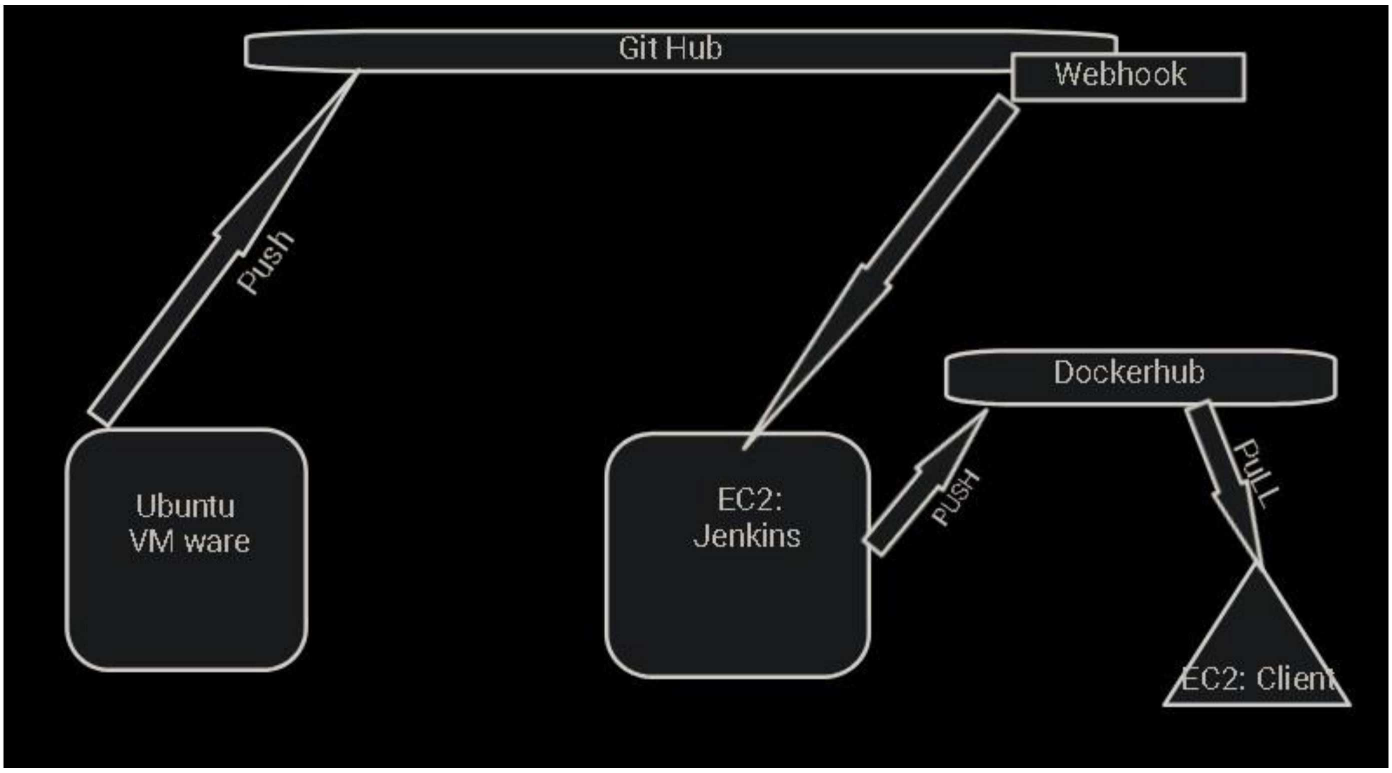
Objective : to upload index.html from local repo and output should be visible on apache web server .in container

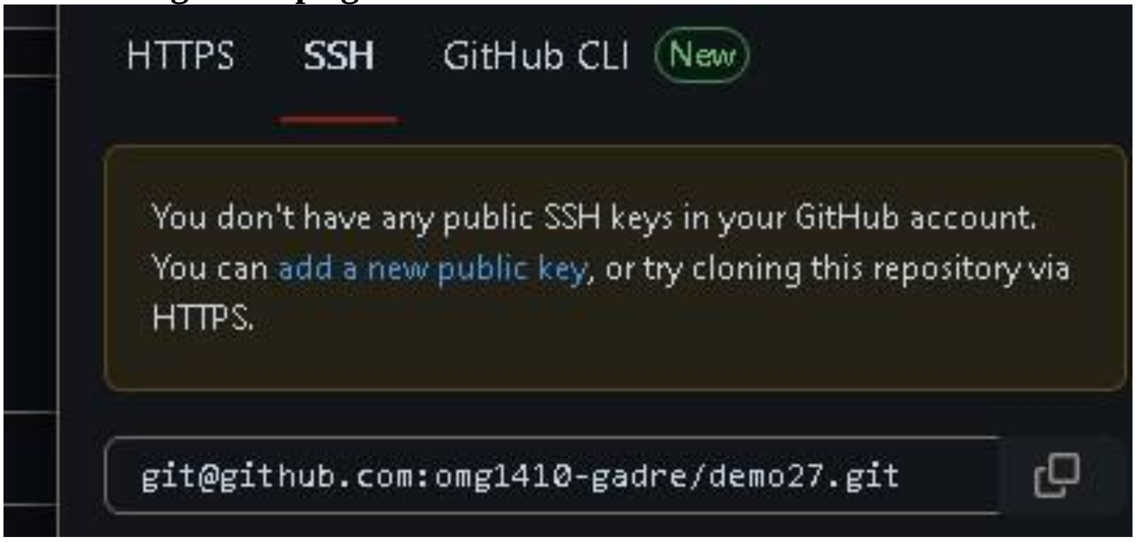
Pipeline :



Local repo: Also create a repo in Github account

Ubuntu:

```
$sudo apt-get install git
$mkdir ____
$cd ____
$git init
$git remote add origin "https/git"
```

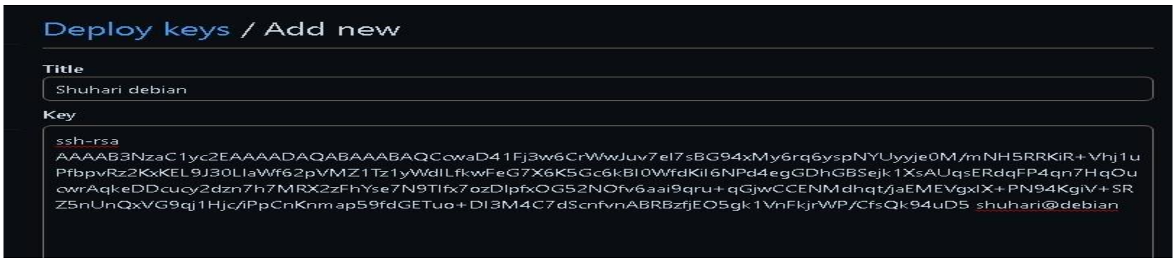


You don't have any public SSH keys in your GitHub account. You can add a new public key, or try cloning this repository via HTTPS.

```
git@github.com:omg1410-gadre/demo27.git
```

```
$ssh-keygen
```

Add public key into github settin g-> deploy keys)

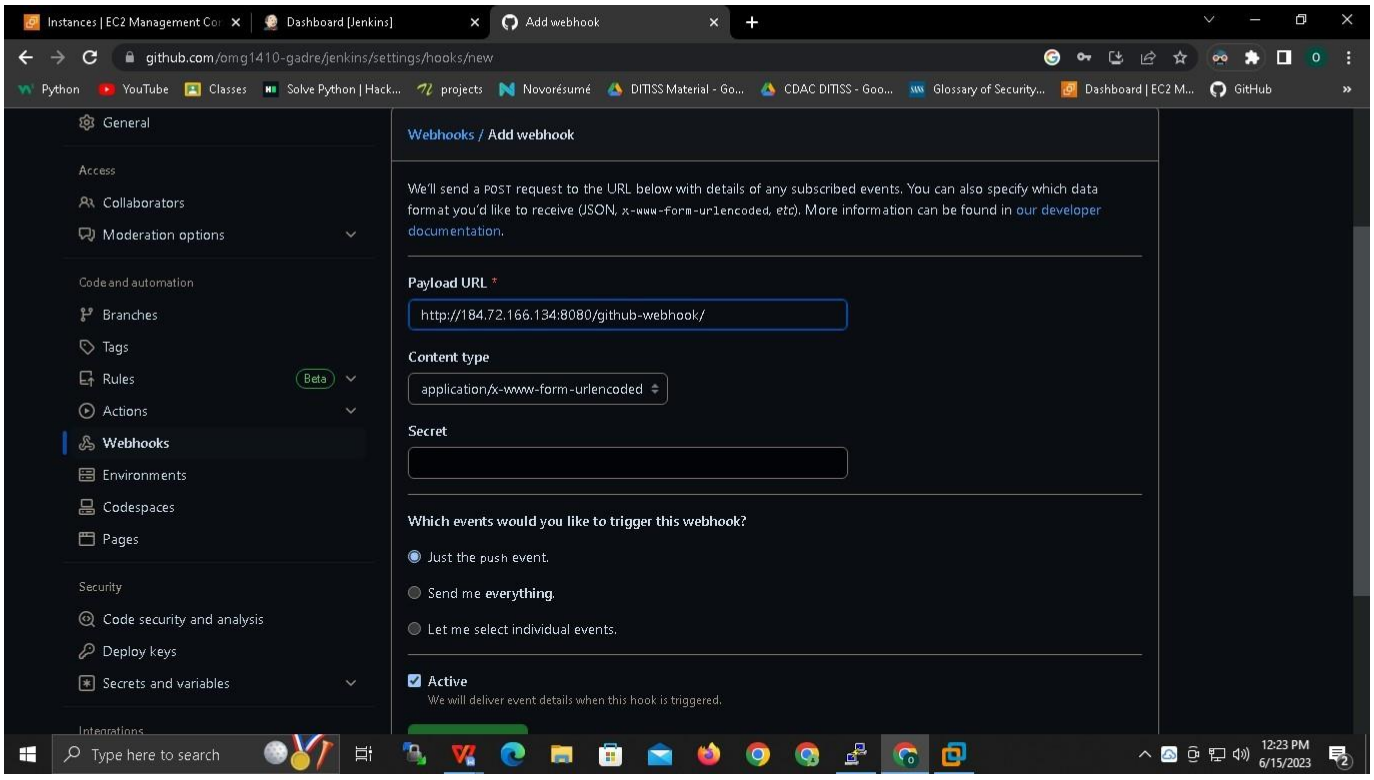

Deploy keys / Add new

Title: Shuhari debian

Key:
```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCcwaD41Fj3w6CrWwJuv7el7sBG94xMy6rq6yspNYUyyje0M/mNH5RRKiR+Vhj1u
PfbpvRz2KxKEL9J30LIaWf62pVMZ1Tz1yWdlLfkwFeG7X6K5Gc6kBI0WfdKiI6NPd4egGDhGBSejk1XsAUqsERdqFP4qn7HqOu
cwrAqkeDDcucy2dzn7h7MRX2zFhYse7N9TIfx7ozDlpfxOG52NOfv6aai9qru+qGjwCCENMdhqt/jaEMEVgxlX+PN94KgiV+SR
Z5nUnQxVG9qj1Hjc/iPpCnKnmap59fdGETuo+DI3M4C7dScnfvnABRBzfjEO5gk1VnFkjrWP/CfsQk94uD5 shuhari@debian
```

$git pull origin master
$git --config global user.name "…"
$git --config global user.email "…."
$git add Dockerfile
$git add index.html
$git commit -m "msg"
$git push origin master

Allow webhook
Setting :
Payload url : (……..url of jenkins…..)/github-webhook/
Content type : json
which event (just push)



Webhooks / Add webhook

We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, x-www-form-urlencoded, etc). More information can be found in our developer documentation.

Payload URL *: http://184.72.166.134:8080/github-webhook/

Content type: application/x-www-form-urlencoded

Secret:

Which events would you like to trigger this webhook?

- [✓ Just the push event.]
- Send me everything.
- Let me select individual events.

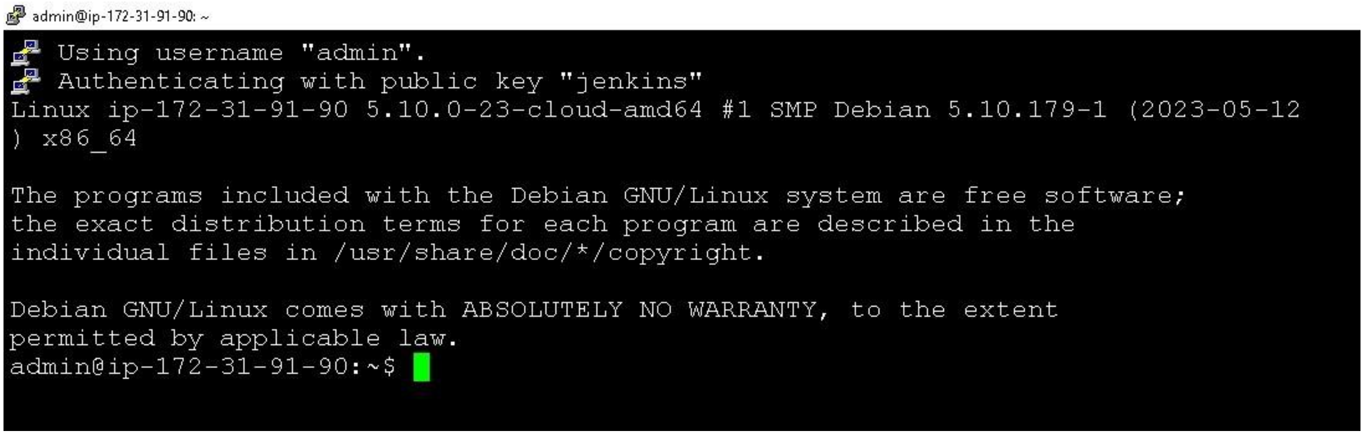
☑ Active
We will deliver event details when this hook is triggered.

2nd Machine :

Jenkins

Create EC2 instance , allow http and https traffic as well .

Login in putty :



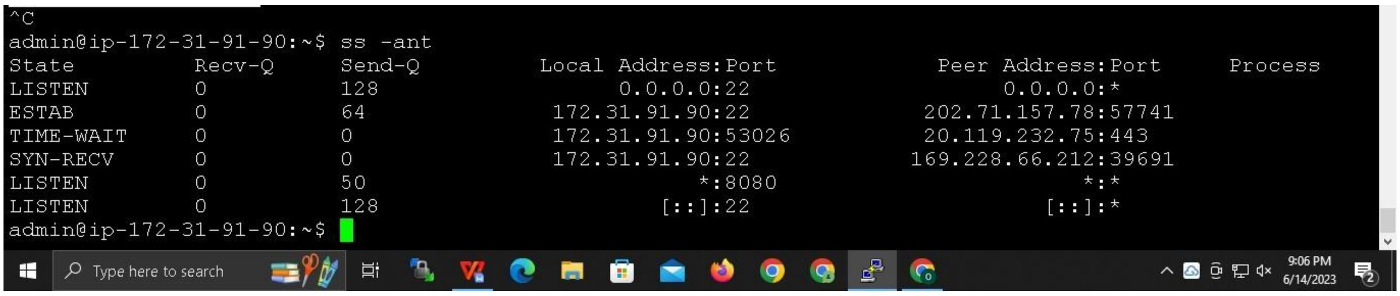
admin@ip-172-31-91-90: ~

```
Using username "admin".
Authenticating with public key "jenkins"
Linux ip-172-31-91-90 5.10.0-23-cloud-amd64 #1 SMP Debian 5.10.179-1 (2023-05-12
) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
admin@ip-172-31-91-90:~$
```

$ apt-get update -y
$ apt-get upgrade -y
$ apt-get install gnupg -y *(it's a basic and simple tool for encrypting files)*
$ apt-get install default-jre -y (*used for compilig and launching java programm*)
$ curl -fsSL https://pkg.jenkins.io/debian/jenkins.io-2023.key | sudo tee /usr/share/keyrings/jenkins-keyring.asc > /dev/null *(generating key with curl command)*
**$ echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] **
**https://pkg.jenkins.io/debian binary/ | sudo tee **
/etc/apt/sources.list.d/jenkins.list > /dev/null *(updating repo file)*
$ apt update -y
$ apt-get install jenkins -y
$ sudo usermod -a -G root jenkins *(adding jenkins users in root group I.e secondry)*
$ systemctl status jenkins
$sudo apt-get install git
$ apt-get install docker.io
$sudo chown nobody:nogroup docker
$sudo chown
$ sudo usermod -a -G docker jenkins (add docker user into group jenkins)
$ sudo systemctl restart Jenkins
$ sudo chown nobody:nogroup /var/run/docker.sock
$sudo chown nobody:nogroup /var/lib/docker
$sudo chmod 777 -R /var/run/docker.sock
$sudo chmod 777 -R /var/lib/docker
$sudo passwd jenkins
$ su jenkins
$ssh-keygen

Copy public key and copy it to client

Port of jenkins is 8080

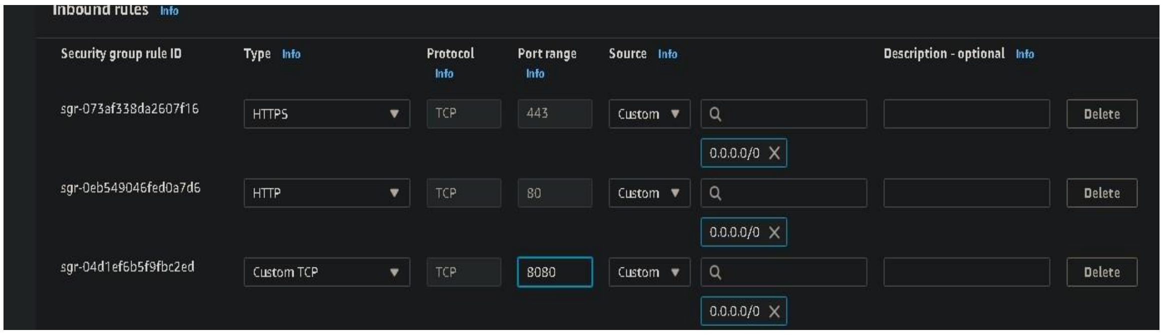


```
^C
admin@ip-172-31-91-90:~$ ss -ant
State         Recv-Q     Send-Q          Local Address:Port          Peer Address:Port     Process
LISTEN        0          128                   0.0.0.0:22                 0.0.0.0:*
ESTAB         0          64                172.31.91.90:22           202.71.157.78:57741
TIME-WAIT     0          0                 172.31.91.90:53026         20.119.232.75:443
SYN-RECV      0          0                 172.31.91.90:22          169.228.66.212:39691
LISTEN        0          50                          *:8080                     *:*
LISTEN        0          128                      [::]:22                    [::]:*
admin@ip-172-31-91-90:~$
```

Type here to search

9:06 PM
6/14/2023

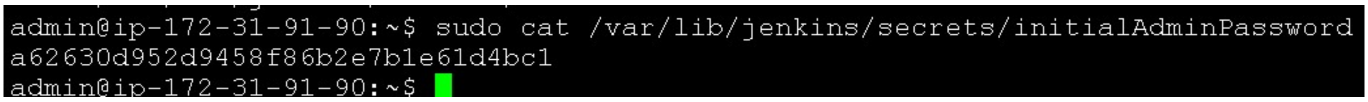
Go to aws -> Security -> edit inbound rules -> allow all port (ALL TCP)



Inbound rules Info

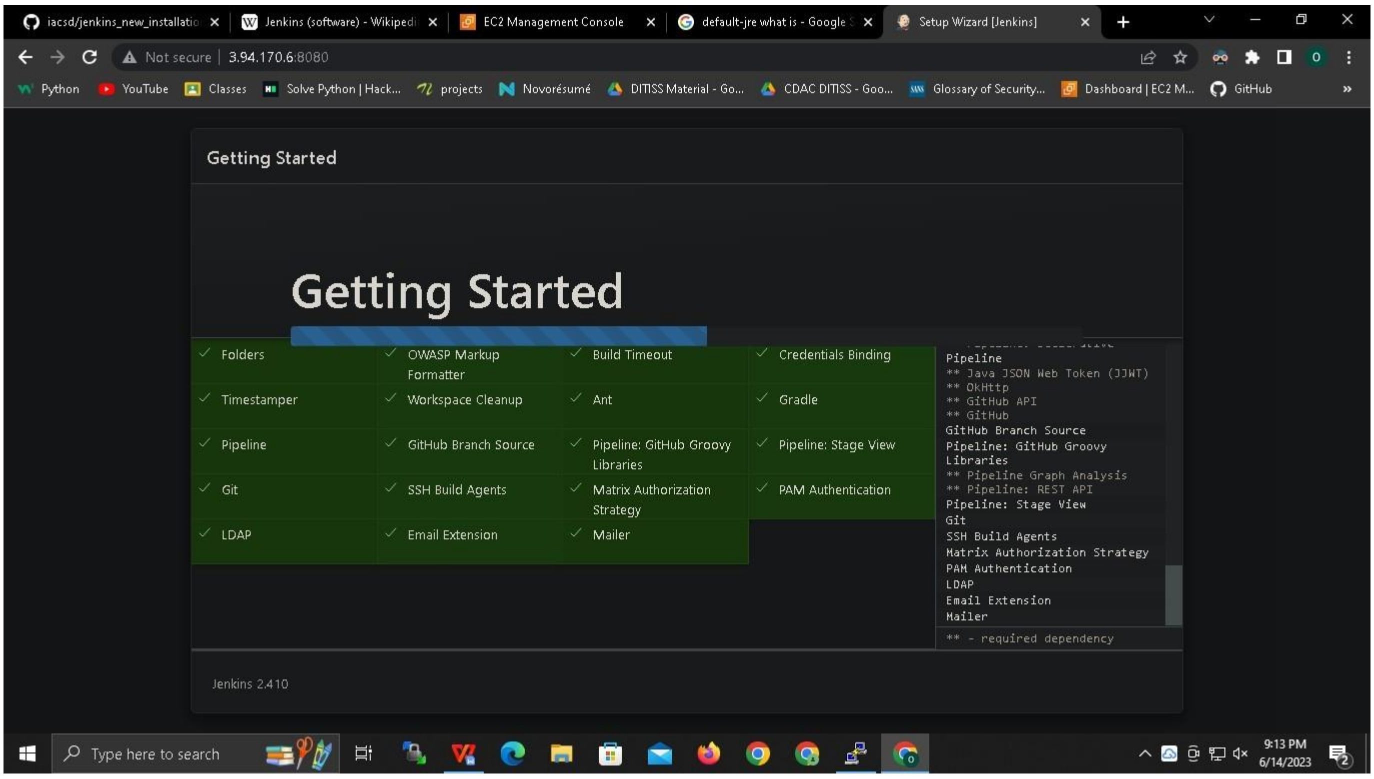
Security group rule ID	Type	Protocol	Port range	Source		Description - optional	
sgr-073af338da2607f16	HTTPS	TCP	443	Custom	0.0.0.0/0		Delete
sgr-0eb549046fed0a7d6	HTTP	TCP	80	Custom	0.0.0.0/0		Delete
sgr-04d1ef6b5f9fbc2ed	Custom TCP	TCP	8080	Custom	0.0.0.0/0		Delete

Go to browser -> public ip : 8080



```
admin@ip-172-31-91-90:~$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
a62630d952d9458f86b2e7b1e61d4bc1
admin@ip-172-31-91-90:~$
```

Install plugings



Getting Started

Getting Started

✓ Folders	✓ OWASP Markup Formatter	✓ Build Timeout	✓ Credentials Binding
✓ Timestamper	✓ Workspace Cleanup	✓ Ant	✓ Gradle
✓ Pipeline	✓ GitHub Branch Source	✓ Pipeline: GitHub Groovy Libraries	✓ Pipeline: Stage View
✓ Git	✓ SSH Build Agents	✓ Matrix Authorization Strategy	✓ PAM Authentication
✓ LDAP	✓ Email Extension	✓ Mailer	

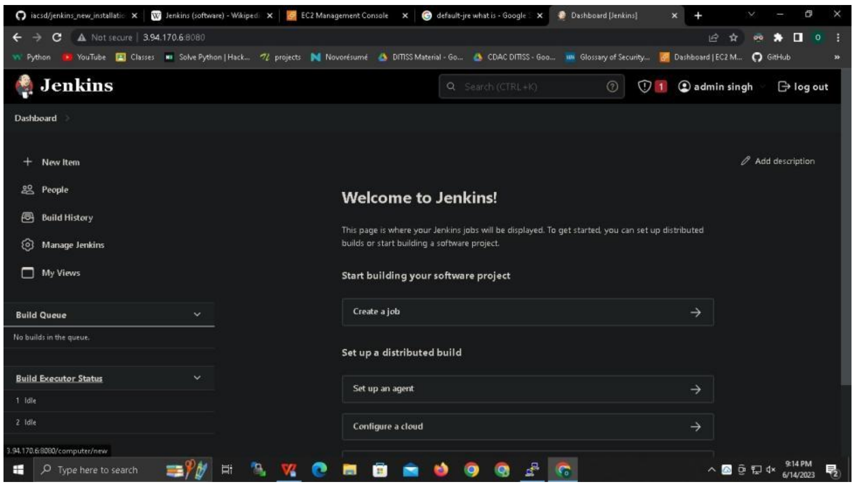
```
Pipeline
** Java JSON Web Token (JJWT)
** OkHttp
** GitHub API
** GitHub
GitHub Branch Source
Pipeline: GitHub Groovy Libraries
** Pipeline Graph Analysis
** Pipeline: REST API
Pipeline: Stage View
Git
SSH Build Agents
Matrix Authorization Strategy
PAM Authentication
LDAP
Email Extension
Mailer
** - required dependency
```

Jenkins 2.410

Type here to search

9:13 PM
6/14/2023

New Item:



Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

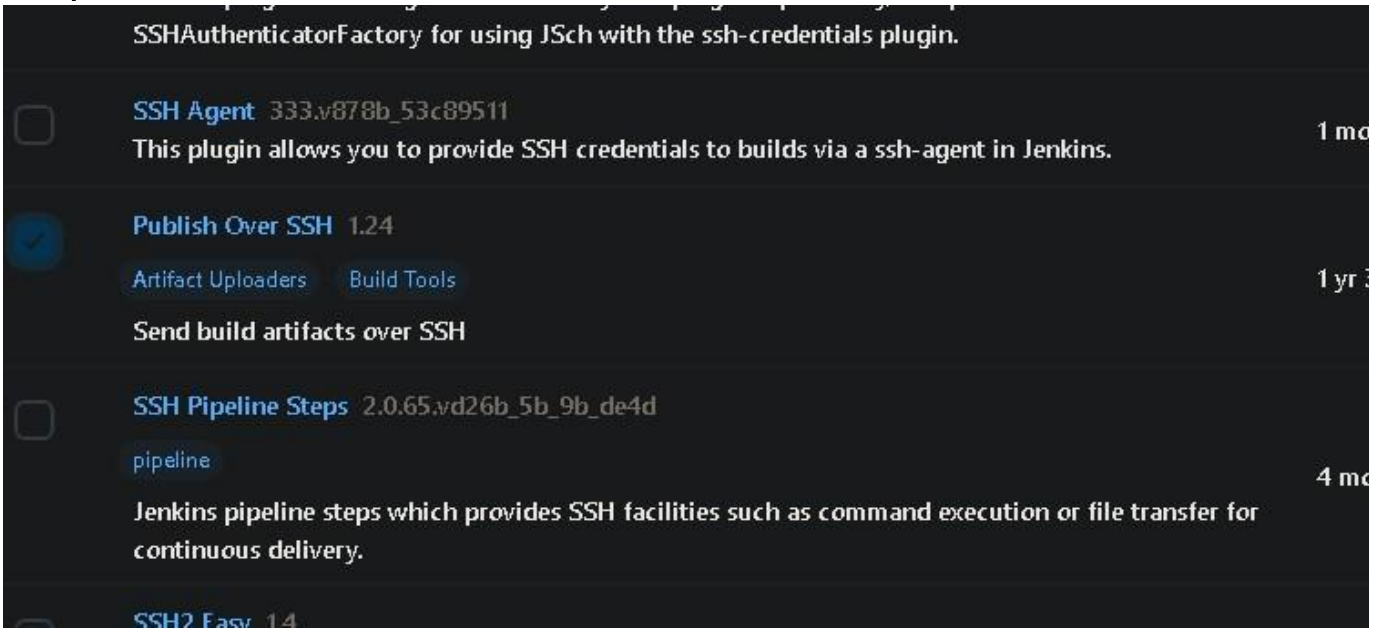
Start building your software project

Set up a distributed build

No builds in the queue.

Go to Jenkins dashboard -> manage Jenkins -> plugins
3 plugin

a) **Publish over ssh**



SSHAuthenticatorFactory for using JSch with the ssh-credentials plugin.

SSH Agent 333.v878b_53c89511
This plugin allows you to provide SSH credentials to builds via a ssh-agent in Jenkins.

Publish Over SSH 1.24
Artifact Uploaders Build Tools
Send build artifacts over SSH

SSH Pipeline Steps 2.0.65.vd26b_5b_9b_de4d
pipeline
Jenkins pipeline steps which provides SSH facilities such as command execution or file transfer for continuous delivery.

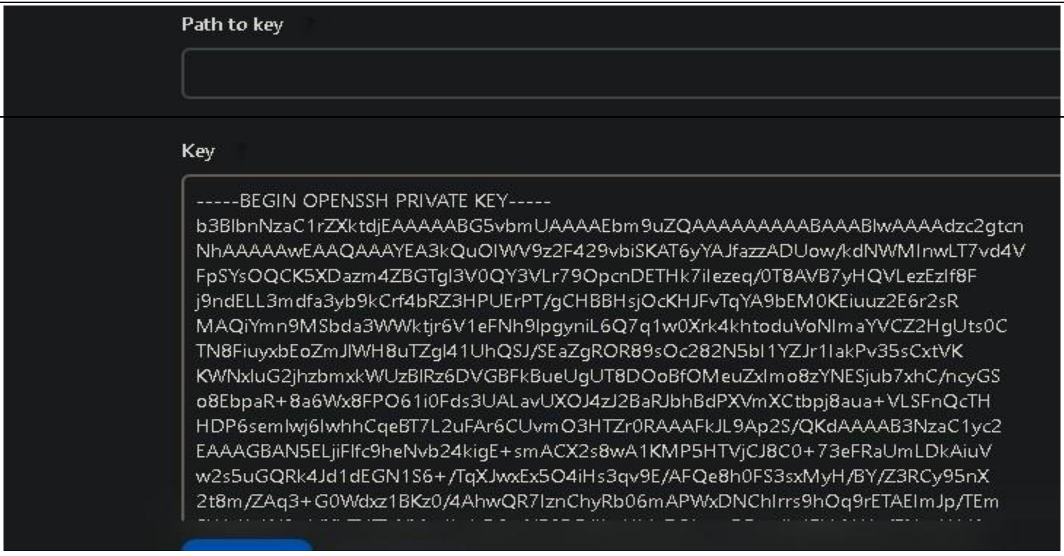
SSH2 Easy 1.4

b) **Docker only**

c) **Post build (do check if bash execution is presnt)**

Restart ..

Configure plugins :
Publish over ssh : add pvt key of Jenkins

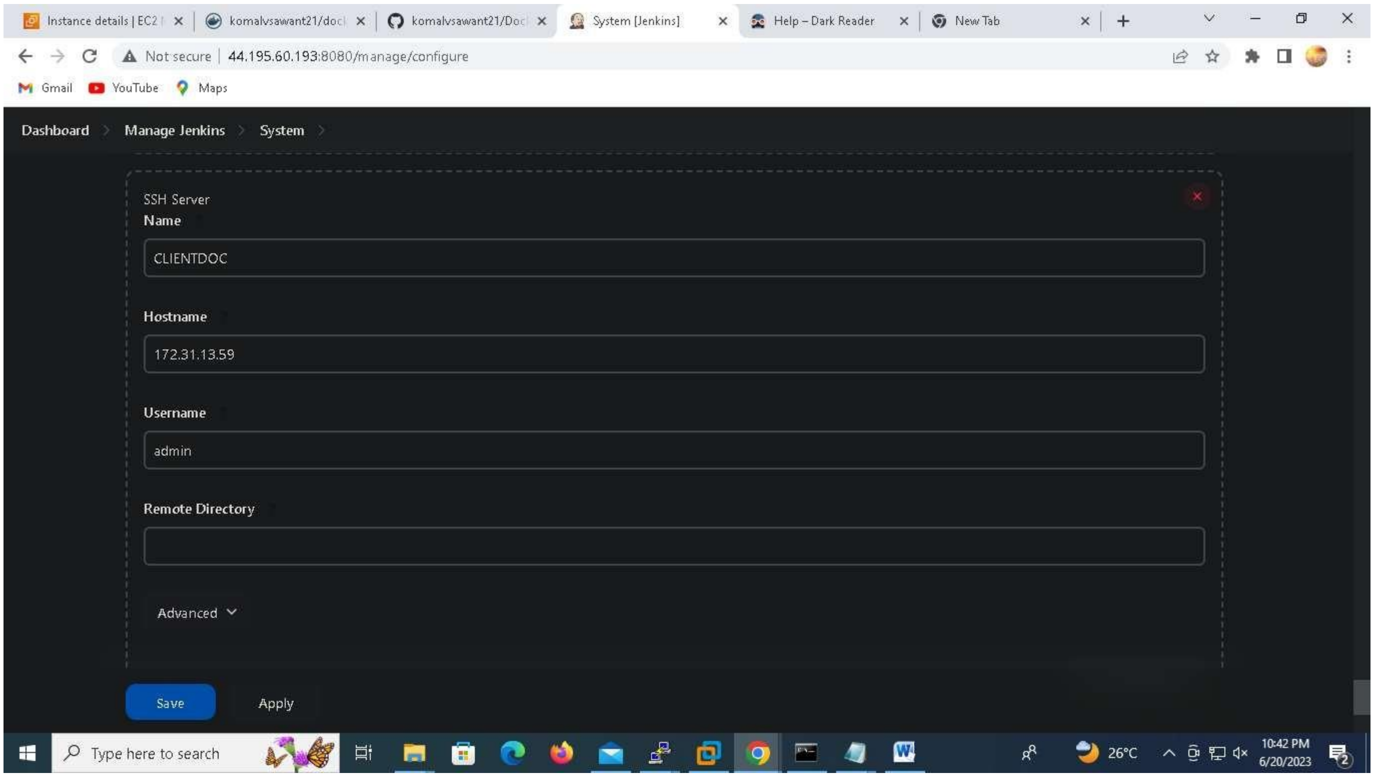

Path to key

Key

```
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEA3kQuOIWV9z2F429vbiSKAT6yYAJfazzADUow/kdNWMInwLT7vd4V
FpSYsOQCK5XDazm4ZBGTgl3V0QY3VLr79OpcnDETHk7ilezeq/0T8AVB7yHQVLezEzlf8F
j9ndELL3mdfa3yb9kCrf4bRZ3HPUErPT/gCHBBHsjOcKHJFvTqYA9bEM0KEiuuz2E6r2sR
MAQiYmn9MSbda3WWktjr6V1eFNh9lpgyniL6Q7q1w0Xrk4khtoduVoNImaYVCZ2HgUts0C
TN8FiuyxbEoZmJlWH8uTZgl41UhQSJ/SEaZgROR89sOc282N5bl1YZJr1IakPv35sCxtVK
KWNxluG2jhzbmxkWUzBlRz6DVGBFkBueUgUT8DOoBfOMeuZxlmo8zYNESjub7xhC/ncyGS
o8EbpaR+8a6Wx8FPO61i0Fds3UALavUXOJ4zJ2BaRJbhBdPXVmXCtbpj8aua+VLSFnQcTH
HDP6semlwj6lwhhCqeBT7L2uFAr6CUvmO3HTZr0RAAAFkJL9Ap2S/QKdAAAAB3NzaC1yc2
EAAAGBAN5ELjiFlfc9heNvb24kigE+smACX2s8wA1KMP5HTVjCJ8C0+73eFRaUmLDkAiuV
w2s5uGQRk4Jd1dEGN1S6+/TqXJwxEx5O4iHs3qv9E/AFQe8h0FS3sxMyH/BY/Z3RCy95nX
2t8m/ZAq3+G0Wdxz1BKz0/4AhwQR7IznChyRb06mAPWxDNChIrrs9hOq9rETAElmJp/TEm
```

Expand server

Add name of instance + add usrname as admin + add hostname i.e pvt ip of server



SSH Server

Name

CLIENTDOC

Hostname

172.31.13.59

Username

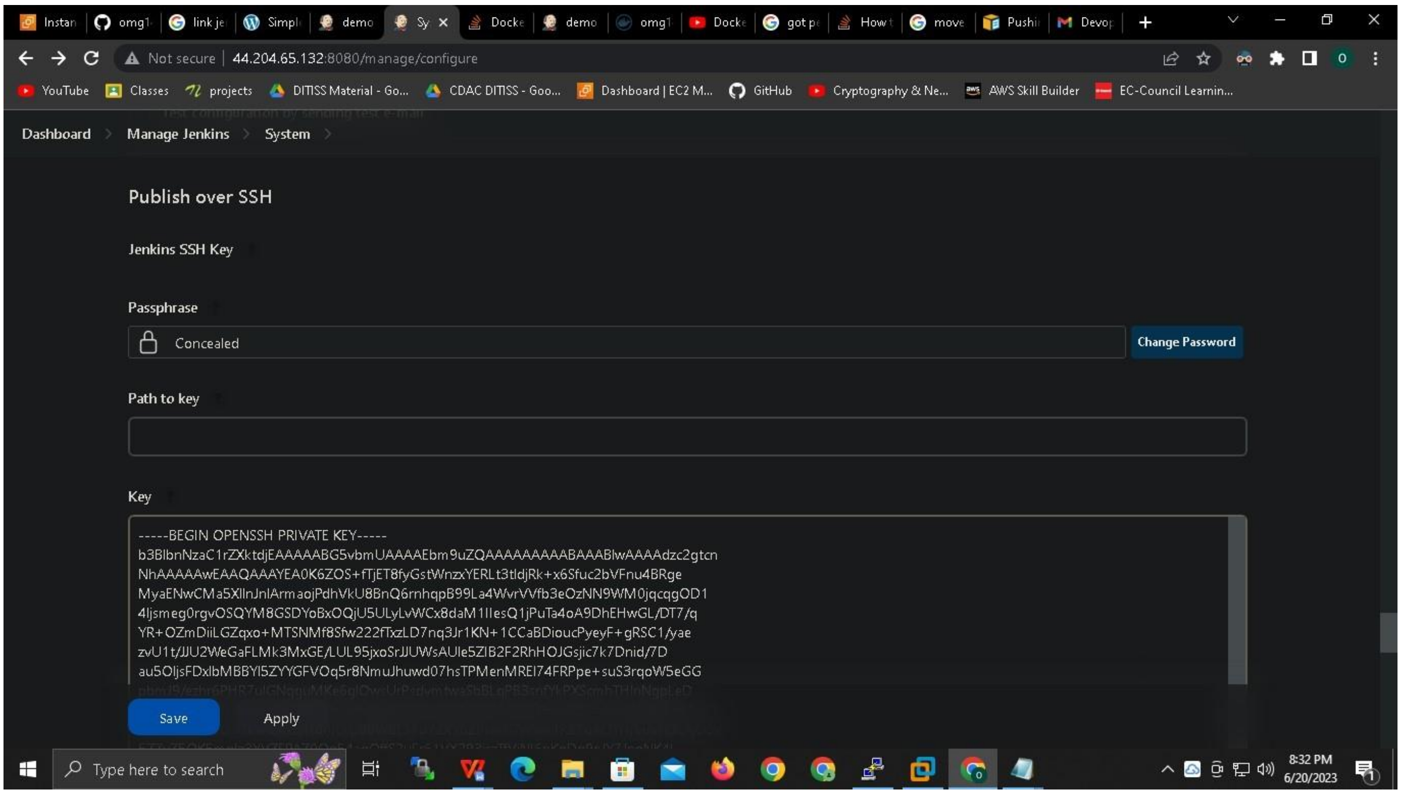
admin

Remote Directory

Define a project

Pipeline Project -> add git hub url -> Pipeline Script:

Manage jenkins -> credentials -> add



Dashboard > Manage Jenkins > System

Publish over SSH

Jenkins SSH Key

Passphrase: Concealed

Path to key

Key

```
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEA0K6ZOS+fTjET8fyGstWnzxYERLt3tldjRk+x6Sfuc2bVFnu4BRge
MyaENwCMa5XIlnJnlArmaojPdhVkU8BnQ6rnhqpB99La4WvrVVfb3eOzNN9WM0jqcqgOD1
4ljsmeg0rgvOSQYM8GSDYoBxOQjU5ULyLvWCx8daM1IIesQ1jPuTa4oA9DhEHwGL/DT7/q
YR+OZmDiiLGZqxo+MTSNMf8Sfw222fTxzLD7nq3Jr1KN+1CCaBDioucPyeyF+gRSC1/yae
zvU1t/JJU2WeGaFLMk3MxGE/LUL95jxoSrJJUWsAUle5ZIB2F2RhHOJGsjic7k7Dnid/7D
au5OljsFDxlbMBBYI5ZYYGFVOq5r8NmuJhuwd07hsTPMenMREI74FRPpe+suS3rqoW5eGG
```

pipeline {

agent any (anyone can login)

environment {

DOCKERHUB_CREDENTIALS= credentials('dockerhubcredentials')

} (id which we have given during setting up credentials)

stages {

stage("Git Checkout"){

steps{

git credentialsId: 'github', url: 'https://github.com/omg1410-gadre/docker.git'

echo 'Git Checkout Completed'

}

} (adding github https url)

stage('Build Docker Image') {

steps{

sh 'docker build -t omg1410/docker2 .' (docker build -t uname/repo)

echo 'Build Image Completed'

}

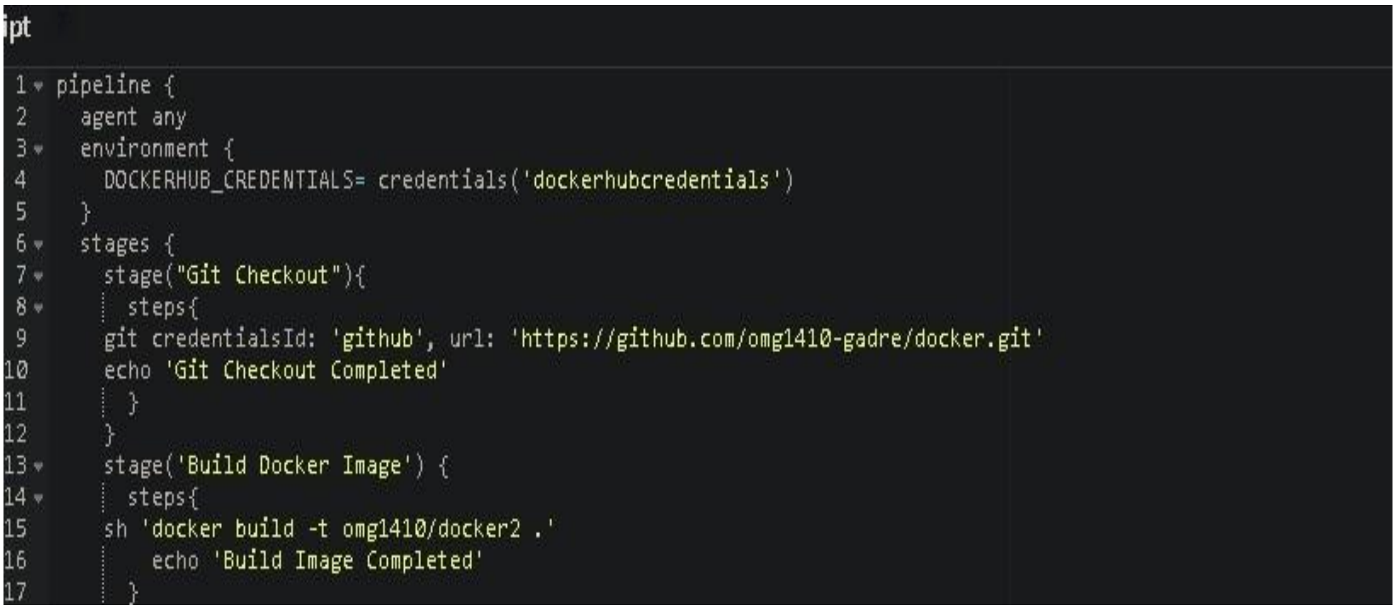
}

stage('Login to Docker Hub') {

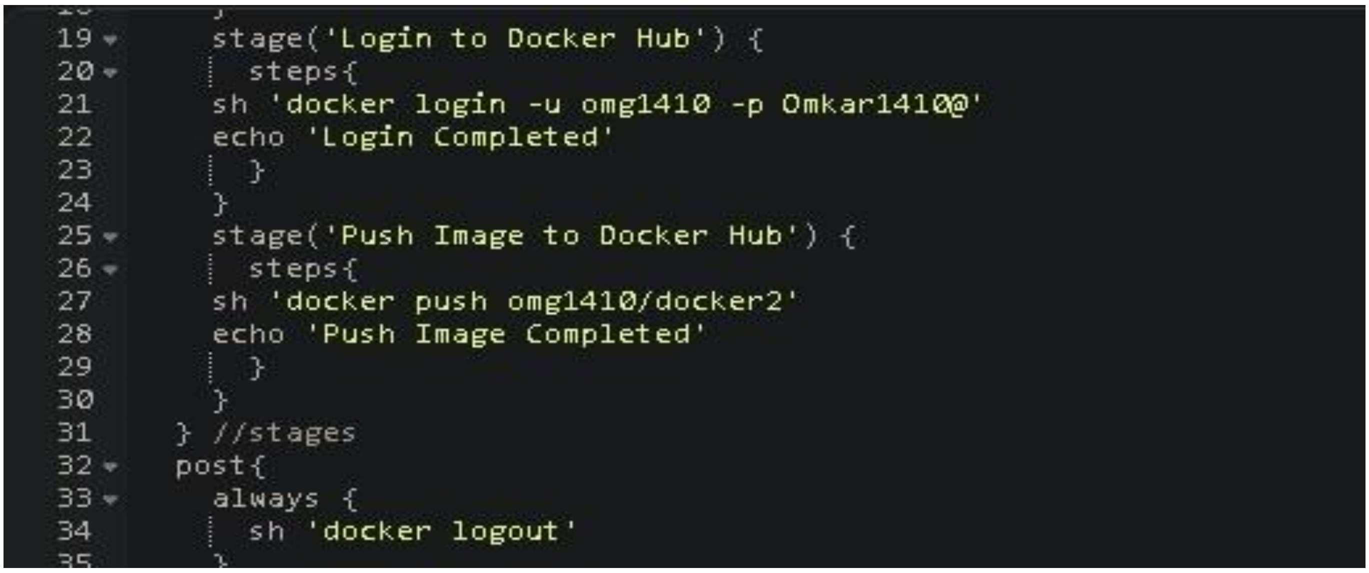
steps{

sh 'docker login -u omg1410 -p Omkar1410@' (login in docker)

```
        echo 'Login Completed'
    }
  }
  stage('Push Image to Docker Hub') {
    steps{
        sh 'docker push omg1410/docker2'        (push image)
        echo 'Push Image Completed'
    }
  }
   stage('SSH login') {                          (login to client )
       steps{
        sh 'ssh admin@172.31.31.147'
        echo 'SSH Login Completed'
       }
     }
```



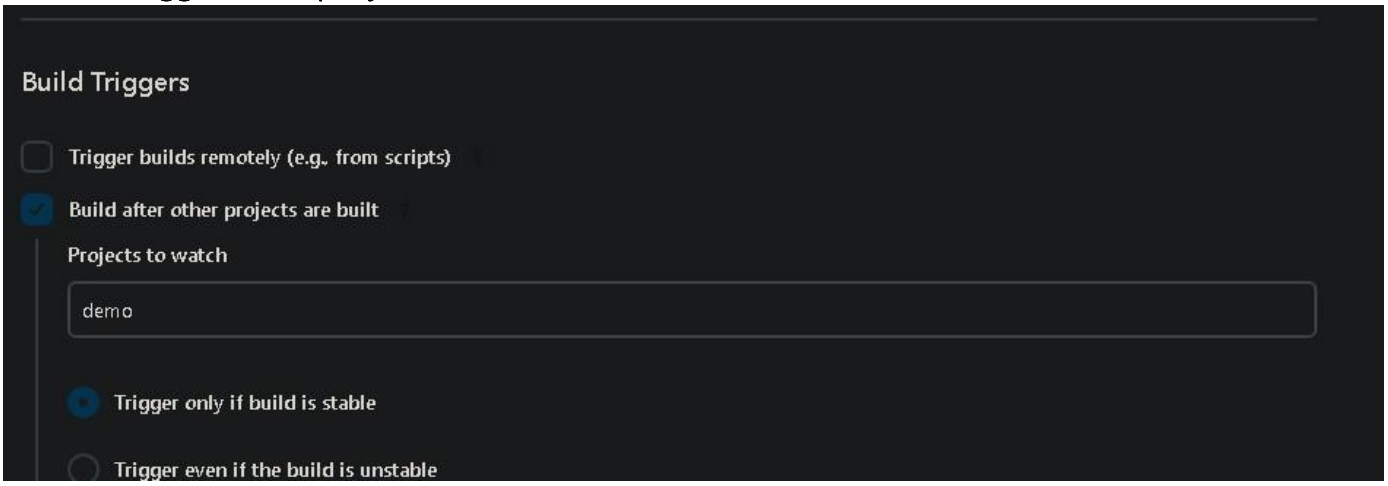
```
ipt
pipeline {
  agent any
  environment {
    DOCKERHUB_CREDENTIALS= credentials('dockerhubcredentials')
  }
  stages {
    stage("Git Checkout"){
      steps{
    git credentialsId: 'github', url: 'https://github.com/omg1410-gadre/docker.git'
    echo 'Git Checkout Completed'
      }
    }
    stage('Build Docker Image') {
      steps{
    sh 'docker build -t omg1410/docker2 .'
        echo 'Build Image Completed'
      }
```



```
      stage('Login to Docker Hub') {
        steps{
      sh 'docker login -u omg1410 -p Omkar1410@'
      echo 'Login Completed'
        }
      }
      stage('Push Image to Docker Hub') {
        steps{
      sh 'docker push omg1410/docker2'
      echo 'Push Image Completed'
        }
      }
    } //stages
    post{
      always {
        sh 'docker logout'
```

2nd Pipeline :

In build trigger : add project to watch



Build Triggers

☐ Trigger builds remotely (e.g., from scripts)

☑ Build after other projects are built

Projects to watch

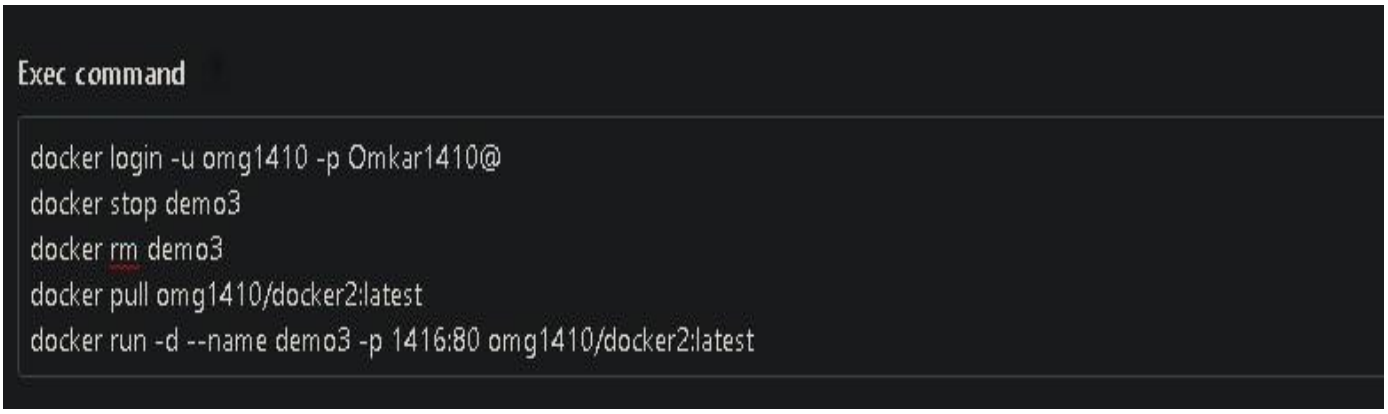
demo

● Trigger only if build is stable

○ Trigger even if the build is unstable

Add this in SSH : publish over ssh -> exec command
This all will be performed in client side remotely

docker login -u omg1410 -p Omkar1410@
docker stop demo3 (*to remove or stop existing images)*
docker rm demo3
docker pull omg1410/docker2:latest
docker run -d --name demo3 -p 1416:80 omg1410/docker2:latest



Exec command

```
docker login -u omg1410 -p Omkar1410@
docker stop demo3
docker rm demo3
docker pull omg1410/docker2:latest
docker run -d --name demo3 -p 1416:80 omg1410/docker2:latest
```

Client side:
$sudo apt-get install docker
$sudo chown nobody:nogroup /var/run/docker.sock
$sudo chown nobody:nogroup /var/lib/docker
$sudo chmod 7777 /var/run/docker.sock
$sudo chmod 7777 /var/lib/docker
$cd .ssh
$sudo nano auth___key
Add jenkins pvt key into it .

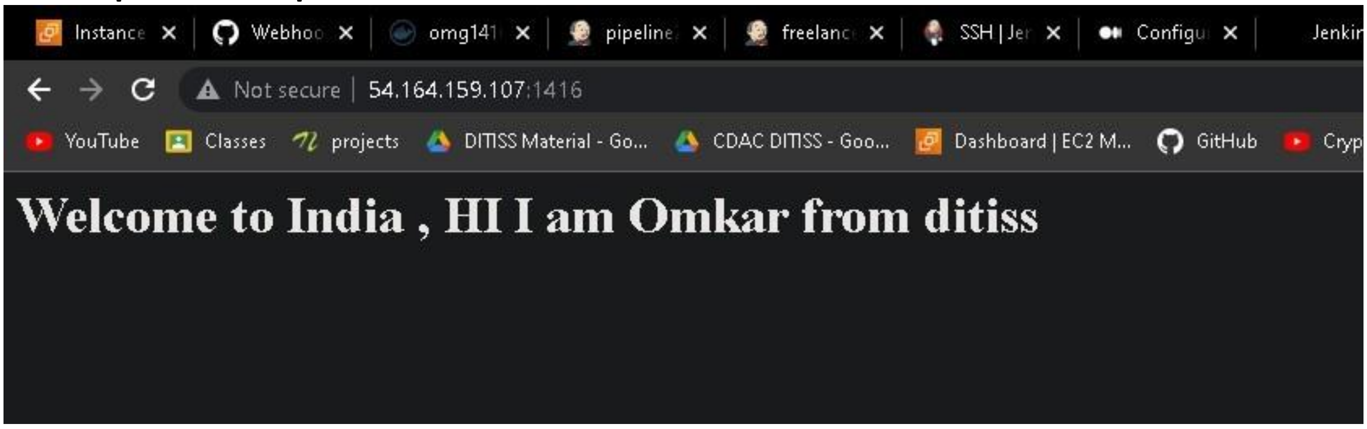
Testing :

Ubuntu machine :

```
$nano index.html
$git add index.html
$git commit -m “ggg”
$git push origin master
```

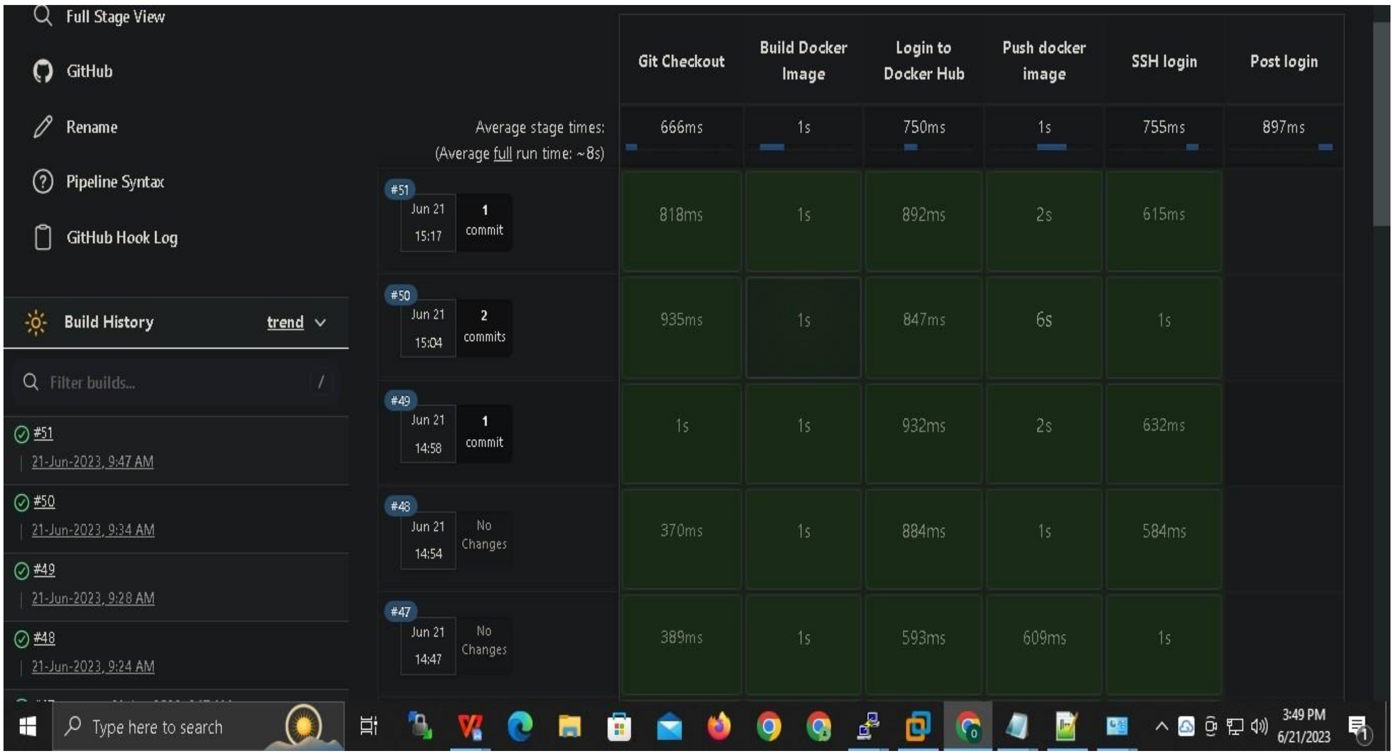
Check on Browser

Public ip of client : port

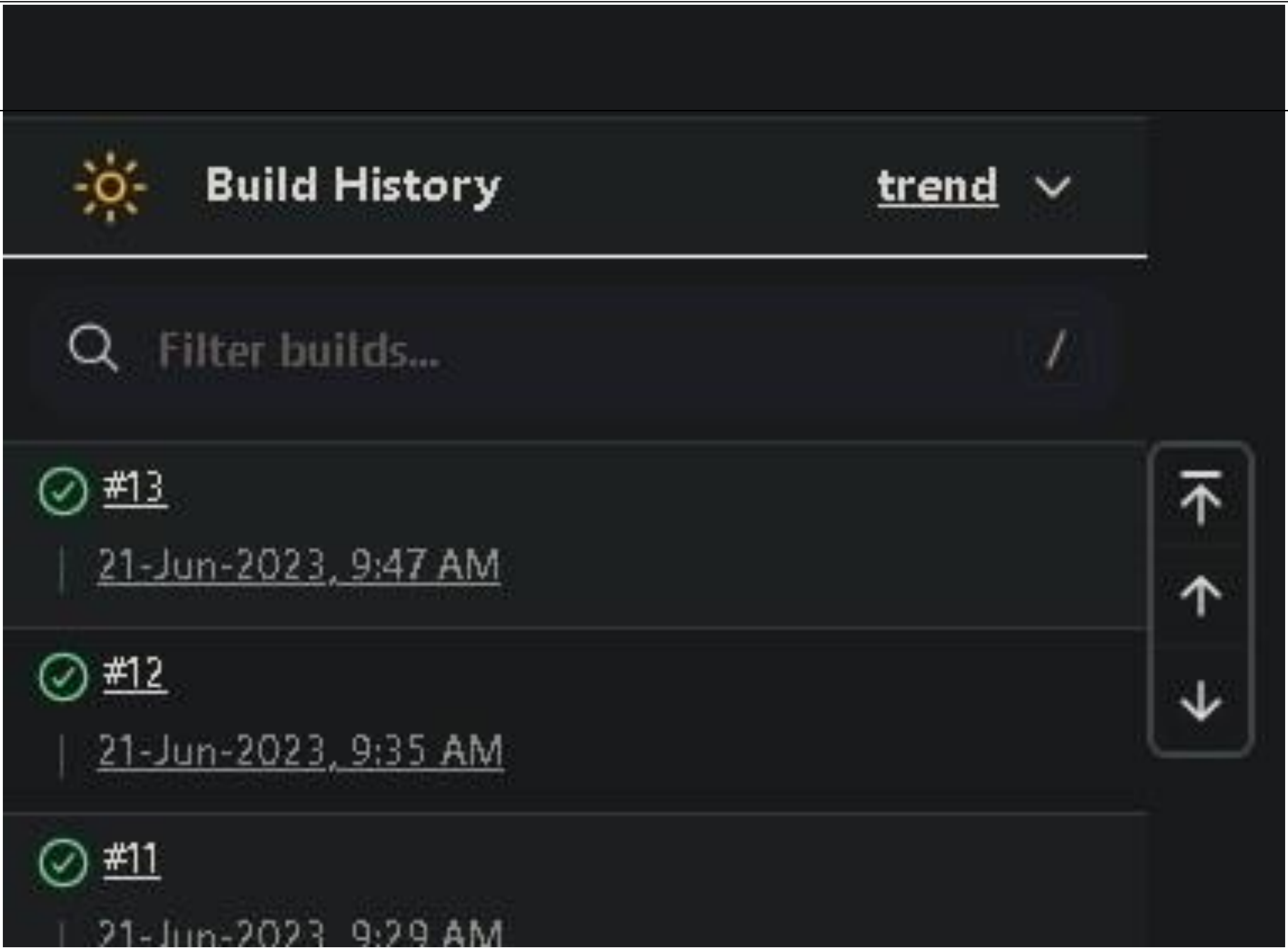


Welcome to India , HI I am Omkar from ditiss

Pipeline View :



	Git Checkout	Build Docker Image	Login to Docker Hub	Push docker image	SSH login	Post login
Average stage times: (Average full run time: ~8s)	666ms	1s	750ms	1s	755ms	897ms
#51 Jun 21 15:17 1 commit	818ms	1s	892ms	2s	615ms	
#50 Jun 21 15:04 2 commits	935ms	1s	847ms	6s	1s	
#49 Jun 21 14:58 1 commit	1s	1s	932ms	2s	632ms	
#48 Jun 21 14:54 No Changes	370ms	1s	884ms	1s	584ms	
#47 Jun 21 14:47 No Changes	389ms	1s	593ms	609ms	1s	



#13
21-Jun-2023, 9:47 AM
#12
21-Jun-2023, 9:35 AM
#11
21-Jun-2023, 9:29 AM

Sticky bit : (interview ques diff b/w 777 and 7777)

In computing, the sticky bit is a user ownership access right flag that can be assigned to files and directories on Unix-like systems.

There are two definitions: one for files, one for directories.

For files, particularly executables, superuser could tag these as to be retained in main memory, even when their need ends, to minimize swapping that would occur when another need arises, and the file now has to be reloaded from relatively slow secondary memory.[1] This function has become obsolete due to swapping optimization.

For directories, when a directory's sticky bit is set, the filesystem treats the files in such directories in a special way so only the file's owner, the directory's owner, or root user can rename or delete the file Without the sticky bit set, any user with write and execute permissions for the directory can rename or delete contained files, regardless of the file's owner. Typically this is set on the /tmp directory to prevent ordinary users from deleting or moving other users' files.

The modern function of the sticky bit refers to directories, and protects directories and their content from being hijacked by non-owners this is found in most modern Unix-like systems. Files in a shared directory such as /tmp belong to individual owners, and non-owners may not delete, overwrite or rename them.

Summary :

Ubuntu :

```
$sudo apt-get install git && apt-get install docker.io
$mkdir docker
$cd docker
$nano Dockerfile
$nano index.html
$git config --global user.name “”
$git config --global user.email””
$git remote add “”
```

Git page : create git repo
Add webhook

Client :EC2 instance **(allow all tcp port in inbound rules)**

```
$sudo apt-get update
$sudo apt-get install docker.io
$sudo chown nobody:nogroup /var/lib/docker
$sudo chmod 7777 /var/lib/docker
$sudo usermod -a -G docker admin
$cd .ssh
$ sudo  nano authorised_keys
        Add jenkins public key here
```

Jenkins : EC2 instance(allow all tcp ports inbound rules)

```
$sudo apt-get update
$install jenkins ..
$sudo apt-get install git
$sudo apt-get install docker.io
$sudo chown nobody:nogroup /var/run/docker.sock
$sudo chown nobody:nogroup /var/lib/docker
$sudo chmod 7777 /var/run/docker.sock
$sudo chmod 7777 /var/lib/docker
$sudo passwd jenkins
$su jenkins
$ssh-keygen
```

Add public key into client authorised key file
Also copy paste th pvt key .

Jenkins Dashboard : port 8080

Real game starts here
Install plugins default
Install ssh + docker + publish over ssh plugin also and restart ..

Go to manage jenkins -> credentials -> name given by me is dockerhubcredentials
Go to manage jenkins -> system -> publish over ssh
    Add pvt key of jenkins user only
    Add server -> name -> pvt ip of client -> username
Do testing :

1st project :Select as pipeline

Create pipeline :
Creating a environment : No need just do agent any

```
pipeline  {
  agent any
  environment {
    DOCKERHUB_CREDENTIALS= credentials('dockerhubcredentials')
  }
```

Git repo:

```
  stages {
    stage("Git Checkout"){
      steps{
           git credentialsId: 'github', url: 'https://github.com/omg1410-gadre/docker.git'
           echo 'Git Checkout Completed'
      }
  }
```

Build Docker image;

Image name : omg1410/docker2

```
    stage('Build Docker Image') {
      steps{
           sh 'docker build -t omg1410/docker2 .'
        echo 'Build Image Completed'
      }
    }
```

Login to docker hub:

```
    stage('Login to Docker Hub') {
    steps{
           sh 'docker login -u omg1410 -p Omkar1410@'
           echo 'Login Completed'
      }
    }
```

Push docker image :

Docker repo : omg1410/docker2

```
     stage('Push docker image') {
      steps{
           sh 'docker push omg1410/docker2:latest'
           echo 'Push Completed'
      }
```

```
   }
```

SSH login:

```
   stage('SSH
   login') {steps{
        sh 'ssh admin@172.31.31.147'                          pvt ip  of client
        echo 'SSH Login completed'
     }
   }
```

Post SSH :

Go to pipeline syntax -> select publish over ssh -> add commands which you wanna run after ssh -> generate script . copy

Go to pipeline

```
  Stage("Post SSH ") {
                Steps{
                Copied content
                  }
        }
```

Email alert :

Go to pipeline syntax : email extended option ->
Add all required details -> generate pipeline script

```
Stage ("email") {
       Steps {
                Copied content
                  }
     }
  } //stages
} //pipeline
```

Or 2nd Project : Freelance

Build trigger : add trigger that when 1st pipeline is executed successfully then run 2nd pipeline .(build project after othe rproject build)

Publish over ssh -> exec shell

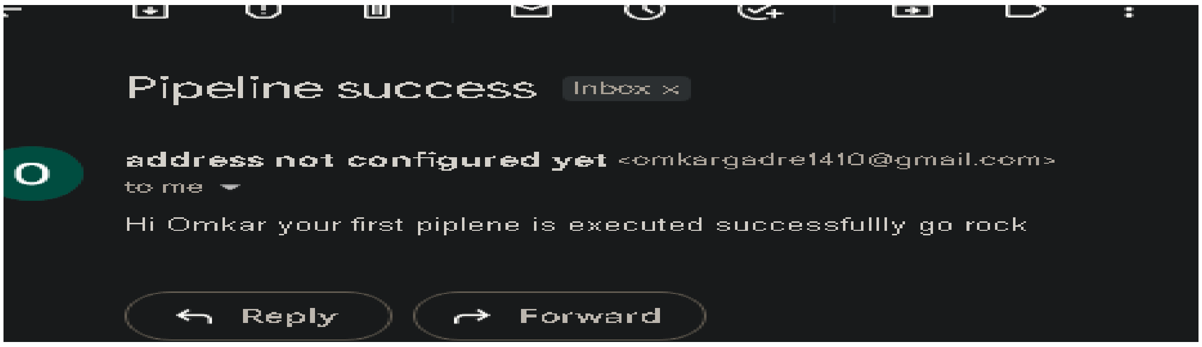
docker login -u omg1410 -p Omkar1410@
docker stop *demo3* (*to remove or stop existing images)*
docker rm *demo3*
docker pull *omg1410/docker2:latest*
docker run -d --name *demo3* -p *1416:80 omg1410/docker2:latest*

(Testing)

```
$git add Dockerfile
$git commit -m “”
$git add index.html
$git commit -m “test..”
$git push origin master
```

Apache ip : port you have assigne

Check email:



Pipeline success

address not configured yet <omkargadre1410@gmail.com>

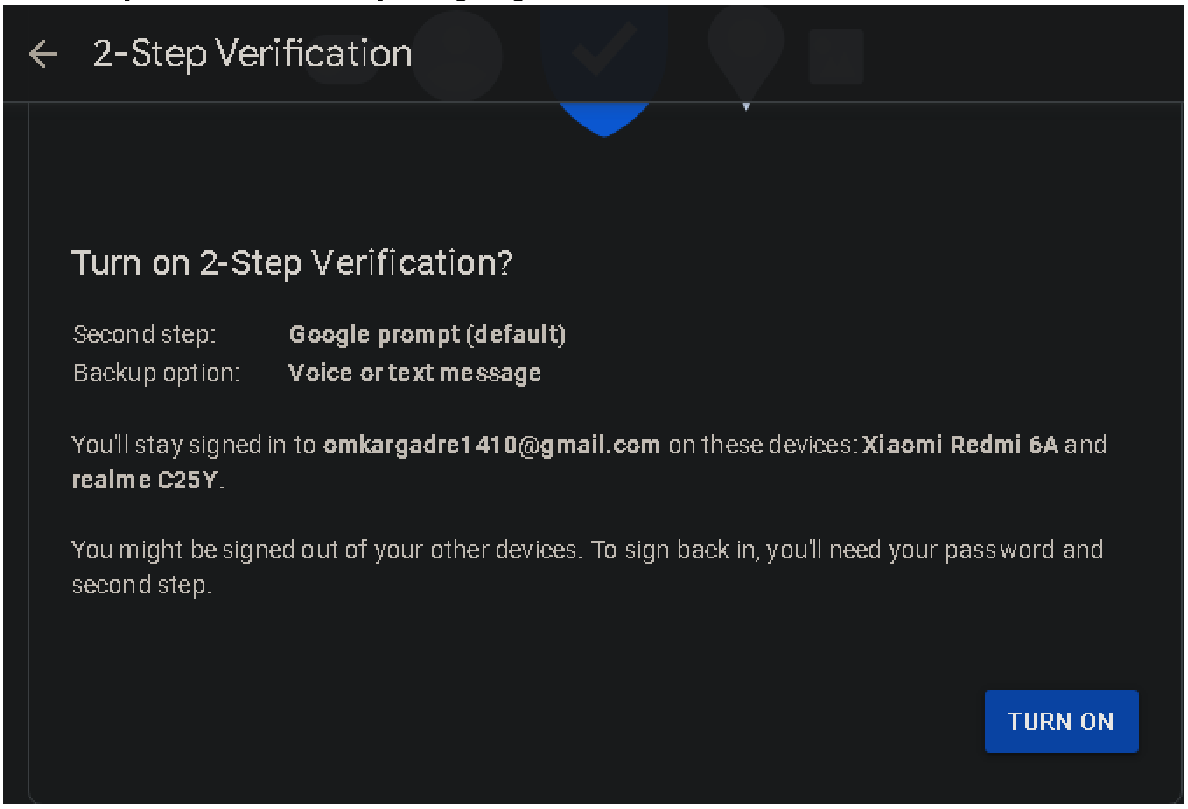
to me

Hi Omkar your first piplene is executed successfullly go rock

Docker access token :

Email alert :

Allow 2 step verification in your google account :



← 2-Step Verification

Turn on 2-Step Verification?

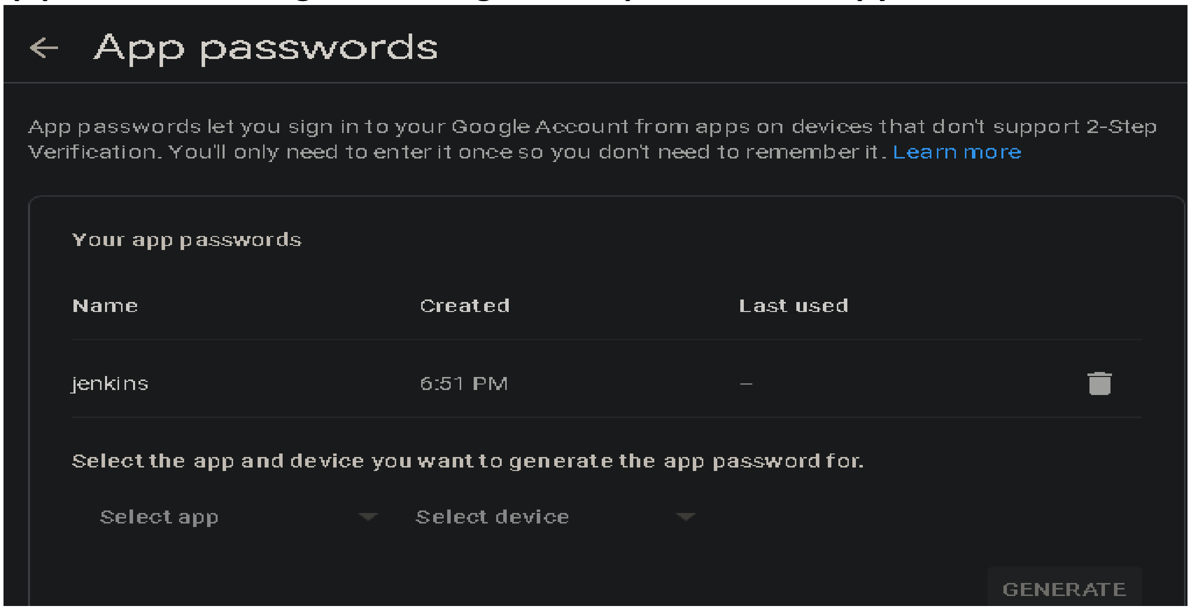
Second step: **Google prompt (default)**
Backup option: **Voice or text message**

You'll stay signed in to **omkargadre1410@gmail.com** on these devices: **Xiaomi Redmi 6A** and **realme C25Y**.

You might be signed out of your other devices. To sign back in, you'll need your password and second step.

TURN ON

App password -> assign name -> generate password -> copy it



← App passwords

App passwords let you sign in to your Google Account from apps on devices that don't support 2-Step Verification. You'll only need to enter it once so you don't need to remember it. Learn more

Your app passwords

Name	Created	Last used
jenkins	6:51 PM	–

Select the app and device you want to generate the app password for.

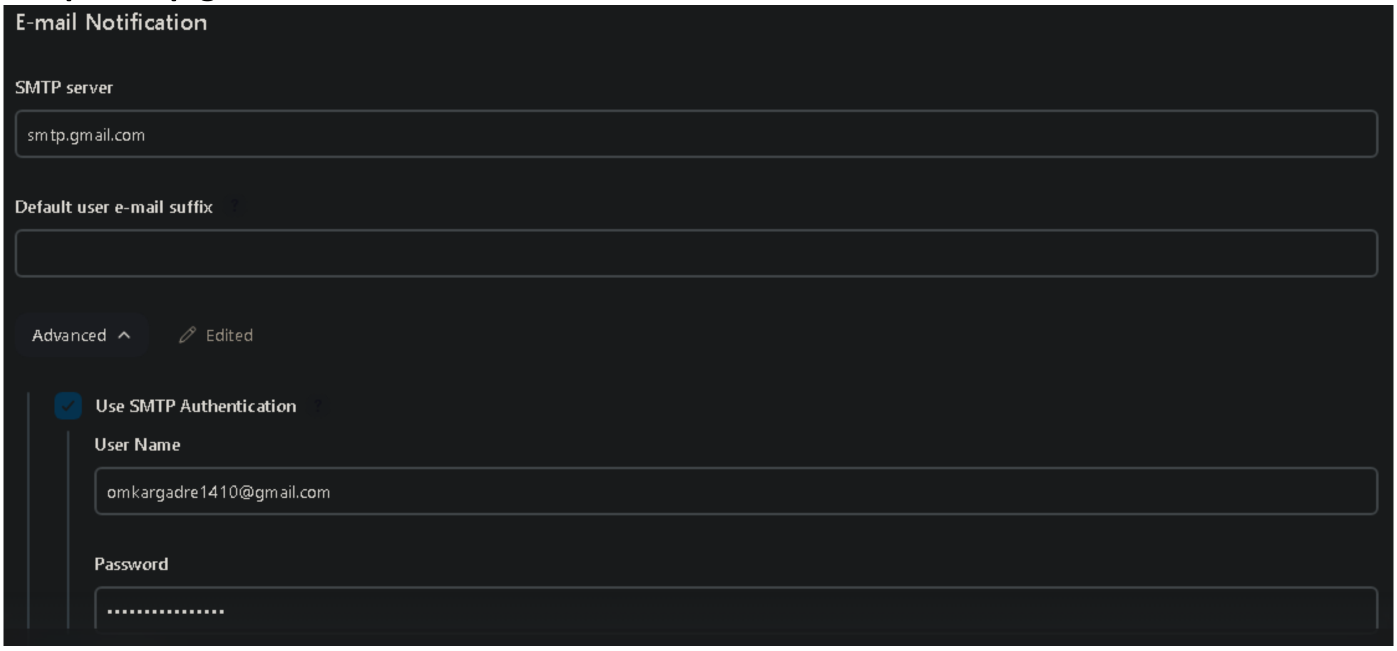
Select app ▾ Select device ▾

GENERATE

Jenkins dashboard -> manage jenkins -> available plugin -> instal email extension template

Manage jenkins -> system -> extended email

Username - mail
Password -> which we got assigned
Smtp : smtp.gmail.com



E-mail Notification

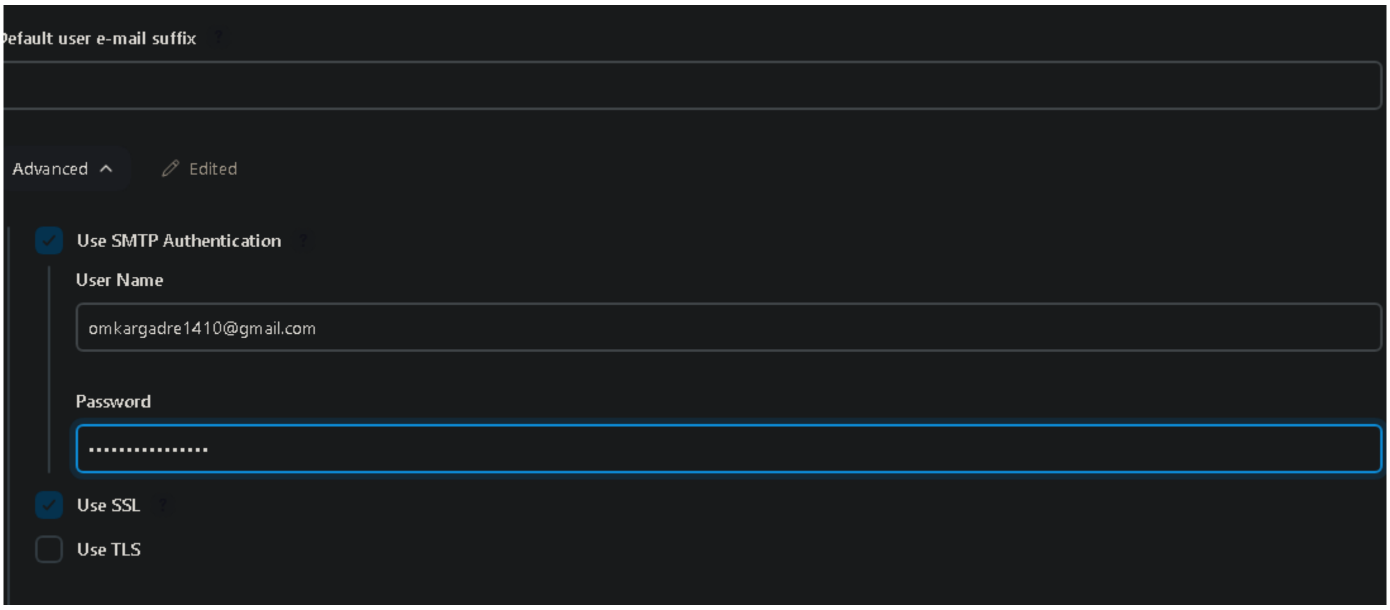
SMTP server
smtp.gmail.com

Default user e-mail suffix

☑ Use SMTP Authentication
User Name
omkargadre1410@gmail.com

Password
················

Port 465
Enable ssl



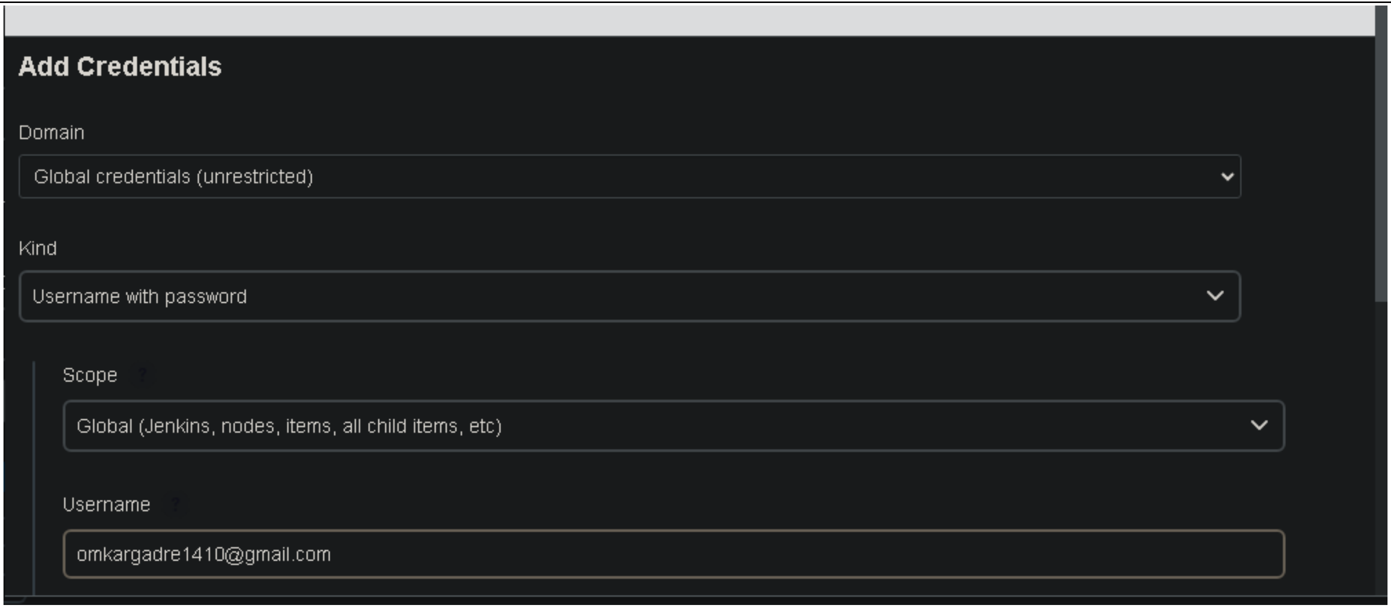
efault user e-mail suffix

☑ Use SMTP Authentication
User Name
omkargadre1410@gmail.com

Password
················

☑ Use SSL
☐ Use TLS

Smtp server
Port
Add credentials
- **Uname -mail**
- **Pwd -which we got assigned**



Add Credentials

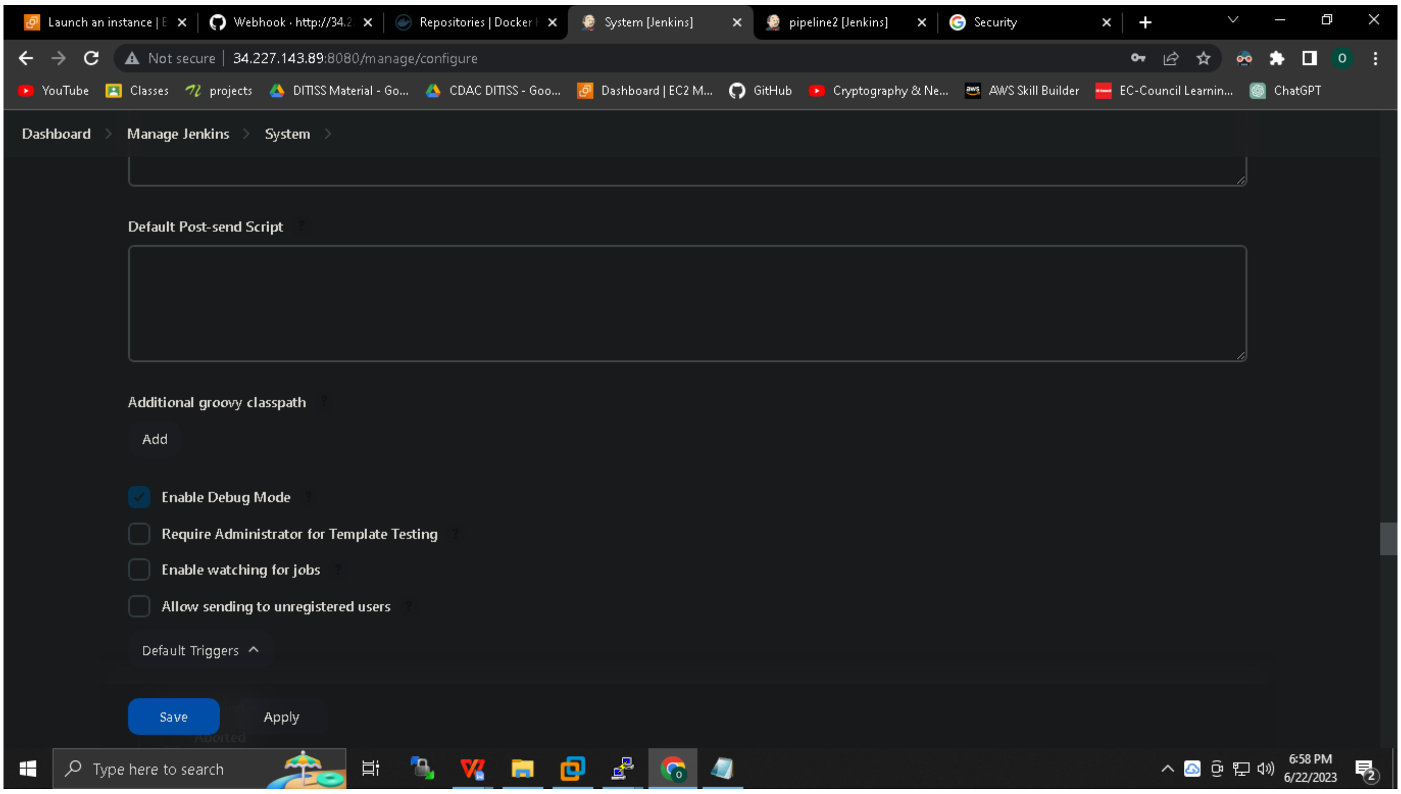
Domain: Global credentials (unrestricted)

Kind: Username with password

Scope: Global (Jenkins, nodes, items, all child items, etc)

Username: omkargadre1410@gmail.com

Remove already in deafult pre sent and post sent script



Dashboard > Manage Jenkins > System

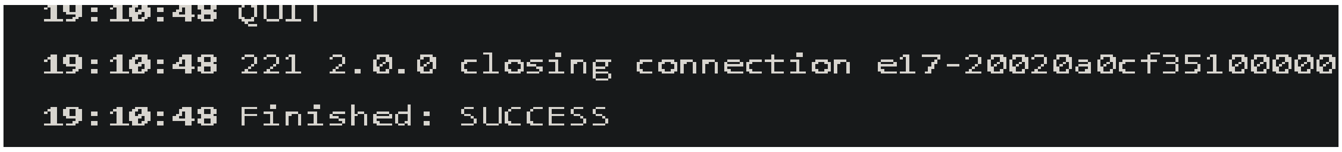
Default Post-send Script

Additional groovy classpath

- [✓] Enable Debug Mode
- [ ] Require Administrator for Template Testing
- [ ] Enable watching for jobs
- [ ] Allow sending to unregistered users

**Go to your project -> add post build -> enable editable email notification ->
Go to that section -> editable email notification**

**Project receipent list -> add email
In trigger add receipents list , remove developers
Advanced -> add receipent list**



```
19:10:48 QUIT
19:10:48 221 2.0.0 closing connection e17-20020a0cf35100000
19:10:48 Finished: SUCCESS
```