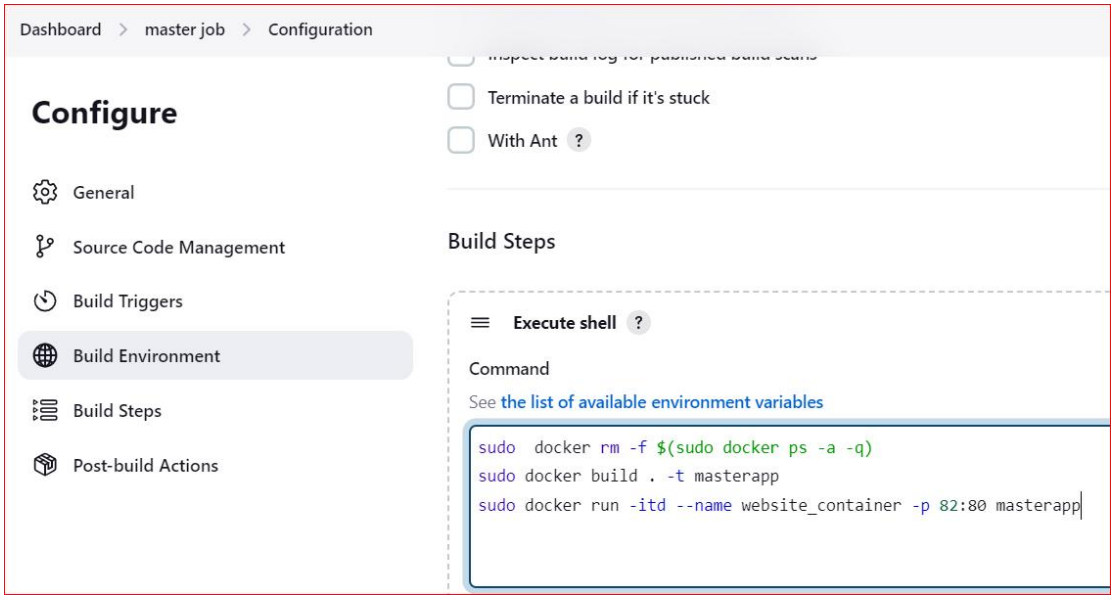
Now we have two jobs and now click build now:-



* Now in master job add this:- Master Job->build environment->build steps-> add build step -> execute -> in execute shell -> command ->

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

-> apply-> save -> build now



Now we have dockerfile in both server:-

$ ls
$ cd workspace
$ ls
$ cd develop\ job
$ ls

```
ubuntu@prod-server:~/jenkins$ ls
remoting  remoting.jar
ubuntu@prod-server:~/jenkins$ ls
remoting  remoting.jar  workspace
ubuntu@prod-server:~/jenkins$ cd workspace/
ubuntu@prod-server:~/jenkins/workspace$ ls
'develop job'  'develop job@tmp'
ubuntu@prod-server:~/jenkins/workspace$ cd develop\ job
ubuntu@prod-server:~/jenkins/workspace/develop job$ ls
dockerfile  images  index.html
ubuntu@prod-server:~/jenkins/workspace/develop job$
```

$ ls
$ cd workspace
$ ls
$ cd master\ job
$ ls

```
ubuntu@test-server:~/jenkins$ ls
remoting  remoting.jar  workspace
ubuntu@test-server:~/jenkins$ cd workspace/
ubuntu@test-server:~/jenkins/workspace$ ls
'master job'  'master job@tmp'
ubuntu@test-server:~/jenkins/workspace$ cd master\ job
ubuntu@test-server:~/jenkins/workspace/master job$ ls
dockerfile  images  index.html
ubuntu@test-server:~/jenkins/workspace/master job$
```

* Browser-> http://34.229.44.2:82