DevSecOps Training

DAY – 4

Today's IP Address details:

S.No	Ip Address- Day4	User Name	Password	Participants Name
Trainer	44.203.116.198	**root**	**redhat**	**Trainer**
1	44.204.230.147	root	redhat	alpana
2	44.203.45.243	root	redhat	baljeet
3	34.205.31.36	root	redhat	Devaraj
4	54.174.196.5	root	redhat	ganesh
5	44.203.179.83	root	redhat	kaustubh
6	3.93.81.178	root	redhat	manish
7	54.89.225.252	root	redhat	manoj
8	44.204.227.63	root	redhat	naveen
9	54.210.181.95	root	redhat	neha
10	3.83.49.123	root	redhat	pravallika
11	18.212.191.61	root	redhat	ramanand sai
12	44.204.52.144	root	redhat	revanth
13	3.82.212.219	root	redhat	rohit
14	3.83.119.15	root	redhat	rudra
15	35.171.3.106	root	redhat	sahitya
16	44.201.131.236	root	redhat	sampat
17	54.144.166.182	root	redhat	sangamesh
18	54.89.211.79	root	redhat	sashi
19	3.87.214.222	root	redhat	shashidhar
20	54.163.16.149	root	redhat	shweta
21	52.87.213.141	root	redhat	sudheer
22	3.84.5.212	root	redhat	uday
23	3.82.197.38	root	redhat	vaishnavi
24	18.204.206.0	root	redhat	vishali
25	54.166.60.234	root	redhat	sankalp
26	34.203.188.178	root	redhat	balakrishna

Details of my machine:

IP Address: 18.204.206.0

Username: root

Password: redhat

Available RAM space for the trainer is 8GB

Available RAM space: 2GB (For participants)

```
[root@vishali ~]# free -m
              total        used        free      shared  buff/cache   available
Mem:           1971          91        1351           0         527        1742
Swap:             0           0           0
[root@vishali ~]#
```

Machines are stopped due to the less RAM Space. We need atleast 4GB for Jenkins. We can use less GB but the performance will be less

---

Agenda:

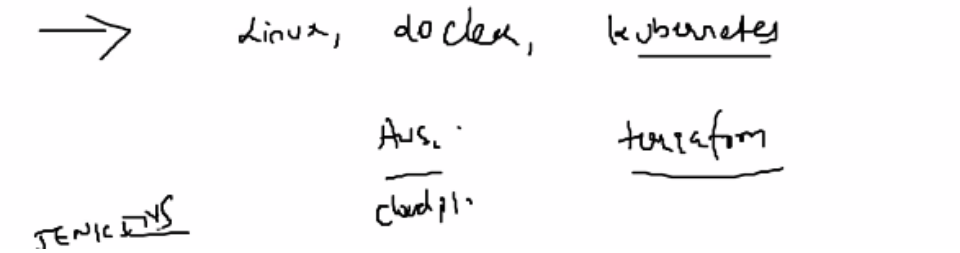
- Jenkins Pipeline
- Pipeline Security

What is the language of Jenkins?

Jenkins is an open-source continuous integration/continuous delivery and deployment (CI/CD) automation software DevOps tool written in the **Java programming language**.

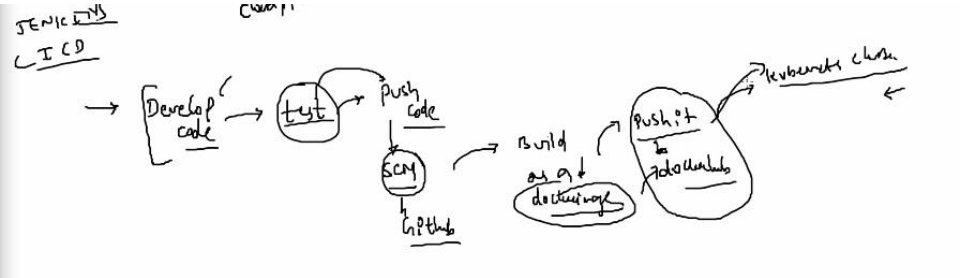
What tools we have used till now:

Linux, docker, Kubernetes, AWS(cloud), Terraform (to automate)



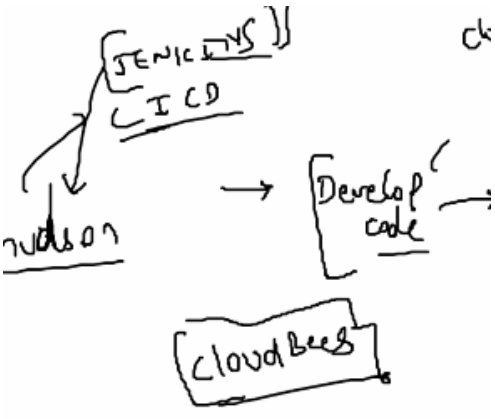
Jenkins – CI/CD:

- Develop the code
- Test the code
- Push the code – Github Account
- Build a docker image – can do it
- Push it to docker Hub
- Deploy it in Kubernetes cluster platform



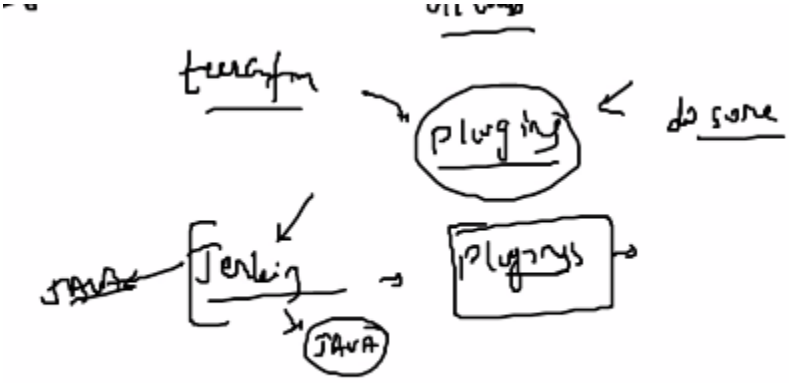
All these things are manually done. To automate this process, we can do it in Jenkins.

Jenkins is managed by: Cloud Bees



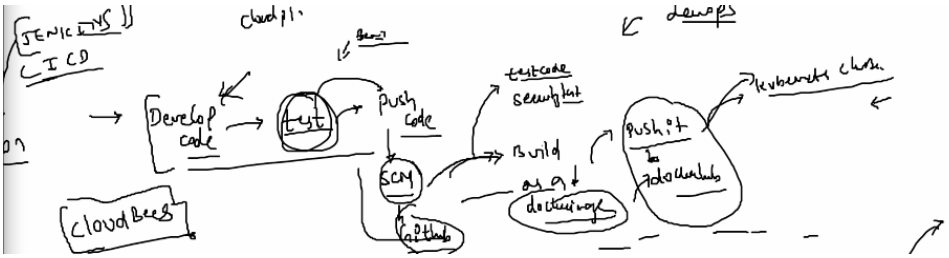
Jenkins mainly depends on the Plugins.

Jenkins created by language JAVA.



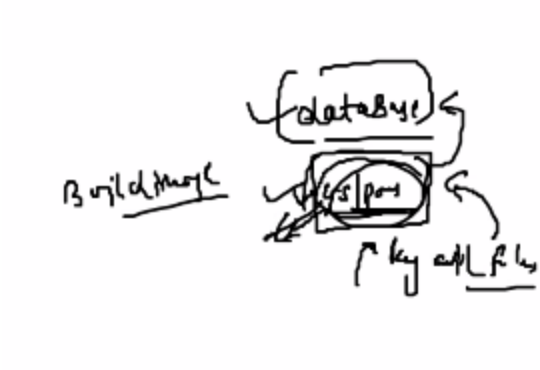
Where do we need security?

We can apply on the test phase, push, ….



If you create a database, you need a password and username to access it? Yes

It isn't a security breach? Yes

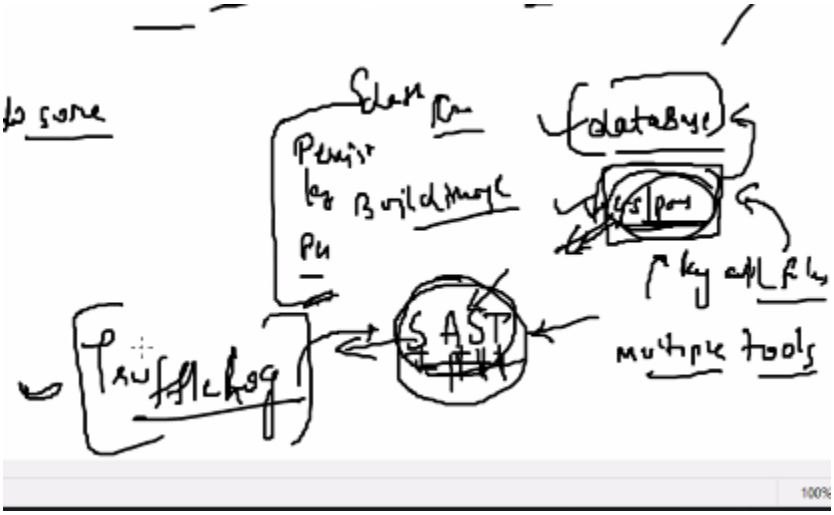


SAST – Static analysis security Testing – The above process is called SAST

Who will do it? Multiple tools

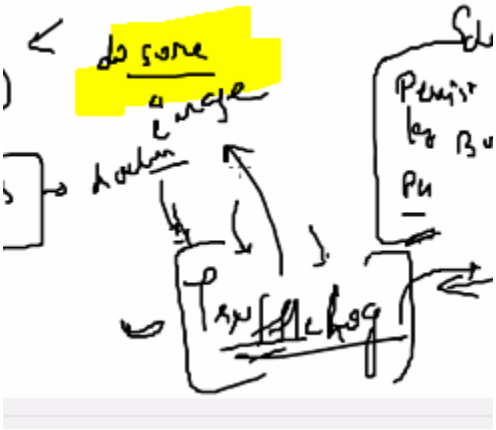
Truffle Log: Tool help to test SAST/ security testing – used to test the password, username.

This is the later stage once we have the pipeline.



Problem with this tool? Complex while install:

Solution: User the docker image:



Before using this above tool, we need to build the pipeline:

Flow: Develop – push – dockerfile – image – push it – deploy – Kubernetes.

Jenkins default port number:

The default Jenkins installation runs on ports **8080 and 8443**. Typically, HTTP/HTTPS servers run on ports 80 and 443, respectively.

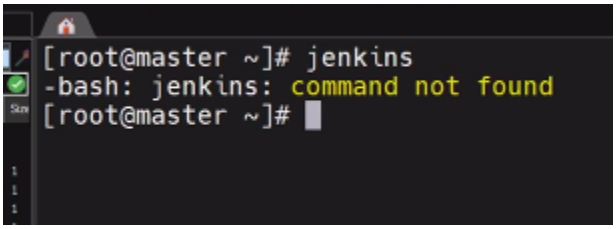
Jenkins installation in AWS Linux:

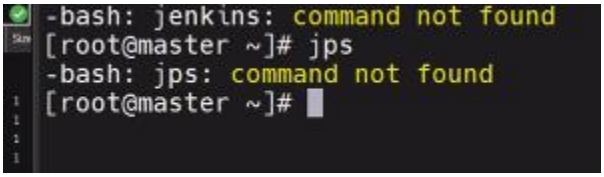
Dh –h command – for checking storage:

dh –h

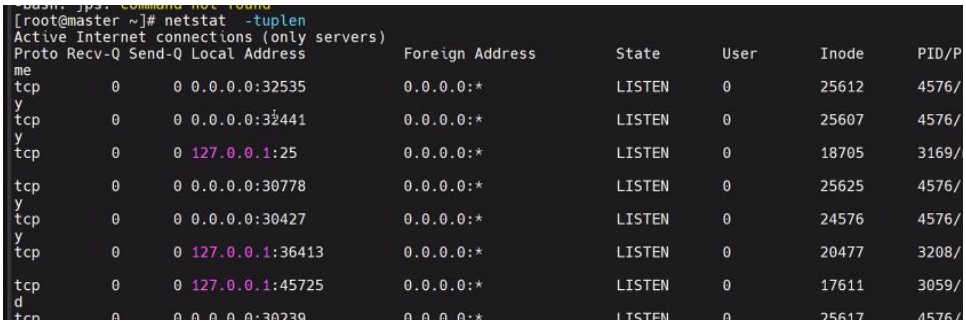
We are good with the storage.

Check whether Jenkins is present in the machine:





How many ports are currently occupied?



Check what is running in port 22:

```
[root@master ~]# netstat  -tuplen | grep -i 22
tcp        0      0 0.0.0.0:31776           0.0.0.0:*               LISTEN
y
tcp        0      0 0.0.0.0:31522           0.0.0.0:*               LISTEN
y
tcp        0      0 0.0.0.0:32232           0.0.0.0:*               LISTEN
y
tcp        0      0 172.31.11.86:2379       0.0.0.0:*               LISTEN

tcp        0      0 127.0.0.1:2379          0.0.0.0:*               LISTEN

tcp        0      0 172.31.11.86:2380       0.0.0.0:*               LISTEN

tcp        0      0 127.0.0.1:10259         0.0.0.0:*               LISTEN
duler
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN

tcp6       0      0 :::6443                 :::*                    LISTEN
erver
tcp6       0      0 :::22                   :::*                    LISTEN

udp6       0      0 fe80::c2ff:feb3:fb7:546 :::*
```

To check what is running in port 80?

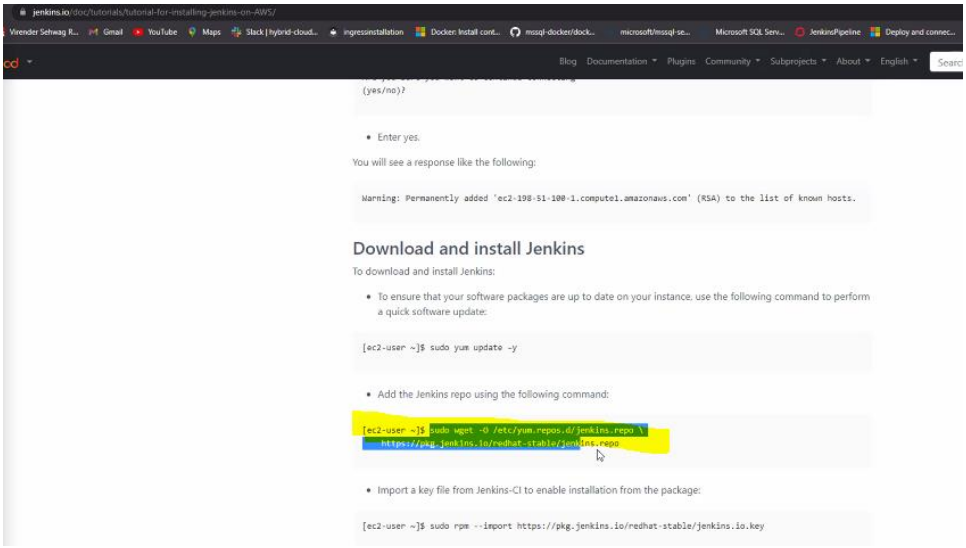
```
[root@master ~]# netstat  -tuplen | grep -i 80
tcp        0      0 172.31.11.86:2380       0.0.0.0:*               LISTEN      0          22939

tcp        0      0 0.0.0.0:30480           0.0.0.0:*               LISTEN      0          25609
y
udp6       0      0 fe80::c2ff:feb3:fb7:546 :::*                                0          16224
```

To check what is running in port 8080?

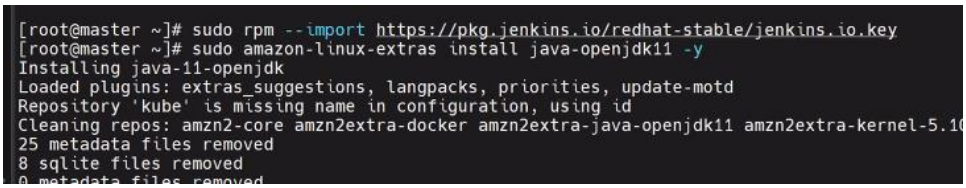
```
[root@master ~]# netstat  -tuplen | grep -i 8080
[root@master ~]#
```

Run this command: (Follow the below site)

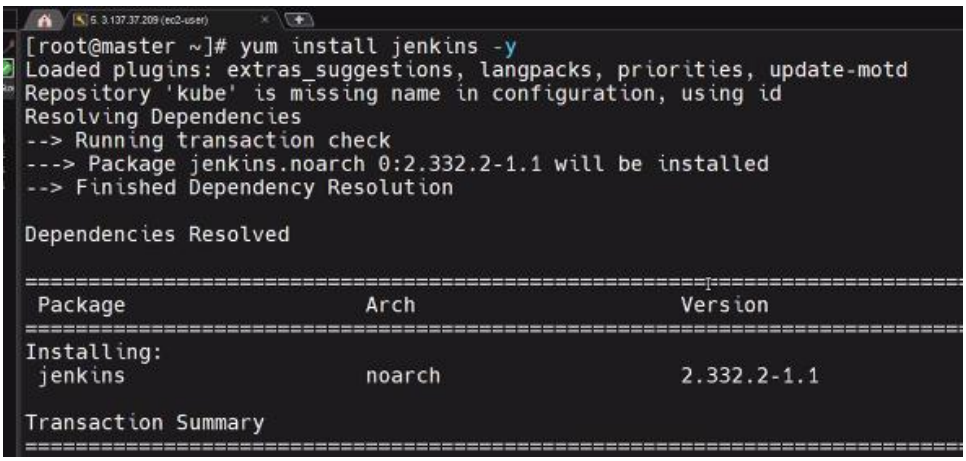
Link:https://www.jenkins.io/doc/tutorials/tutorial-for-installing-jenkins-on-AWS/#download-and-install-jenkins



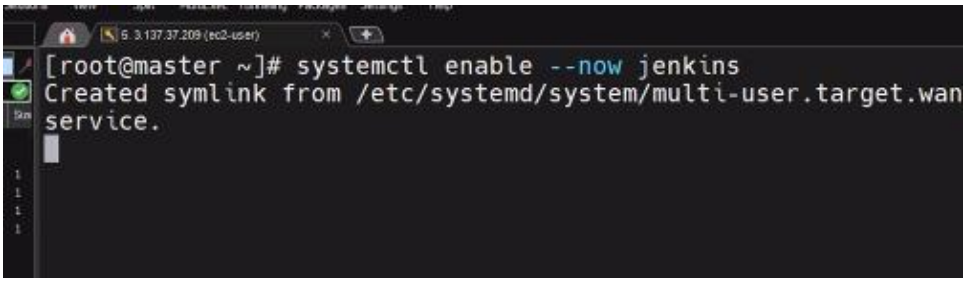
Install java:



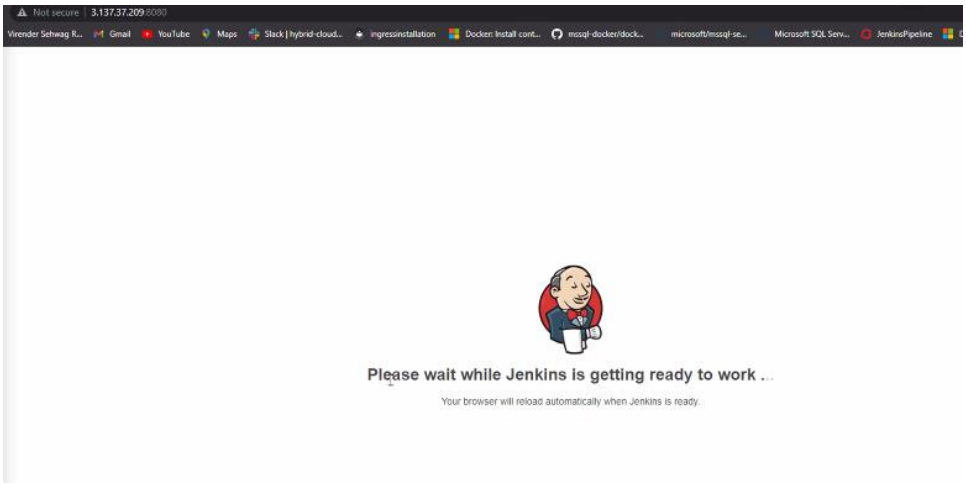
Install Jenkins:



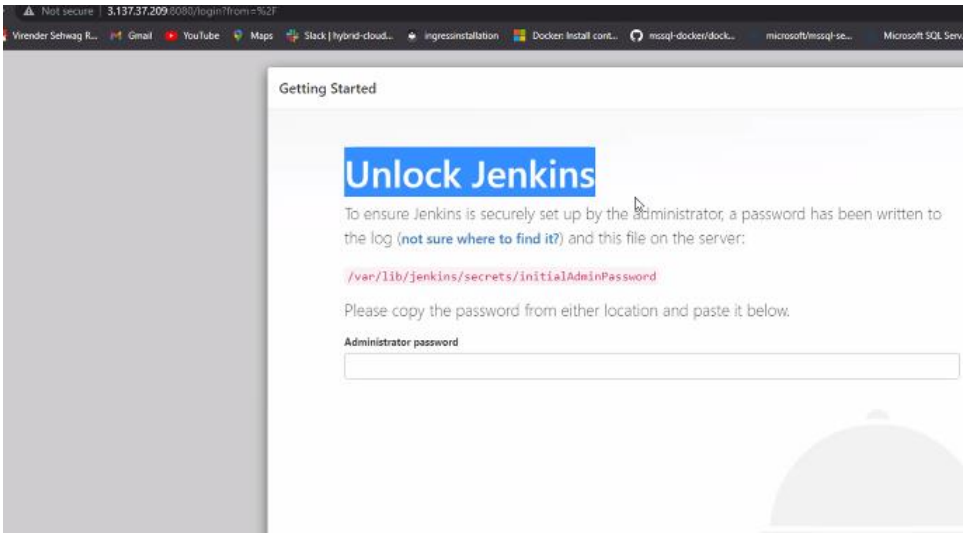
Enable the service:



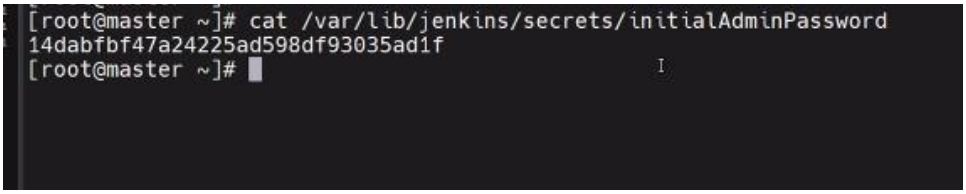
Jenkins is starting:



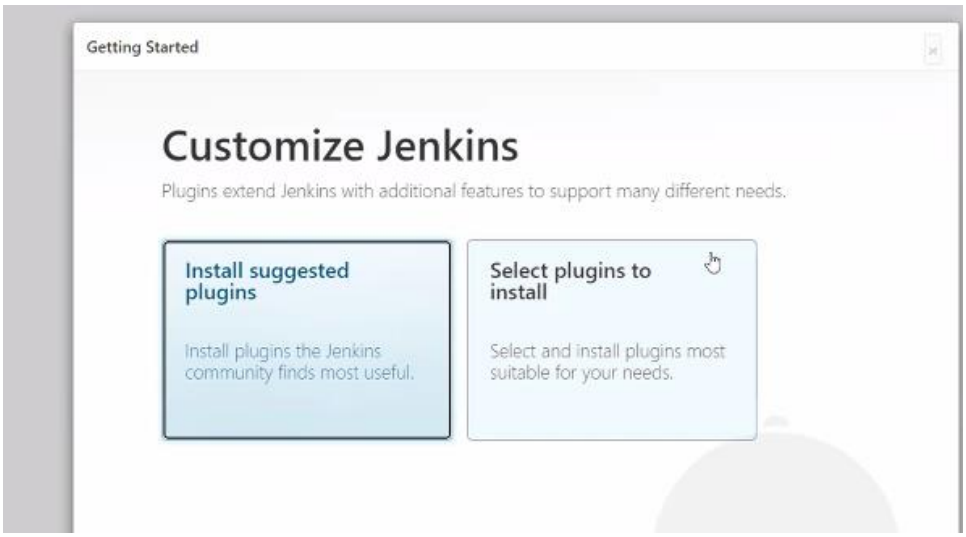
Authentication:



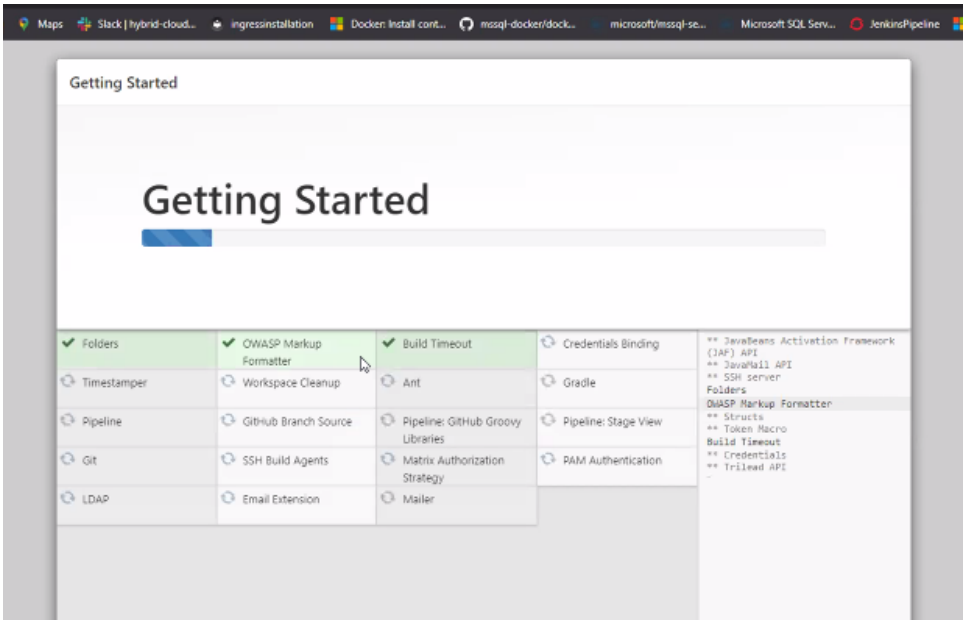
Copy this initial password and paste it:



Plugins: select suggested plugins



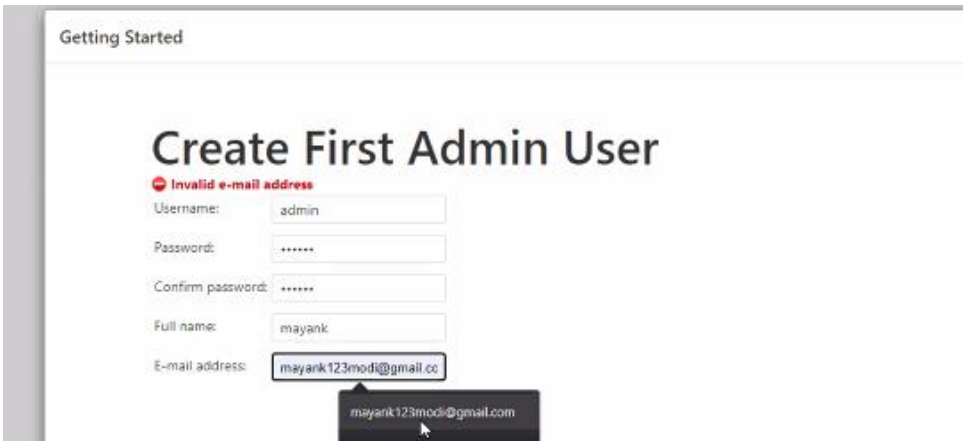
It's installing plugins:

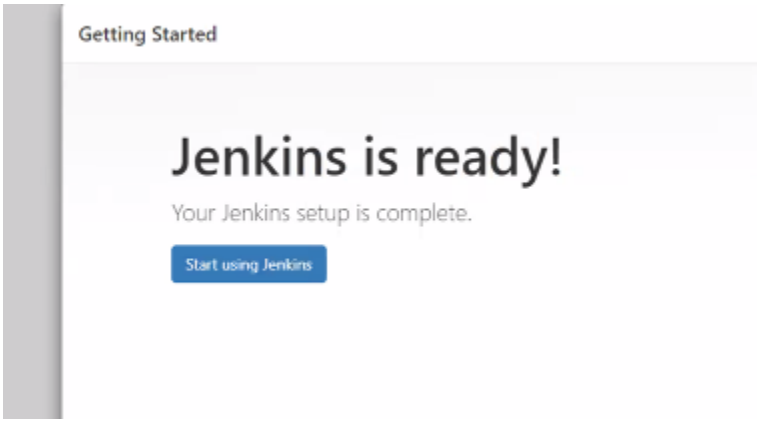


How to configure Apache server:

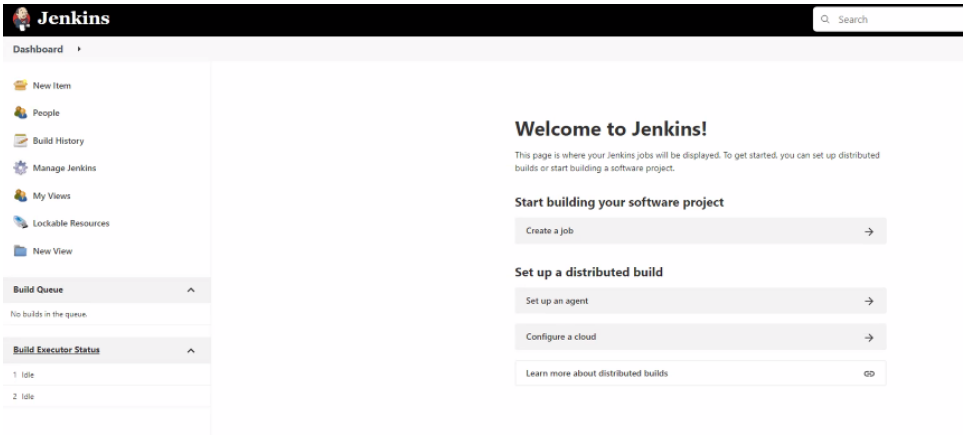
Apache is not yet configured. We will do it later

Provide the username and password here:



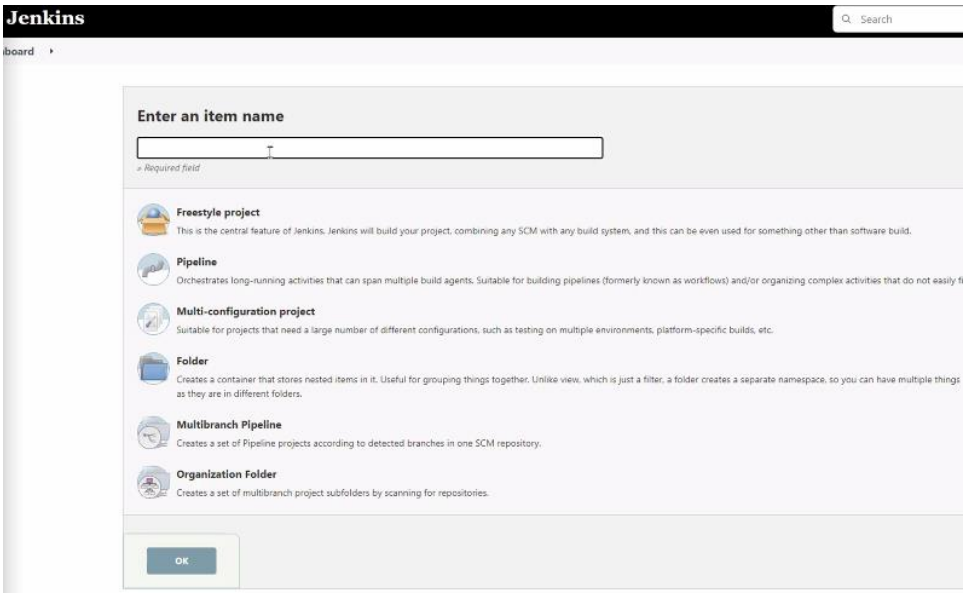


Jenkins dashboard is ready now!

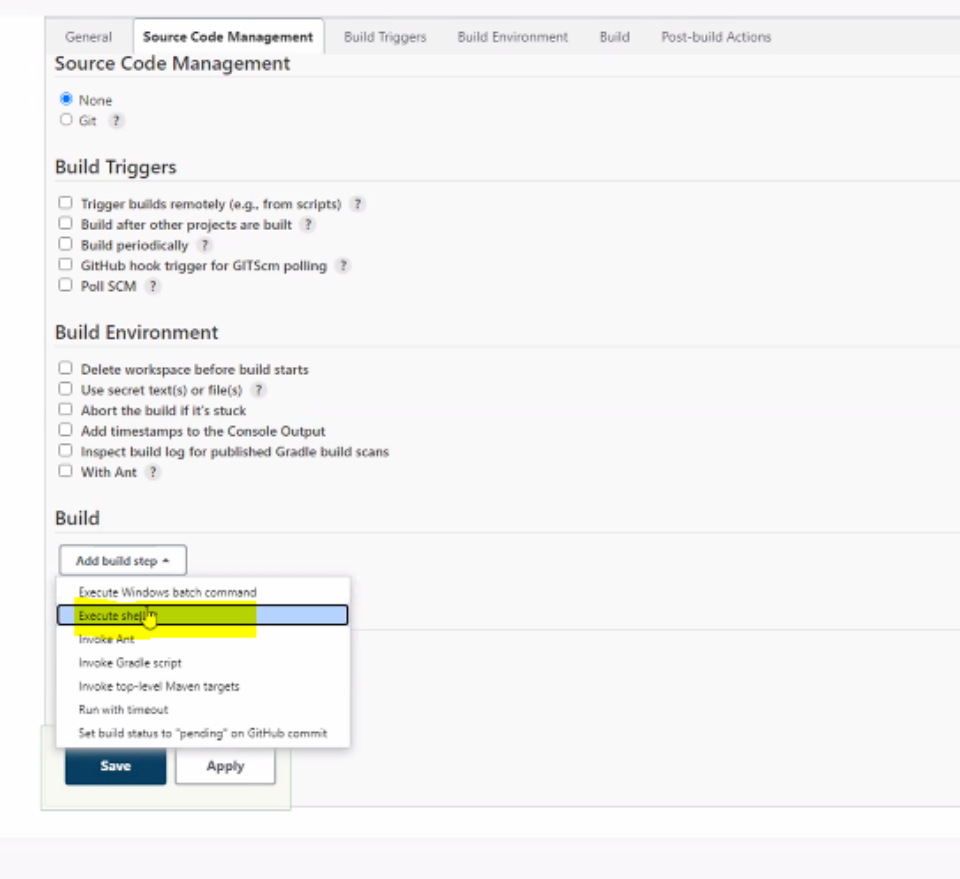


We need to create a job: to create pipeline

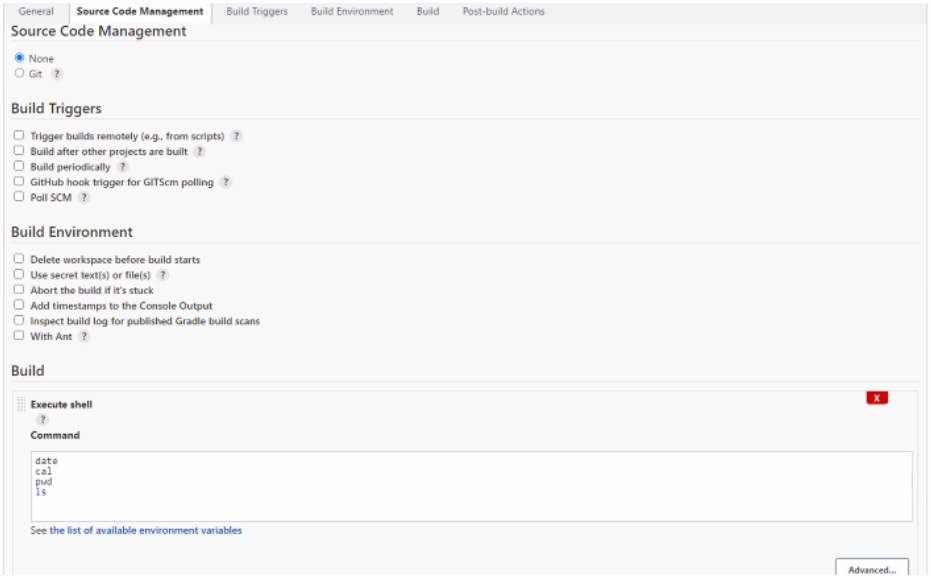
Click on create new job: (As a beginner, select freestyle project)



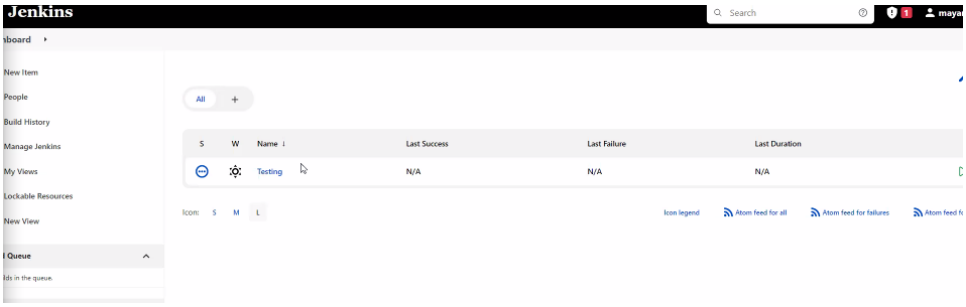
Select execute command: to exe some commands



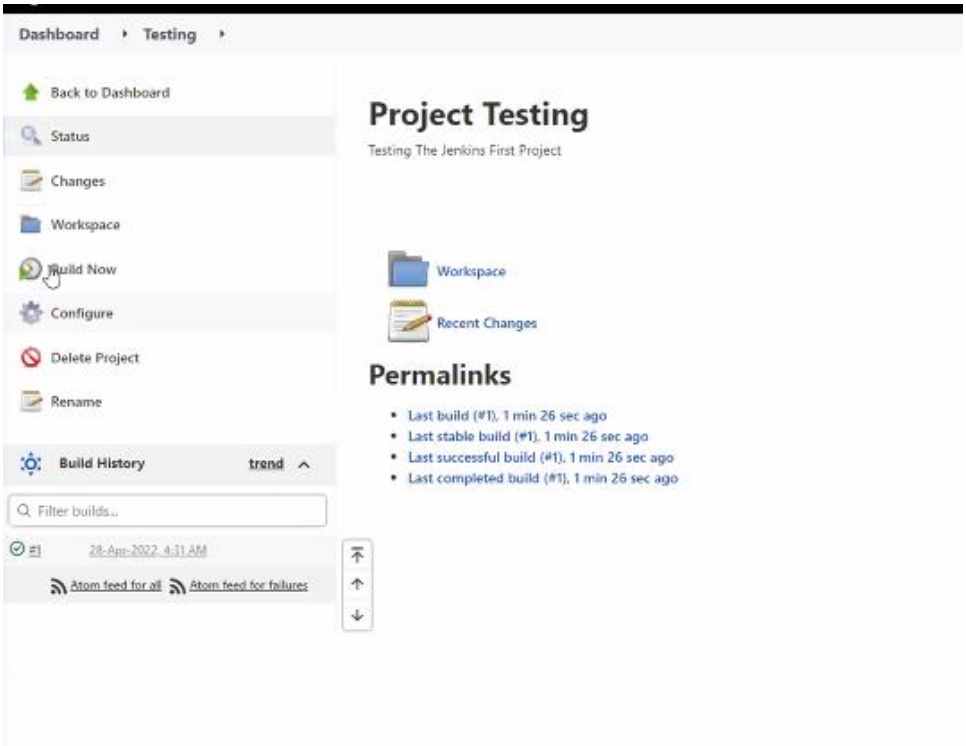
Type some commands here:



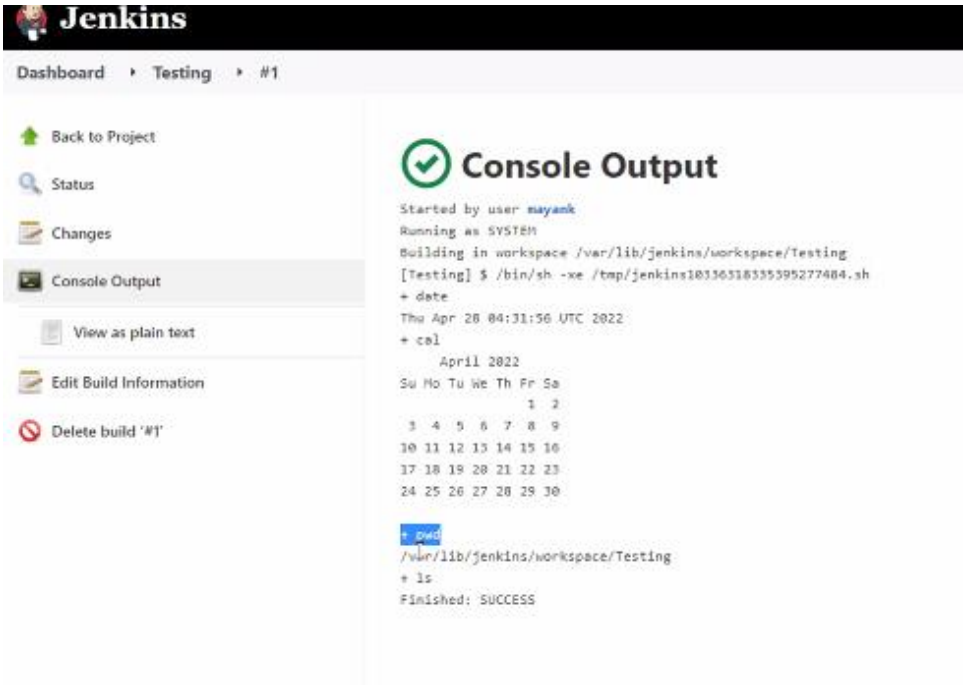
You will see the job has been created now:



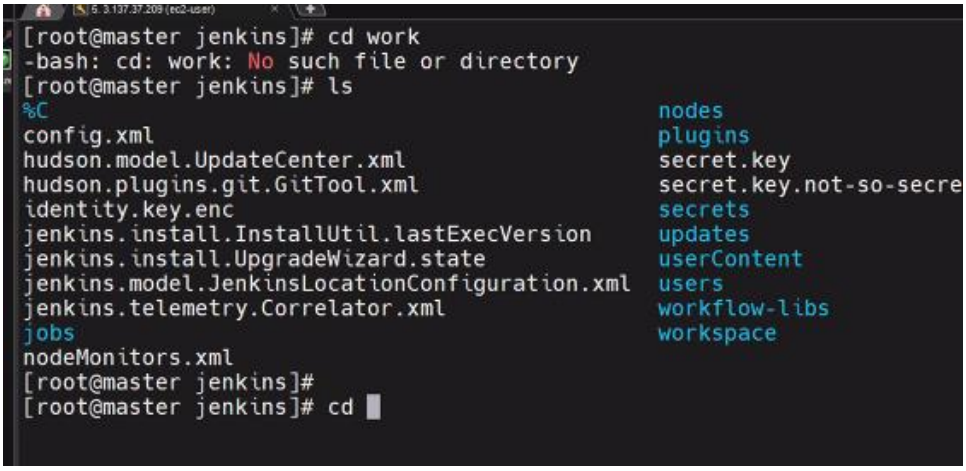
2 ways to run it: Click it directly /Build now

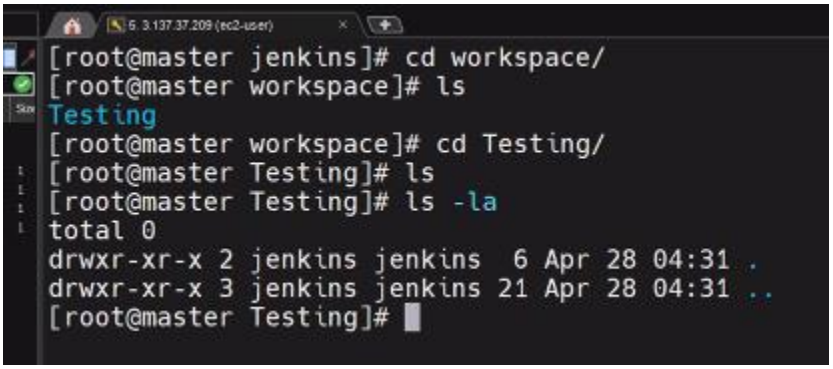


Check the console output:



They will create a workspace:

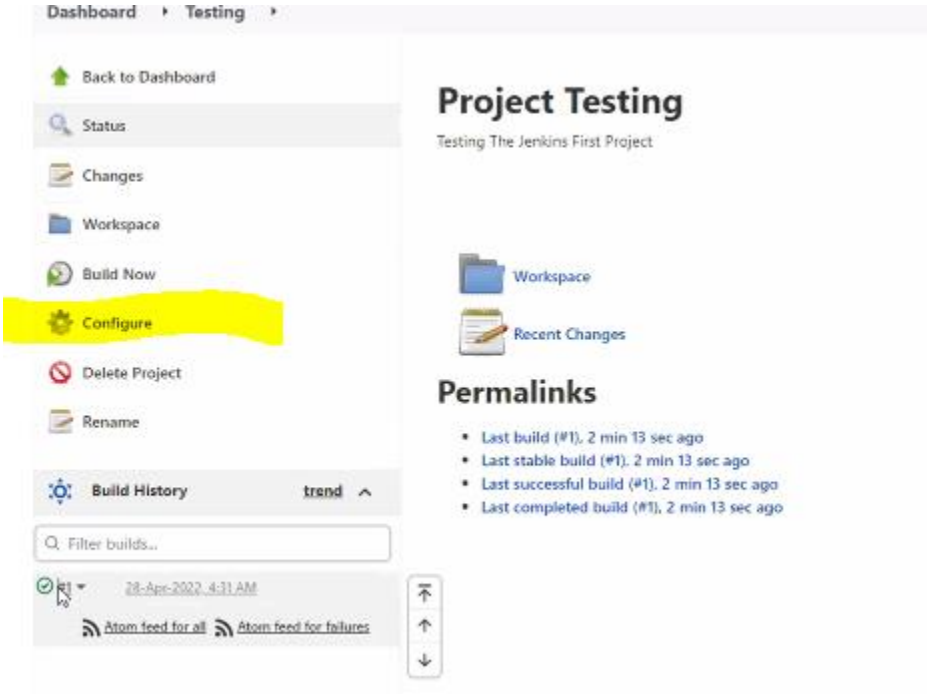


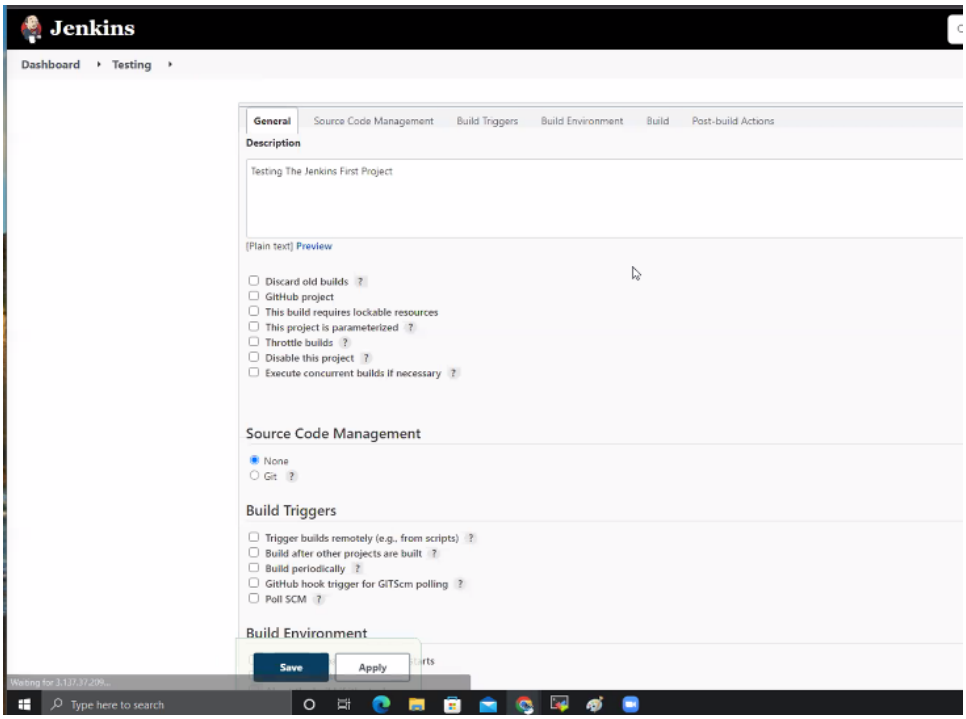


```
drwxr-xr-x 3 jenkins jenkins 21 Apr 28 04:31 ..
[root@master Testing]#
[root@master Testing]#
[root@master Testing]# cd ..
[root@master workspace]# ls
Testing
[root@master workspace]#
```

If you want to edit it:

Go to configure:

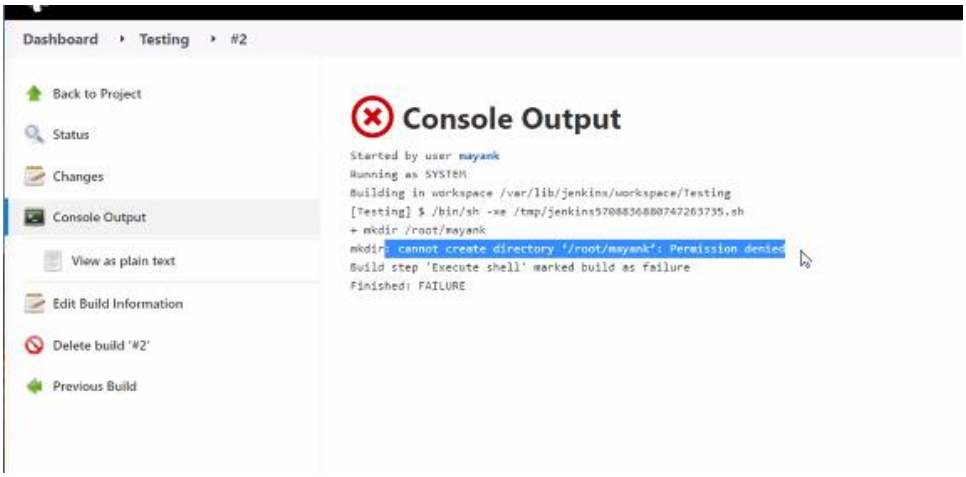




Previous:

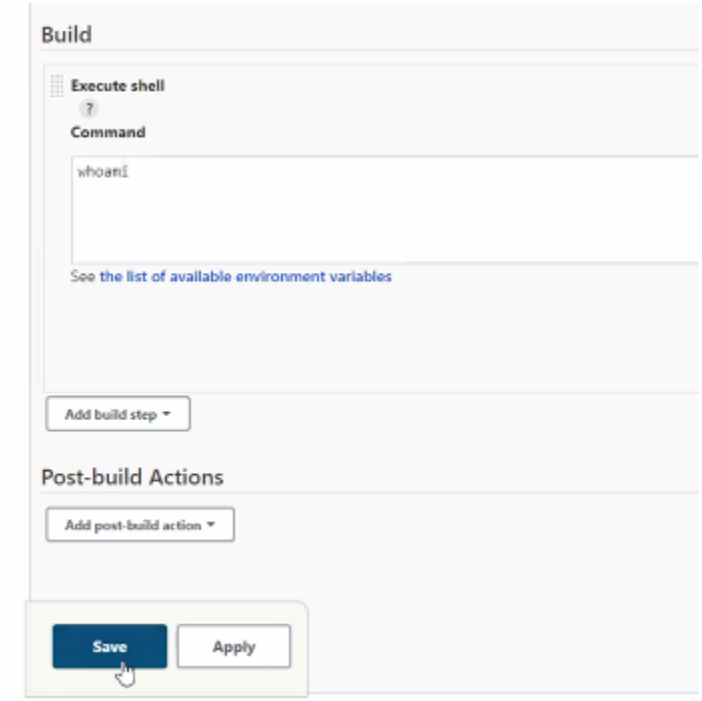
Change it to: mkdir /root/mayank

Permission denied error:

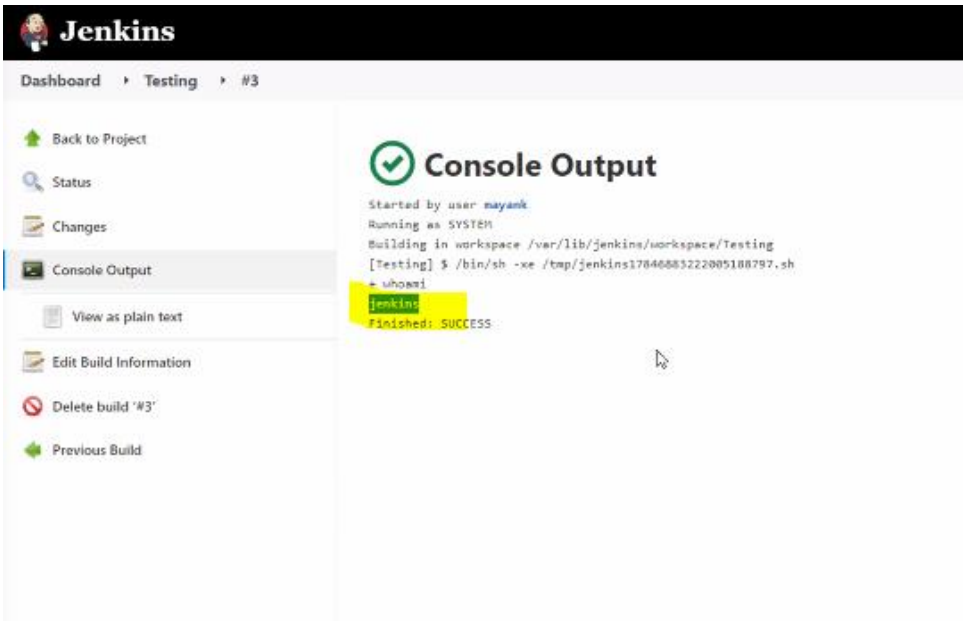


Check for the user:

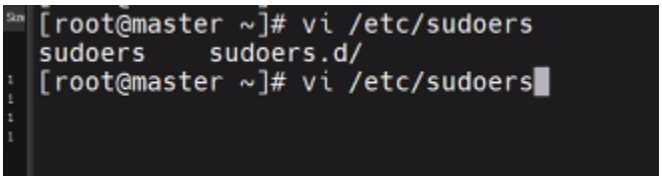
If it's not root, then we need to change it.



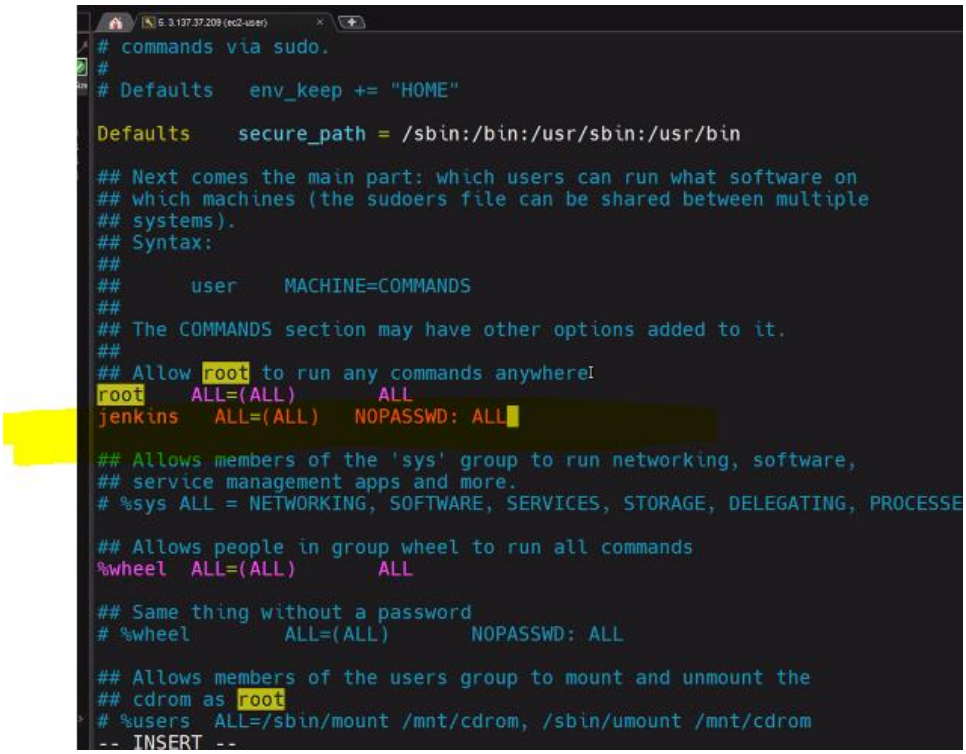
Check the user: by building it. Its Jenkins user.



Make it root: give the permission for Jenkins



Edit the file:

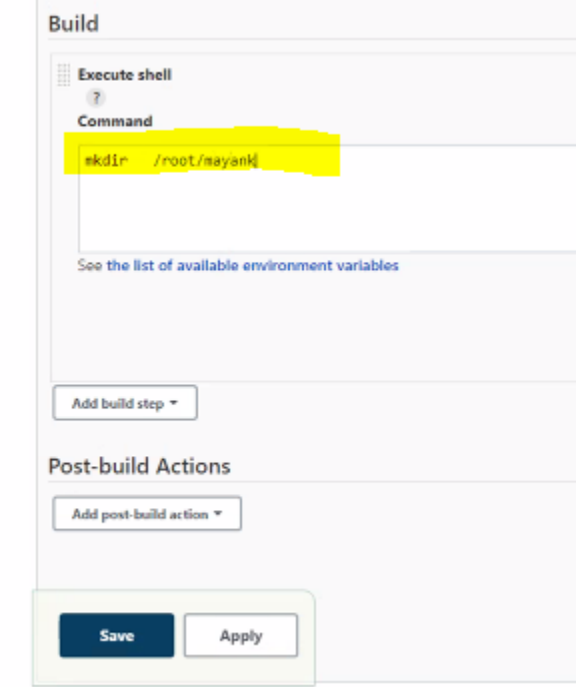


To save it :wq!

Check it:

```
[root@master ~]# cat /etc/sudoers  | grep -i jenkin
jenkins    ALL=(ALL)    NOPASSWD: ALL
[root@master ~]#
```

Now, can this run?



NO! If you build it again, then it won't work. We need to provide sudo:

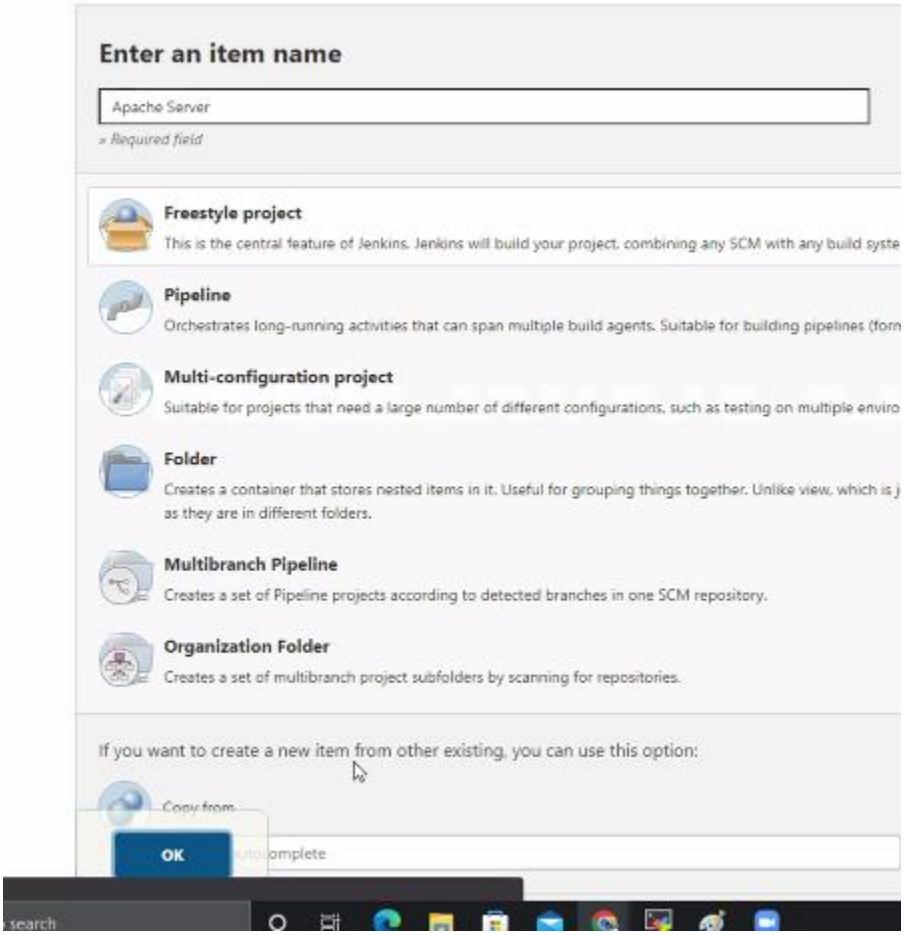
Sudo – gives privilege of root

```
[root@master ~]# ls
calico.yaml  code  docker  harsh.py  mayank  mayank.py  terracode
[root@master ~]# ls -la
total 272
dr-xr-x--- 12 root root    323 Apr 28 04:39 .
```

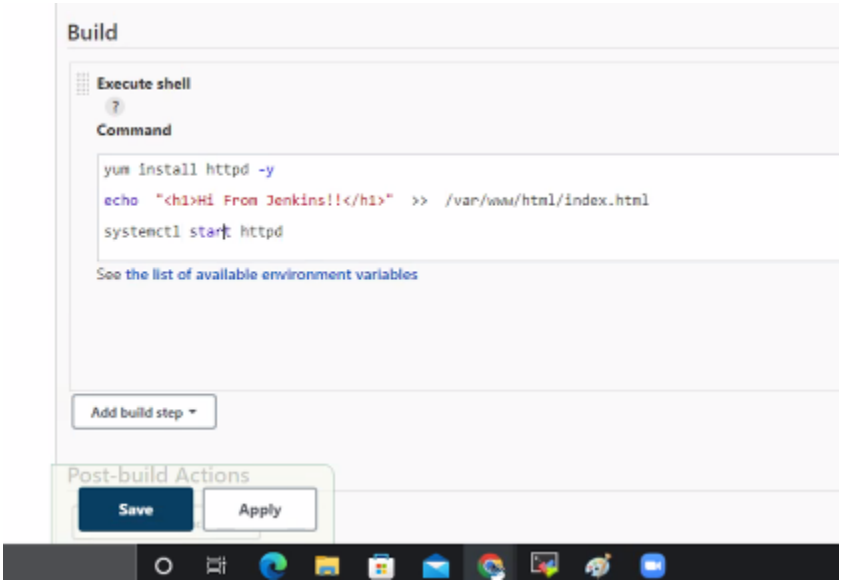
Remove the folder:

Sudo rmdir /root/mayank

Configure the Apache server: Create a new project:

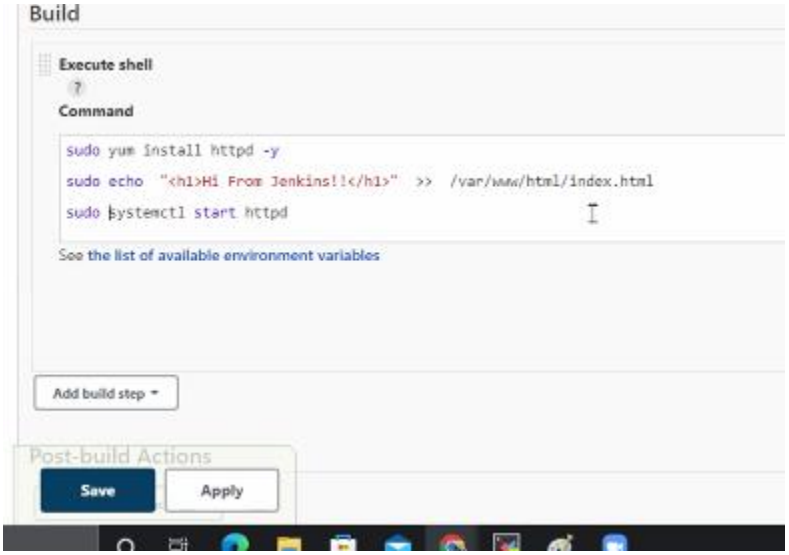


Provide commands now:

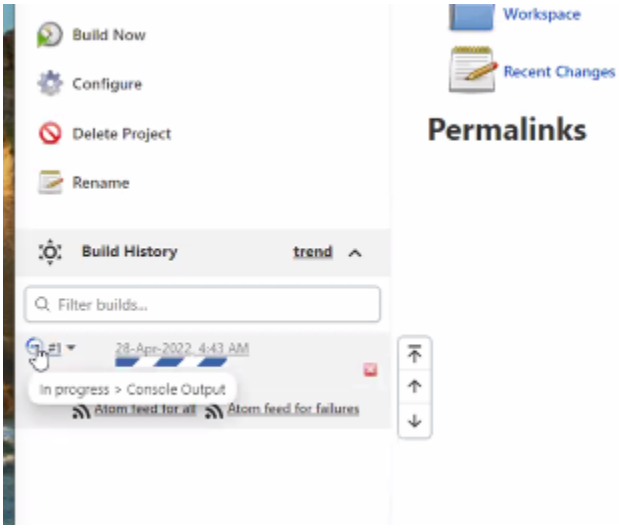


What are we missing above?

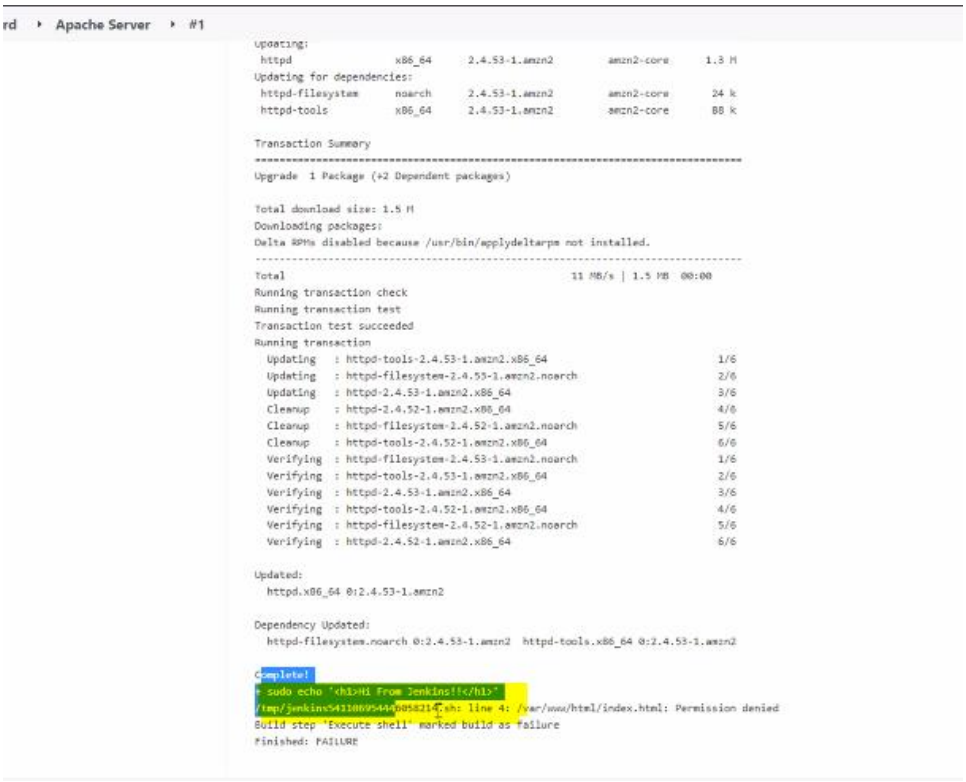
Need sudo for all commands:



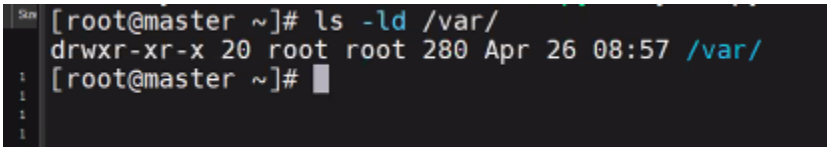
Build is running!



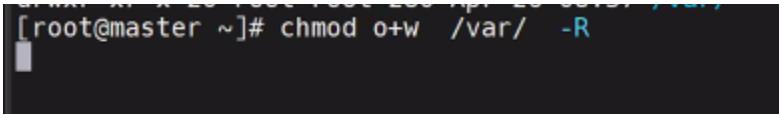
Error: Permission denied: You need to change the permission:



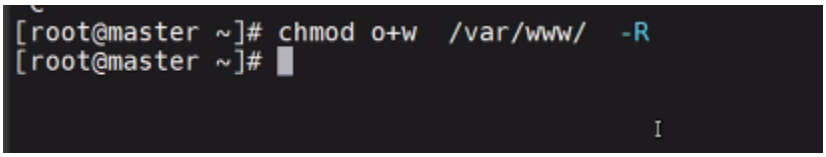
Now, you can see only root has the permission: We need to add for Jenkins



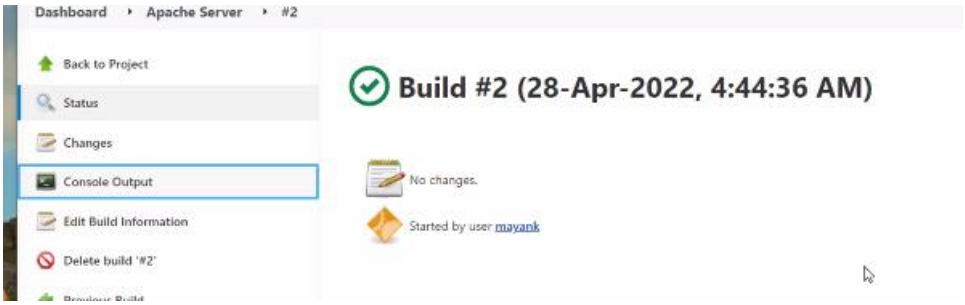
This command takes long time to change the permission:



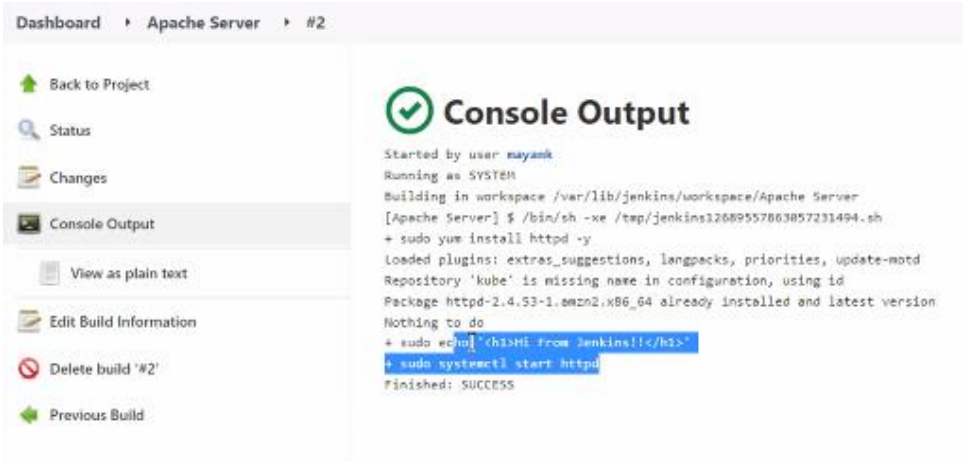
Use this command to change the permission:



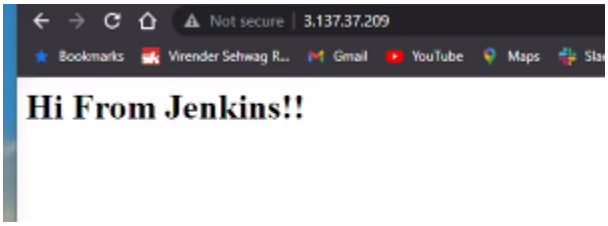
Build is success now:



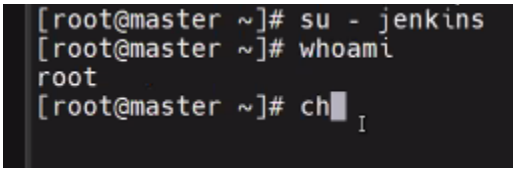
Check the console output:



Check the website:

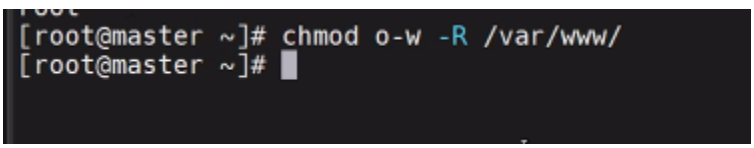


You can’t change the user directly:

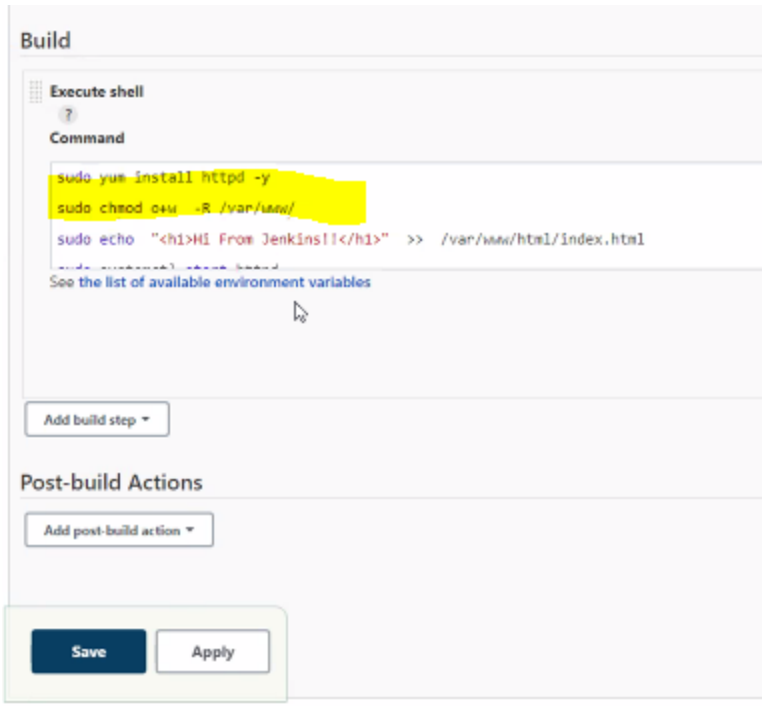


Remove the permission in the machine manually:

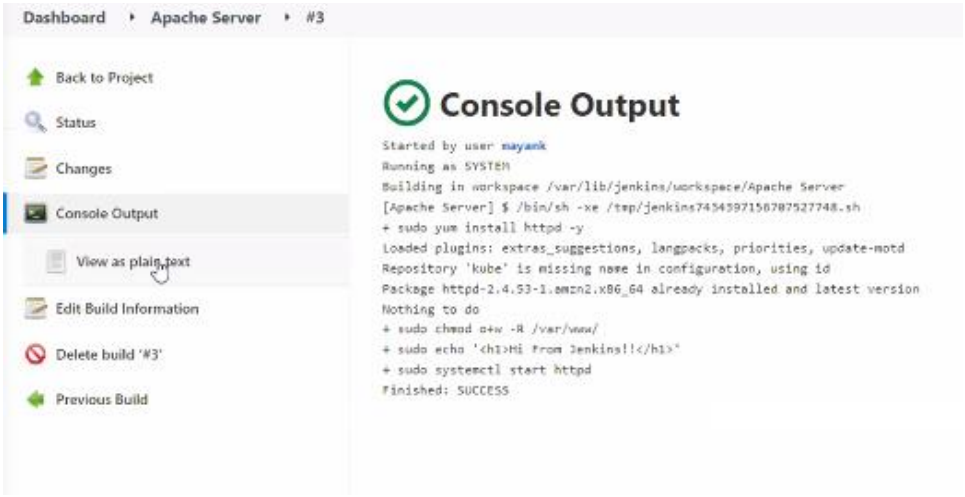
What is the meaning of recursive? It includes all the files and folder inside and inside



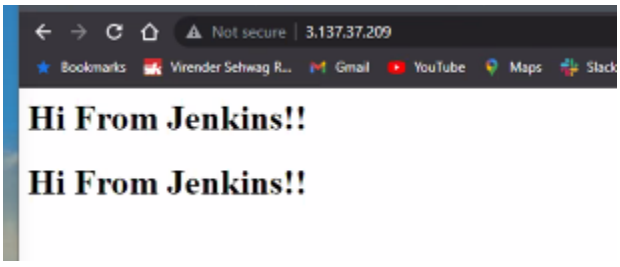
Provide permission in the command itself:



Build is success! Check the console output:

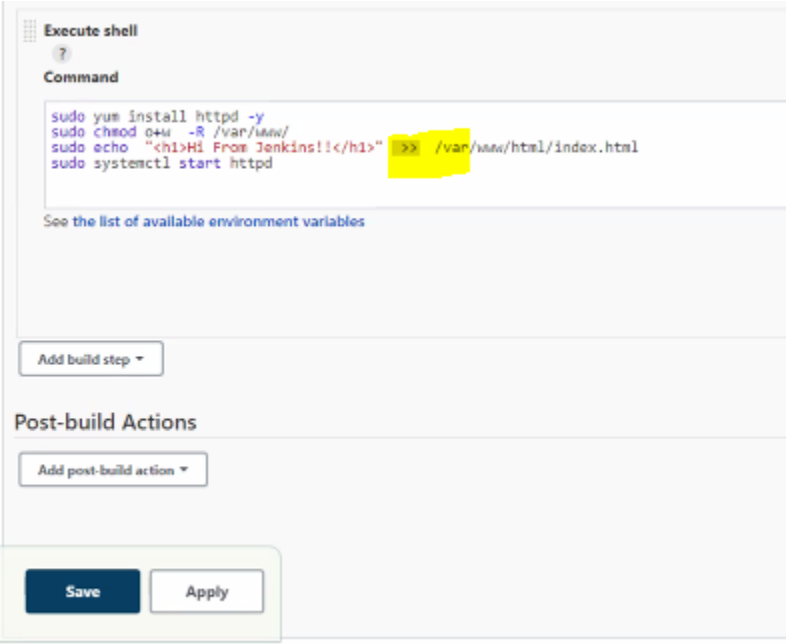


Check the website:

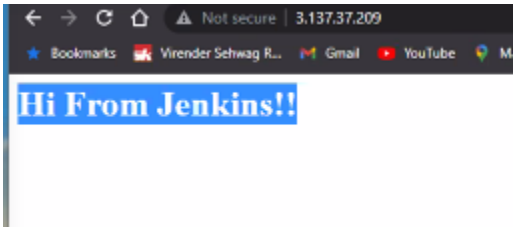


Why >> ?

It will append the text



If > -----override: You can see only one

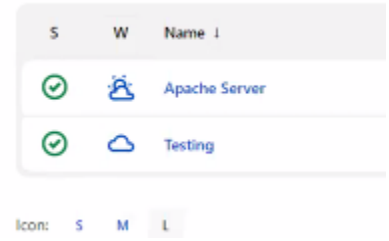


What is W? Stability – weather report

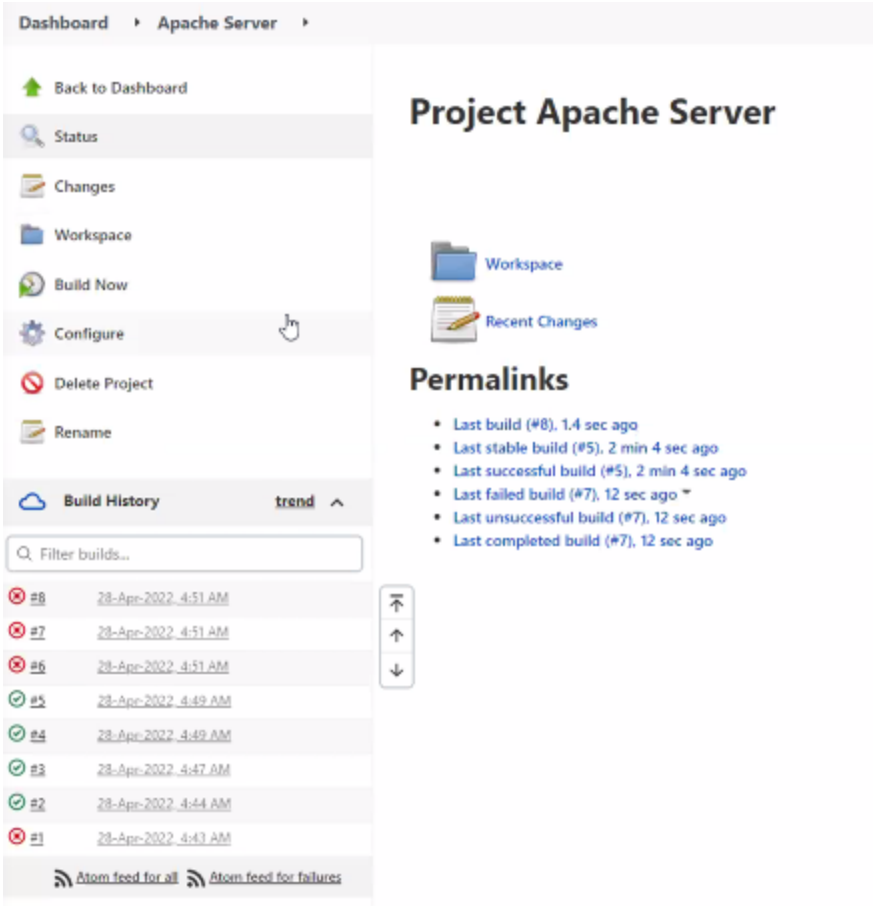
Rain – stability poor

Sun – Stability is only run one time

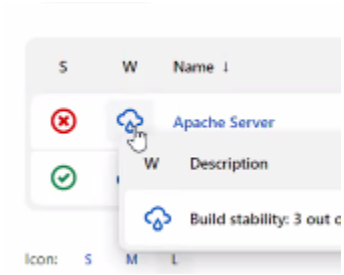
After failing the build, the weather report will change.



Look into the status of the build:



Kind of rain – indicates – it's not a stable build!



Why it is 3 out of 5? We have built 8 times?

It takes last 5 builds only for the stability analysis

Heavy rain:

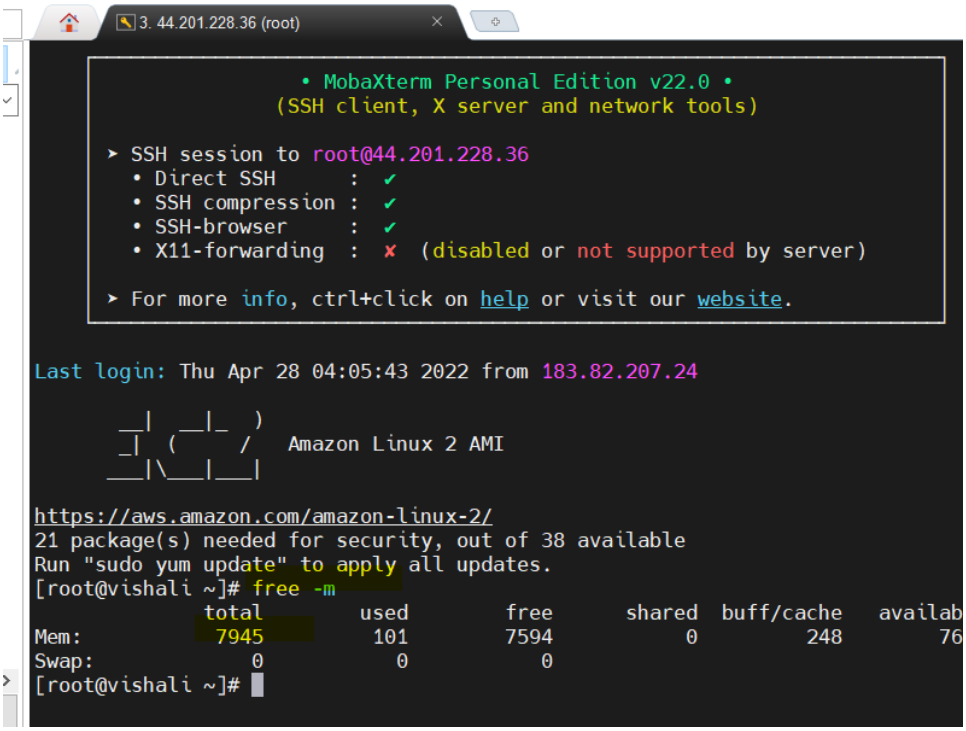
IP Address details:

	A	B	C	D	E
1	S.No	**Ip Address- Day4**	User Name	Password	Participants Name
2	**Trainer**	54.227.220.217	**root**	**redhat**	**Trainer**
3	1	54.159.152.129	root	redhat	alpana
4	2	54.236.250.13	root	redhat	baljeet
5	3	54.90.85.126	root	redhat	Devaraj
6	4	3.89.115.152	root	redhat	ganesh
7	5	3.86.28.136	root	redhat	kaustubh
8	6	54.152.185.61	root	redhat	manish
9	7	54.209.111.20	root	redhat	manoj
10	8	54.166.111.131	root	redhat	naveen
11	9	35.174.109.145	root	redhat	neha
12	10	54.208.171.183	root	redhat	pravallika
13	11	18.207.217.0	root	redhat	ramanand sai
14	12	44.201.132.21	root	redhat	revanth
15	13	3.86.33.138	root	redhat	rohit
16	14	54.208.19.204	root	redhat	rudra
17	15	44.203.182.80	root	redhat	sahitya
18	16	54.165.127.214	root	redhat	sampat
19	17	3.83.46.56	root	redhat	sangamesh
20	18	52.91.174.220	root	redhat	sashi
21	19	18.233.152.22	root	redhat	shashidhar
22	20	44.203.147.13	root	redhat	shweta
23	21	107.23.59.118	root	redhat	sudheer
24	22	54.159.160.221	root	redhat	uday
25	23	54.208.29.194	root	redhat	vaishnavi
26	24	44.201.228.36	root	redhat	vishali
27	25	54.211.178.69	root	redhat	sankalp
28	26	52.87.130.58	root	redhat	balakrishna

Details of my machine:

IP: 44.201.228.36

Now, its 8GB:



TASK:

1. **INSTALL JENKINS**

Link: https://www.jenkins.io/doc/tutorials/tutorial-for-installing-jenkins-on-AWS/#download-and-install-jenkins

- Add the Jenkins repo using the following command:

sudo wget -O /etc/yum.repos.d/jenkins.repo \
https://pkg.jenkins.io/redhat-stable/jenkins.repo

- Import a key file from Jenkins-CI to enable installation from the package:

sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key

- Install Java:

sudo amazon-linux-extras install java-openjdk11 -y

- Install Jenkins:

sudo yum install jenkins -y

- Enable the Jenkins service to start at boot:

sudo systemctl enable jenkins

- Start Jenkins as a service:

sudo systemctl start jenkins

You can check the status of the Jenkins service using the command:

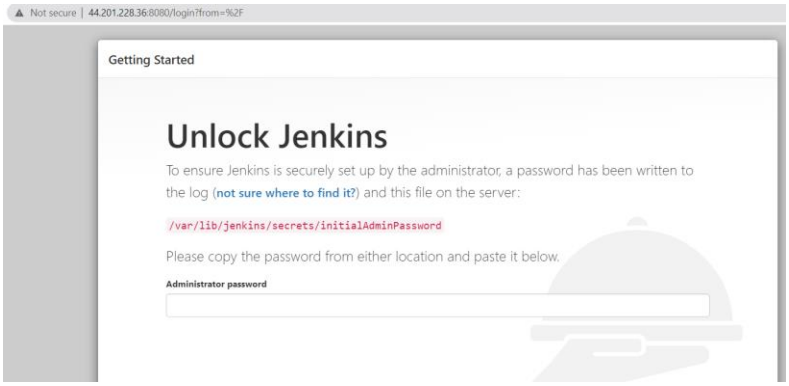
sudo systemctl status jenkins

```
[root@vishali ~]# sudo wget -O /etc/yum.repos.d/jenkins.repo \
>     https://pkg.jenkins.io/redhat-stable/jenkins.repo
--2022-04-28 05:01:18--  https://pkg.jenkins.io/redhat-stable/jenkins.repo
Resolving pkg.jenkins.io (pkg.jenkins.io)... 146.75.34.133, 2a04:4e42:78::
Connecting to pkg.jenkins.io (pkg.jenkins.io)|146.75.34.133|:443... connec
HTTP request sent, awaiting response... 200 OK
Length: 85
Saving to: '/etc/yum.repos.d/jenkins.repo'

100%[======================================>] 85          --.-K/s   in 0s

2022-04-28 05:01:19 (6.23 MB/s) - '/etc/yum.repos.d/jenkins.repo' saved [8
```

```
[root@vishali ~]# sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins
.io.key
[root@vishali ~]#
```

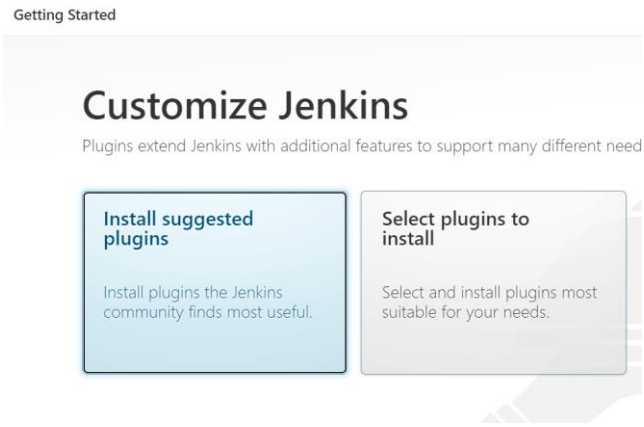
```
Installed:
  jenkins.noarch 0:2.332.2-1.1

Complete!
[root@vishali ~]# sudo systemctl enable jenkins
Created symlink from /etc/systemd/system/multi-user.target.wants/je
 to /usr/lib/systemd/system/jenkins.service.
[root@vishali ~]# sudo systemctl start jenkins
[root@vishali ~]#
```

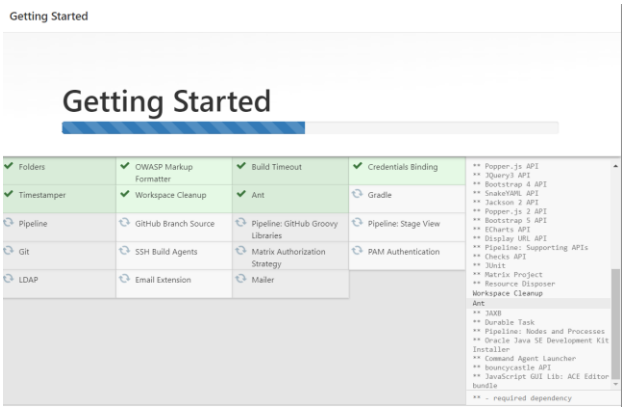
2. **Access it:**




```
[root@vishali ~]# cat /var/lib/jenkins/secrets/initialAdminPassword
e7446e5ecd15430c84e7c6ade95cd3d2
[root@vishali ~]#
```

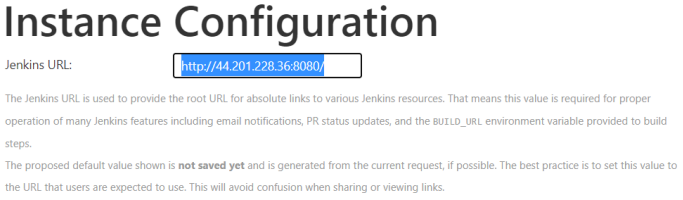
e7446e5ecd15430c84e7c6ade95cd3d2



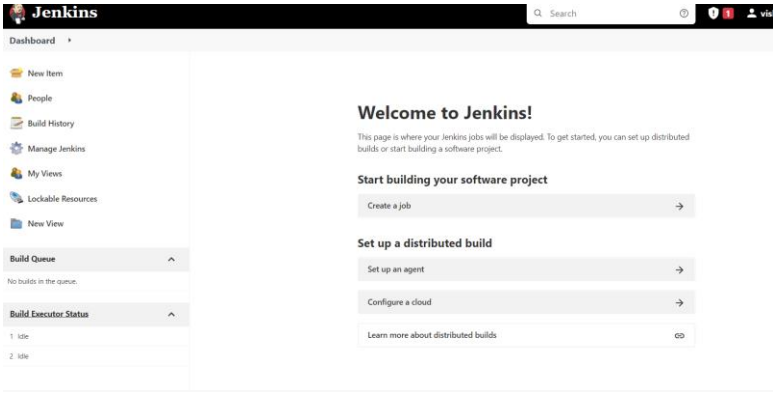


Password: redhat

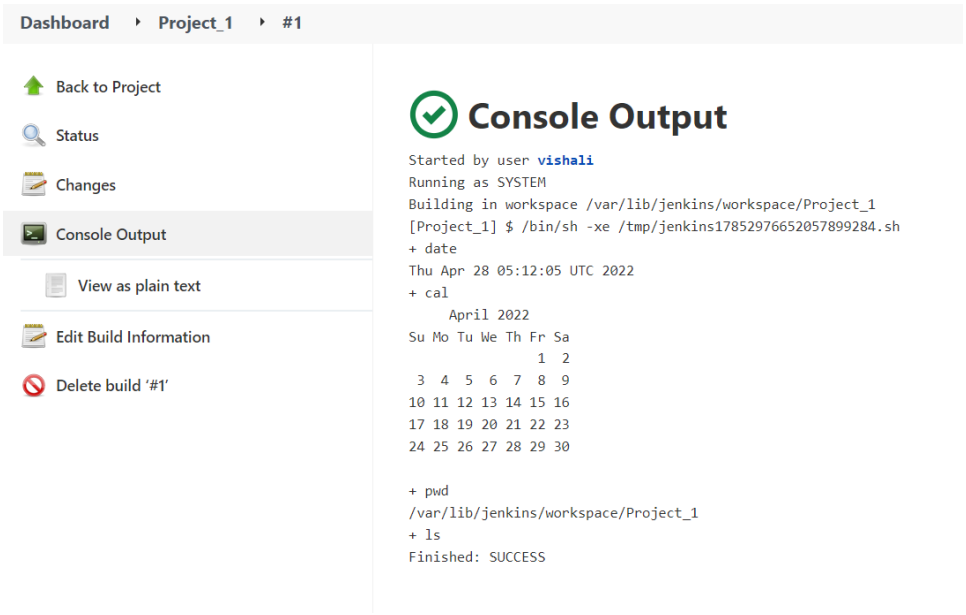
http://44.201.228.36:8080/



Jenkins dashboard is ready!

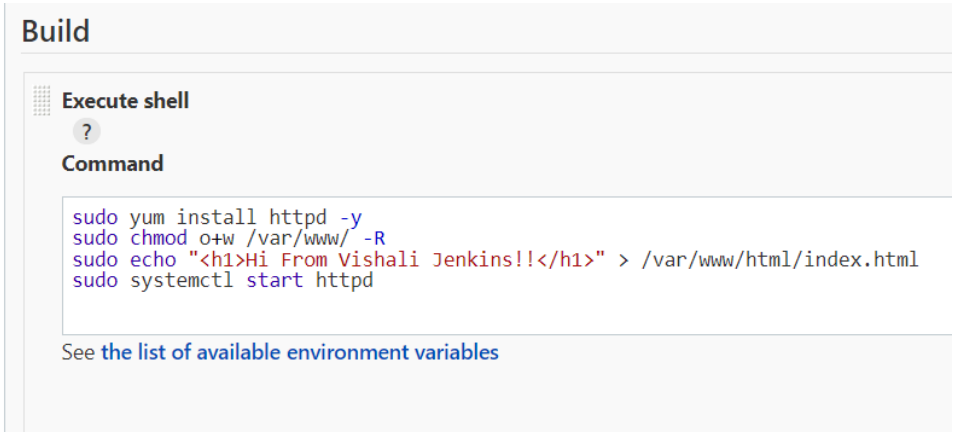


3. Create two jobs

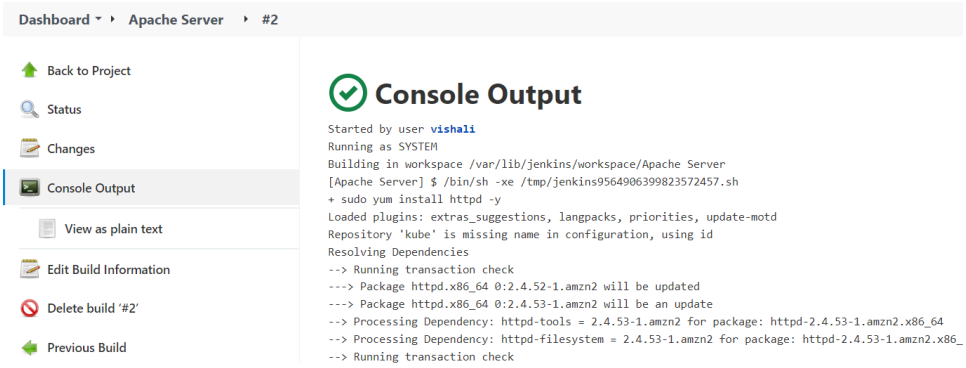


- One job will configure Apache server with some code and then share me the URL

Build:



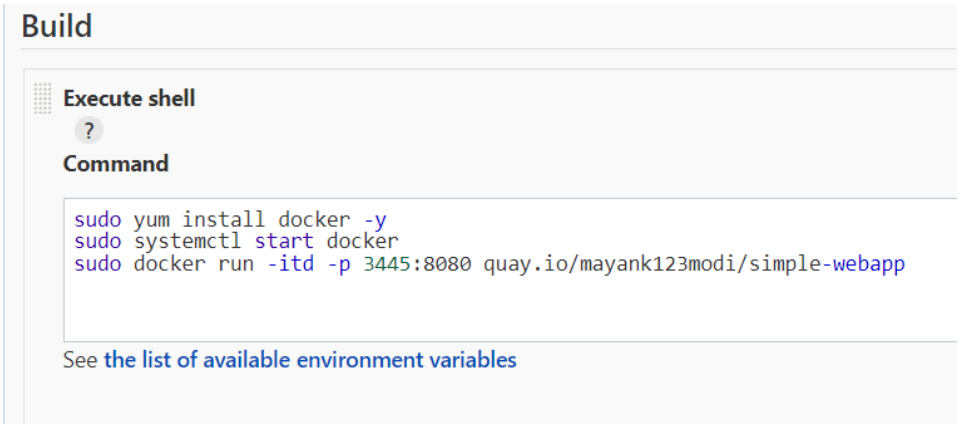
Console output:



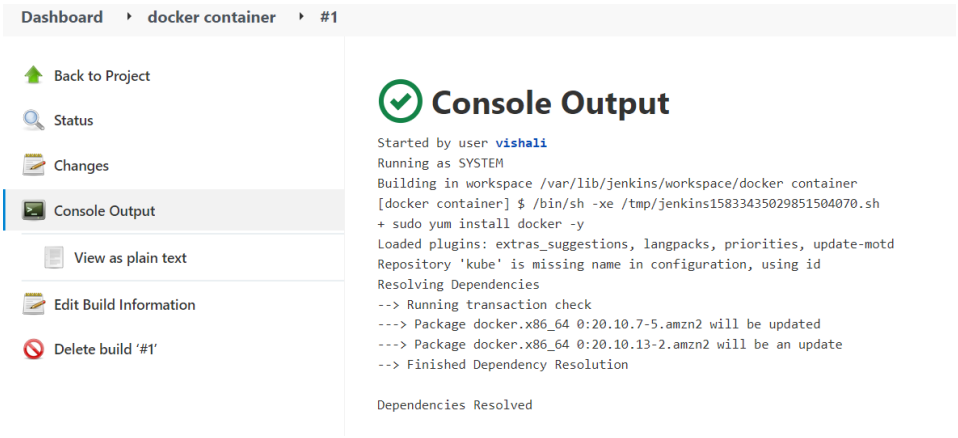
Website:

- Second job will create a container and expose int on 3445 port no. using quay.io/mayank123modi/simple-webapp

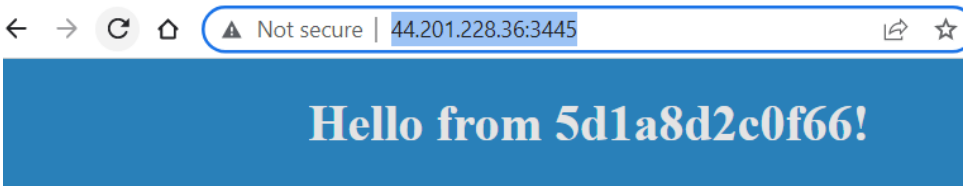
Build:



Console output:

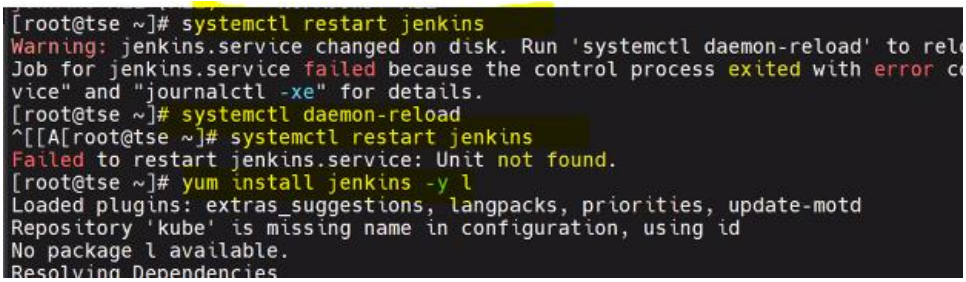


Website:



Troubleshooting one of the participant error:

Reinstall jenkins:



Check the status:

```
[root@tse ~]# systemctl restart jenkins
Failed to restart jenkins.service: Unit not found.
[root@tse ~]# yum install jenkins -y l
^CPlugin "extras_suggestions" can't be imported
Loaded plugins: langpacks, priorities, update-motd
Repository 'kube' is missing name in configuration, using id
^[[A^CPackage jenkins-2.332.2-1.1.noarch already installed and latest version
^[[A[root@tse yum install jenkins -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Repository 'kube' is missing name in configuration, using id
Package jenkins-2.332.2-1.1.noarch already installed and latest version
Nothing to do
[root@tse ~]# systemctl status jenkins
● jenkins.service
   Loaded: not-found (Reason: No such file or directory)
   Active: failed (Result: start-limit) since Thu 2022-04-28 06:20:04 UTC; 59s ago
 Main PID: 31482 (code=exited, status=203/EXEC)

Apr 28 06:20:03 tse systemd[1]: jenkins.service: main process exited, code=exited, status=203/EXEC
Apr 28 06:20:03 tse systemd[1]: Failed to start Jenkins Continuous Integration Server.
Apr 28 06:20:03 tse systemd[1]: Unit jenkins.service entered failed state.
Apr 28 06:20:03 tse systemd[1]: jenkins.service failed.
Apr 28 06:20:04 tse systemd[1]: jenkins.service holdoff time over, scheduling restart.
Apr 28 06:20:04 tse systemd[1]: Stopped Jenkins Continuous Integration Server.
Apr 28 06:20:04 tse systemd[1]: start request repeated too quickly for jenkins.service
Apr 28 06:20:04 tse systemd[1]: Failed to start Jenkins Continuous Integration Server.
Apr 28 06:20:04 tse systemd[1]: Unit jenkins.service entered failed state.
Apr 28 06:20:04 tse systemd[1]: jenkins.service failed.
[root@tse ~]#
```

Something got locked:

```
[root@tse ~]# rpm -e jenkins
warning: Unable to get systemd shutdown inhibition lock
[root@tse ~]#
```

```
[root@tse ~]# ps -aux | grep jenkins
root     32549  0.0  0.0 123560   964 pts/1    S+   06:21   0:00 grep --color=auto jenkins
[root@tse ~]# kill -9 32549
-bash: kill: (32549) - No such process
[root@tse ~]# kill -9 32549
-bash: kill: (32549) - No such process
[root@tse ~]# ps -aux | grep jenkins
root     32612  0.0  0.0 123560  1048 pts/1    S+   06:21   0:00 grep --color=auto jenkins
[root@tse ~]# kill -9 32612
-bash: kill: (32612) - No such process
[root@tse ~]# ps -aux | grep jenkins
root       323  0.0  0.0 123560  1000 pts/1    S+   06:21   0:00 grep --color=auto jenkins
[root@tse ~]# jenkins
-bash: jenkins: command not found
[root@tse ~]#
```

Remove java:

```
[root@tse ~]# rpm -qa | grep -i java
javapackages-tools-3.4.1-11.amzn2.noarch
java-11-openjdk-11.0.13.0.8-1.amzn2.0.3.x86_64
python-javapackages-3.4.1-11.amzn2.noarch
tzdata-java-2022a-1.amzn2.noarch
java-11-openjdk-headless-11.0.13.0.8-1.amzn2.0.3.x86_64
[root@tse ~]#
```

Problem might be with the sudo file:

```
-r--r----- 1 root root 4364 Apr 28 05:18 /etc/sudoers
[root@tse ~]# chmod 777 /etc/sudoers
[root@tse ~]# docker
-bash: /usr/bin/docker: Permission denied
[root@tse ~]# chmod 777 /etc/sudoers  -R
[root@tse ~]#
[root@tse ~]#
[root@tse ~]#
[root@tse ~]# logout
Connection to 54.152.185.61 closed.
[root@master ~]# scp /etc/sudoers  root@54.152.185.61
[root@master ~]#
```

So, trainer is sharing with his file to him

```
[root@tse ~]# docker
-bash: /usr/bin/docker: Permission denied
[root@tse ~]# chmod 777 -R /usr/bin/
chmod: changing permissions of '/usr/bin/ssm-agent-worker': Operation not permitted
chmod: changing permissions of '/usr/bin/docker': Operation not permitted
[root@tse ~]# sudo chmod 777 -R /usr/bin/
sudo: unable to stat /etc/sudoers: Permission denied
sudo: no valid sudoers sources found, quitting
sudo: unable to initialize policy plugin
[root@tse ~]#
```

Trainer: Created new machine and configured the password.

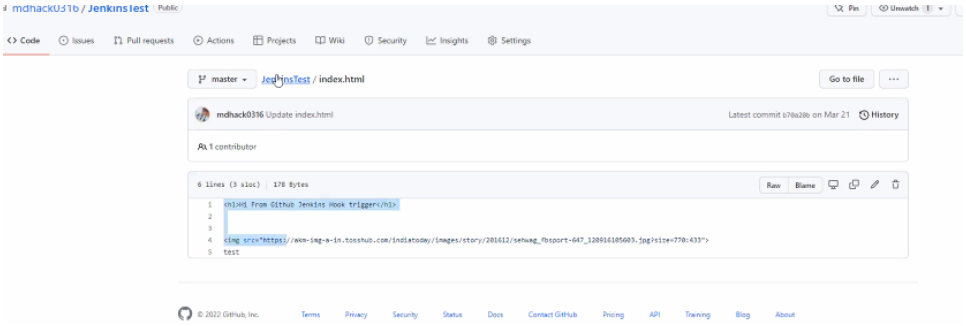
```
https://aws.amazon.com/amazon-linux-2/
16 package(s) needed for security, out of 32 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-17-207 ~]$ sudo -i
[root@ip-172-31-17-207 ~]# passwd root
Changing password for user root.
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
passwd: all authentication tokens updated successfully.
[root@ip-172-31-17-207 ~]# vi /etc/ssh/sshd_config
[root@ip-172-31-17-207 ~]# systemctl restart sshd
[root@ip-172-31-17-207 ~]#
```

Inside the sshd_config file, you have to enable the password authentication as yes

---

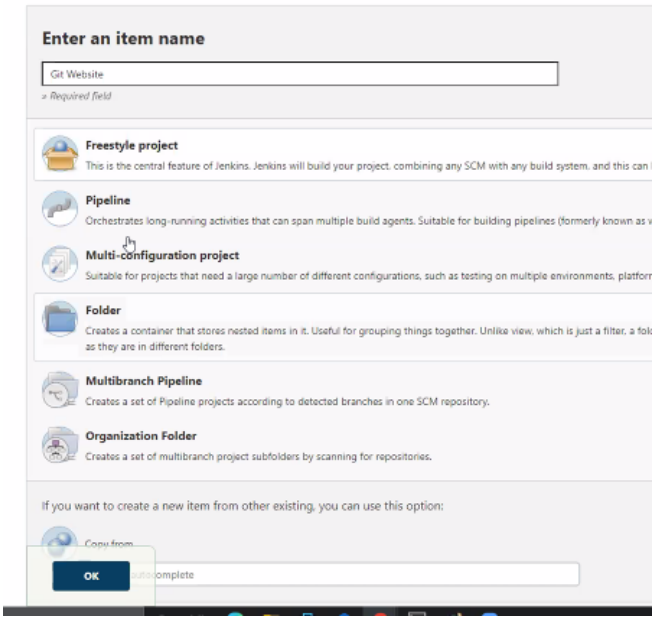
Let's see how to push into the GitHub code:

Trainer's GitHub Repo – code:



Create a website using this html file:

Create a new item:



Paste the URL of the repo here:

Provide the URL of the github repo:

Why there is a problem? You need to install git in the machine

yum install git -y

Check for the branch:

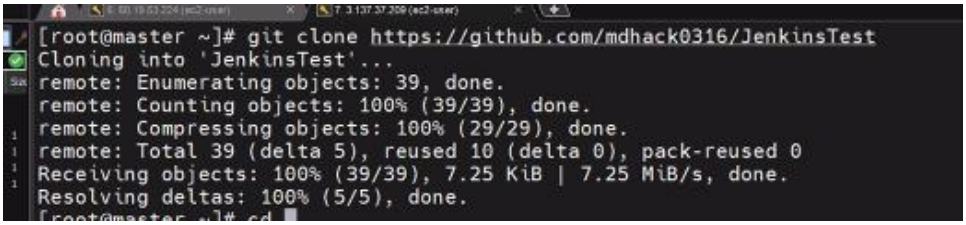
This will automatically clone.

Apache is already configured!

You just must copy the html into location.

Refresh and check:

Clone the repo:

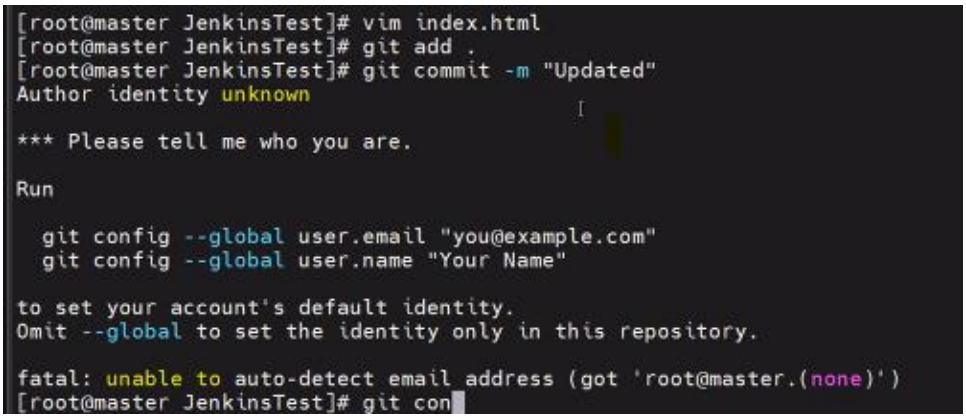


Go inside the file:

Change the content inside it:

Change it in my local machine. Git doesn't know about it

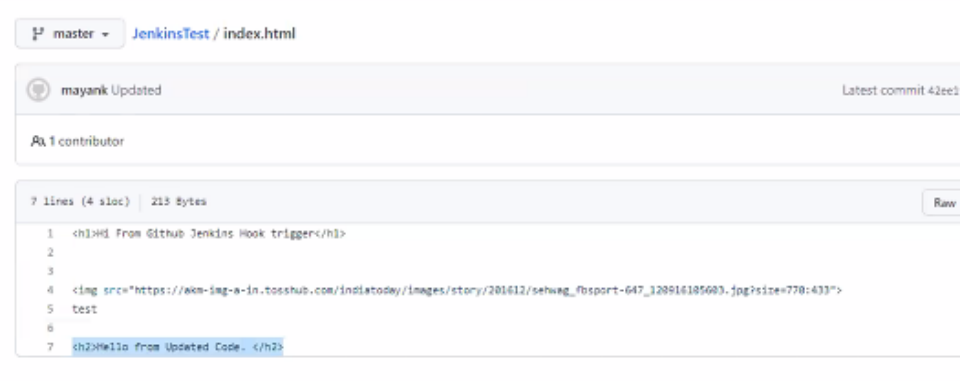
Push that code:



Assign the user email and name:

```
fatal: unable to auto-detect email address (got 'root@master.(none)')
[root@master JenkinsTest]# git config --global user.email mayank@example.com
[root@master JenkinsTest]# git config --global user.name mayank
[root@master JenkinsTest]# git config --global user.name mayank
[root@master JenkinsTest]# git commit -m "Updated"
[master 42ee194] Updated
 1 file changed, 1 insertion(+)
```

While pushing we can't use the password, you must use the token:



Will the change reflect now? No

We must trigger the build.

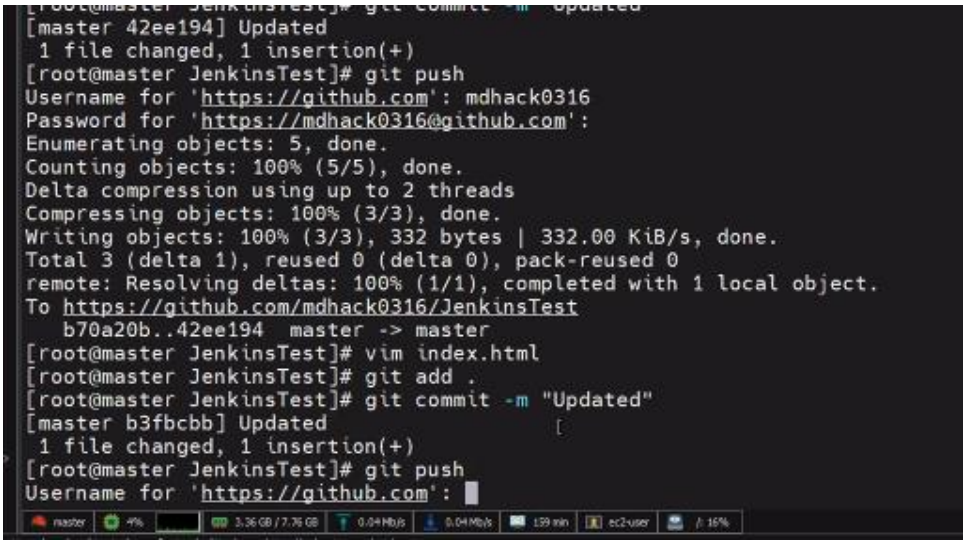
Change my code in Github account: it should automatically reflect!

Automatically trigger the build.

Every minute they will verify it.

5 start means every minute.

Now, change the html and push it and verify it:

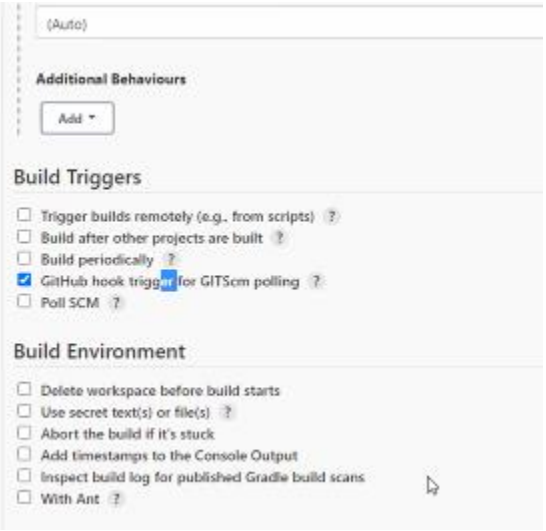


Enter the token:

Problem faced here:

Load issue: Every time it will check if there is a change. Resource is being utilized every minute. This should be avoided.

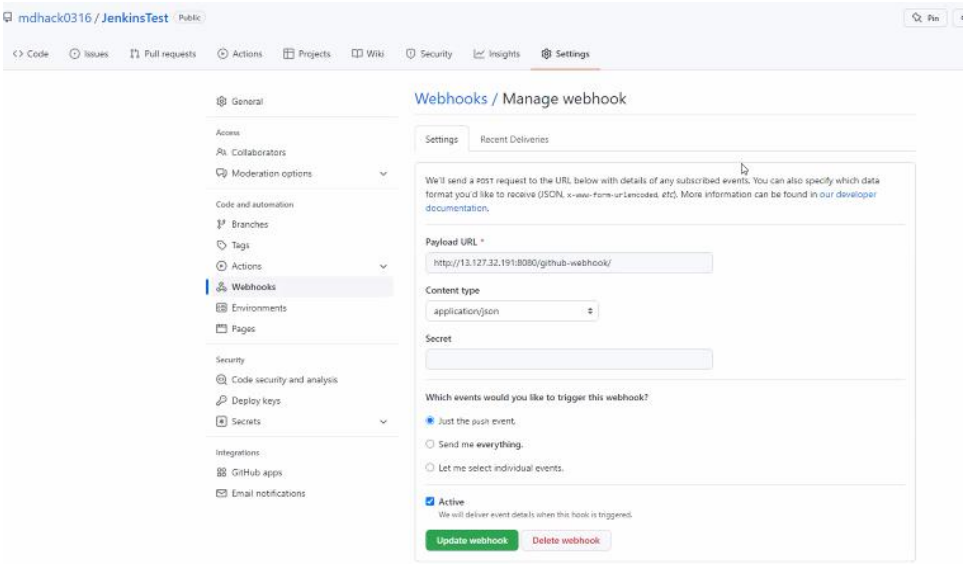
Best thing to use: GitHub hook trigger for GITScm polling



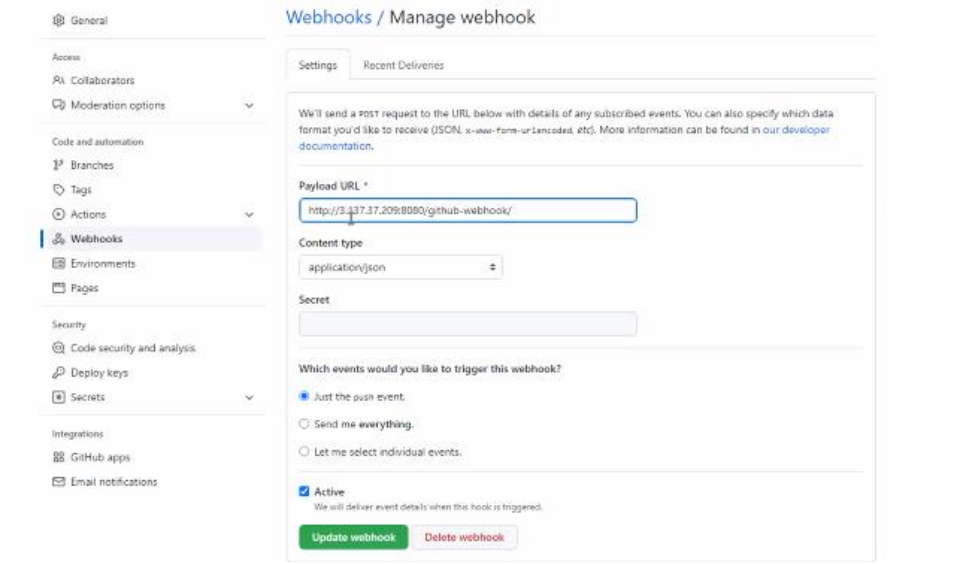
Whenever there is a Github change, then it will trigger.

Create a webhook:

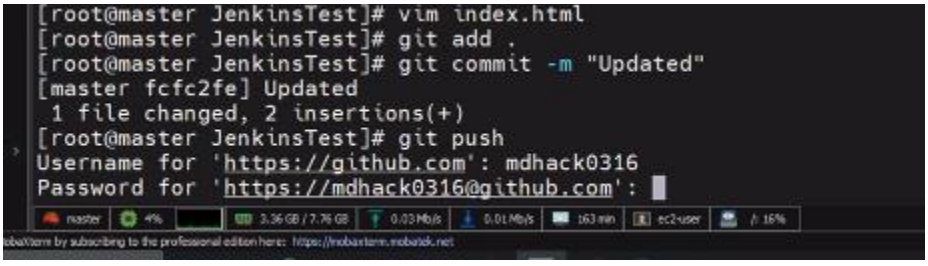
This is used to automatically trigger the Jenkins build, when there is a change in the github code.



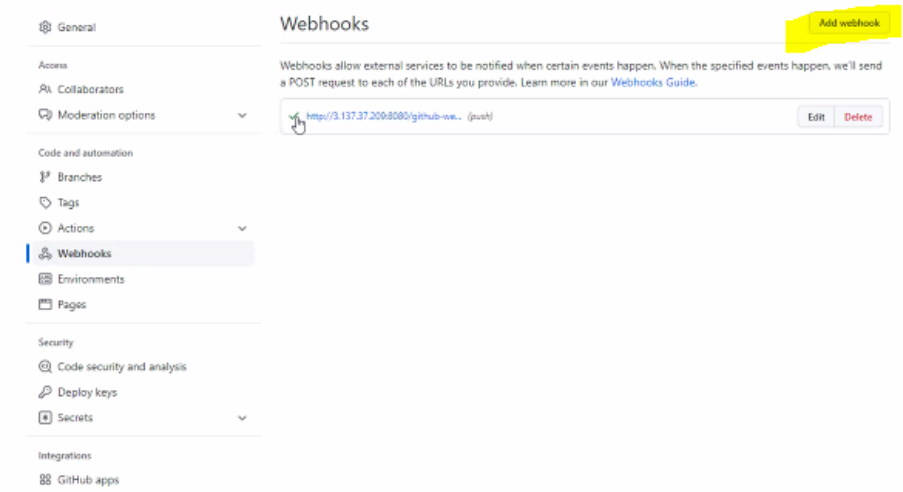
Mention the Plugin: That is the URL



Change the content:



You can create multiple webhook:



What is the fork?

A fork is **a copy of a repository that you manage**. Forks let you make changes to a project without affecting the original repository. You can fetch updates from or submit changes to the original repository with pull requests.

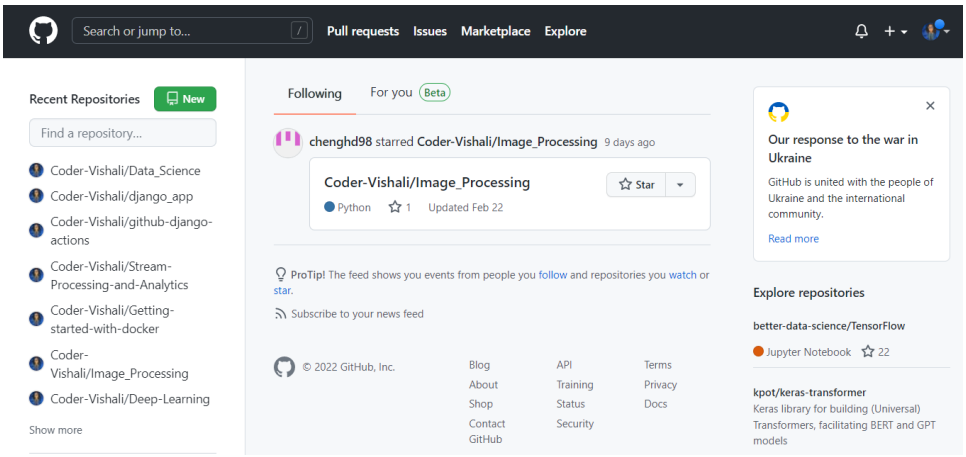
Commands shared by Trainer:

541 yum installl git -y

```
542  yum install git -y
543  git
544  git clone https://github.com/mdhack0316/JenkinsTest
545  cd JenkinsTest/
546  ls
547  cat Dockerfile
548  cat index.html
549  vim index.html
550  git add .
551  git commit -m "Updated"
552  git config --global user.email mayank@example.com
553  git config --global user.name mayank
554  git commit -m "Updated"
555  git push
556  vim index.html
557  git add .
558  git commit -m "Updated"
559  git push
560  vim index.html
561  git add .
562  git commit -m "Updated"
```

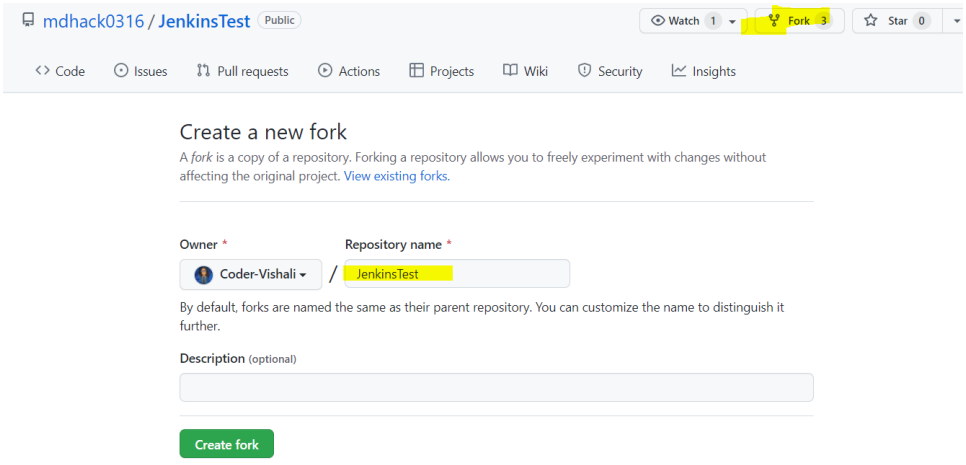
TASK:

Task 1: Create a GitHub account.

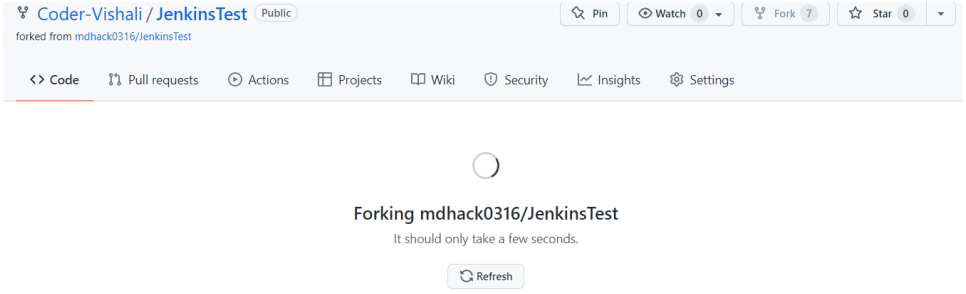


Task 2: Fork this repo https://github.com/mdhack0316/JenkinsTest

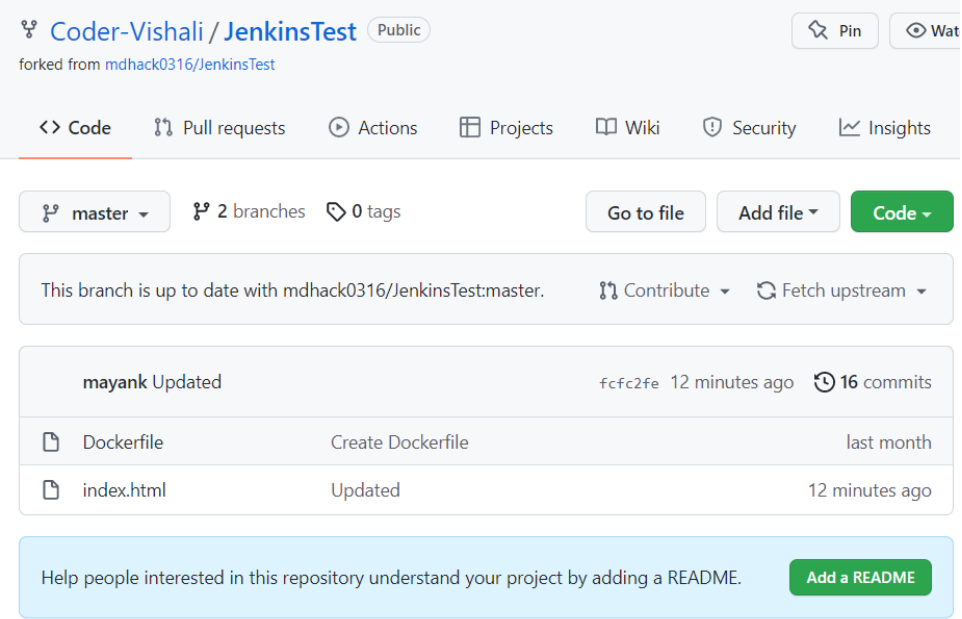
Click on Fork:



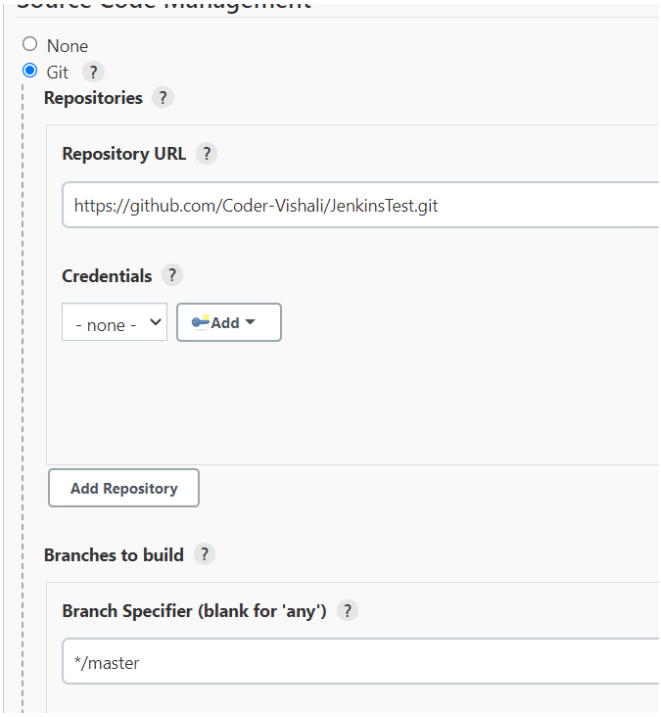
Processing:



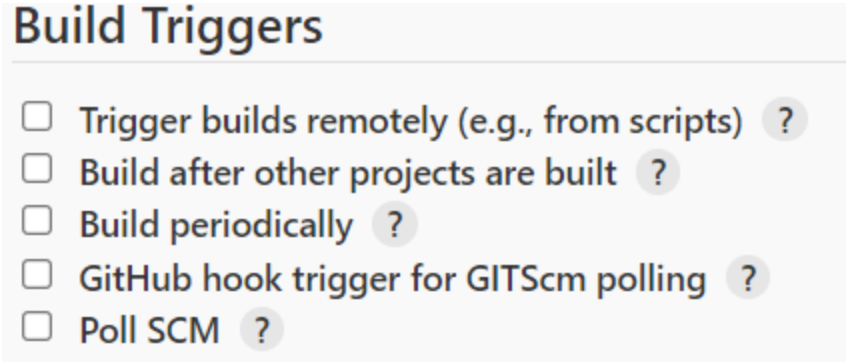
Now you can see the repo now:



Task 3: On Jenkins create a job that will help you to use above GitHub code in your Apache server

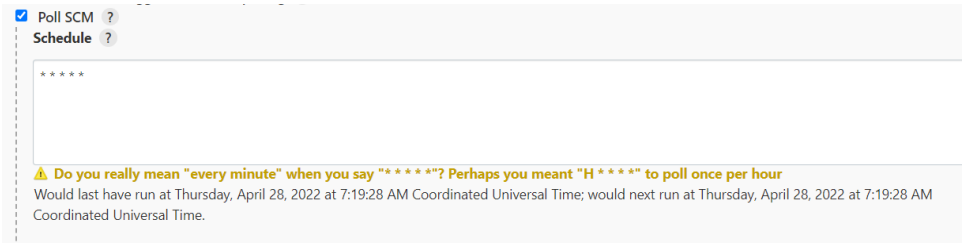


Manual click on the build: (no build trigger is selected at this step)




```
[root@vishali JenkinsTest]# git push
Username for 'https://github.com': Coder-Vishali
Password for 'https://Coder-Vishali@github.com':
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 2 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 328 bytes | 328.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/Coder-Vishali/JenkinsTest.git
   fcfc2fe..43d9efb  master -> master
[root@vishali JenkinsTest]#
```

Task 4: Make it every minute build, when you change the code, it should show automatically updated code



Task 5: Do the same task as above using GitHub Webhook trigger.

Build Triggers

- [ ] Trigger builds remotely (e.g., from scripts) ?
- [ ] Build after other projects are built ?
- [ ] Build periodically ?
- [x] GitHub hook trigger for GITScm polling ?
- [ ] Poll SCM ?

Build Environment

- [ ] Delete workspace before build starts
- [ ] Use secret text(s) or file(s) ?
- [ ] Abort the build if it's stuck
- [ ] Add timestamps to the Console Output
- [ ] Inspect build log for published Gradle build scans
- [ ] With Ant ?

Build

Execute shell ?

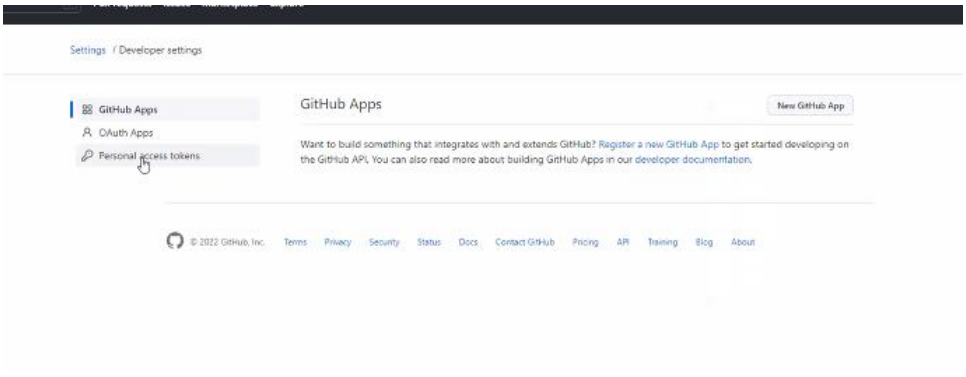
Command

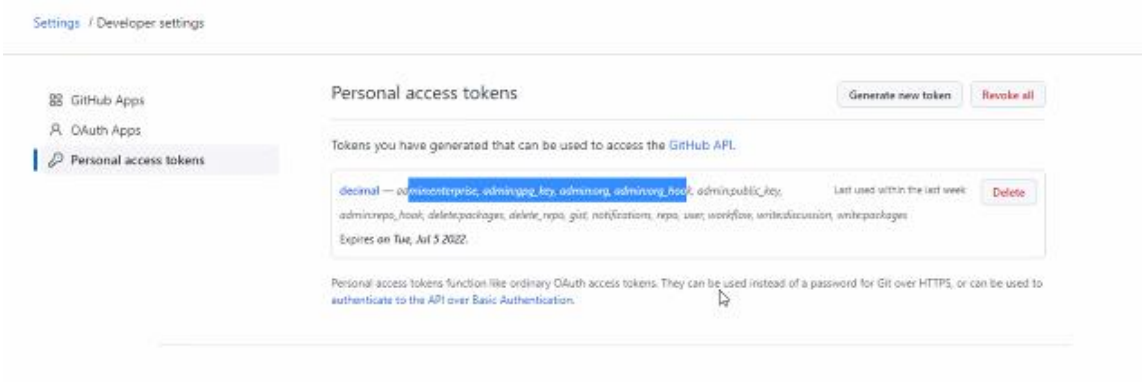
```
sudo cp index.html /var/www/html/
```

Check the website:

(Make sure to install GIT in your machine)

Generation of token: Go to setting -> Personal access -> token



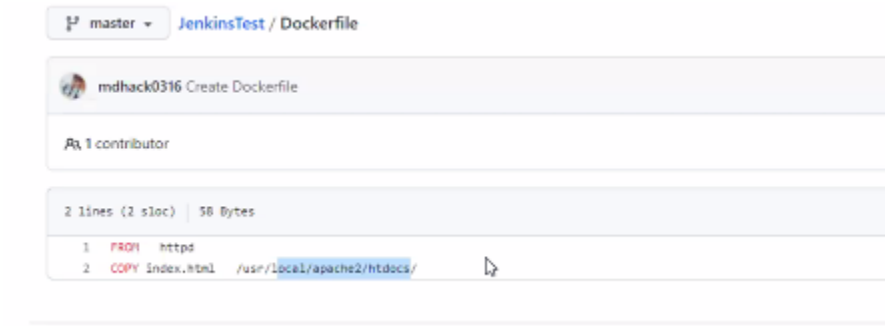


Containerization part of Docker:

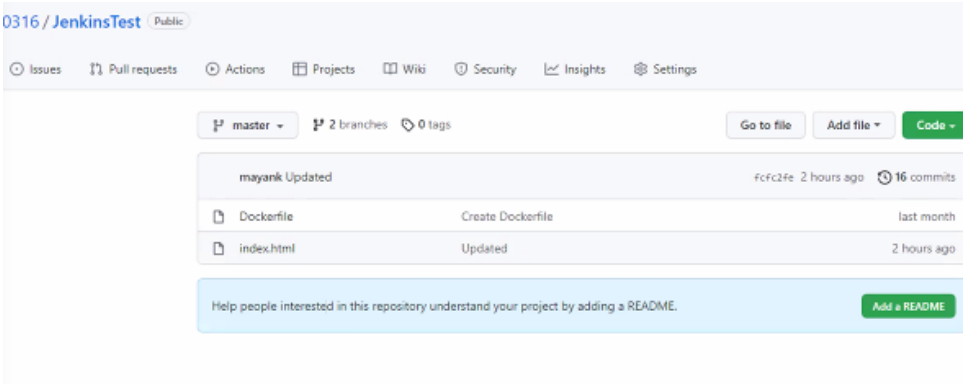
Embed docker, container, Jenkins and other platforms.

How to create a website in the Jenkins into the container?

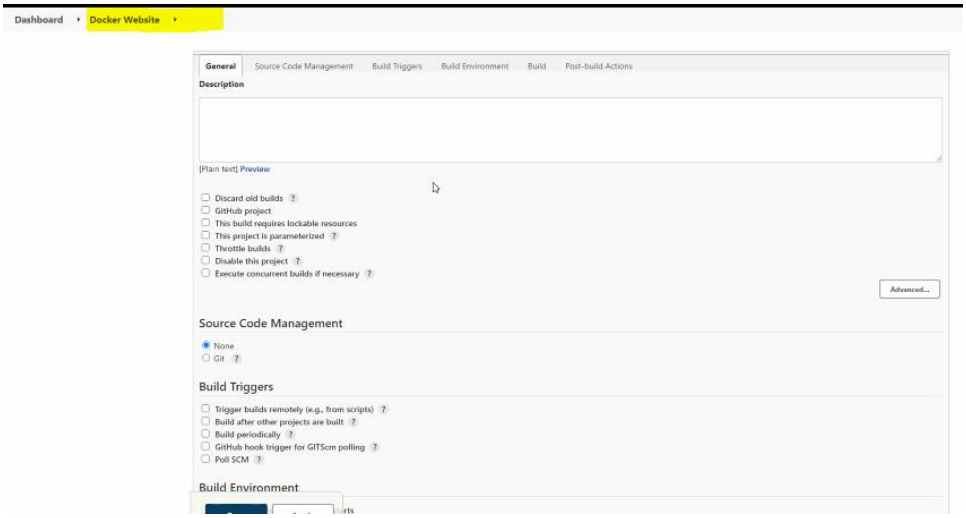
Dockerfile:



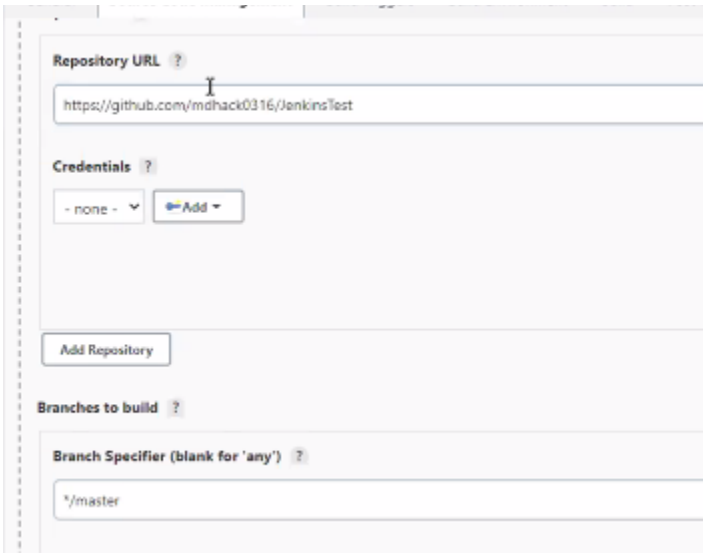
Same Repo:



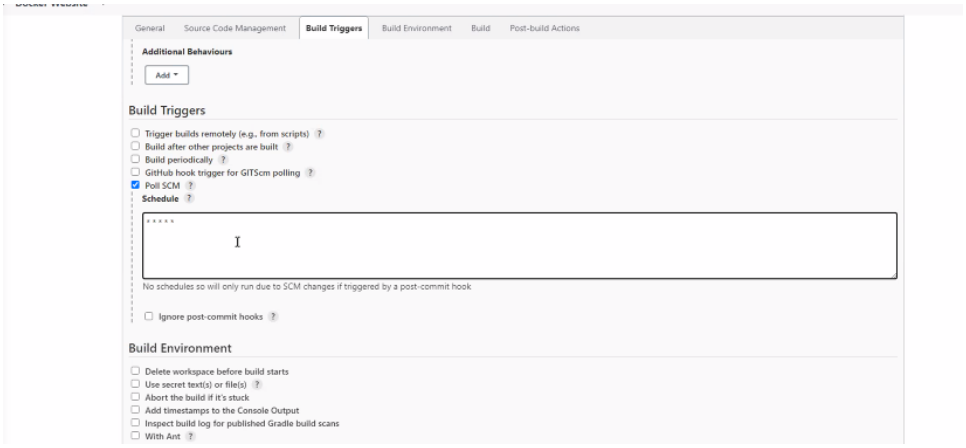
Create a new job:



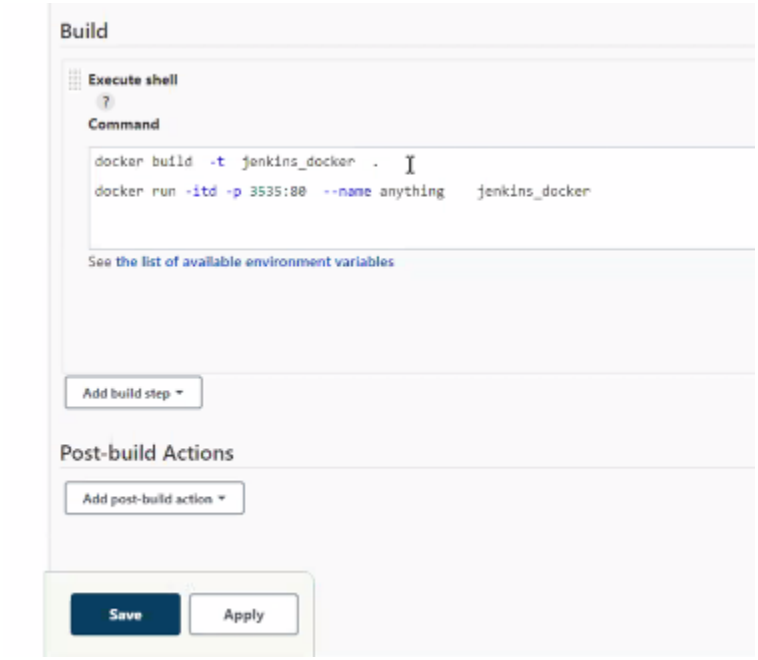
Clone the repo:



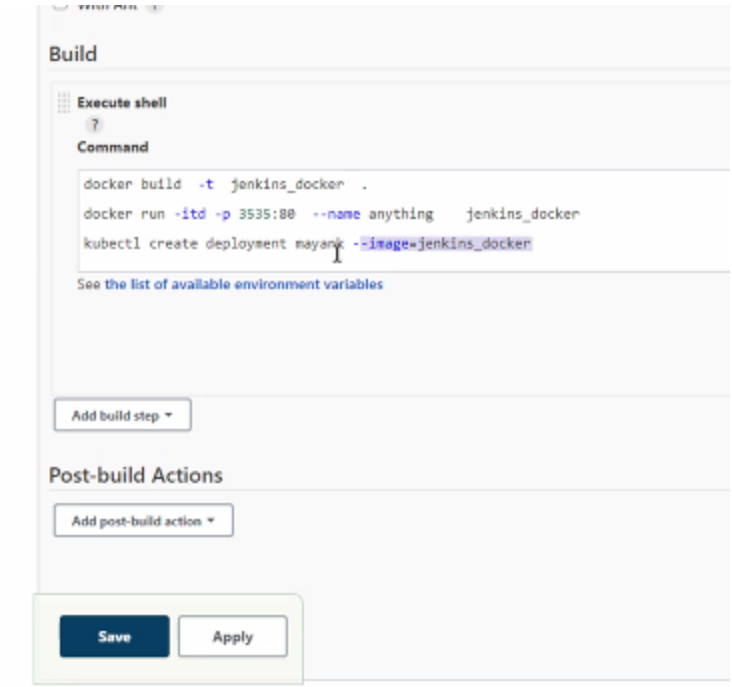
Trigger builds:

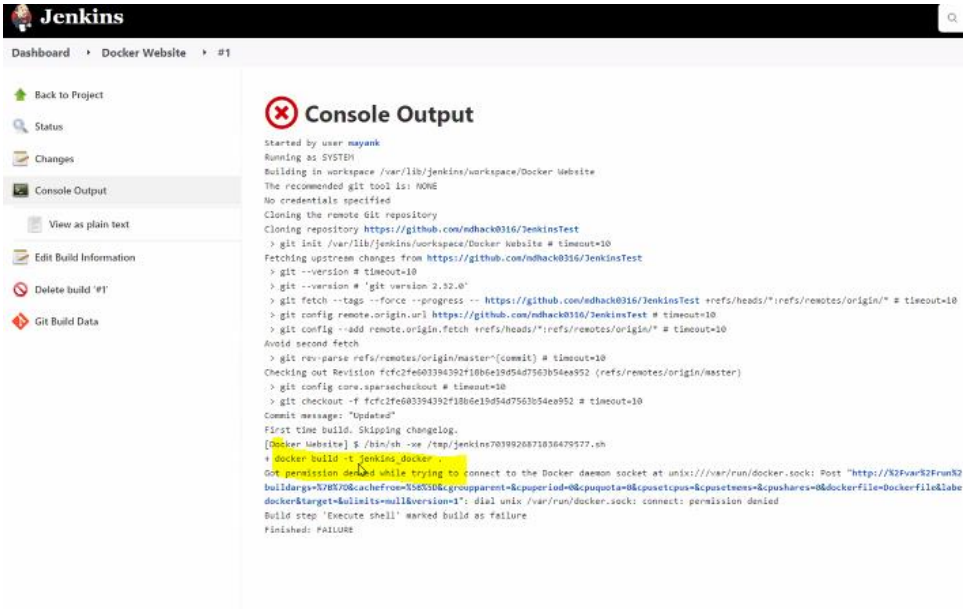


Commands:

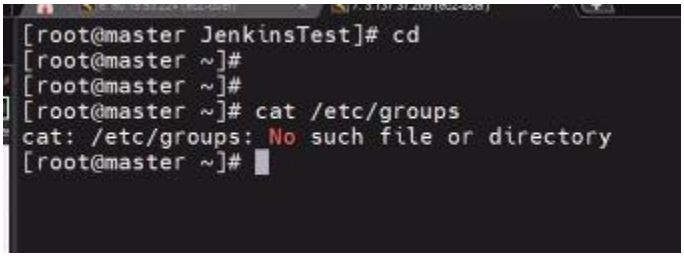


Problem here is – it will get the image only from docker hub.





Docker is a root less container:

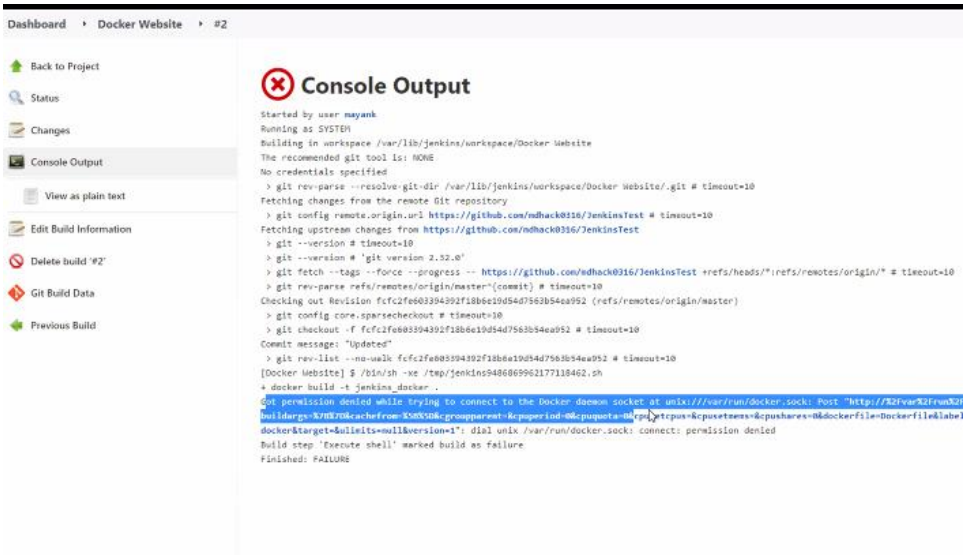


Creating the docker groups:

```
jenkins:x:991:
[root@master ~]# usermod -aG docker  jenkins
```

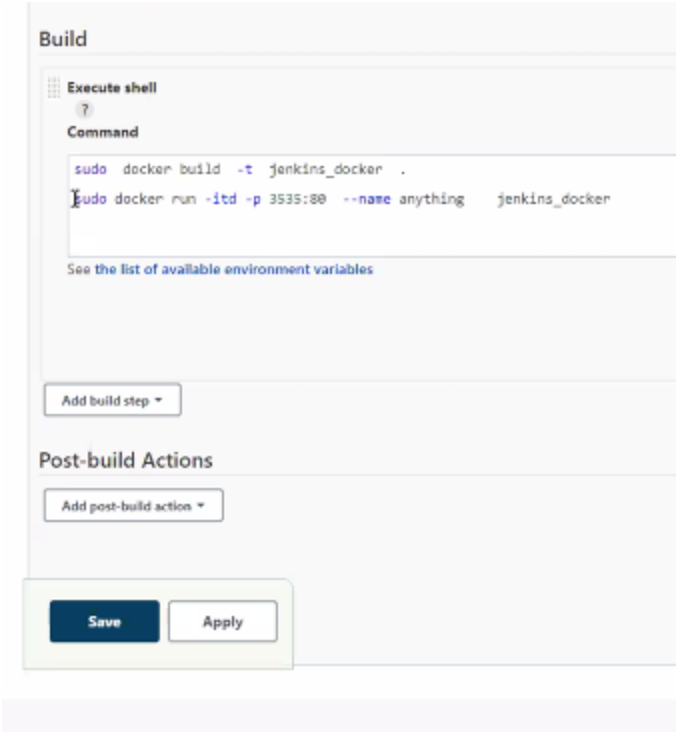
```
stapusr:x:156:
stapsys:x:157:
stapdev:x:158:
screen:x:84:
tcpdump:x:72:
ec2-user:x:1000:
cgred:x:993:
docker:x:992:jenkins
apache:x:48:
jenkins:x:991:
[root@master ~]#
```

Build is failing still:

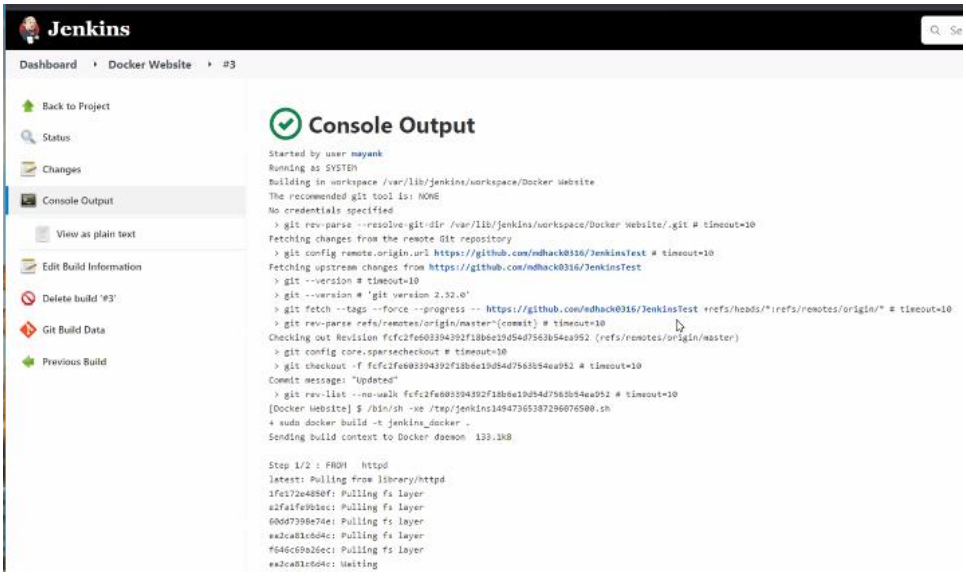



```
[root@master ~]# systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; vendor preset:
   Active: active (running) since Thu 2022-04-28 03:59:48 UTC; 5h 3min ago
     Docs: https://docs.docker.com
 Main PID: 3227 (dockerd)
    Tasks: 18
   Memory: 141.0M
   CGroup: /system.slice/docker.service
           └─3227 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd

Apr 28 03:59:47 master dockerd[3227]: time="2022-04-28T03:59:47.694873664Z" level=
Apr 28 03:59:48 master dockerd[3227]: time="2022-04-28T03:59:48.161407508Z" level=
Apr 28 03:59:48 master dockerd[3227]: time="2022-04-28T03:59:48.253157616Z" level=
Apr 28 03:59:48 master dockerd[3227]: time="2022-04-28T03:59:48.383364210Z" level=
Apr 28 03:59:48 master dockerd[3227]: time="2022-04-28T03:59:48.384343189Z" level=
Apr 28 03:59:48 master systemd[1]: Started Docker Application Container Engine.
Apr 28 03:59:48 master dockerd[3227]: time="2022-04-28T03:59:48.412227849Z" level=
Apr 28 04:00:01 master dockerd[3227]: time="2022-04-28T04:00:01.876726062Z" level=
Apr 28 04:00:05 master dockerd[3227]: time="2022-04-28T04:00:05.103976536Z" level=
Apr 28 04:00:05 master dockerd[3227]: time="2022-04-28T04:00:05.901081779Z" level=
Hint: Some lines were ellipsized, use -l to show in full.
[root@master ~]#
```

Add sudo:



Build is success now!



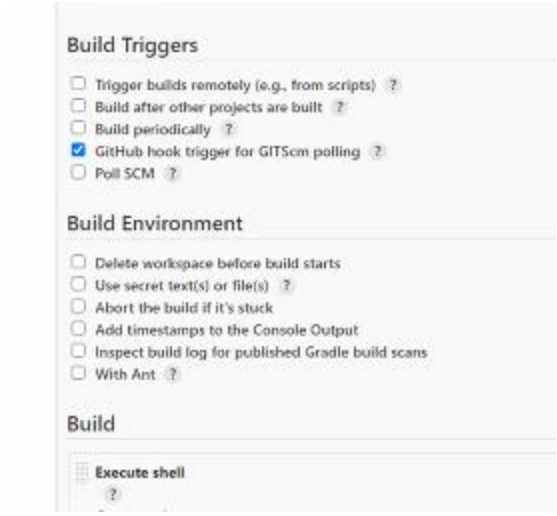
On docker container now this website is running!

If there is a change in the content of html, but the build will fail

Why?

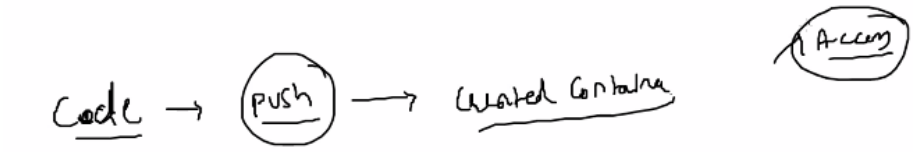
Port is already name of the container is already in use

Add webhook:



Edit html file:

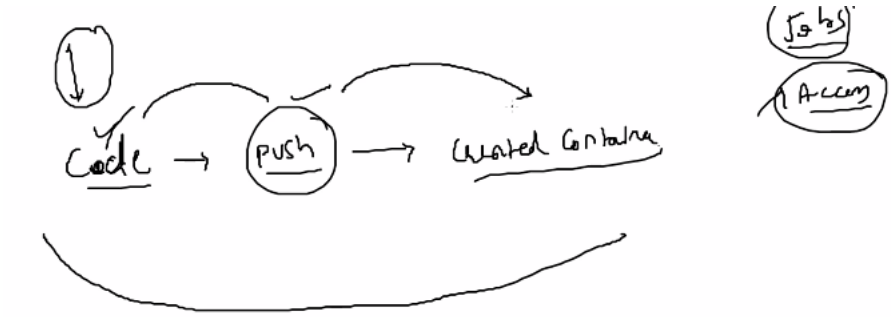
Check the website: (build will automatically generate)

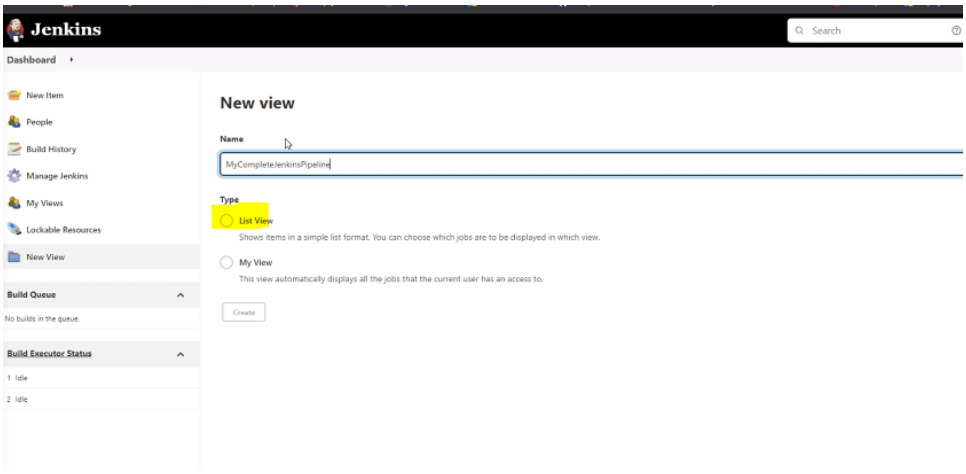


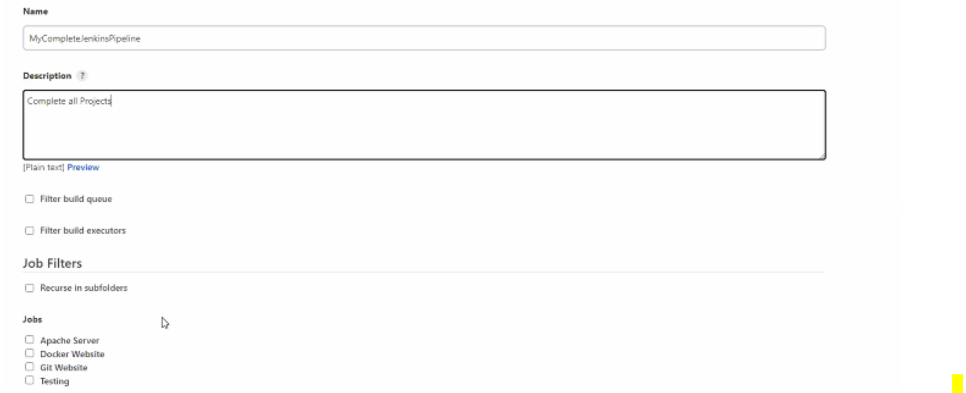
But it is not a continuous integration:

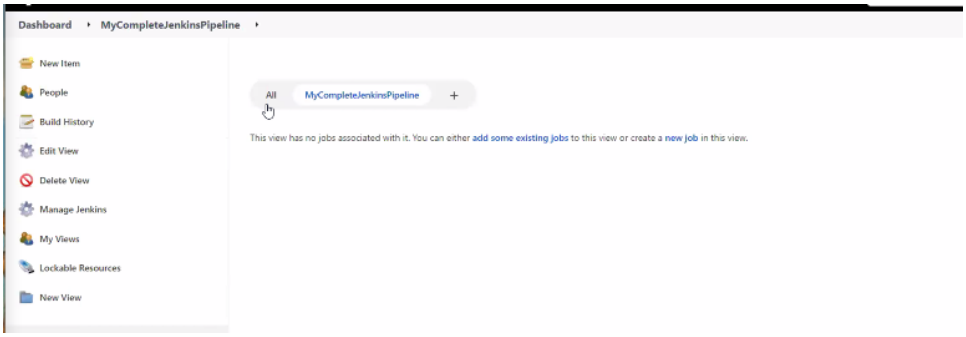
We will try to create a multiple job and dump into one job .

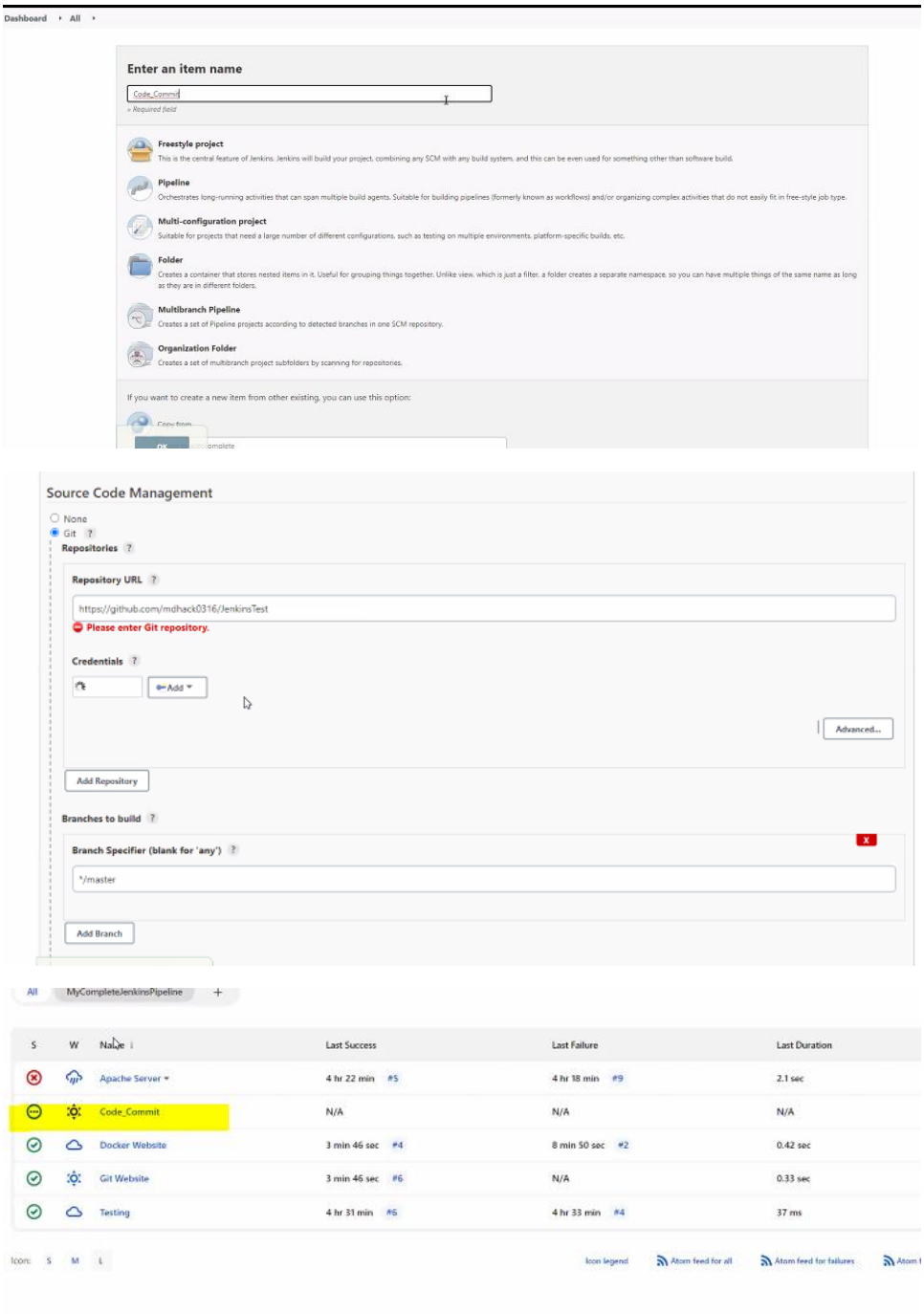
If the first one passes, then go to the next step... like that



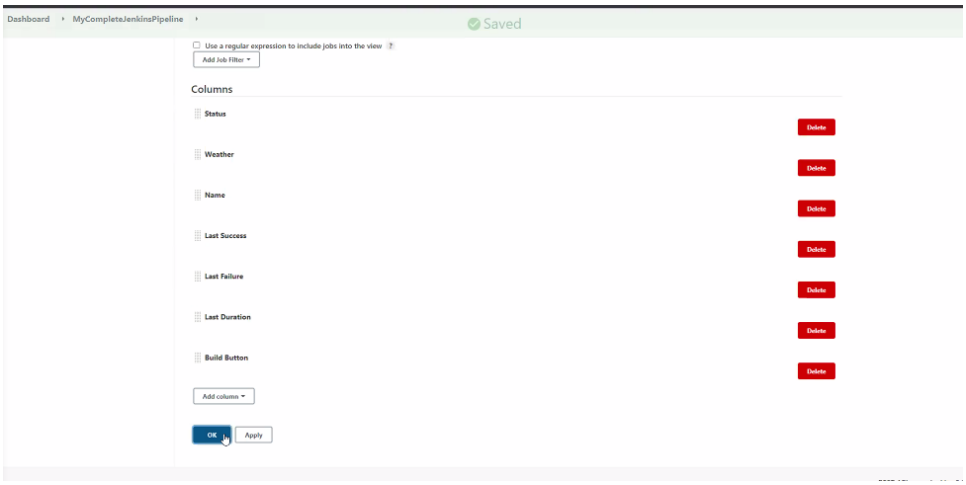

Jenkins
Search
Dashboard
New Item
People
Build History
Manage Jenkins
My Views
Lockable Resources
New View
Build Queue
No builds in the queue.
Build Executor Status
1 Idle
2 Idle
New view
Name
MyCompleteJenkinsPipeline
Type
List View
Shows items in a simple list format. You can choose which jobs are to be displayed in which view.
My View
This view automatically displays all the jobs that the current user has an access to.
Create

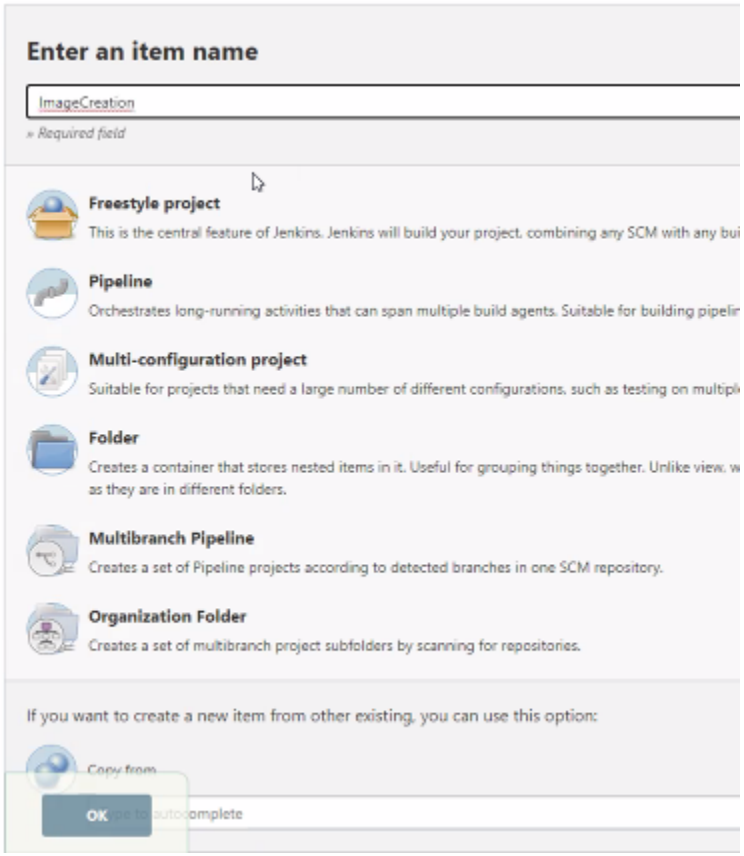

Name
MyCompleteJenkinsPipeline
Description
Complete all Projects
[Plain text] Preview
Filter build queue
Filter build executors
Job Filters
Recurse in subfolders
Jobs
Apache Server
Docker Website
Git Website
Testing

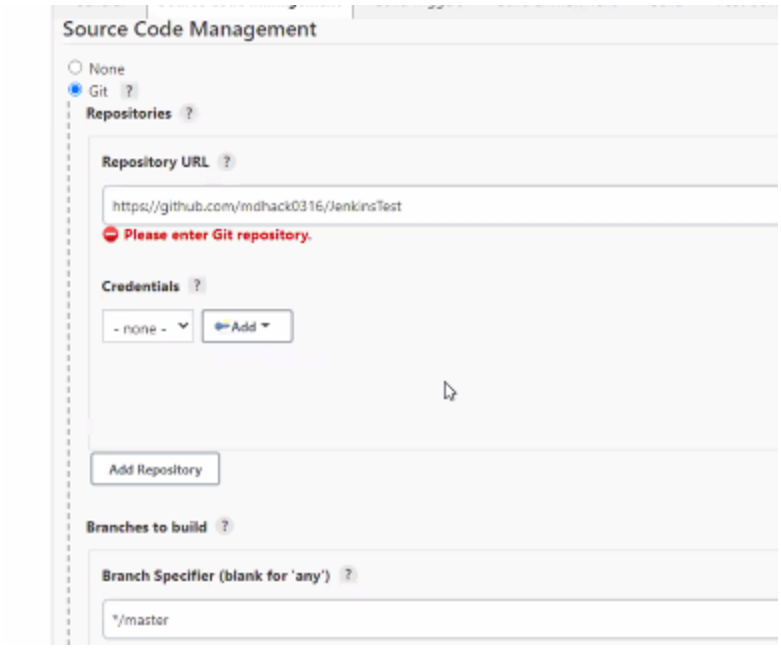

Dashboard
MyCompleteJenkinsPipeline
New Item
People
Build History
Edit View
Delete View
Manage Jenkins
My Views
Lockable Resources
New View
All
MyCompleteJenkinsPipeline
This view has no jobs associated with it. You can either add some existing jobs to this view or create a new job in this view.

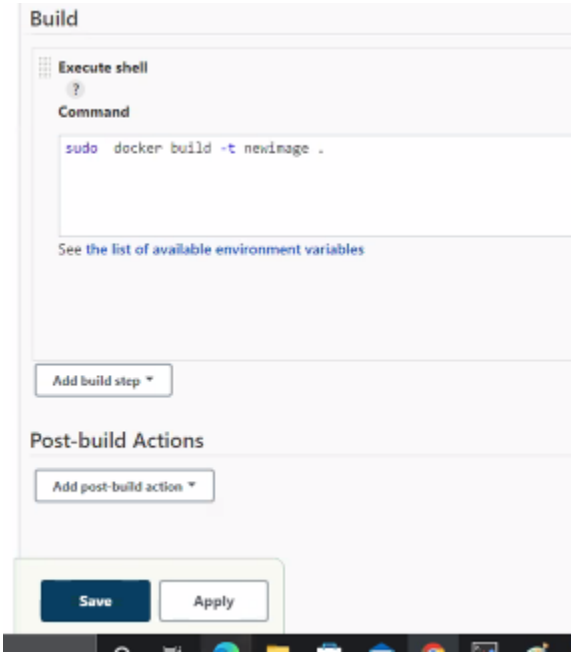


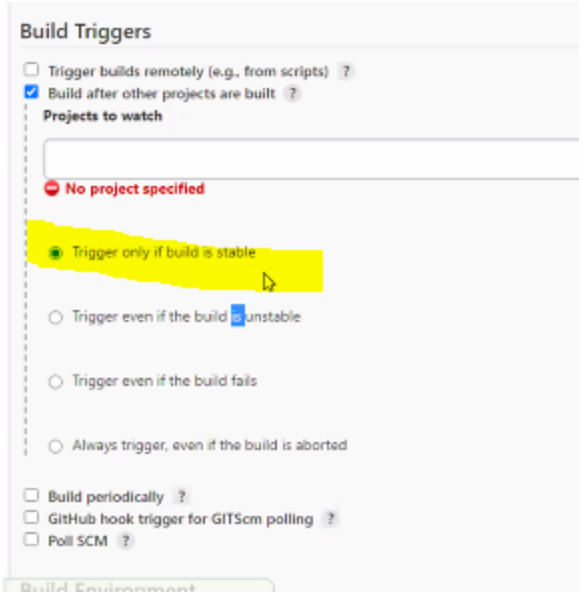
Add it in the views:

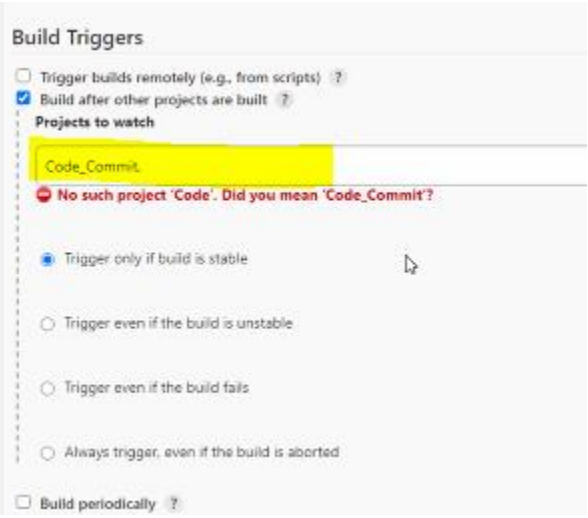

Dashboard
MyCompleteJenkinsPipeline
Saved
Use a regular expression to include jobs into the view
Add Job Filter
Columns
Status
Weather
Name
Last Success
Last Failure
Last Duration
Build Button
Delete
Add column
OK
Apply

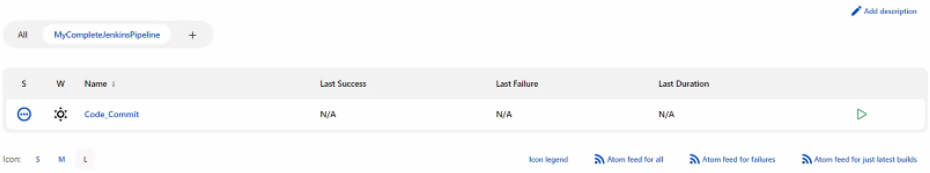

Enter an item name
ImageCreation
» Required field
Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any bui
Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelir
Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multipl
Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, w
as they are in different folders.
Multibranch Pipeline
Creates a set of Pipeline projects according to detected branches in one SCM repository.
Organization Folder
Creates a set of multibranch project subfolders by scanning for repositories.
If you want to create a new item from other existing, you can use this option:
Copy from
OK

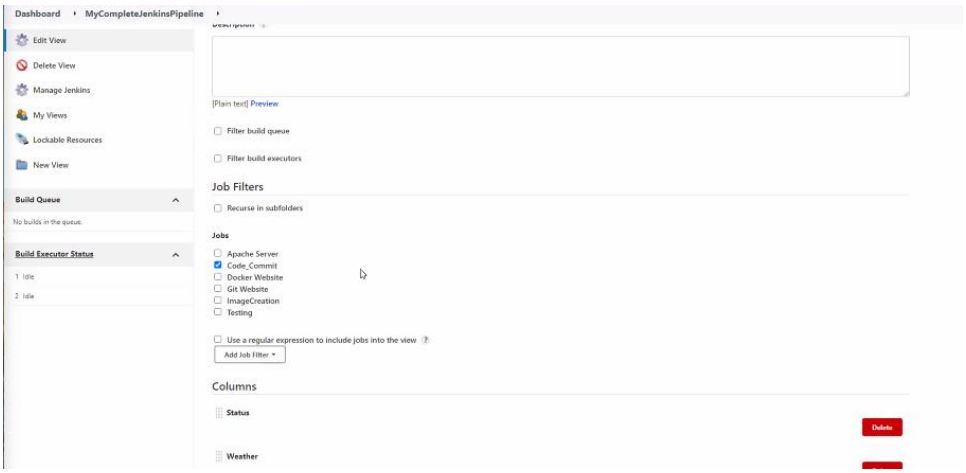

Source Code Management
None
Git
Repositories
Repository URL
https://github.com/mdhack0316/JenkinsTest
Please enter Git repository.
Credentials
- none -
Add
Add Repository
Branches to build
Branch Specifier (blank for 'any')
*/master

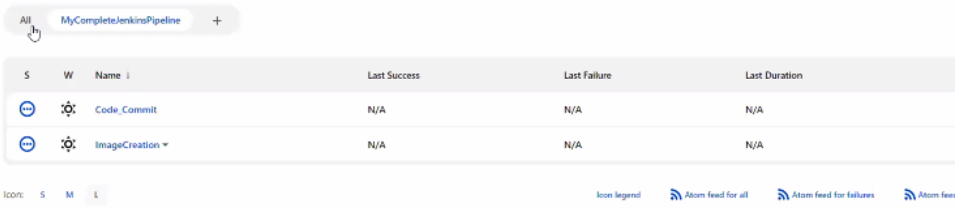

Build
Execute shell
Command
sudo docker build -t newimage .
See the list of available environment variables
Add build step
Post-build Actions
Add post-build action
Save
Apply

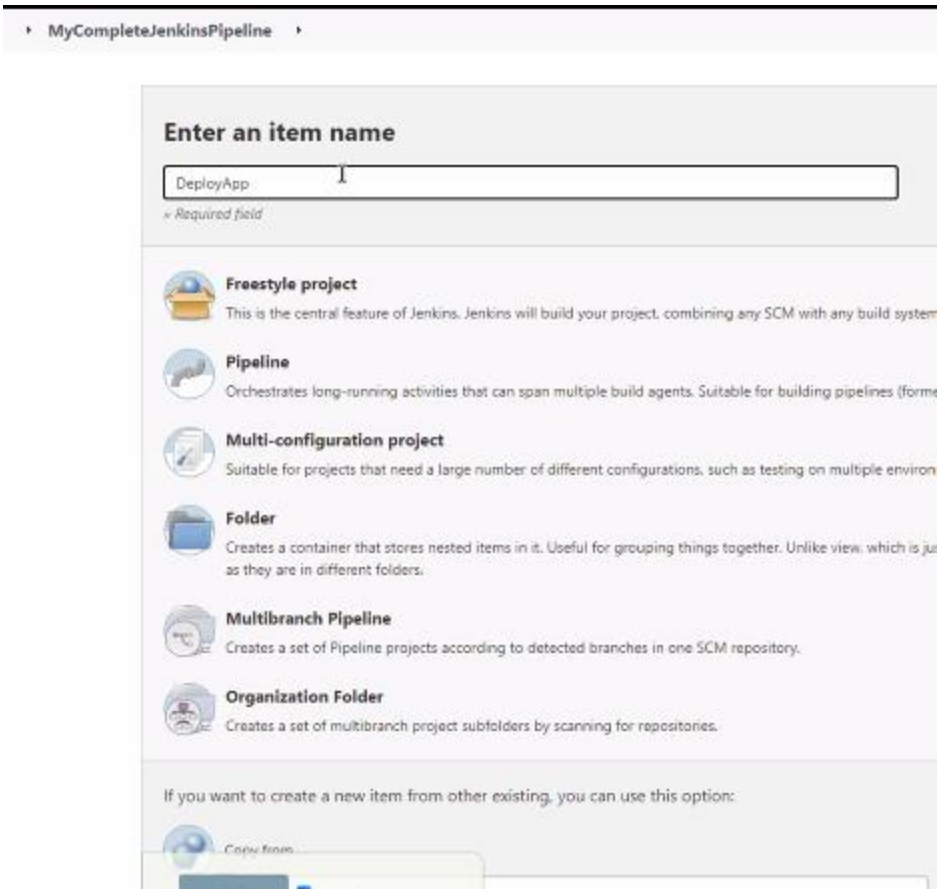

Build Triggers
Trigger builds remotely (e.g., from scripts)
Build after other projects are built
Projects to watch
No project specified
Trigger only if build is stable
Trigger even if the build is unstable
Trigger even if the build fails
Always trigger, even if the build is aborted
Build periodically
GitHub hook trigger for GITScm polling
Poll SCM

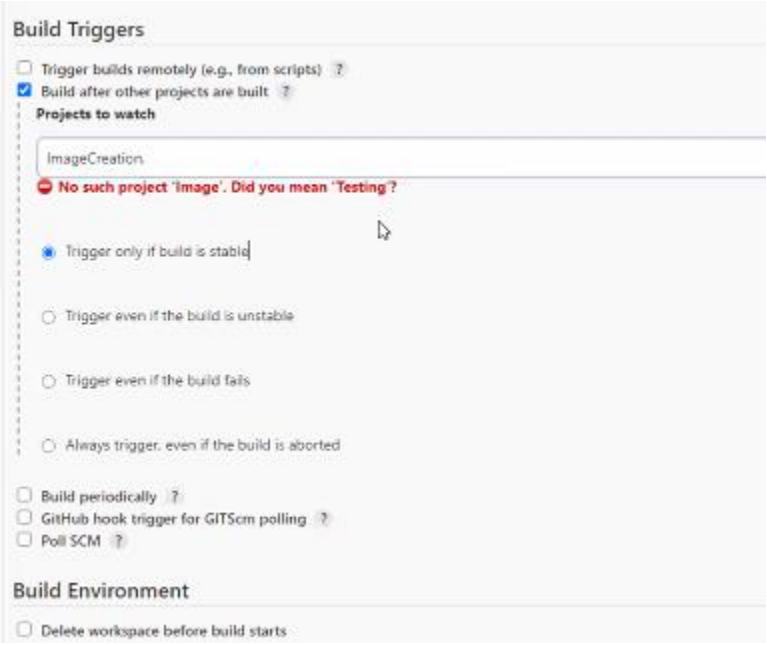

Build Triggers
Trigger builds remotely (e.g., from scripts)
Build after other projects are built
Projects to watch
Code_Commit
No such project 'Code'. Did you mean 'Code_Commit'?
Trigger only if build is stable
Trigger even if the build is unstable
Trigger even if the build fails
Always trigger, even if the build is aborted
Build periodically


Add description
All
MyCompleteJenkinsPipeline
S
W
Name
Last Success
Last Failure
Last Duration
Code_Commit
N/A
N/A
N/A
Icon: S M L
Icon legend
Atom feed for all
Atom feed for failures
Atom feed for just latest builds

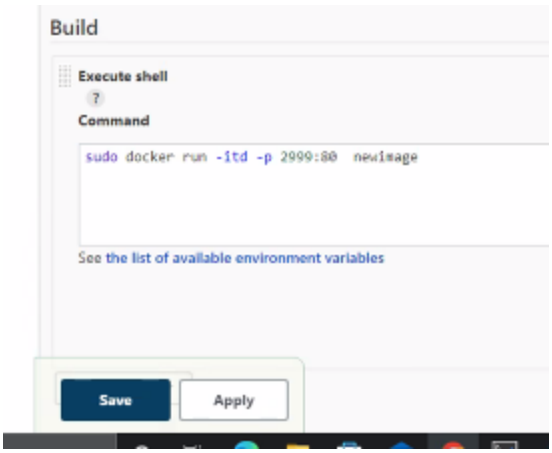

Dashboard
MyCompleteJenkinsPipeline
Edit View
Delete View
Manage Jenkins
My Views
Lockable Resources
New View
Build Queue
No builds in the queue.
Build Executor Status
1 Idle
2 Idle
[Plain text] Preview
Filter build queue
Filter build executors
Job Filters
Recurse in subfolders
Jobs
Apache Server
Code_Commit
Docker Website
Git Website
ImageCreation
Testing
Use a regular expression to include jobs into the view
Add Job Filter
Columns
Status
Weather
Delete


All
MyCompleteJenkinsPipeline
S
W
Name
Last Success
Last Failure
Last Duration
Code_Commit
N/A
N/A
N/A
ImageCreation
N/A
N/A
N/A
Icon: S M L
Icon legend
Atom feed for all
Atom feed for failures

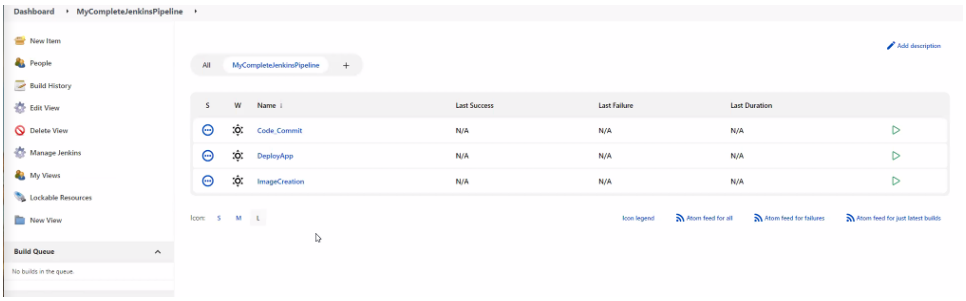


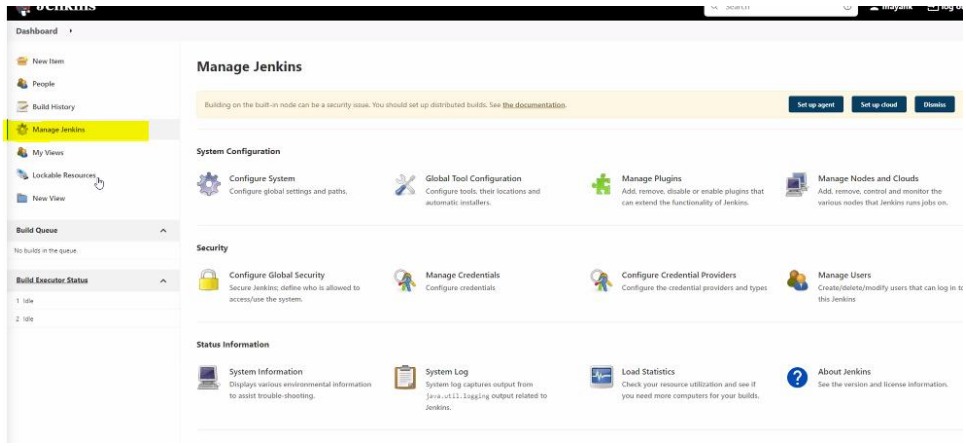


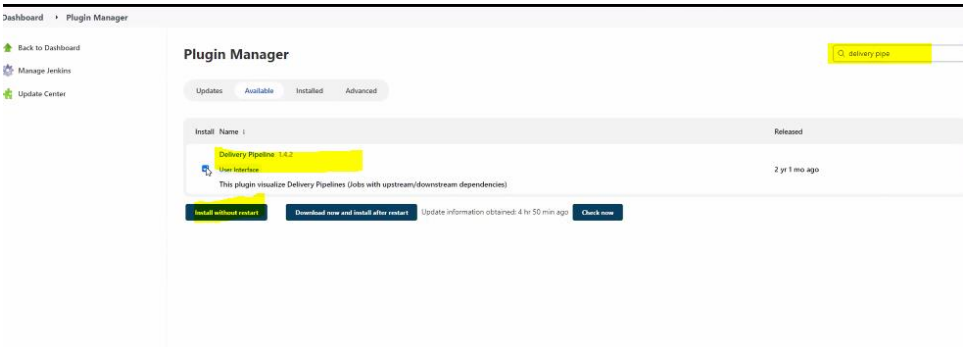
Build command:

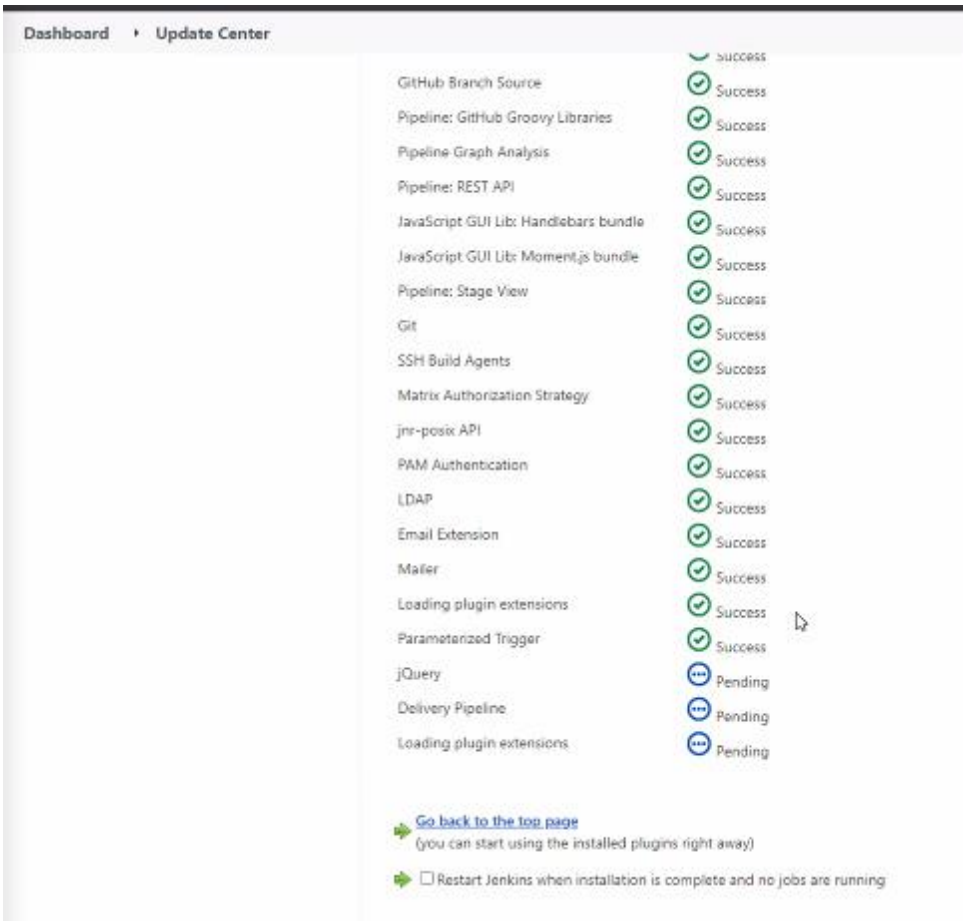


Add it to views:

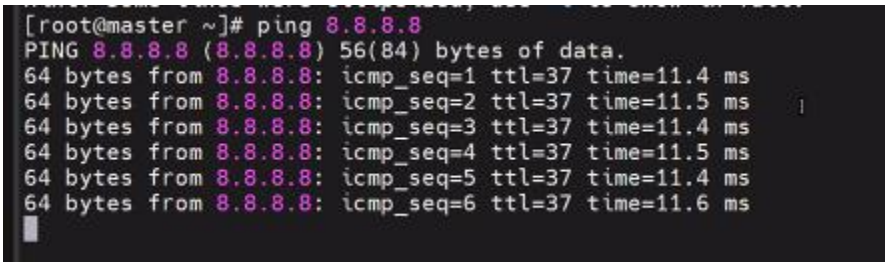




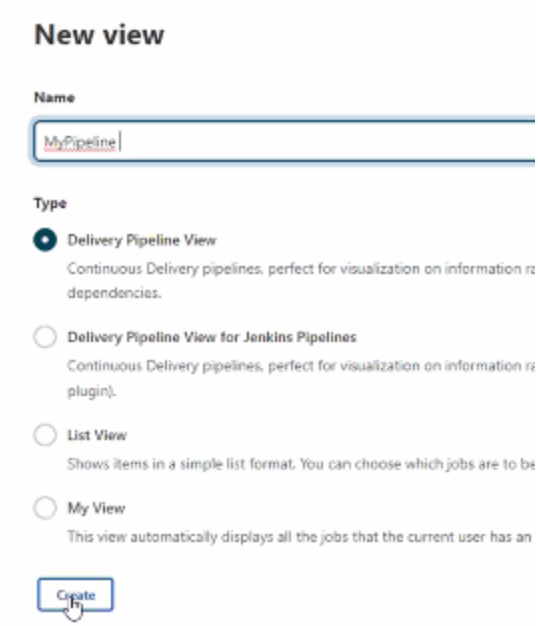


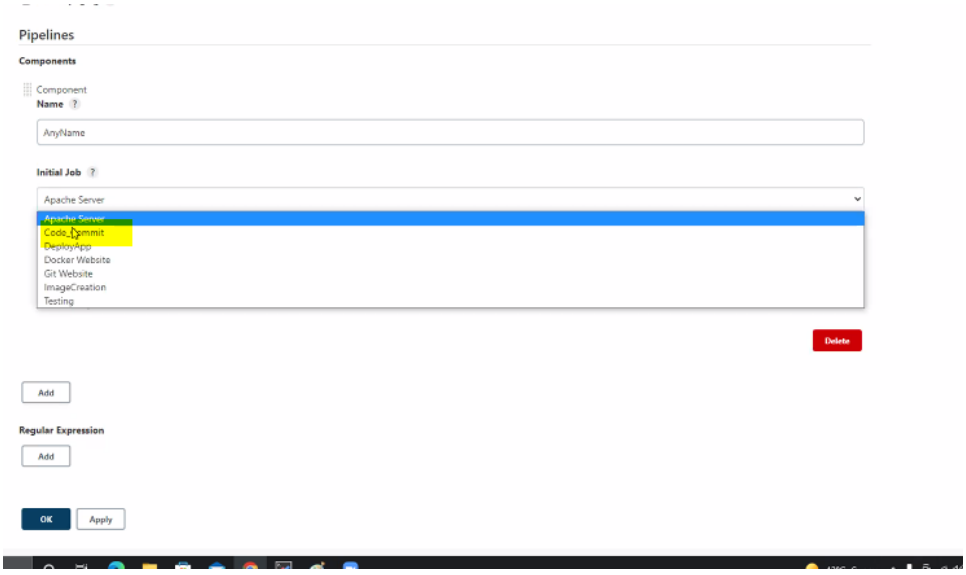


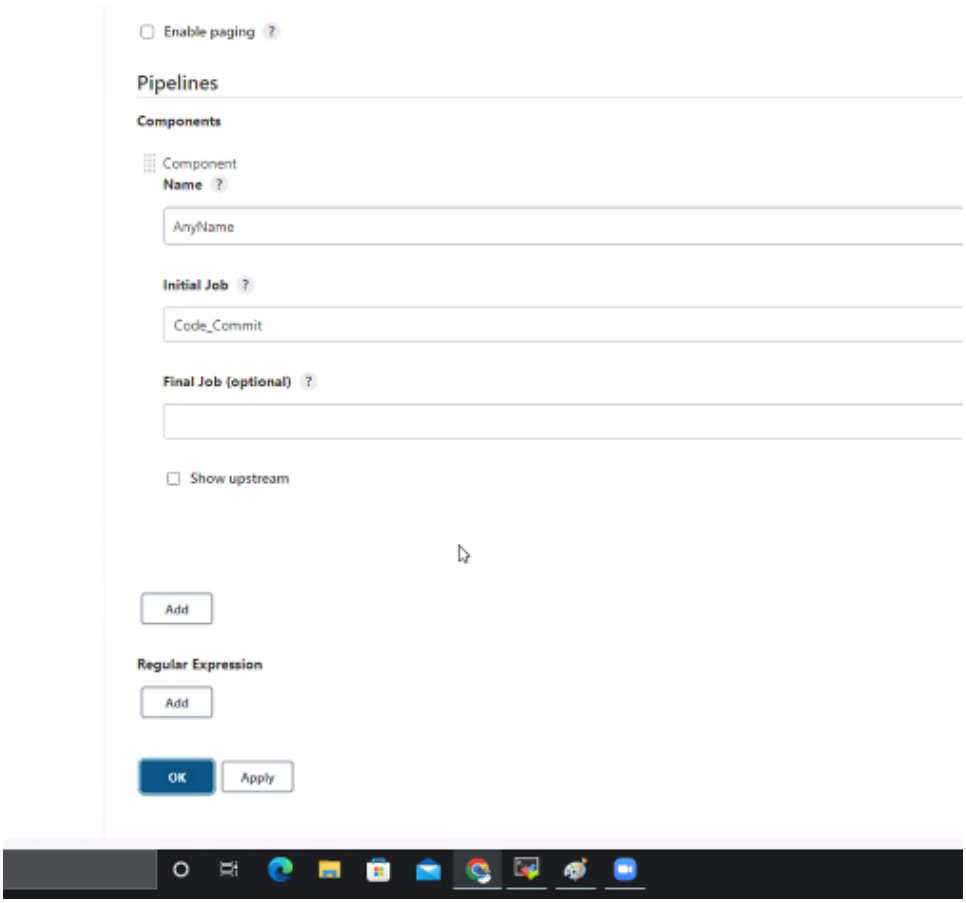
This is optional to do:

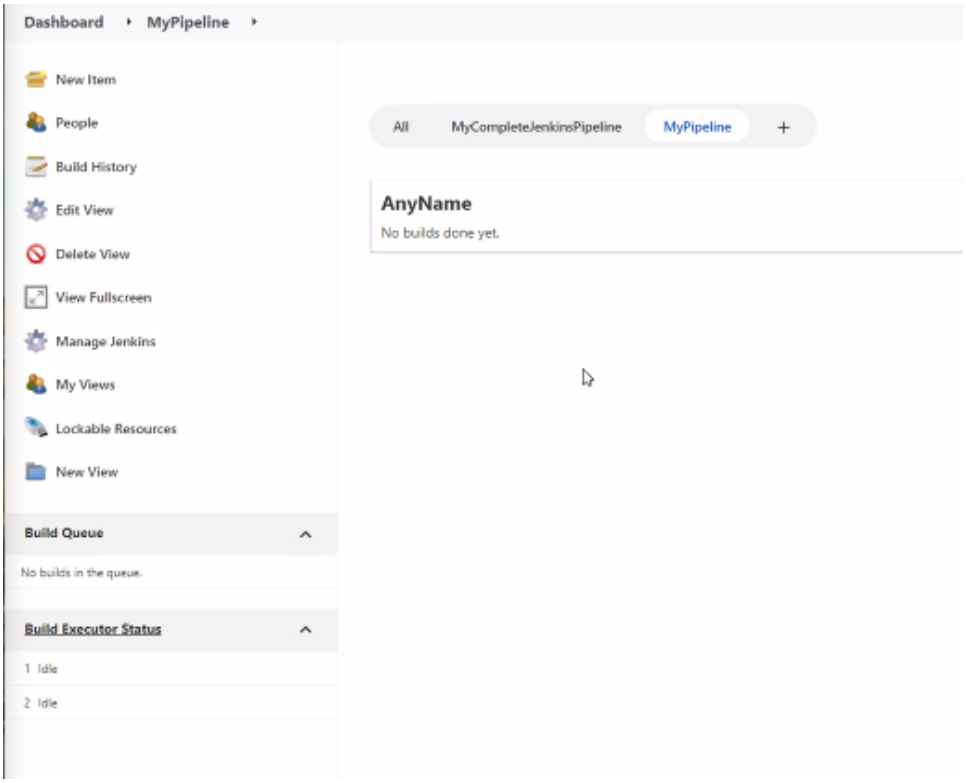


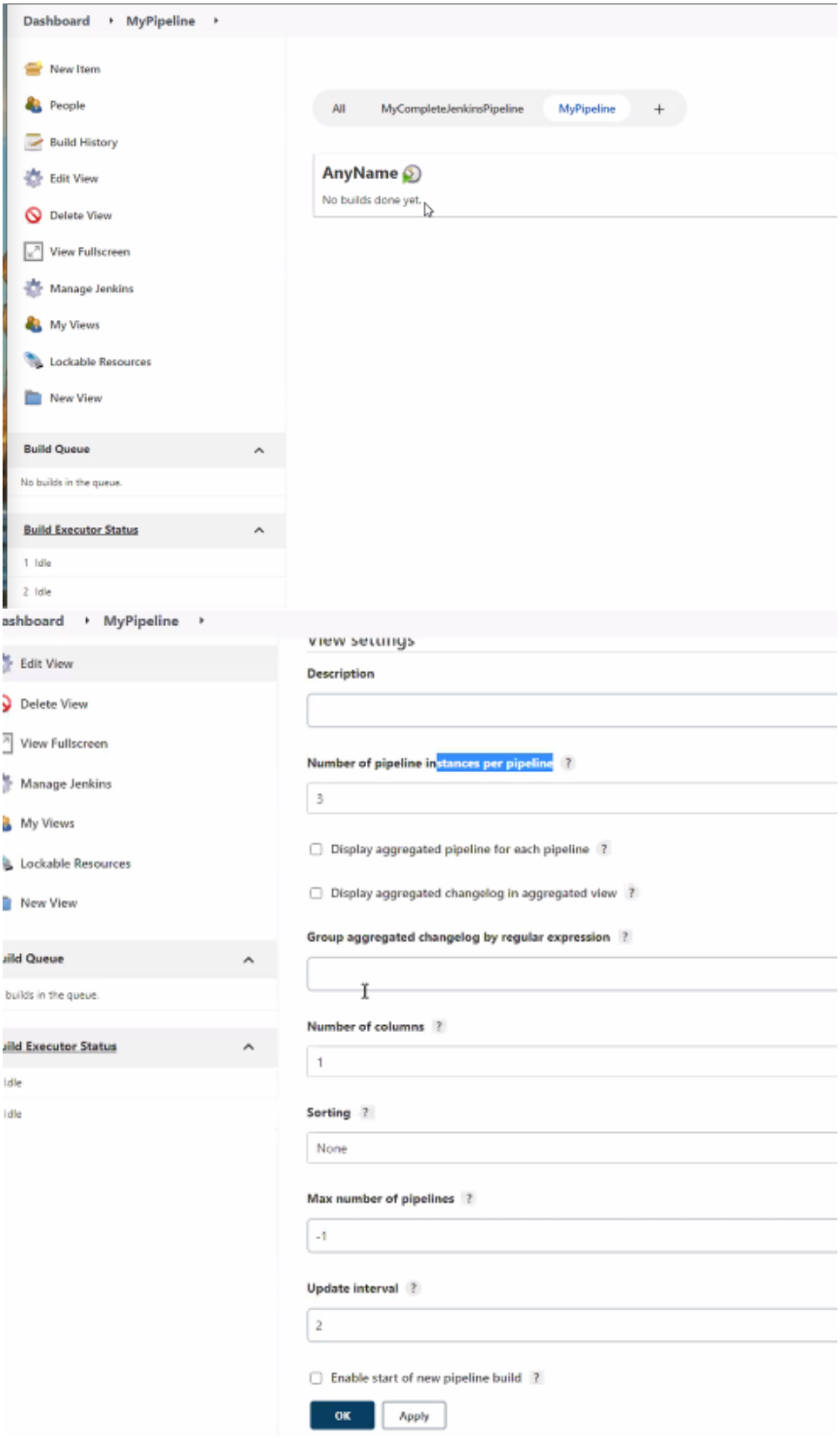
Restart the Jenkins

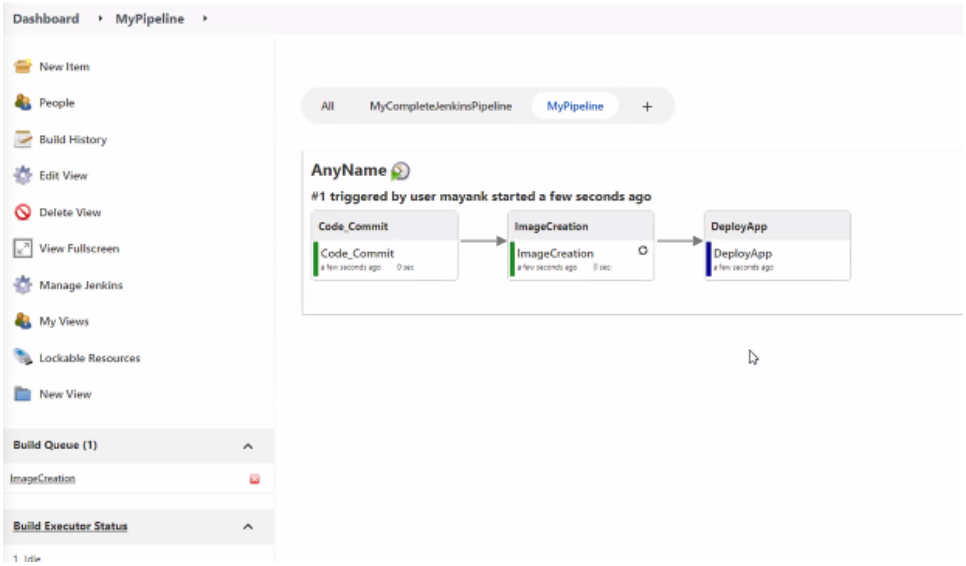

New view
Name
MyPipeline
Type
Delivery Pipeline View
Continuous Delivery pipelines, perfect for visualization on information
dependencies.
Delivery Pipeline View for Jenkins Pipelines
Continuous Delivery pipelines, perfect for visualization on information
plugin).
List View
Shows items in a simple list format. You can choose which jobs are to be
My View
This view automatically displays all the jobs that the current user has an
Create

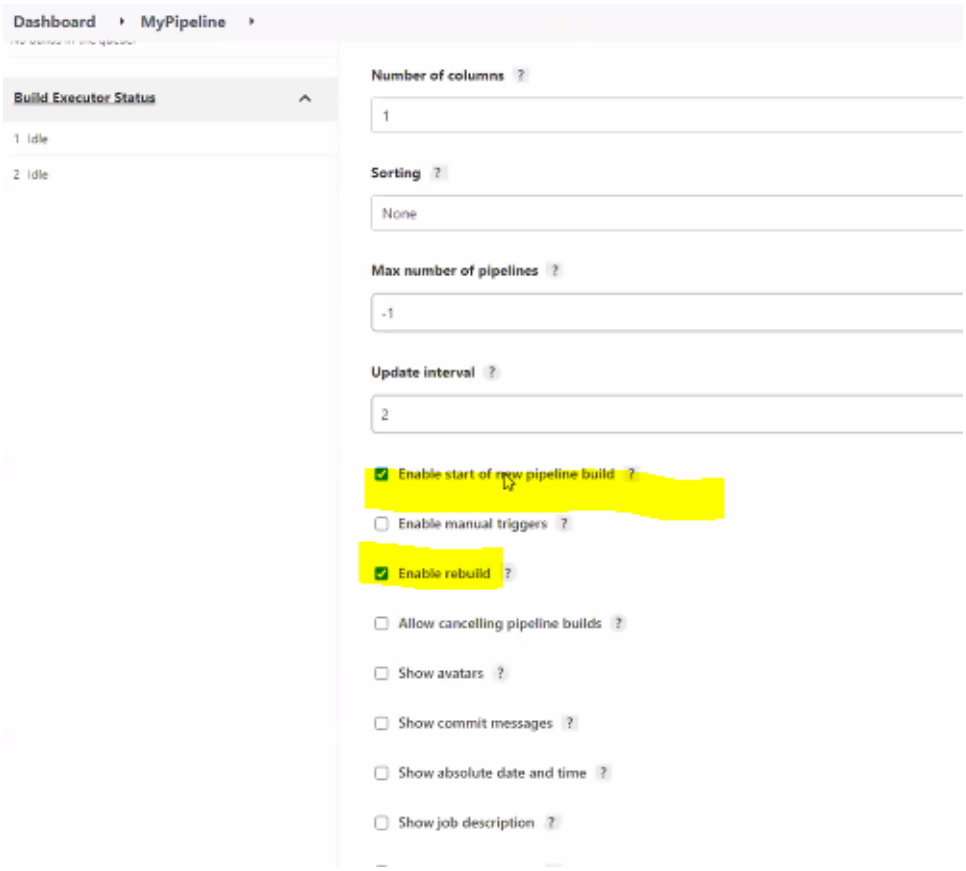

Pipelines
Components
Component
Name
AnyName
Initial Job
Apache Server
Apache Server
Code_Commit
DeployApp
Docker Website
Git Website
ImageCreation
Testing
Delete
Add
Regular Expression
Add
OK
Apply

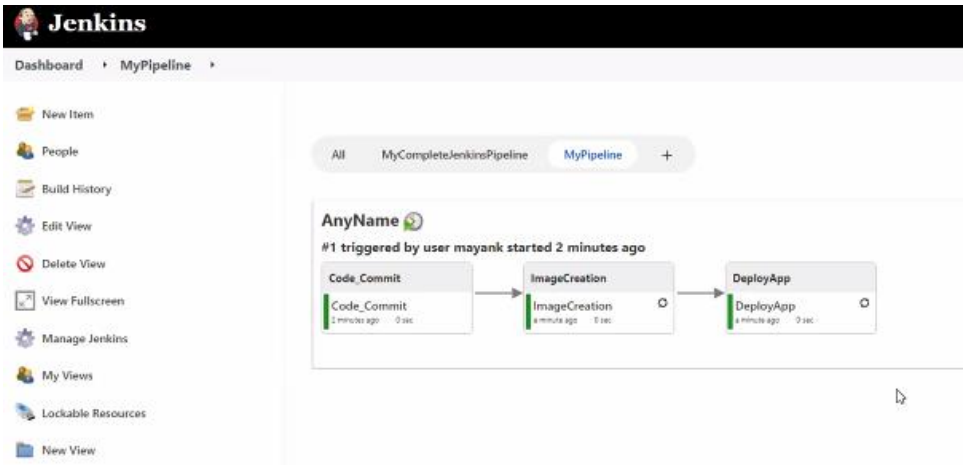

Enable paging
Pipelines
Components
Component
Name
AnyName
Initial Job
Code_Commit
Final Job (optional)
Show upstream
Add
Regular Expression
Add
OK
Apply

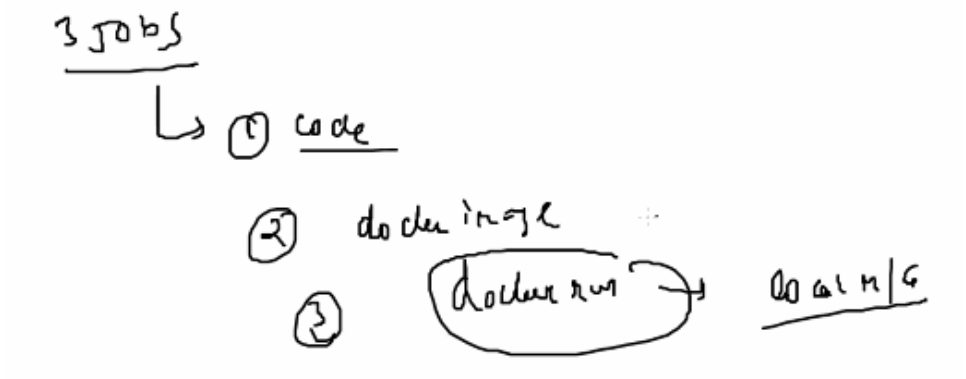

Dashboard
MyPipeline
New Item
People
Build History
Edit View
Delete View
View Fullscreen
Manage Jenkins
My Views
Lockable Resources
New View
Build Queue
No builds in the queue.
Build Executor Status
1 Idle
2 Idle
All
MyCompleteJenkinsPipeline
MyPipeline
AnyName
No builds done yet.


Dashboard
MyPipeline
New Item
People
Build History
Edit View
Delete View
View Fullscreen
Manage Jenkins
My Views
Lockable Resources
New View
Build Queue
No builds in the queue.
Build Executor Status
1 Idle
2 Idle
All
MyCompleteJenkinsPipeline
MyPipeline
AnyName
No builds done yet.
Edit View
Delete View
View Fullscreen
Manage Jenkins
My Views
Lockable Resources
New View
Description
Number of pipeline instances per pipeline
3
Display aggregated pipeline for each pipeline
Display aggregated changelog in aggregated view
Group aggregated changelog by regular expression
Number of columns
1
Sorting
None
Max number of pipelines
-1
Update interval
2
Enable start of new pipeline build
OK
Apply

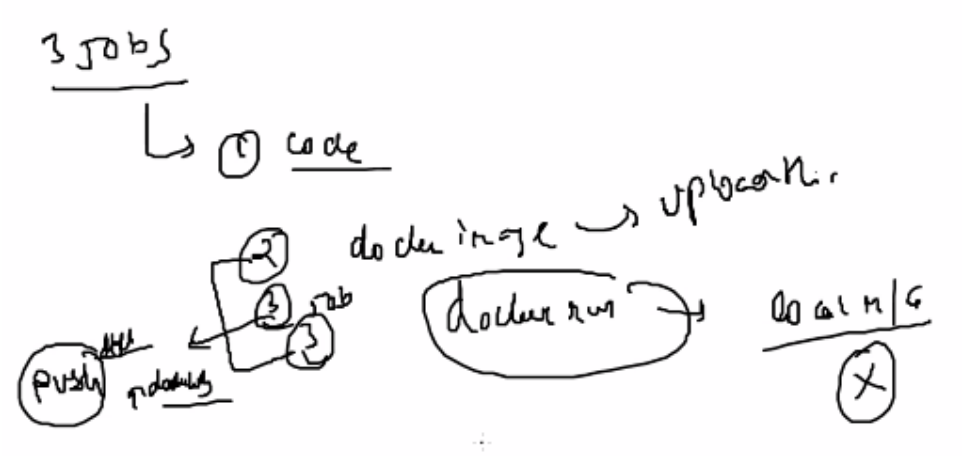

Dashboard
MyPipeline
New Item
People
Build History
Edit View
Delete View
View Fullscreen
Manage Jenkins
My Views
Lockable Resources
New View
Build Queue (1)
ImageCreation
Build Executor Status
All
MyCompleteJenkinsPipeline
MyPipeline
AnyName
#1 triggered by user mayank started a few seconds ago
Code_Commit
ImageCreation
DeployApp


Dashboard
MyPipeline
Build Executor Status
1 Idle
2 Idle
Number of columns
1
Sorting
None
Max number of pipelines
-1
Update interval
2
Enable start of new pipeline build
Enable manual triggers
Enable rebuild
Allow cancelling pipeline builds
Show avatars
Show commit messages
Show absolute date and time
Show job description



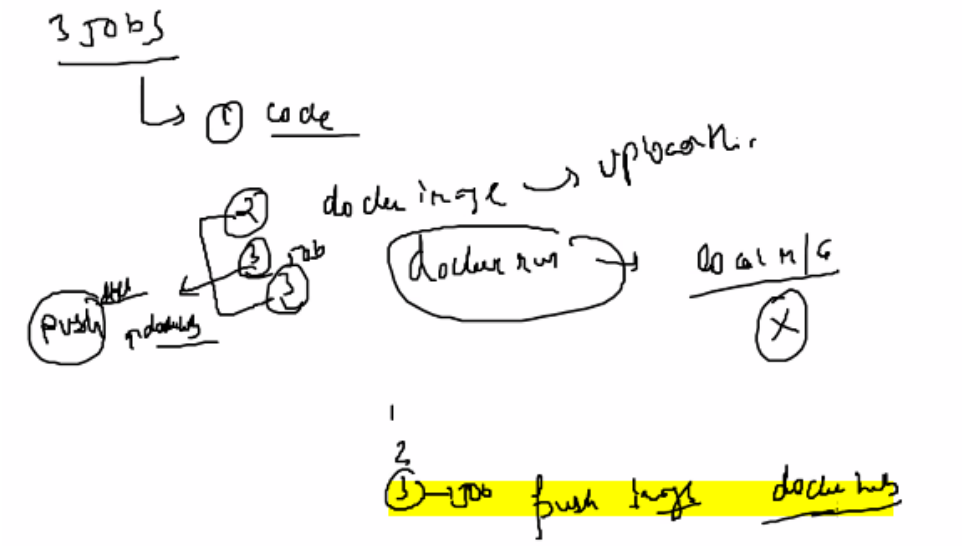


If someone else is using it, there is an error – someone is using it.

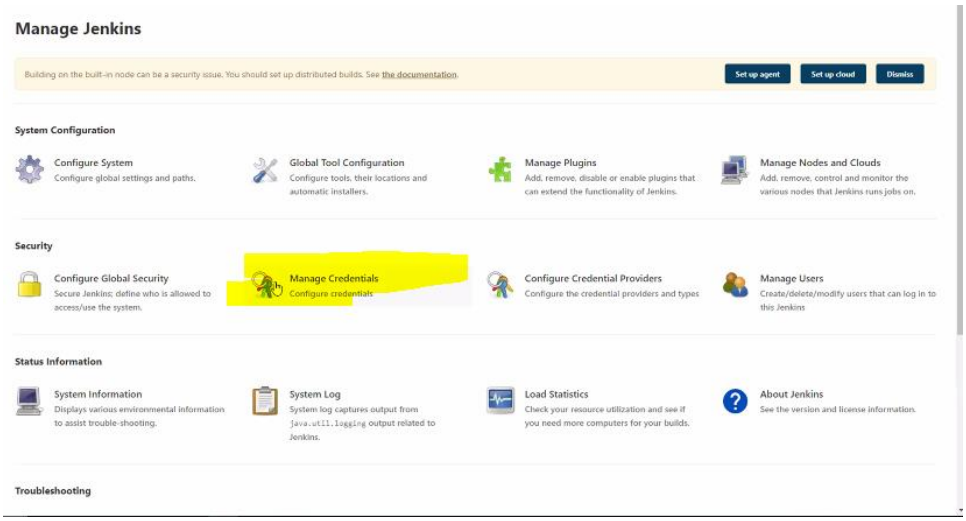


Additional step: Push the image in your account.

When you create a container, you have to add the image into the Dockerhub.



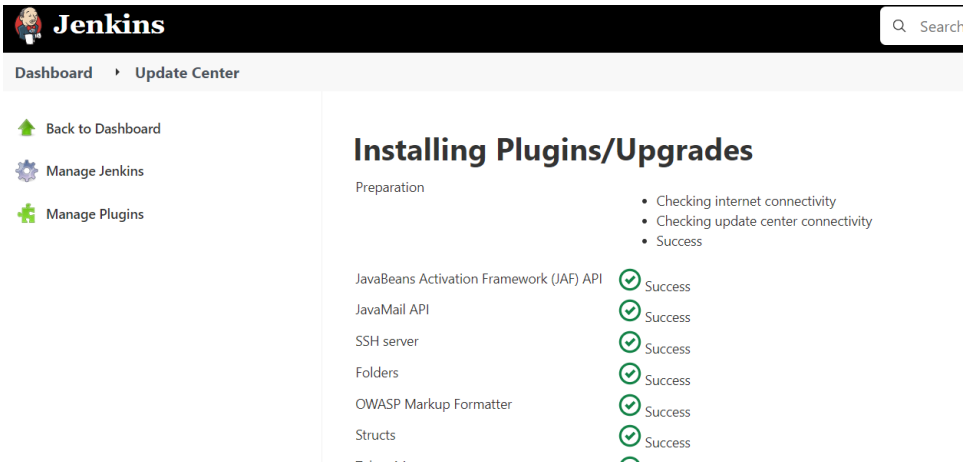
How to provide the docker credentials? (HINT)



TASK:

To try the above process.

Install the plugin:

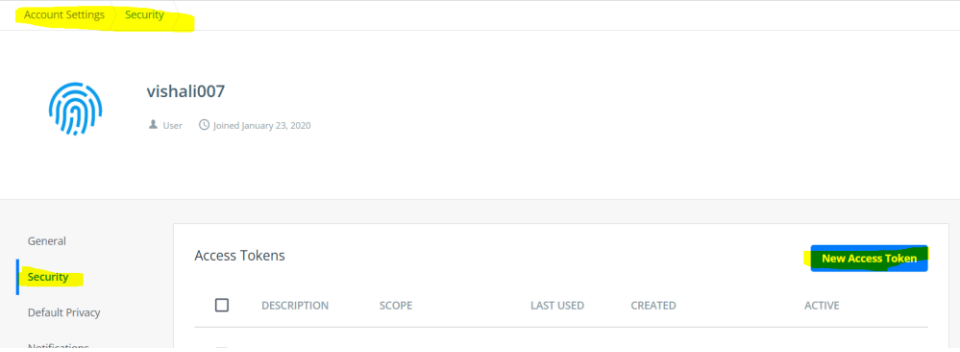


Group: usermod -aG docker jenkins

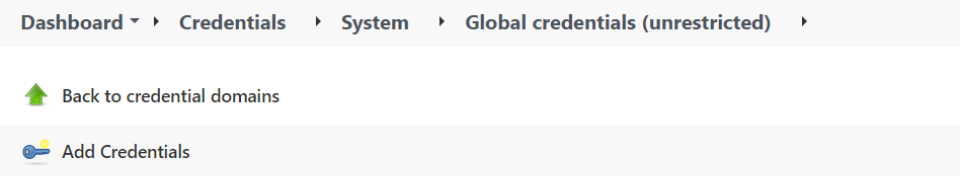
Docker image push:

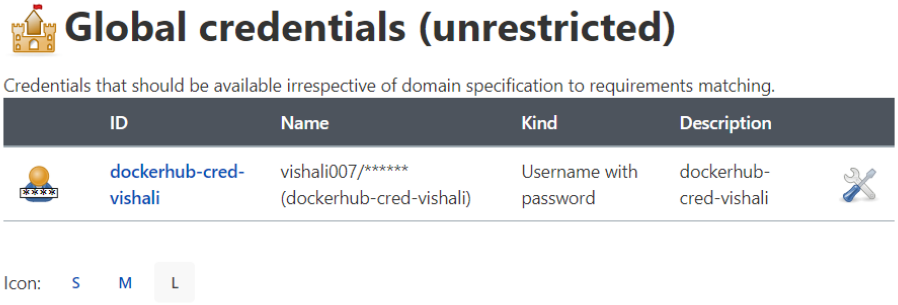
Link for reference: https://medium.com/codex/how-to-push-a-docker-image-to-docker-hub-using-jenkins-487fb1fcbe25#:~:text=First%2C%20we%20need%20to%20add,global)%20%2D%3E%20Add%20Credentials.&text=Alternatively%2C%20you%20can%20use%20the,And%20change%20the%20IP%20address.

Generate the token in DockerHub:



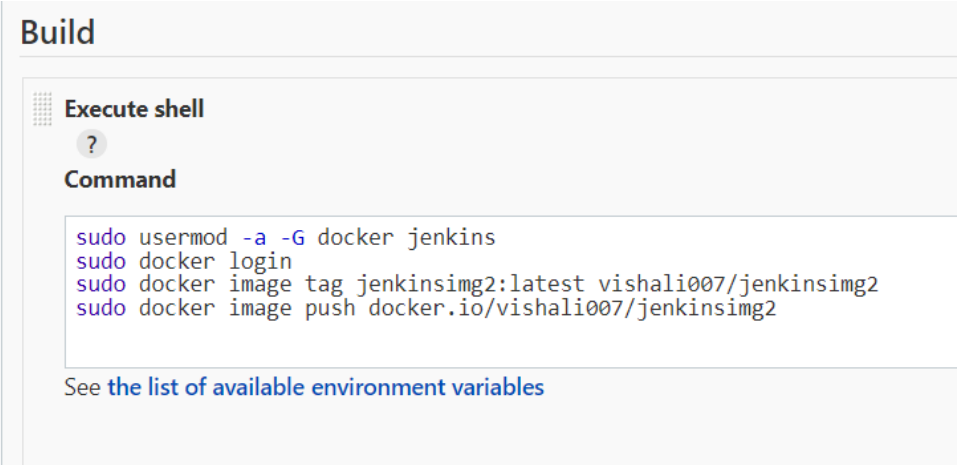
Manage Jenkins -> Manage Credentials -> Jenkins-> Global -> Add credentials -> Type docker hub username -> copy the token from the above step and paste it here - > provide id and description.





Trigger build:

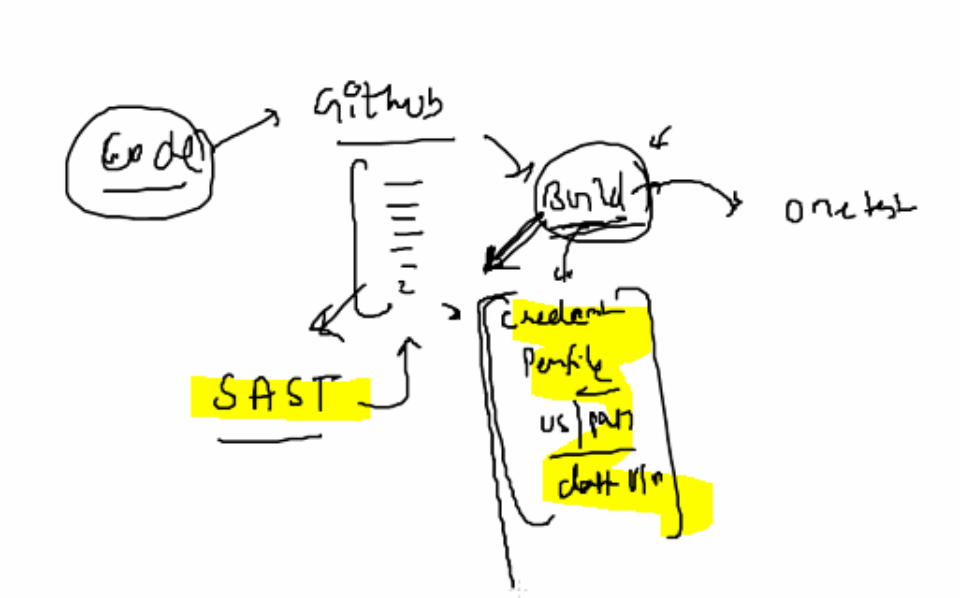
Build commands:



Code -> GitHub

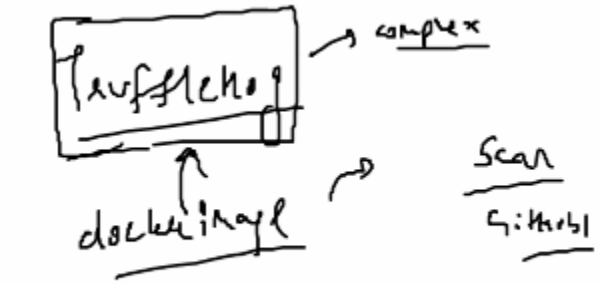
Before building the image and after building the image – Security test

Before build: Perform SAST: Static analysis security test

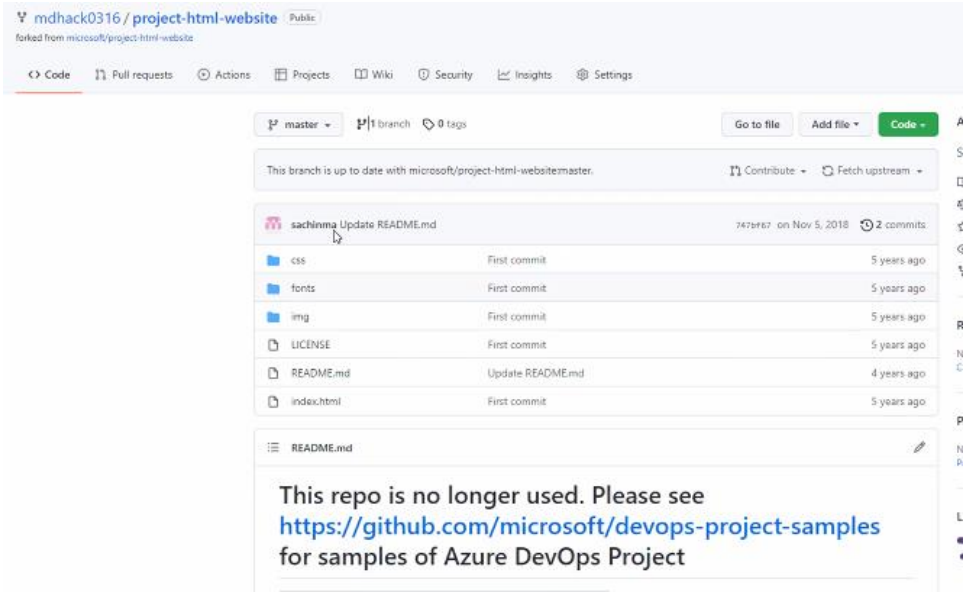


Tool which is used to do it is:

Installing is complex so use the docker image:



Before the build we will test the security, if the security is not good, then don't build the image.

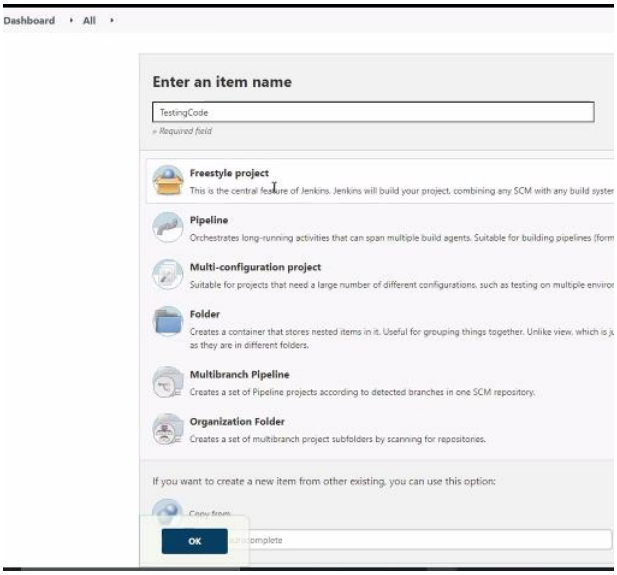


Do I have a problem with the code?

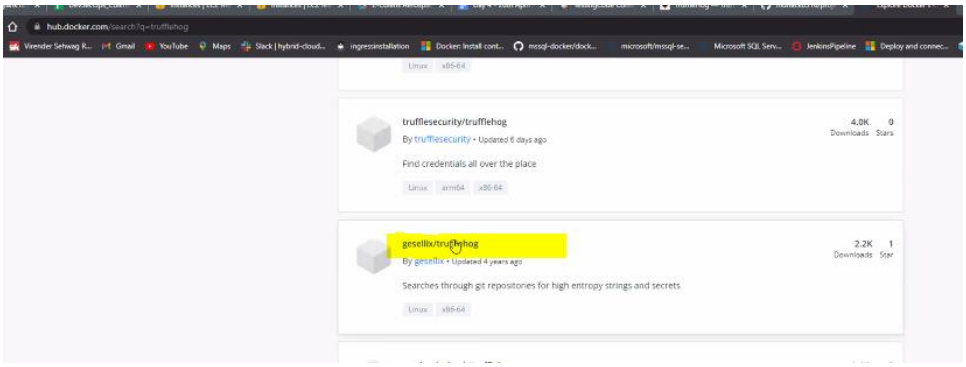
Tell the truffle log to check it.

This repo is fork from Microsoft.

Create an item:

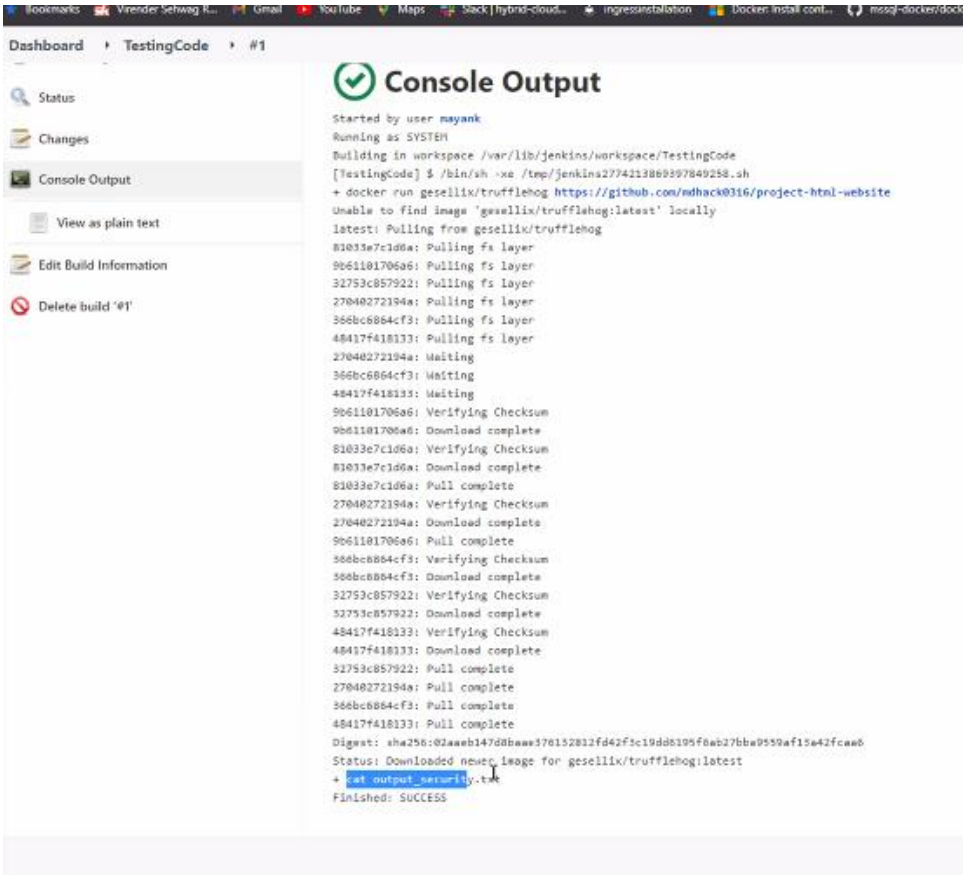


truffle image from docker hub:



Build command:

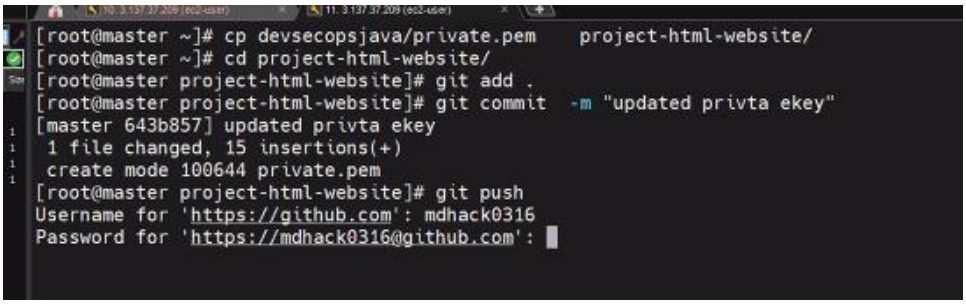
Build now:



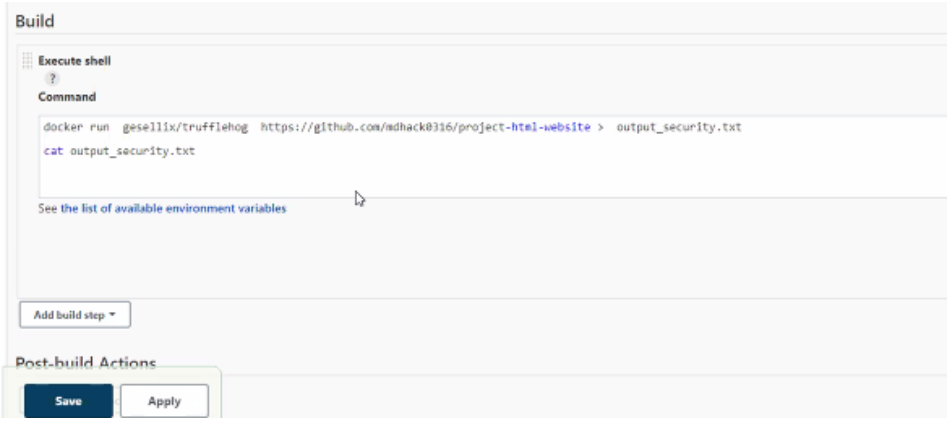
Nothing we got – means no problem is in this code.

Get a different URL which has issues and test it out!

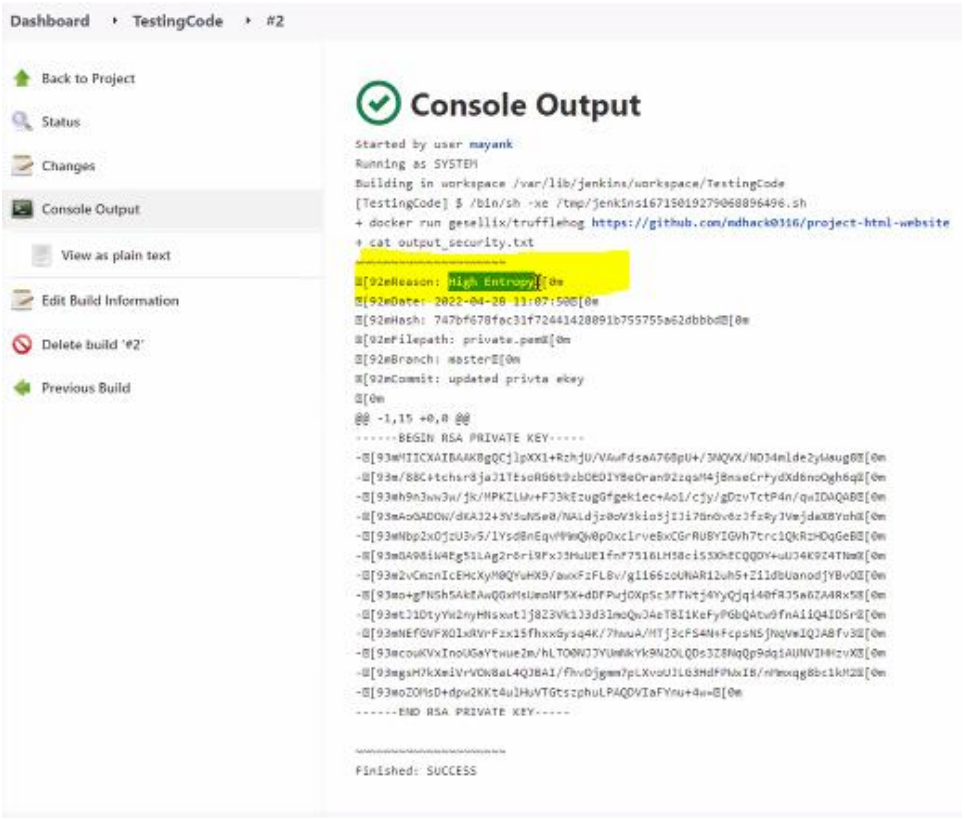
Change the private key:

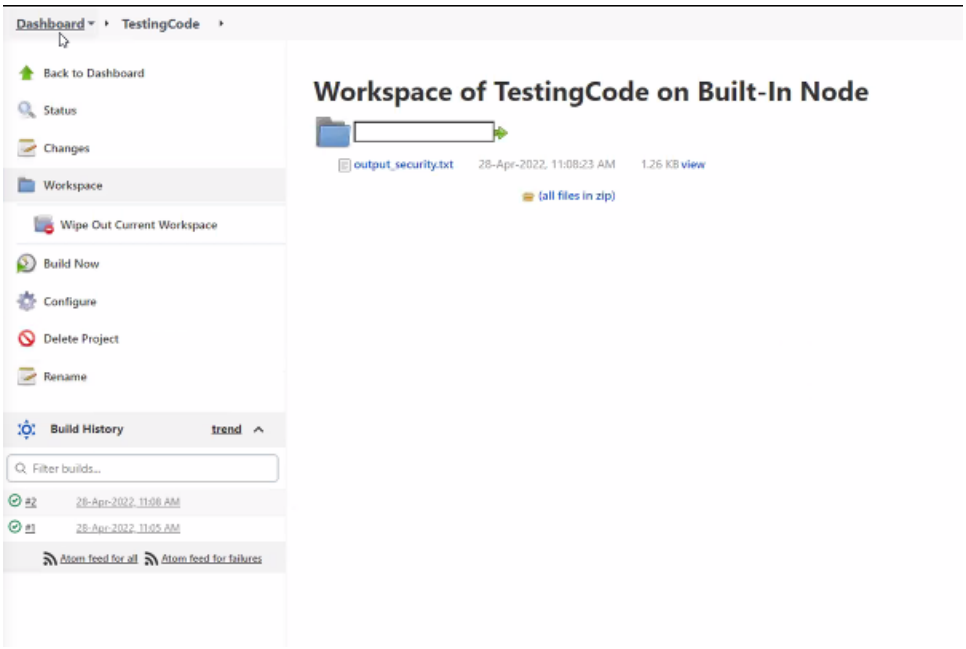


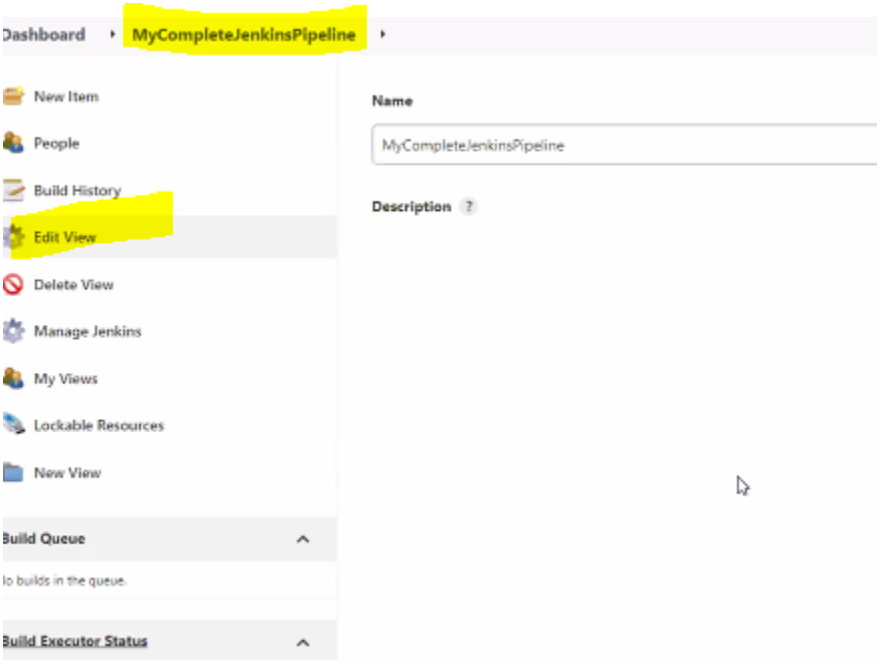
Test the same URL:

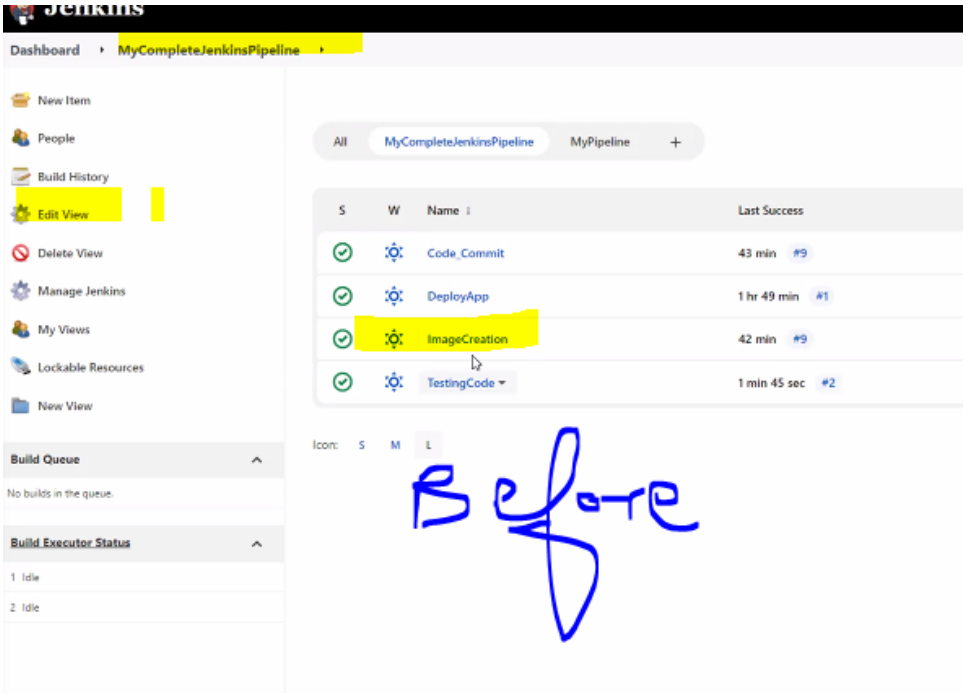


Now, you can see the issue:

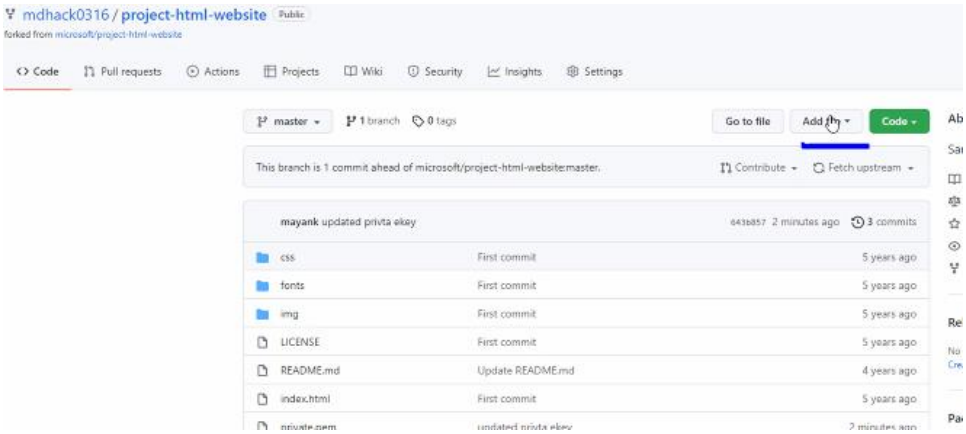


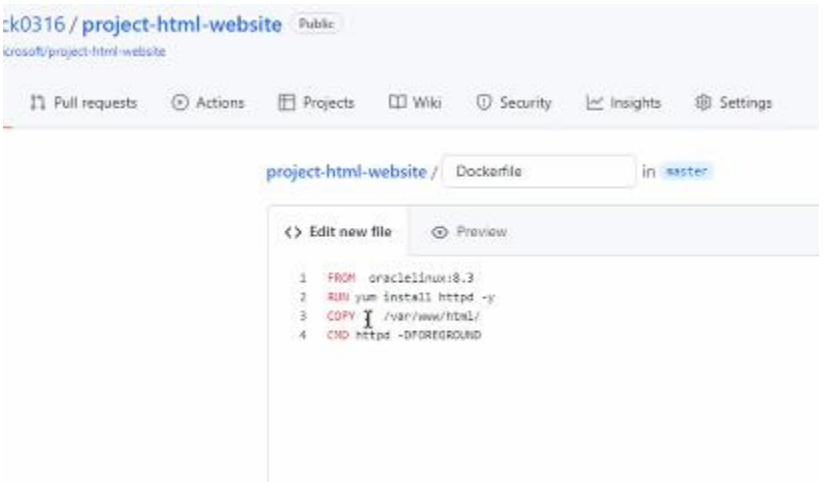

Dashboard
TestingCode
Back to Dashboard
Status
Changes
Workspace
Wipe Out Current Workspace
Build Now
Configure
Delete Project
Rename
Build History
trend
Filter builds...
#2 28-Apr-2022, 11:08 AM
#1 28-Apr-2022, 11:05 AM
Atom feed for all
Atom feed for failures
Workspace of TestingCode on Built-In Node
output_security.txt 28-Apr-2022, 11:08:23 AM 1.26 KB view
(all files in zip)


Dashboard
MyCompleteJenkinsPipeline
New Item
People
Build History
Edit View
Delete View
Manage Jenkins
My Views
Lockable Resources
New View
Build Queue
lo builds in the queue.
Build Executor Status
Name
MyCompleteJenkinsPipeline
Description

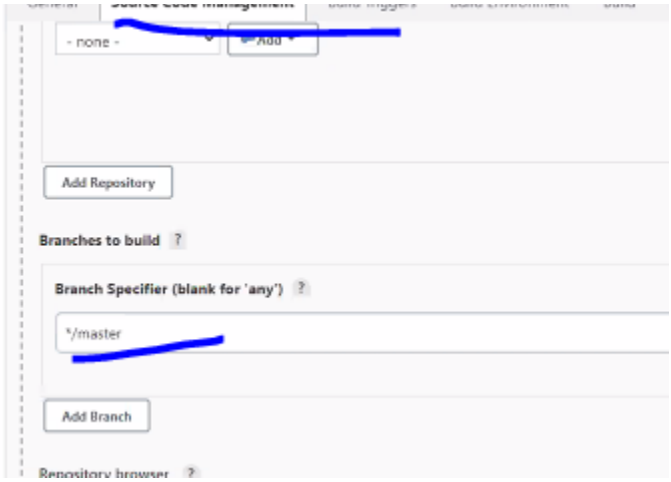


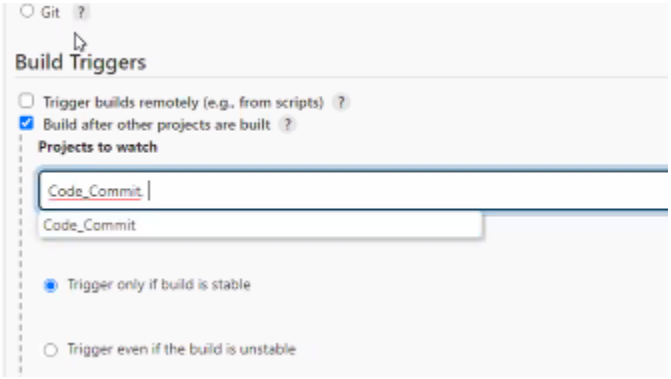
Add a docker file:

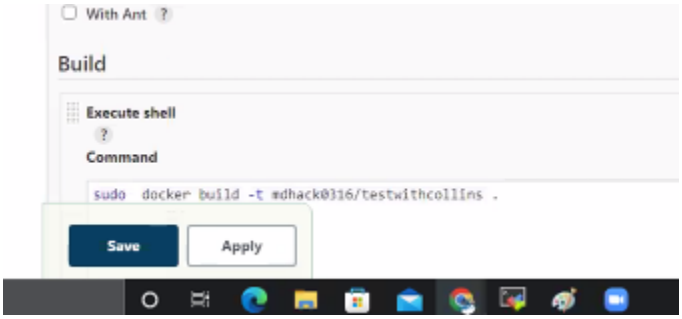




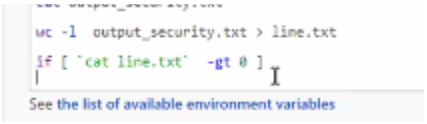
Test the code: (create the job)







Word count:



This build will fail – if there is a problem

```
  echo "Bug Found please check your code"
  exit 1
else
    echo "All Good"

    fi
```

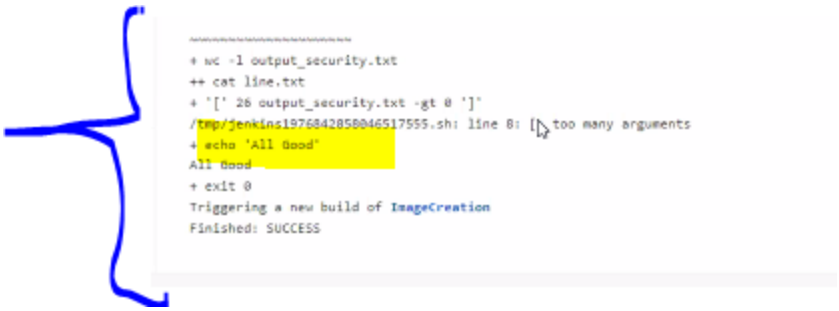
See the list of available environment variables

Add exit 0 in in between echo and fi.

```
[root@master project-html-website]# docker rm -f $(docker ps -q)
60b73d68494a
a2e470b1384b
1cbd21efcffe
3e1b44873f4f
1d8012cbec64
86640c042b54
956412053e91
b22f5faed2a2
6881ff805325
08cf546afec7
b62e0b438716
71015e7a5973
1d9f68fd1551
473f5f78884f
[root@master project-html-website]#
```

Deploy app:

Build now:



Check for the code: There should be some mistake here

New change:

Now, remove the key file and check now:

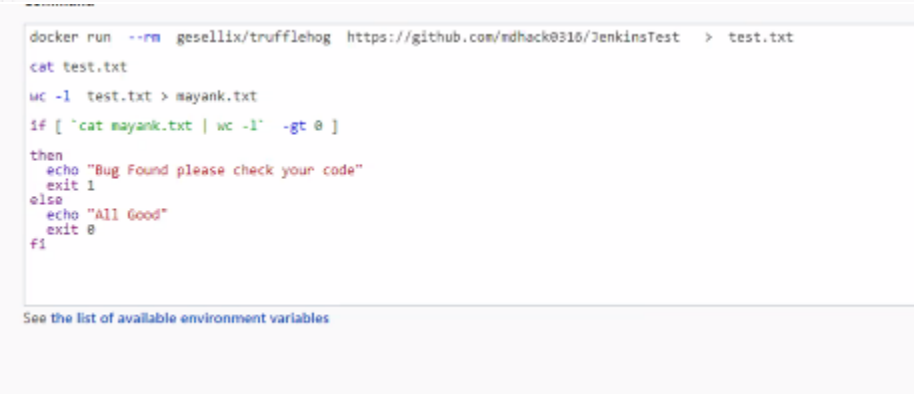
```
[root@master ~]# cd project-html-website/
[root@master project-html-website]# ls
css  fonts  img  index.html  LICENSE  private.pem  README.md
[root@master project-html-website]# rm -rf private.pem
[root@master project-html-website]# git
```

```
[root@master ~]# cd project-html-website/
[root@master project-html-website]# ls
css  fonts  img  index.html  LICENSE  private.pem  README.md
[root@master project-html-website]# rm -rf private.pem
[root@master project-html-website]# git add  .
[root@master project-html-website]# git commit  -m "updated privta ekey"
[master 7927249] updated privta ekey
 1 file changed, 15 deletions(-)
 delete mode 100644 private.pem
[root@master project-html-website]# git push
Username for 'https://github.com':
```

```
[root@master project-html-website]# git fetch https://github.com/mdhack0316/project-html-website
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 5 (delta 2), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (5/5), 1.28 KiB | 1.28 MiB/s, done.
From https://github.com/mdhack0316/project-html-website
 * branch            HEAD       -> FETCH_HEAD
[root@master project-html-website]#
```

```
[root@master project-html-website]# git fetch https://github.com/mdhack0316/project-html-website
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 5 (delta 2), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (5/5), 1.28 KiB | 1.28 MiB/s, done.
From https://github.com/mdhack0316/project-html-website
 * branch            HEAD       -> FETCH_HEAD
[root@master project-html-website]# git pull  https://github.com/mdhack0316/project-html-website
From https://github.com/mdhack0316/project-html-website
 * branch            HEAD       -> FETCH_HEAD
hint: Pulling without specifying how to reconcile divergent branches is
hint: discouraged. You can squelch this message by running one of the following
hint: commands sometime before your next pull:
hint:
hint:   git config pull.rebase false  # merge (the default strategy)
hint:   git config pull.rebase true   # rebase
hint:   git config pull.ff only       # fast-forward only
hint:
hint: You can replace "git config" with "git config --global" to set a default
hint: preference for all repositories. You can also pass --rebase, --no-rebase,
hint: or --ff-only on the command line to override the configured default per
hint: invocation.
Merge made by the 'recursive' strategy.
 Dockerfile | 4 ++++
 1 file changed, 4 insertions(+)
 create mode 100644 Dockerfile
[root@master project-html-website]# ls
css  Dockerfile  fonts  img  index.html  LICENSE  README.md
[root@master project-html-website]#
```

```
[root@master project-html-website]# git commit  -m "updated privta ekey"
On branch master
Your branch is ahead of 'origin/master' by 4 commits.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
[root@master project-html-website]# git push
Username for 'https://github.com': mdhack0316
Password for 'https://mdhack0316@github.com':
remote: Support for password authentication was removed on August 13, 2021. Please use a personal access token instead.
remote: Please see https://github.blog/2020-12-15-token-authentication-requirements-for-git-operations/ for more information.
fatal: Authentication failed for 'https://github.com/mdhack0316/project-html-website/'
```



docker run --rm gesellix/trufflehog https://github.com/mdhack0316/JenkinsTest > test.txt

cat test.txt

if [`cat test.txt | wc -l` -gt 0]

then

echo "Bug Found please check your code"

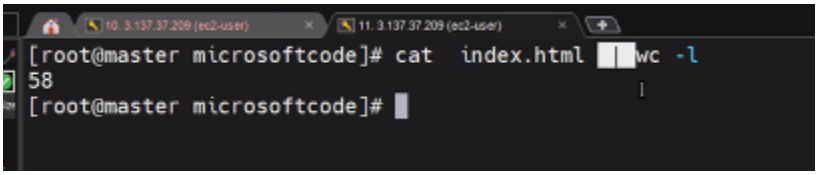
exit 1

else

echo "All Good"

exit 0

fi



TASK:

Try the above process:

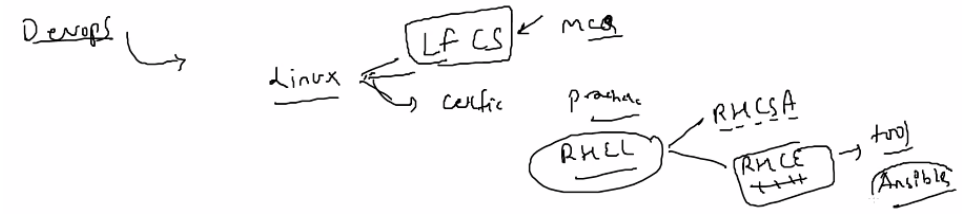
Execute shell

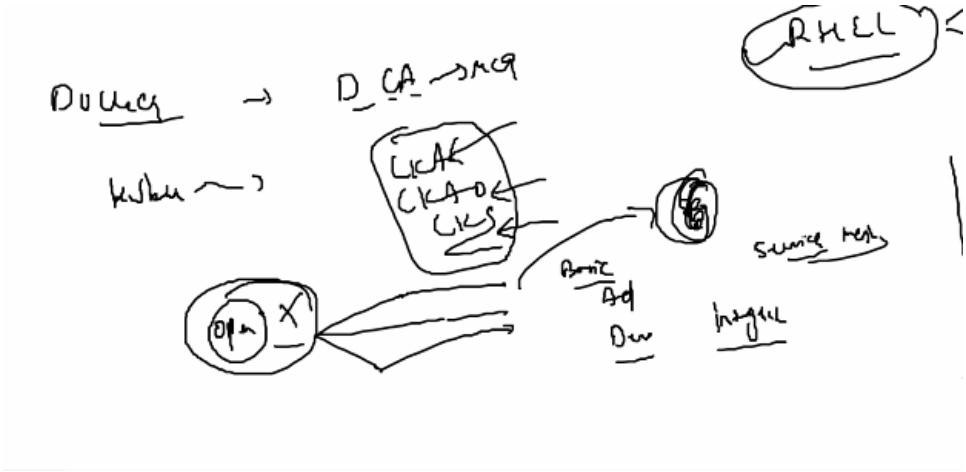
Command

```
sudo docker run  --rm  gesellix/trufflehog  https://github.com/Coder-Vishali/JenkinsTest   >  test.txt
cat test.txt
if [ `cat test.txt | wc -l`  -gt 0 ]
then
  echo "Bug Found please check your code"
  exit 1
else
  echo "All Good"
  exit 0
fi
```

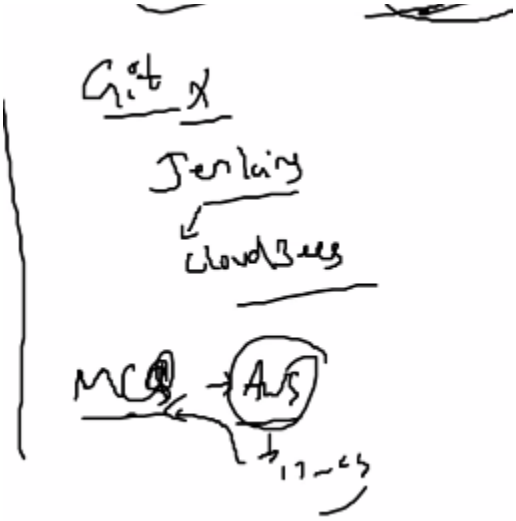
Topic for tomorrow: For secured pipeline:

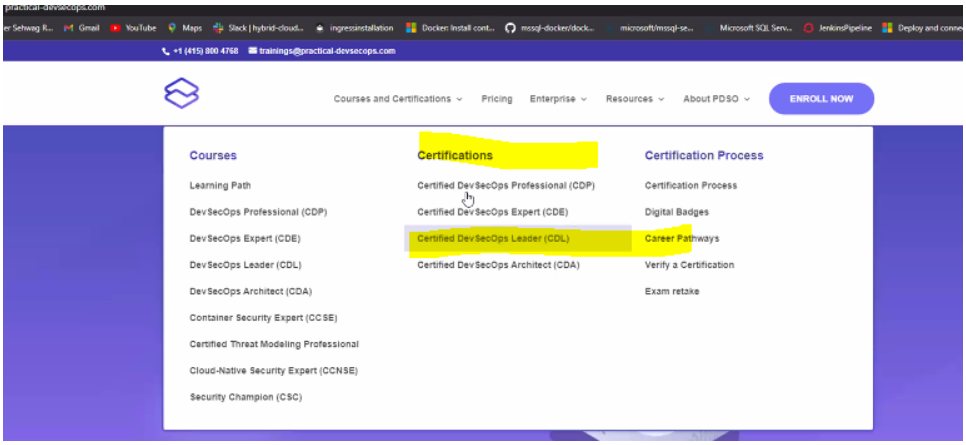
Certification:

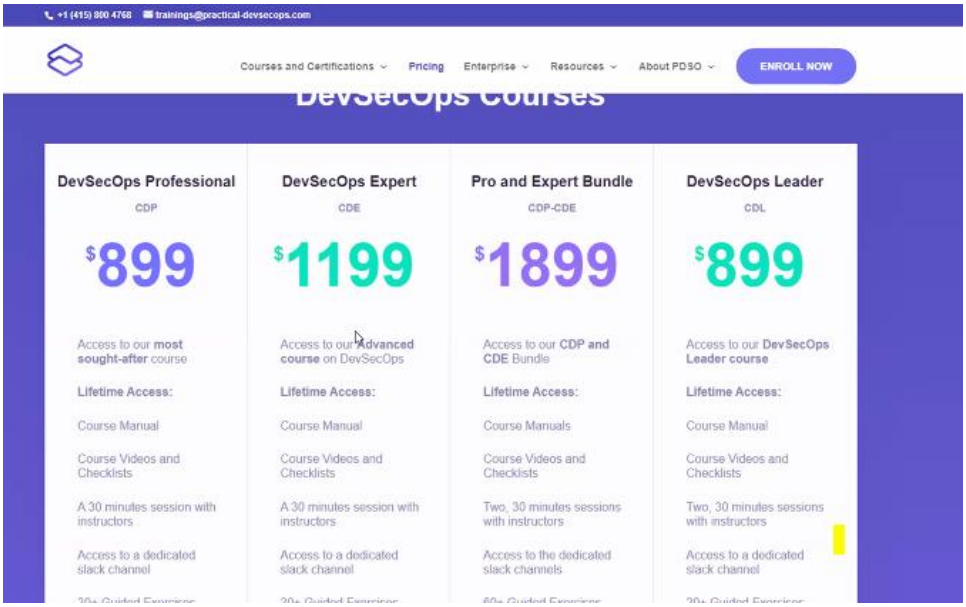


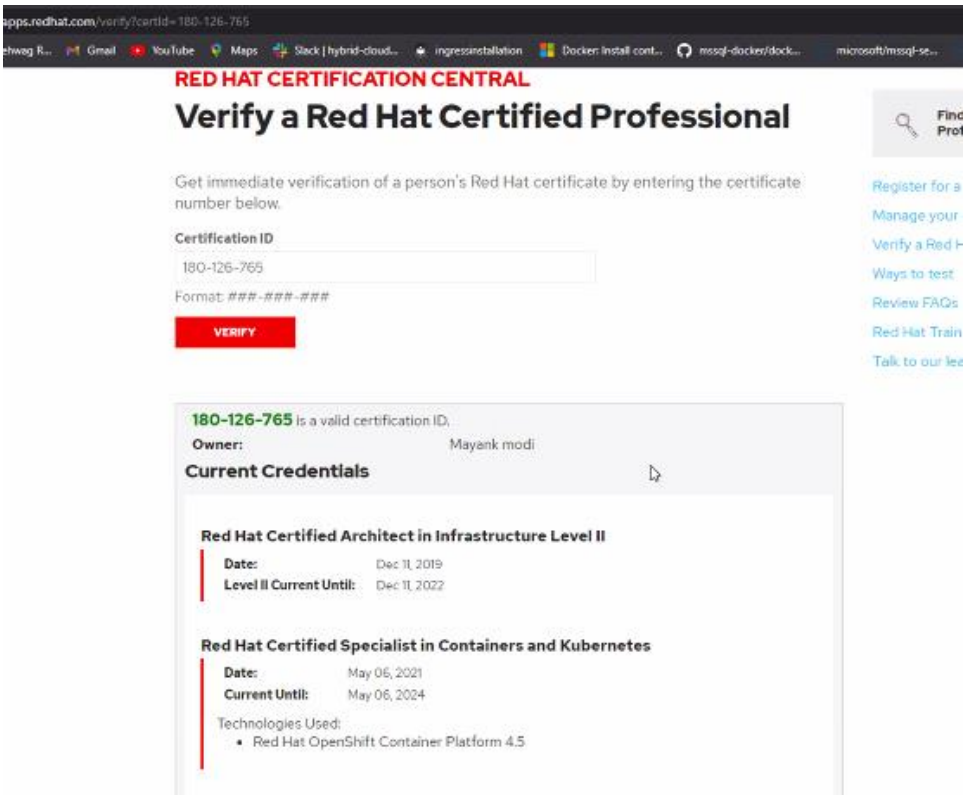


Administrator, operator and security




+1 (415) 800 4768
trainings@practical-devsecops.com
Courses and Certifications
Pricing
Enterprise
Resources
About PDSO
ENROLL NOW
Courses
Learning Path
DevSecOps Professional (CDP)
DevSecOps Expert (CDE)
DevSecOps Leader (CDL)
DevSecOps Architect (CDA)
Container Security Expert (CCSE)
Certified Threat Modeling Professional
Cloud-Native Security Expert (CCNSE)
Security Champion (CSC)
Certifications
Certified DevSecOps Professional (CDP)
Certified DevSecOps Expert (CDE)
Certified DevSecOps Leader (CDL)
Certified DevSecOps Architect (CDA)
Certification Process
Certification Process
Digital Badges
Career Pathways
Verify a Certification
Exam retake


+1 (415) 800 4768
trainings@practical-devsecops.com
Courses and Certifications
Pricing
Enterprise
Resources
About PDSO
ENROLL NOW
DevSecOps Professional
CDP
$899
Access to our most sought-after course
Lifetime Access:
Course Manual
Course Videos and Checklists
A 30 minutes session with instructors
Access to a dedicated slack channel
DevSecOps Expert
CDE
$1199
Access to our advanced course on DevSecOps
Lifetime Access:
Course Manual
Course Videos and Checklists
A 30 minutes session with instructors
Access to a dedicated slack channel
Pro and Expert Bundle
CDP-CDE
$1899
Access to our CDP and CDE Bundle
Lifetime Access:
Course Manuals
Course Videos and Checklists
Two, 30 minutes sessions with instructors
Access to the dedicated slack channels
DevSecOps Leader
CDL
$899
Access to our DevSecOps Leader course
Lifetime Access:
Course Manual
Course Videos and Checklists
Two, 30 minutes sessions with instructors
Access to a dedicated slack channel



Contents:

Containers

Introduction to Container Technology

- Application Management Landscape
- History of Containers
- Containers vs Virtual Machines
- Application Isolation
  - Control Groups
  - Namespaces
  - SELinux
- Resource Measurement and Control
- Container Security
- Container Runtime Alternatives - Docker, CRI-O, RunC
- Podman Architecture
- Installing and Validating Podman

Managing Containers [Podman, Docker and CRI-O]

- Creating a New Container
- Listing Containers
- Getting Information about Containers
- Managing Container Resources
- Running Commands in an Existing Container
- Interacting with a Running Container
- Stopping, Starting, and Removing Containers
- Copying Files in/out of Containers
- Inspecting and Updating Containers
- Running Containers as Services
- Implementing Restart Policies
- Understanding User-Namespaces
- Understanding Rootless Containers Concepts

Understand Container Registry and Managing Container Images

- Introduction to Container Registries
- Listing and Removing Images

- Searching and Downloading for Images
- Image Tagging
- Uploading Images
- Export/Import Images
- Save/Load Images
- Committing Changes
- Working with Skopeo
- Deploying Private Registry

Creating Container Images

- Creating Images with ContainerFile
- Understanding Caching and Image Layers
- Podman Image Build
- Multi-Stage Builds
- Building Images with Buildah

Container Storage

- Understanding Core Concepts of Container Storage - AUFS , OverlayFS
- Internal Volume Drivers
- Podman Volume Command
- Bind Mounting Volumes
- Removing Volumes
- Creating and Using External Volumes with NFS

Container Networking

- Understanding Container to Container Communication
- Networking Drivers
    - Host
    - Bridge
    - Null
- Understanding Pods
- Creating Pods

- Removing Volumes
- Creating and Using External Volumes with NFS

Container Networking

- Understanding Container to Container Communication
- Networking Drivers
    - Host
    - Bridge
    - Null
- Understanding Pods
- Creating Pods

- Attaching Containers to Pods
- Attaching a Container to a Network

Multi-Container Applications

- Understanding the need for Multi-Container Applications
- Introduction to Podman Compose
- Creating a Compose File
- Building a Multi-Container Environment with Compose

Container Security and Observability

- Image Scanning
    - Docker Scan
    - Clair - Quay.io
- Docker Bench
- SECCOMP Profile
- Capabilities
- Privileged Containers
- Portainer

Kubernetes Cluster Architecture

- Master & Nodes (aka...Minions)
- Kube API Server
- Etcd
- Kube Scheduler
- Kube Controller-Manager
- Kubelet
- Kube-Proxy
- Kubectl Client

Installation, Configuration & Validation

- Deploying a Docker Swarm Cluster
- Deploying a Single Node Cluster - Minikube
- Deploying a Multi-Node Cluster - Kubeadm

Introduction to OpenShift

- OpenShift Architecture
- Features
  - API
  - Authentication
  - Networking
  - S2I

OpenShift Installation

- Deploying a Single Node Cluster - CRC
- Deploying OpenShift Cluster using SNI
- Introduction to OpenShift CLI - oc
- Introduction to Projects
- Exploring to OpenShift Web Console

Kubernetes Namespaces

- What are Namespaces?
- Creating Namespaces
- Work with Namespaces