You are hired as a DevOps Engineer for Analytics Pvt Ltd. This company is a product-based organization which uses Docker for their containerization needs within the company. The final product received a lot of traction in the first few weeks of launch. Now with the increasing demand, the organization needs to have a platform for automating deployment, scaling and operations of application containers across clusters of hosts. As a DevOps Engineer, you need to implement a DevOps lifecycle such that all the requirements are implemented without any change in the Docker containers in the testing environment.

Up until now, this organization used to follow a monolithic architecture with just 2 developers. The product is present on: https://github.com/hshar/website.git

Following are the specifications of the lifecycle:

1. Git workflow should be implemented. Since the company follows a monolithic architecture of development, you need to take care of version control. The release should happen only on the 25th of every month.

2. CodeBuild should be triggered once the commits are made in the master branch.

3. The code should be containerized with the help of the Dockerfile. The Dockerfile should be built every time if there is a push to GitHub. Create a custom Docker image using a Dockerfile.

4. As per the requirement in the production server, you need to use the Kubernetes cluster and the containerized code from Docker Hub should be deployed with 2 replicas. Create a NodePort service and configure the same for port 30008.

5. Create a Jenkins Pipeline script to accomplish the above task.

6. For configuration management of the infrastructure, you need to deploy the configuration on the servers to install necessary software and configurations.

7. Using Terraform, accomplish the task of infrastructure creation in the AWS cloud provider.

Architectural Advice: Softwares to be installed on the respective machines using configuration management.

Worker1: Jenkins, Java

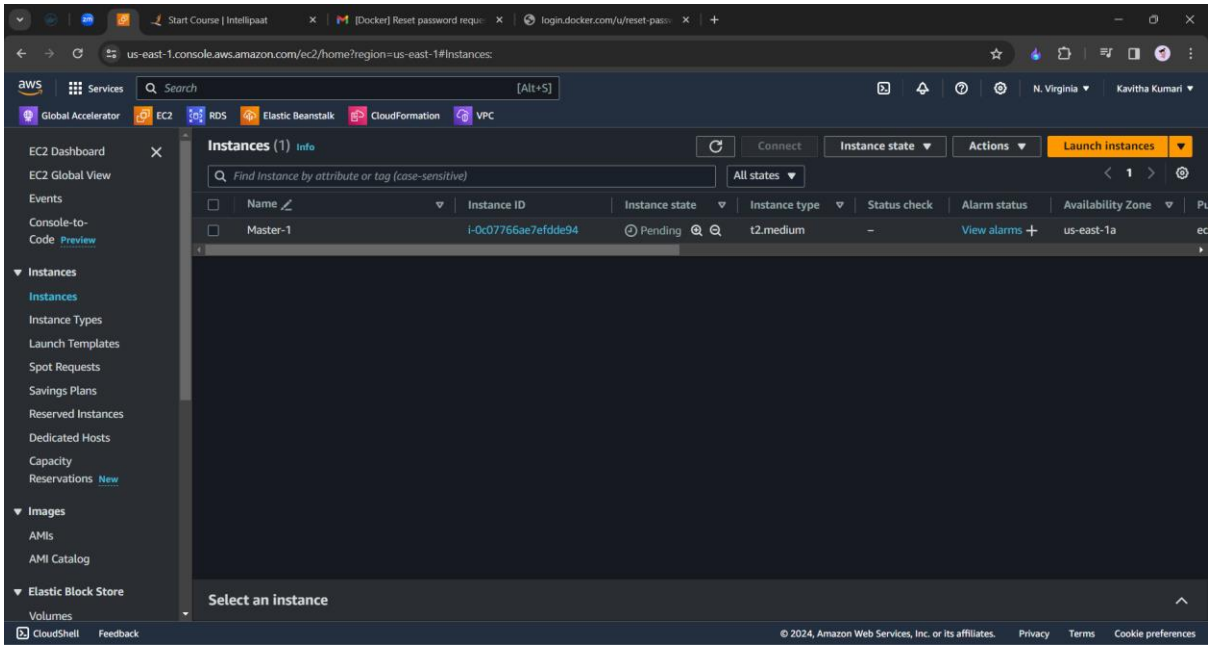
Worker2: Docker, Kubernetes

Worker3: Java, Docker, Kubernetes

Worker4: Docker, Kubernetes

Procedure: -

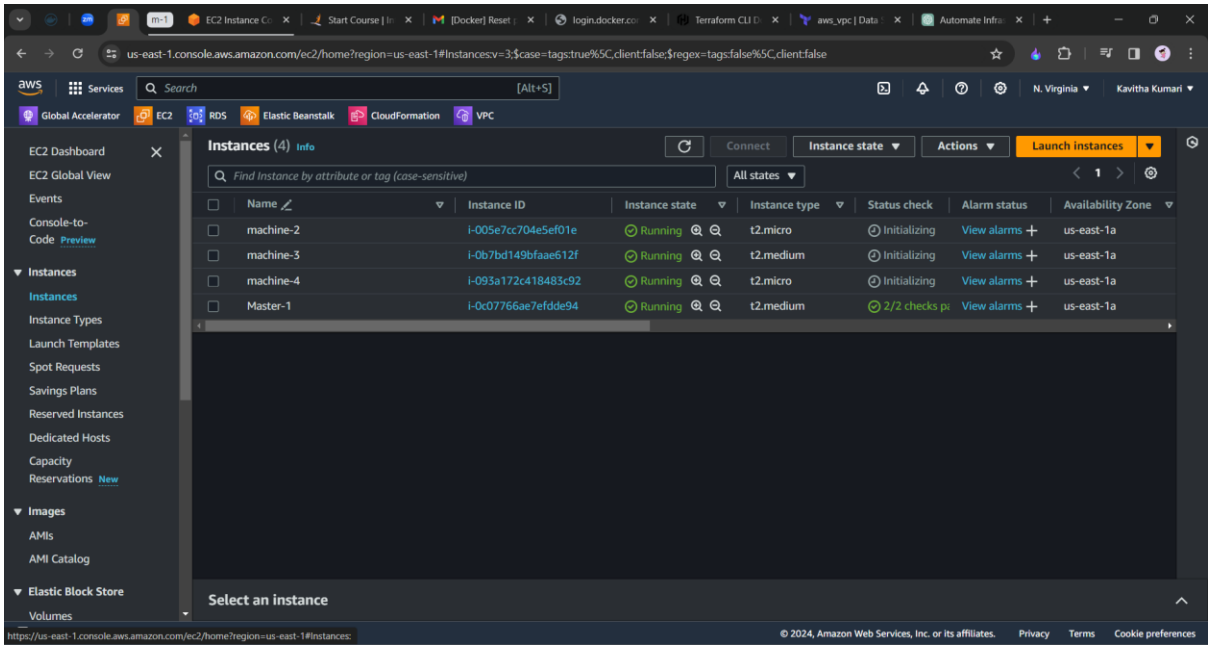
- **We will create one machine on EC2 (main-1)and we will create another 3 with the help of terraform**



- Launch the t2.medium type instance and install terraform “terraform init”,”terraform plan”,”terraform apply” (https://developer.hashicorp.com/terraform/install)
- The below is terraform.tf

```
provider "aws"{
    region="us-east-1"
    access_key=" "
    secret_key=" "
}
resource "aws_instance" "kubernetes-master"{
    ami="ami-0cd59ecaf368e5ccf"
    instance_type="t2.medium"
    subnet_id="subnet-0e8fec508d60f161d"
    vpc_security_group_ids=["sg-0af8f800022ebd9e5"]
    key_name="morningkey"
    associate_public_ip_address = true
    tags={
        Name="machine-3"
    }
}
resource "aws_instance" "kubernetes-slave2"{
    ami="ami-0cd59ecaf368e5ccf"
    instance_type="t2.micro"
    subnet_id="subnet-0e8fec508d60f161d"
    vpc_security_group_ids=["sg-0af8f800022ebd9e5"]
    key_name="morningkey"
    associate_public_ip_address = true
    tags={
        Name="machine-4"
    }
}
resource "aws_instance" "kubernetes-slave1"{
    ami="ami-0cd59ecaf368e5ccf"
```

```
    instance_type="t2.micro"
    subnet_id="subnet-0e8fec508d60f161d"
    vpc_security_group_ids=["sg-0af8f800022ebd9e5"]
    key_name="morningkey"
    associate_public_ip_address = true
    tags={
        Name="machine-2"
    }
}
```



- Now install ansible of the main machine (main-1) and also ssh-keygen and edit the hosts"sudo nano /etc/ansible/hosts"'

- **Now we will create the playbook play.yaml**
- **Also, the script files to install the resources.**
- **Script1.sh**

```
sudo apt update
sudo apt install openjdk-11-jdk -y
//jenkins commands
```

- script2

```
sudo apt update
sudo apt install openjdk-11-jdk -y
sudo apt install docker.io -y
//commands show in the below link
```

https://github.com/Origamini/K8s-Installation/blob/main/New%20K8s%20Installation.txt

- script3.sh

```
sudo apt update
sudo apt install docker.io -y
//commands show in the above link
```

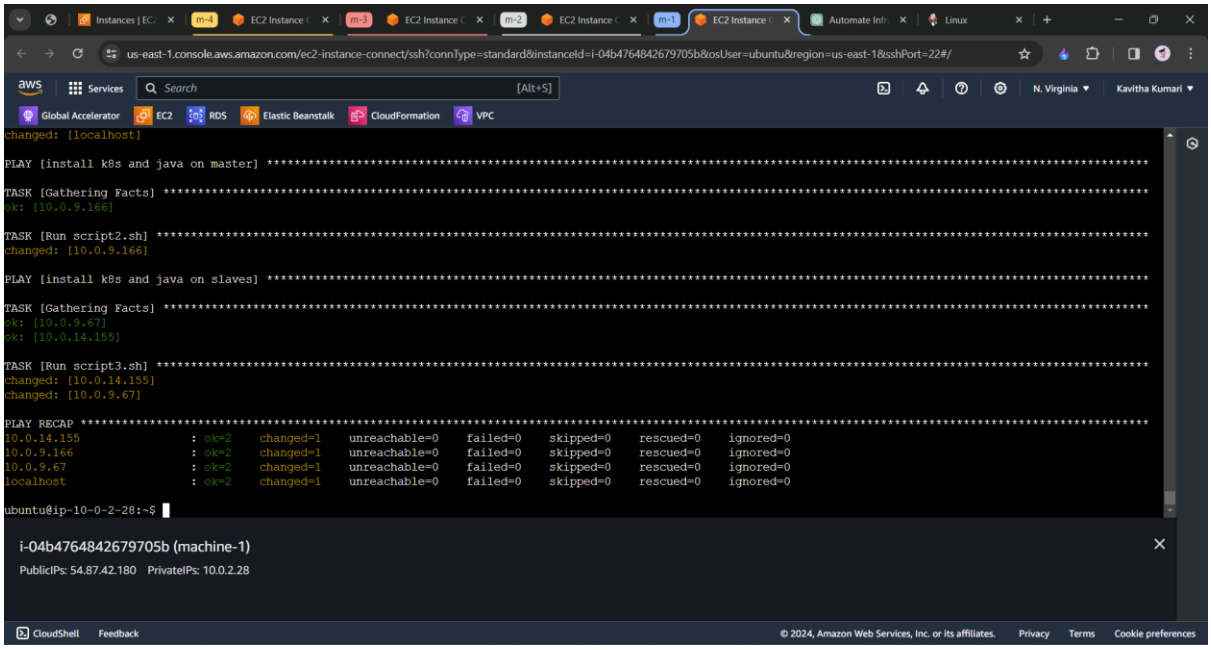
- play.yaml

```
---
- name: install jenkins and java on machine-1
  become: true
  hosts: localhost
  tasks:
  - name: running script
  - script: script1.sh
- name: install k8s and java on machine-1
  become: true
  hosts: master
  tasks:
  - name: running script
  - script: script2.sh
- name: install k8s and java on machine-1
  become: true
  hosts: slaves
  tasks:
  - name: running script
  - script: script3.sh
```

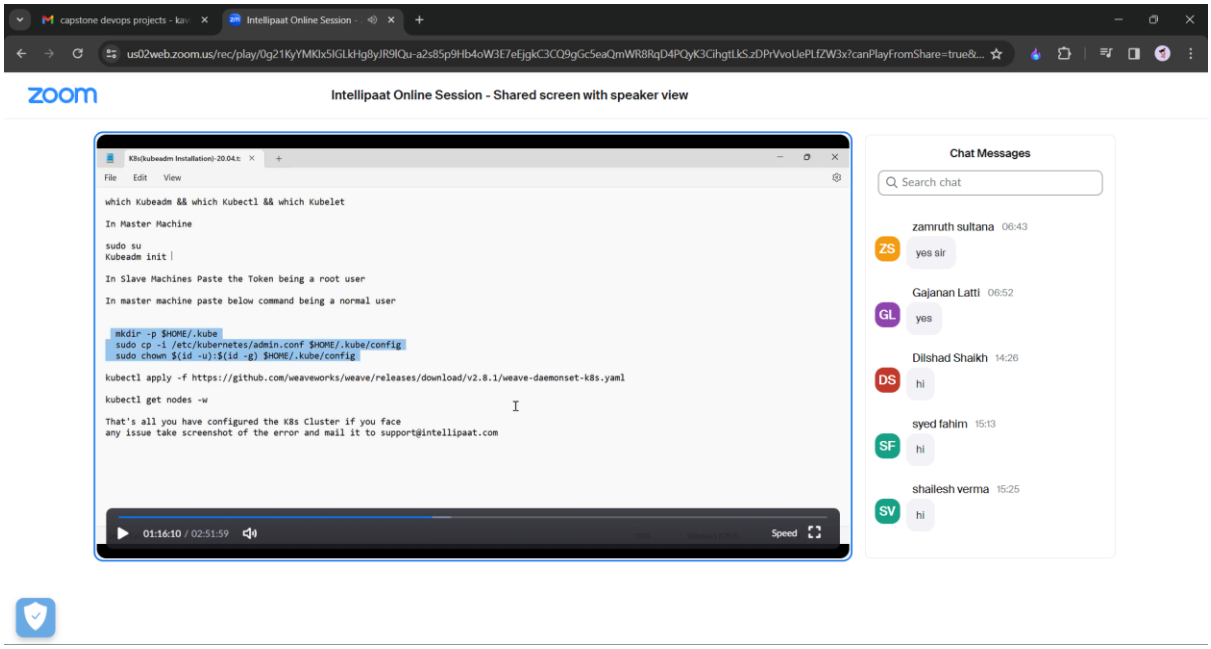
- ansible-playbook play.yaml –check

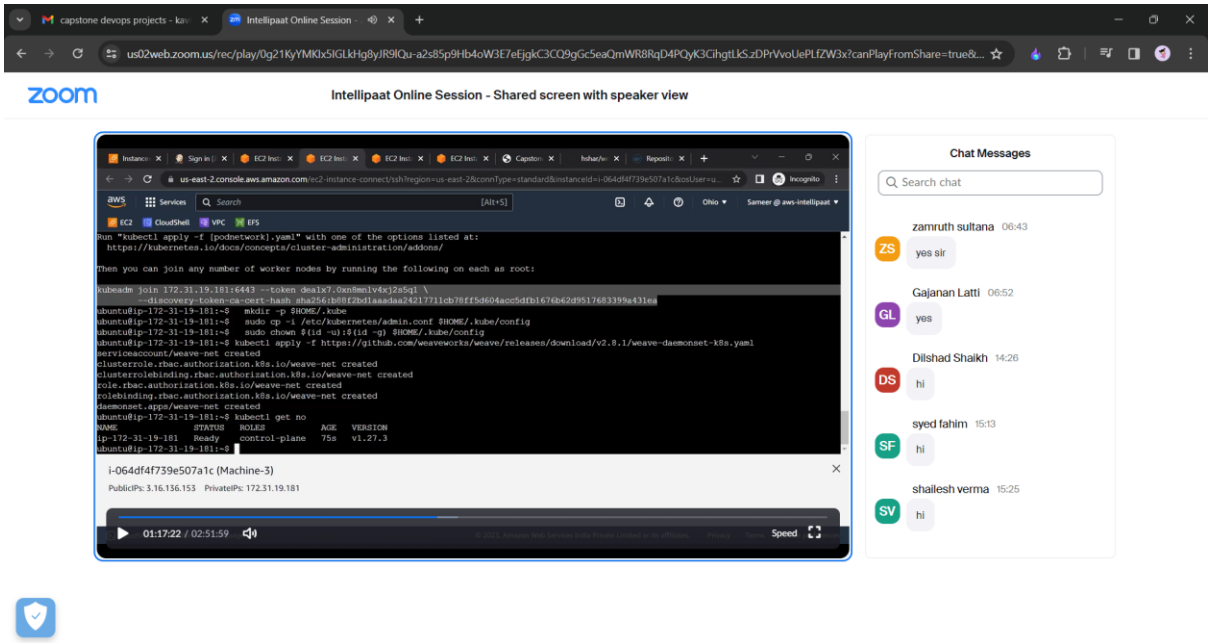
```
sudo apt update
sudo apt upgrade -y
sudo apt install -y curl apt-transport-https ca-certificates
software-properties-common
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo
apt-key add -
sudo add-apt-repository "deb http://apt.kubernetes.io/ kubernetes-
xenial main"
sudo swapoff -a
sudo apt update
sudo apt install -y kubelet kubeadm kubectl
```

- ansible-playbook play.yaml

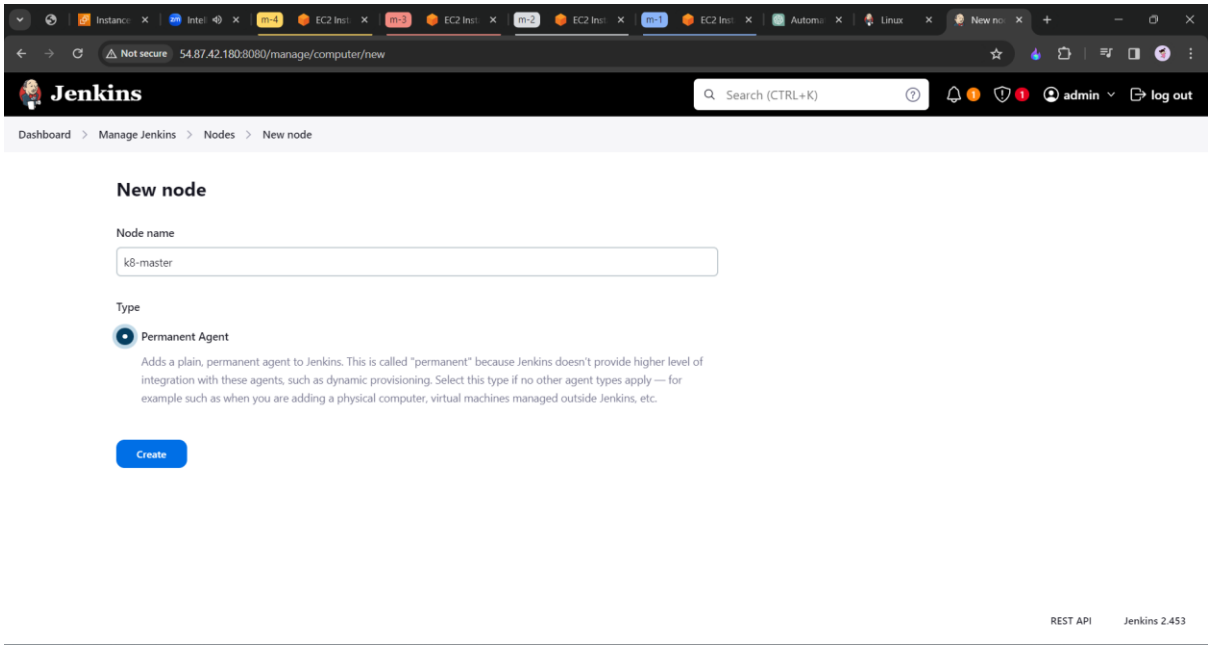


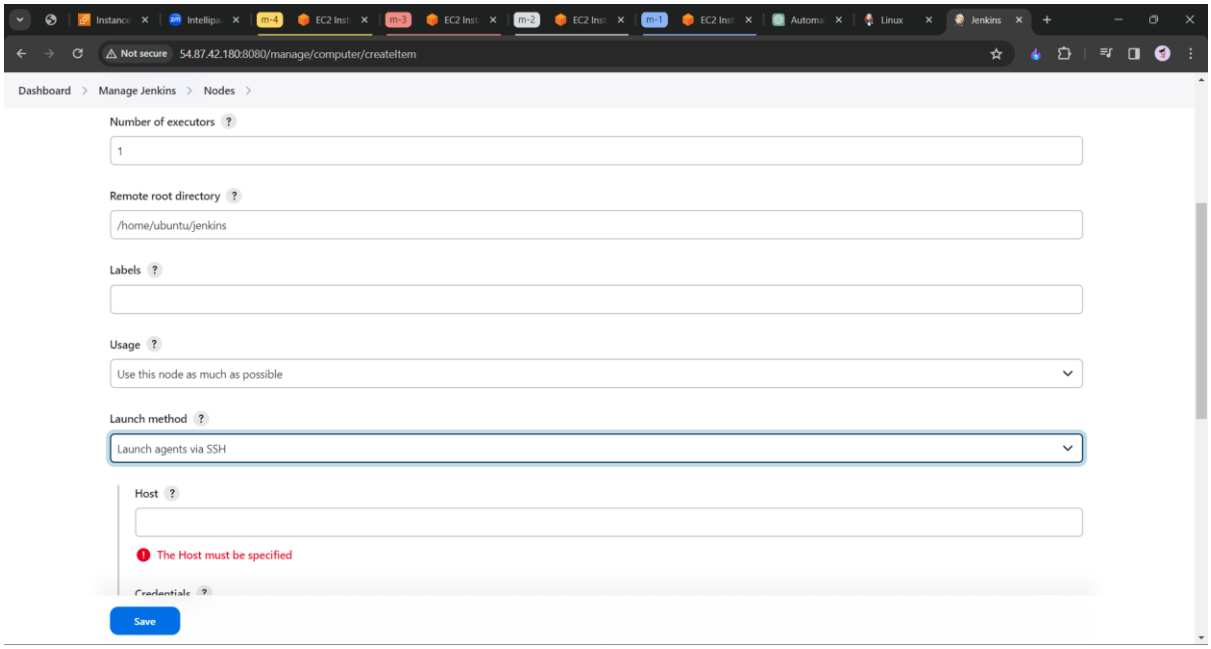
- Now on the master-3 machine we need to run the command “kubeadm init” and then paste the following commands into the slaves in order to join the cluster.

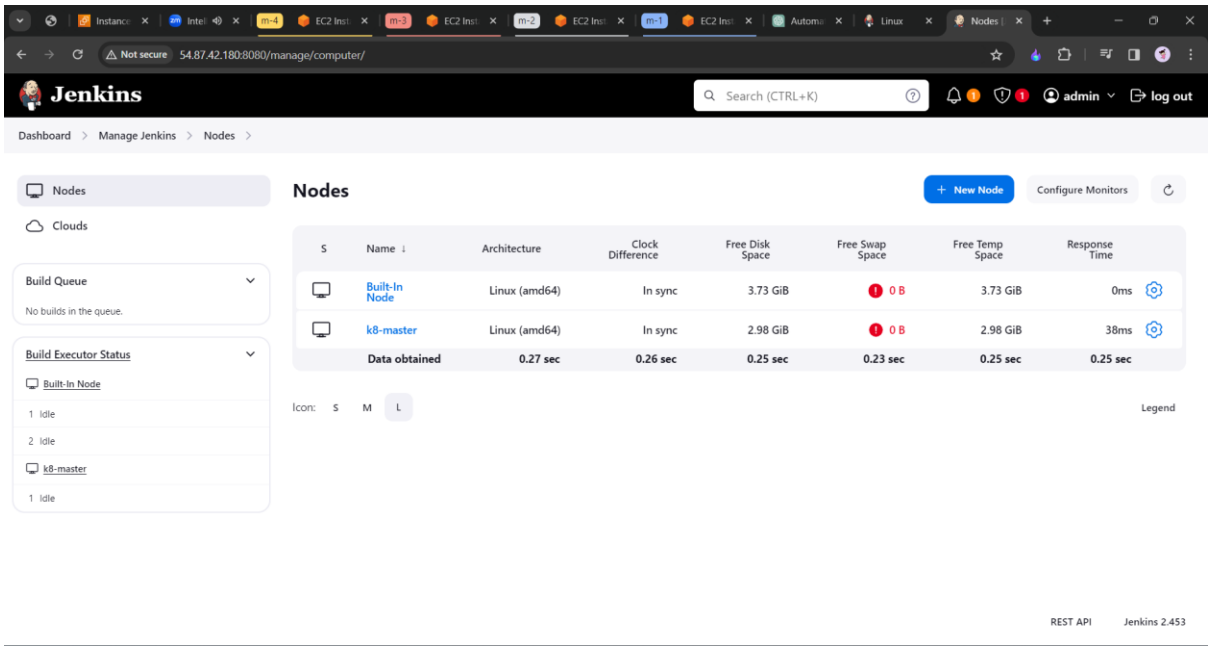




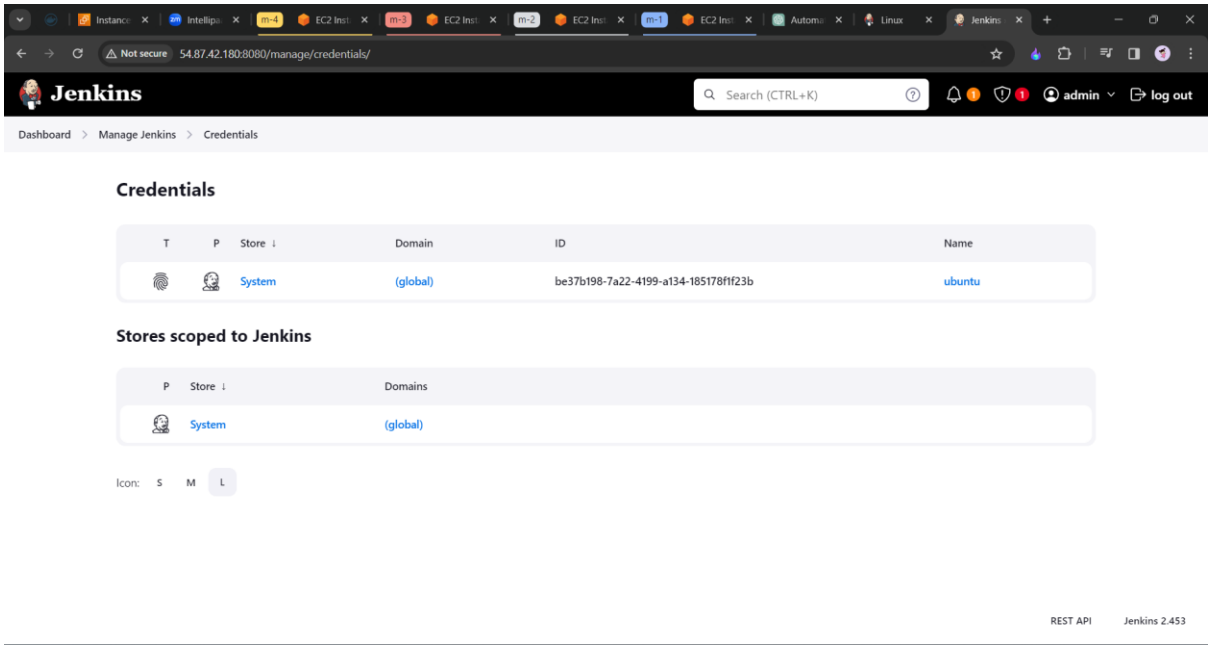
- Copy the generated token from the above command and paste it to the slave machine (machine-2,master-4)
- Configure the jenkins now.
- Add the nodes with the names k8s-master(mater-3),



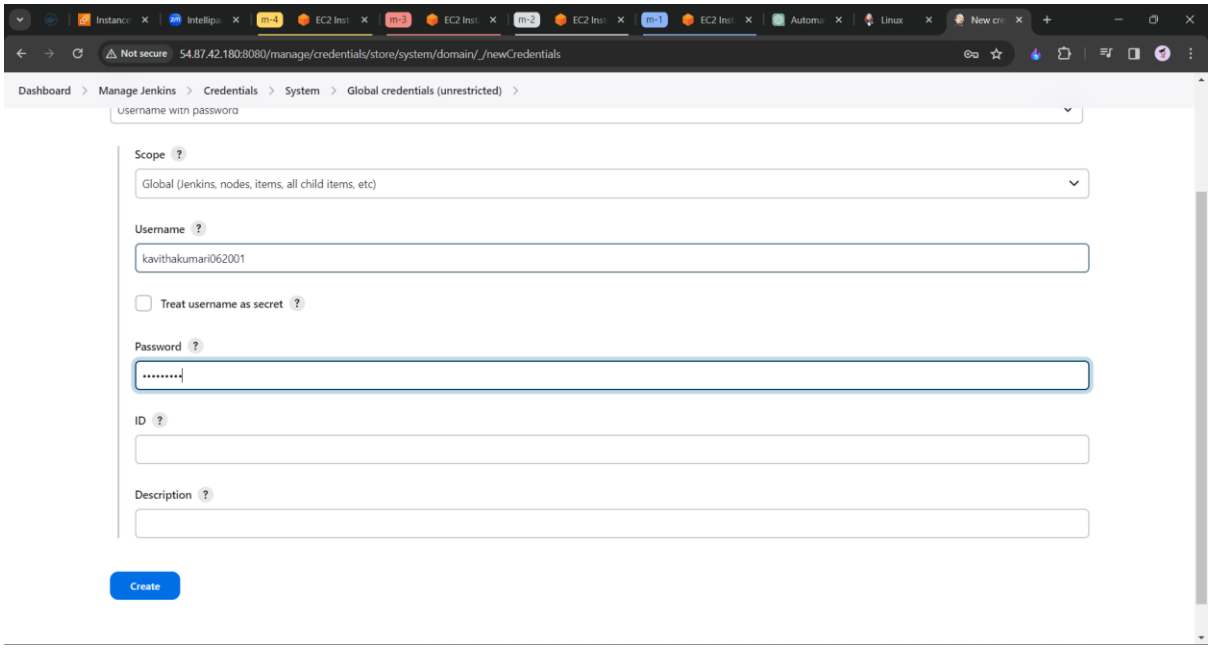




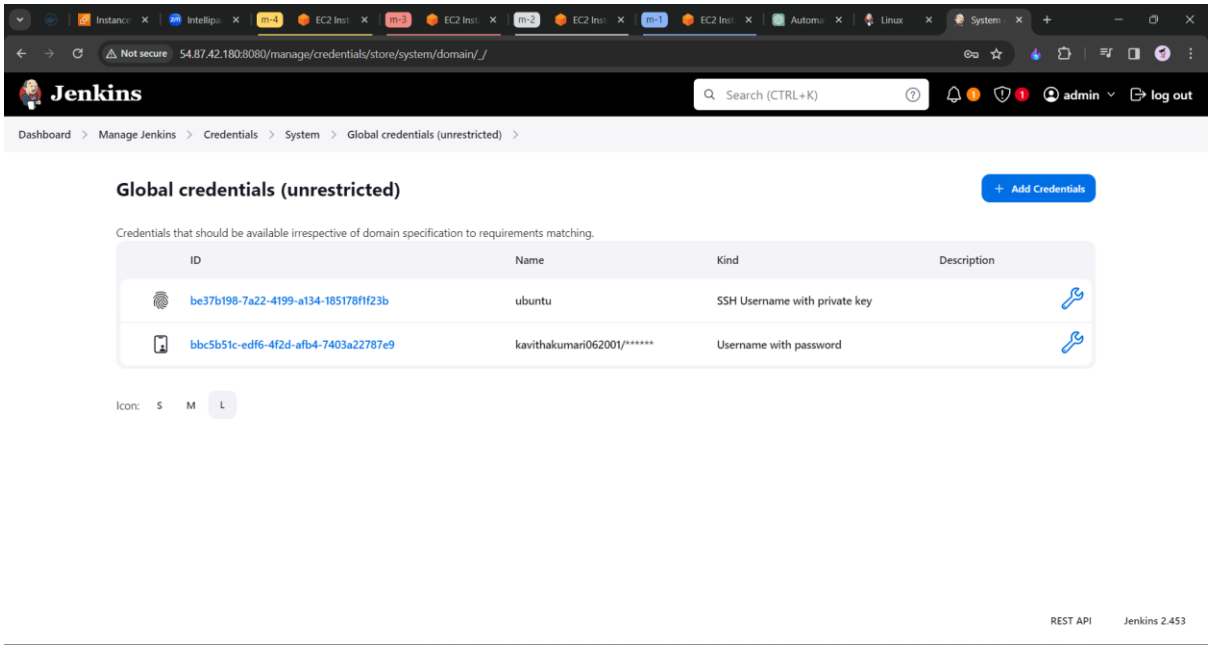
- **Note:** *we can also use the key mentioned in the "id_rsa" to include the node*
- **We need to create credential in the jenkins.**



- Click on global and add credentials



- Give the username and password of the dockerhub account. And create.

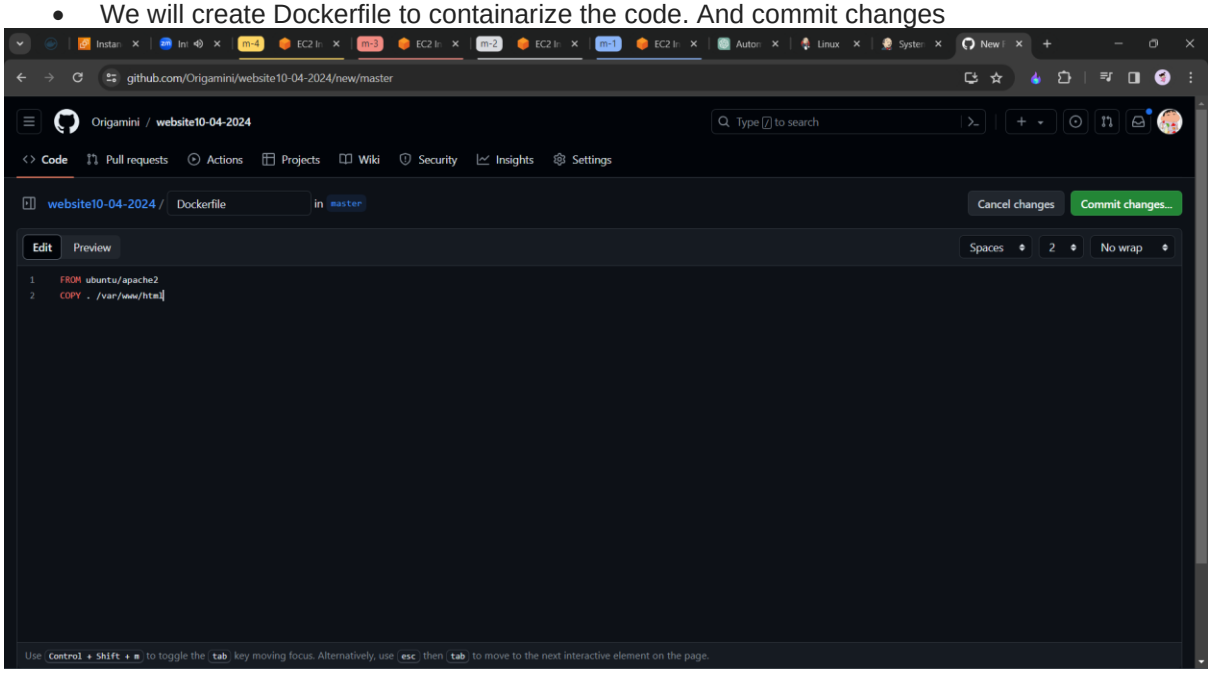


- Goto the path on creating credentials click on global>>add credentials>>username(which is present in the **dockerhub)>>**password(which is used to login in the **dockerhub)>>**click on create
- Fork the repository give by the project. Create Dockerfile

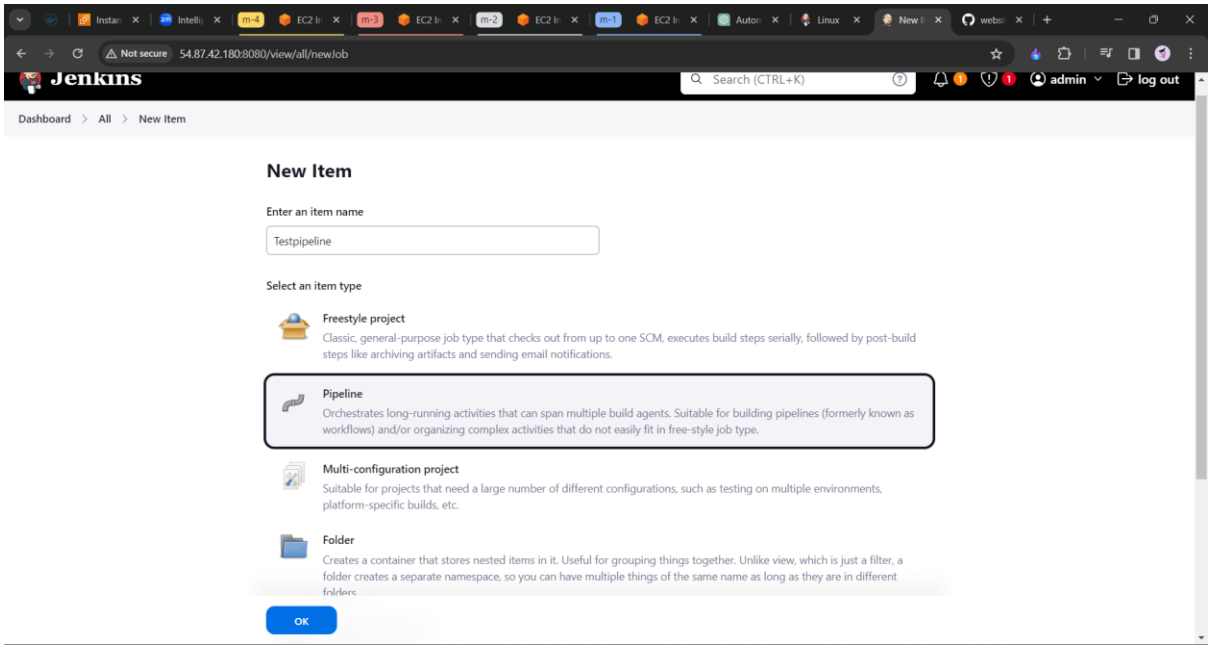
```
FROM ubuntu/apache2
COPY . /var/www/html
```

- Commit changes
- Goto jenkins create a job with the pipeline selected and we need to write script for the pipeline.

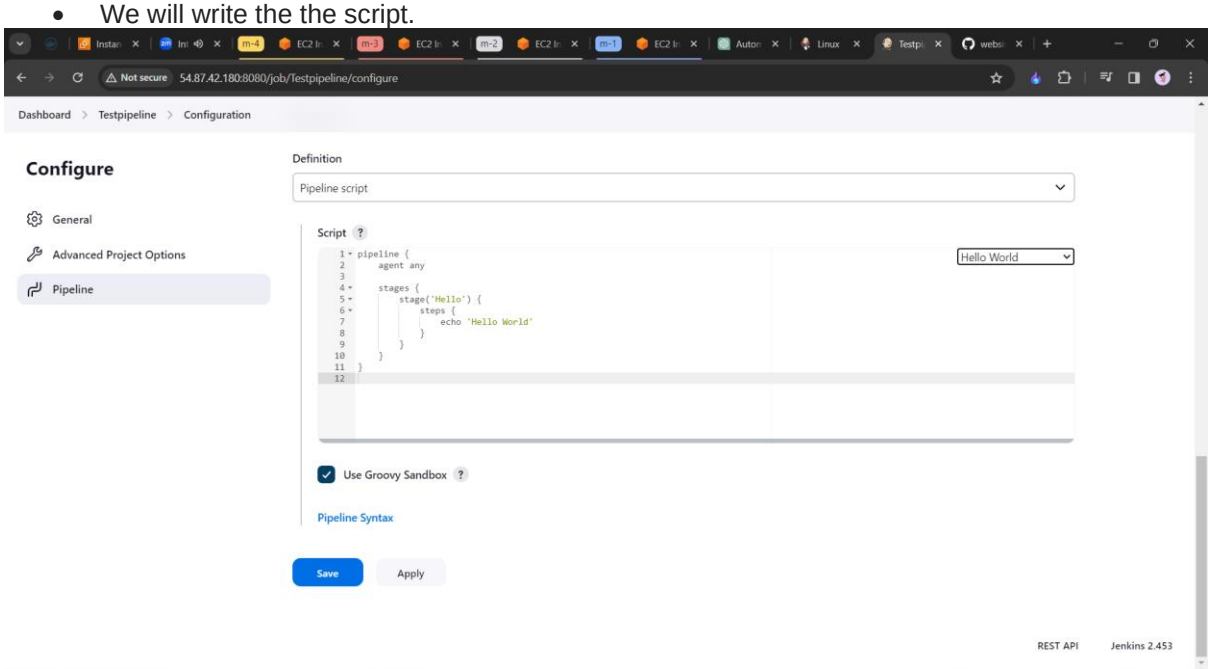
- https://github.com/hshar/website open the repository and fork it
- We will create Dockerfile to containarize the code. And commit changes

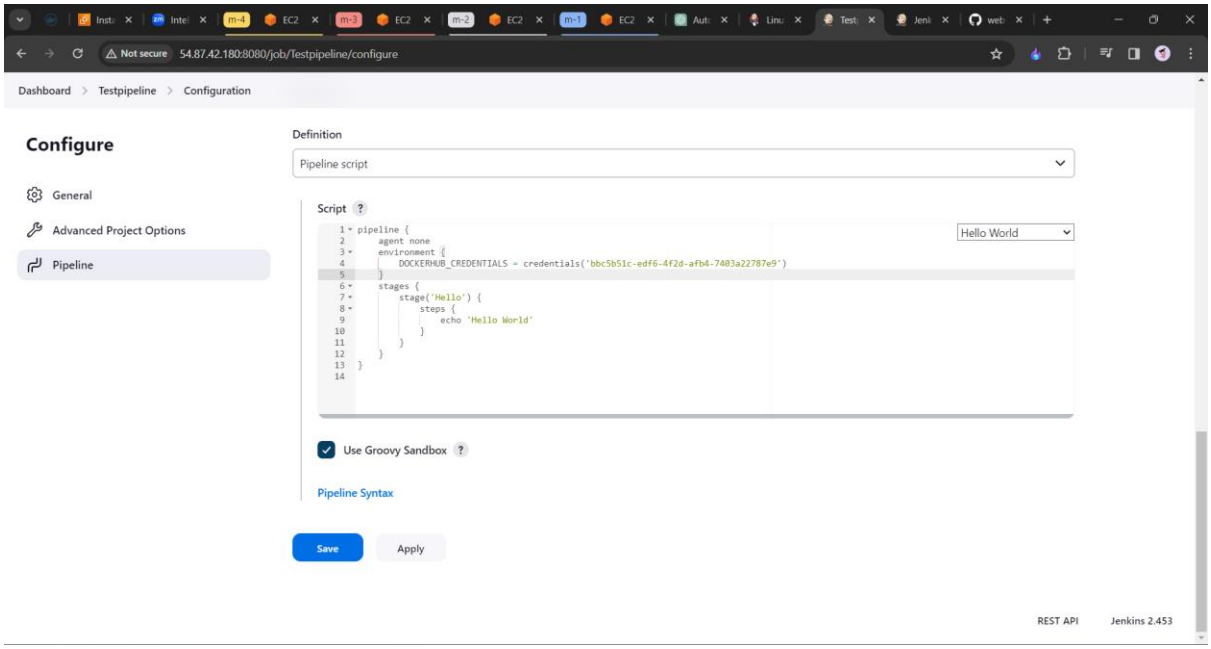


- Come to jenkins and create a pipeline.

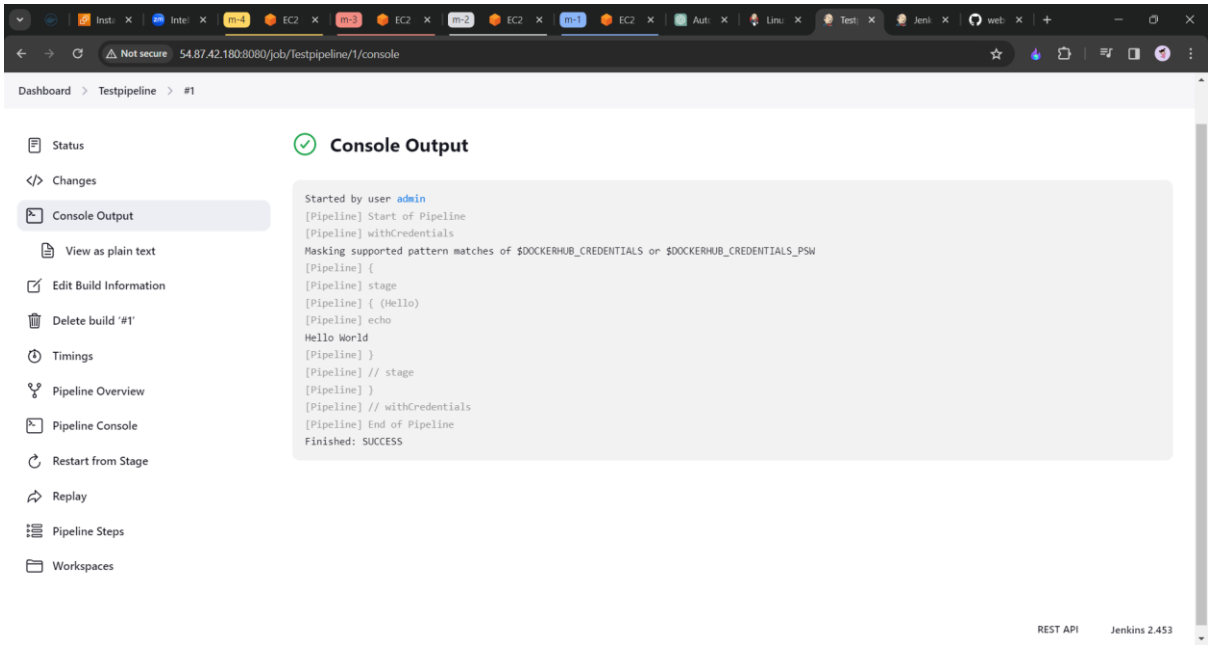


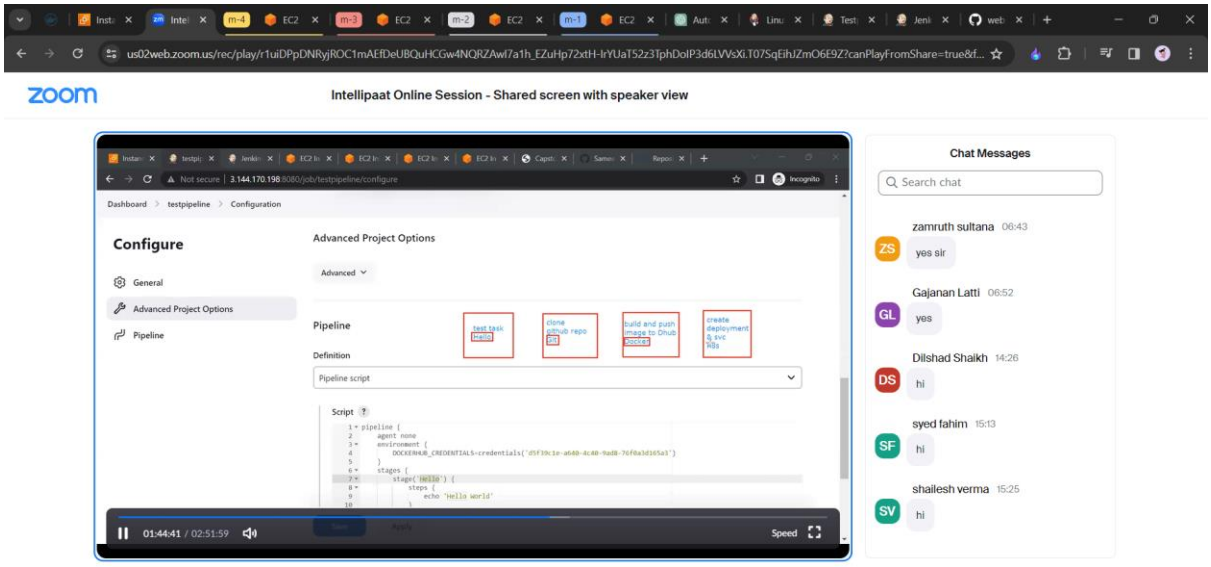
- We will write the the script.

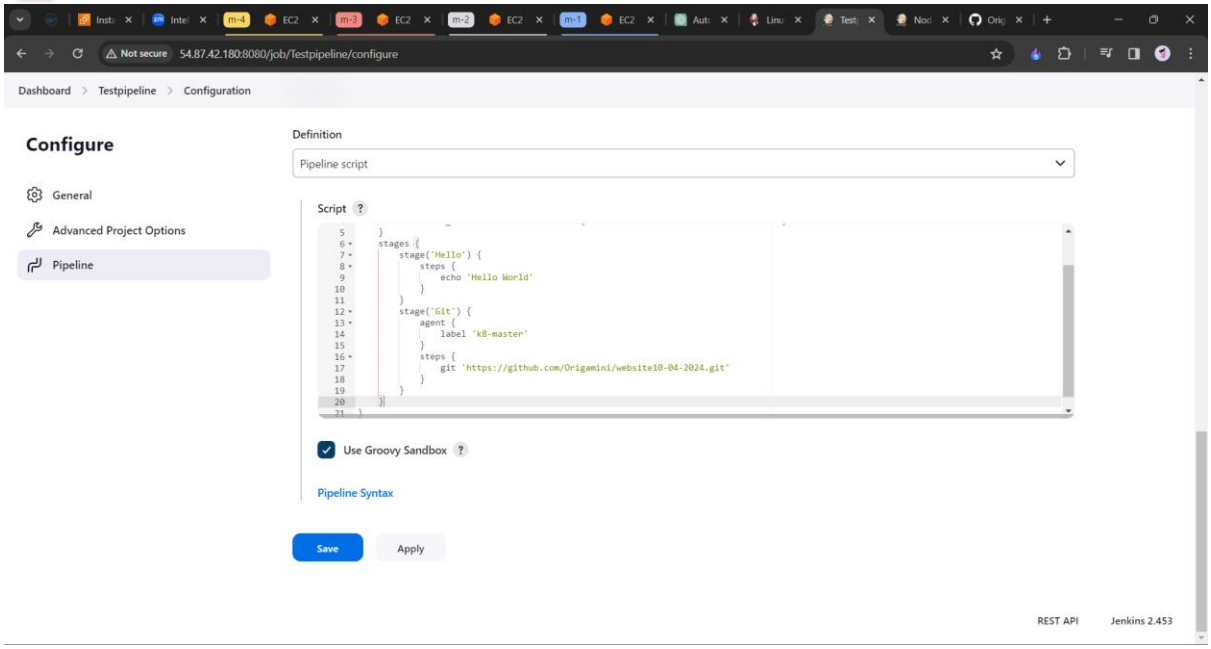




- Click on save




us02web.zoom.us/rec/play/r1uiDPpDNRyjROC1mAEfDeUBQuHCGw4NQRZAwl7a1h_EZuHp72xtH-IrYUaT52z3TphDoIP3d6LVVsXi.T07SqEihUZmO6E9Z?canPlayFromShare=true&f...
zoom
Intellipaat Online Session - Shared screen with speaker view
Dashboard > testpipeline > Configuration
Configure
General
Advanced Project Options
Pipeline
Advanced Project Options
Advanced
Pipeline
Definition
Pipeline script
Script
01:44:41 / 02:51:59
Speed
Chat Messages
Search chat
zamruth sultana 06:43
yes sir
Gajanan Latti 06:52
yes
Dilshad Shaikh 14:26
hi
syed fahim 15:13
hi
shailesh verma 15:25
hi


Not secure 54.87.42.180:8080/job/Testpipeline/configure
Dashboard > Testpipeline > Configuration
Configure
General
Advanced Project Options
Pipeline
Definition
Pipeline script
Script
stages {
stage('Hello') {
steps {
echo 'Hello World'
}
}
stage('Git') {
agent {
label 'k8-master'
}
steps {
git 'https://github.com/Origamini/website10-04-2024.git'
}
}
}
Use Groovy Sandbox
Pipeline Syntax
Save
Apply
REST API
Jenkins 2.453

Not secure 54.87.42.180:8080/job/Testpipeline/3/console

Dashboard > Testpipeline > #3

- Git Build Data
- Pipeline Overview
- Pipeline Console
- Restart from Stage
- Replay
- Pipeline Steps
- Workspaces
- Previous Build

```
[Pipeline] stage
[Pipeline] { (Git)
[Pipeline] node
Running on k8-master in /home/ubuntu/jenkins/workspace/Testpipeline
[Pipeline] {
[Pipeline] git
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Avoid second fetch
Checking out Revision c7898e09c9d96585997127139d23faf8d8190d9c (refs/remotes/origin/master)
Cloning repository https://github.com/Origamini/website10-04-2024.git
 > git init /home/ubuntu/jenkins/workspace/Testpipeline # timeout=10
Fetching upstream changes from https://github.com/Origamini/website10-04-2024.git
 > git --version # timeout=10
 > git --version # 'git version 2.25.1'
 > git fetch --tags --force --progress -- https://github.com/Origamini/website10-04-2024.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://github.com/Origamini/website10-04-2024.git # timeout=10
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
 > git config core.sparsecheckout # timeout=10
 > git checkout -f c7898e09c9d96585997127139d23faf8d8190d9c # timeout=10
 > git branch -a -v --no-abbrev # timeout=10
 > git checkout -b master c7898e09c9d96585997127139d23faf8d8190d9c # timeout=10
Commit message: "Create Dockerfile"
First time build. Skipping changelog.
[Pipeline] }
[Pipeline] // node
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
```

us-east-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=us-east-1&connType=standard&instanceId=i-07d48019716fef42c&osUser=ubuntu&sshPort=22#/

```
clusterrolebinding.rbac.authorization.k8s.io/calico-node created
clusterrolebinding.rbac.authorization.k8s.io/calico-cni-plugin created
daemonset.apps/calico-node created
deployment.apps/calico-kube-controllers created
ubuntu@ip-10-0-9-166:~$
ubuntu@ip-10-0-9-166:~$ kubectl get nodes
NAME              STATUS     ROLES           AGE     VERSION
ip-10-0-14-155    Ready      <none>          104s    v1.28.8
ip-10-0-9-166     NotReady   control-plane   4m21s   v1.28.8
ip-10-0-9-67      NotReady   <none>          92s     v1.28.8
ubuntu@ip-10-0-9-166:~$ kubectl get nodes
NAME              STATUS   ROLES           AGE     VERSION
ip-10-0-14-155    Ready    <none>          3m2s    v1.28.8
ip-10-0-9-166     Ready    control-plane   5m39s   v1.28.8
ip-10-0-9-67      Ready    <none>          2m50s   v1.28.8
ubuntu@ip-10-0-9-166:~$ ls
calico.yaml  jenkins
ubuntu@ip-10-0-9-166:~$ cd jenkins
ubuntu@ip-10-0-9-166:~/jenkins$ ls
remoting  remoting.jar  workspace
ubuntu@ip-10-0-9-166:~/jenkins$ cd workspace
ubuntu@ip-10-0-9-166:~/jenkins/workspace$ ls
Testpipeline
ubuntu@ip-10-0-9-166:~/jenkins/workspace$ cd Testpipeline
ubuntu@ip-10-0-9-166:~/jenkins/workspace/Testpipeline$ ls
Dockerfile  images  index.html
ubuntu@ip-10-0-9-166:~/jenkins/workspace/Testpipeline$
```

i-07d48019716fef42c (machine-3)

PublicIPs: 52.204.146.34 PrivateIPs: 10.0.9.166

Not secure 54.87.42.180:8080/job/Testpipeline/configure

Dashboard > Testpipeline > Configuration

Configure

- General
- Advanced Project Options
- Pipeline

Pipeline

Definition

Pipeline script

Script

```
        }
        steps {
            git 'https://github.com/Origamini/website10-04-2024.git'
        }
    }
    stage('Docker') {
        agent {
            label 'k8-master'
        }
        steps {
            sh 'sudo docker build /home/ubuntu/jenkins/workspace/Testpipeline -t kavithakumari062001/project2'
            sh 'sudo echo $DOCKERHUB_CREDENTIALS_PSW | sudo docker login -u $DOCKERHUB_CREDENTIALS_USR --password-stdin'
            sh 'sudo docker push kavithakumari062001/project2'

        }
    }
}
```

Use Groovy Sandbox

Pipeline Syntax

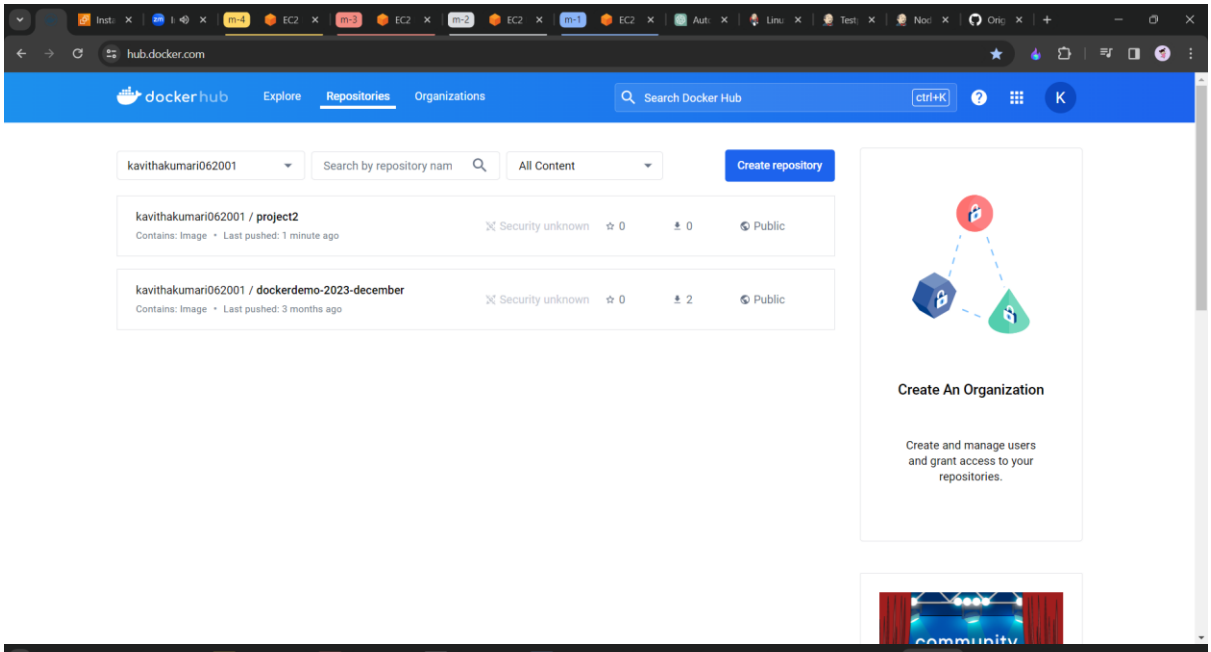
Save Apply

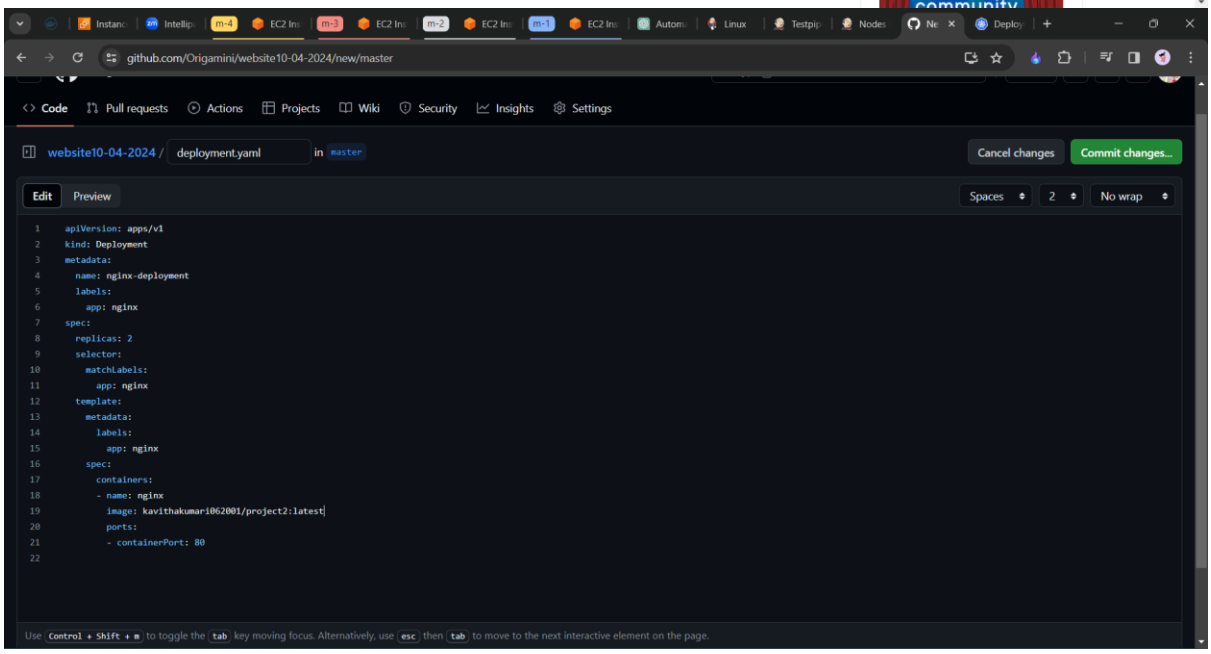
Not secure 54.87.42.180:8080/job/Testpipeline/4/console

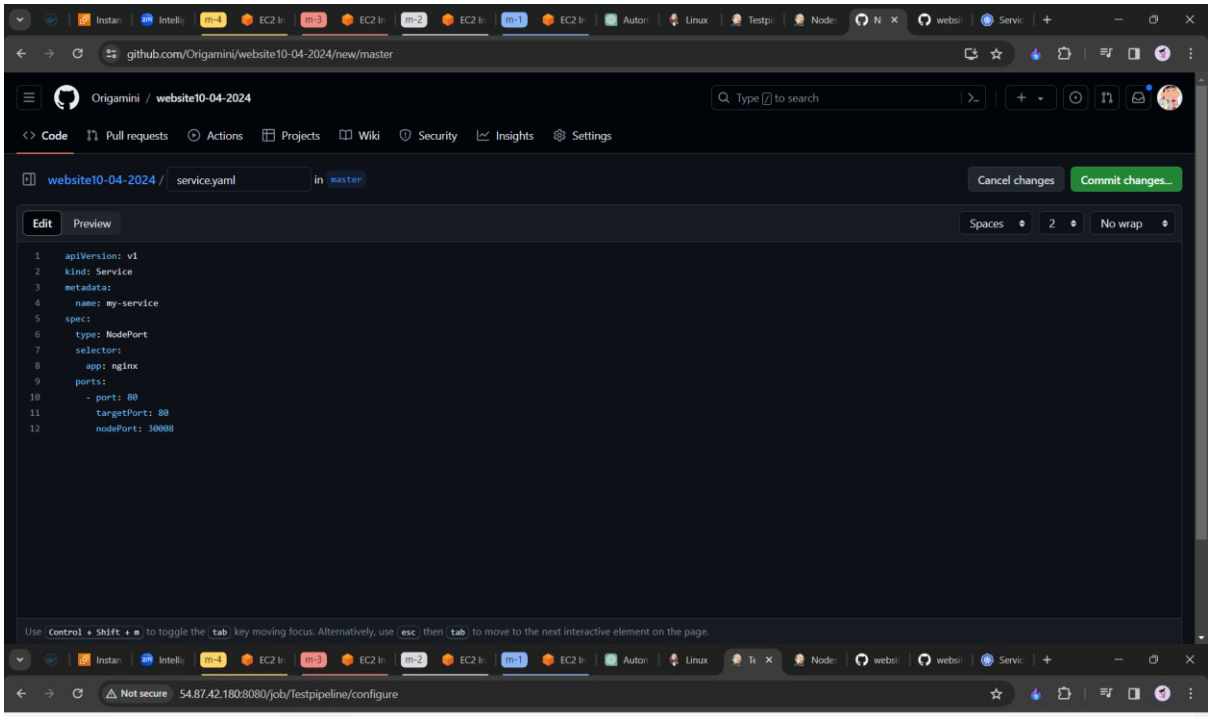
Dashboard > Testpipeline > #4

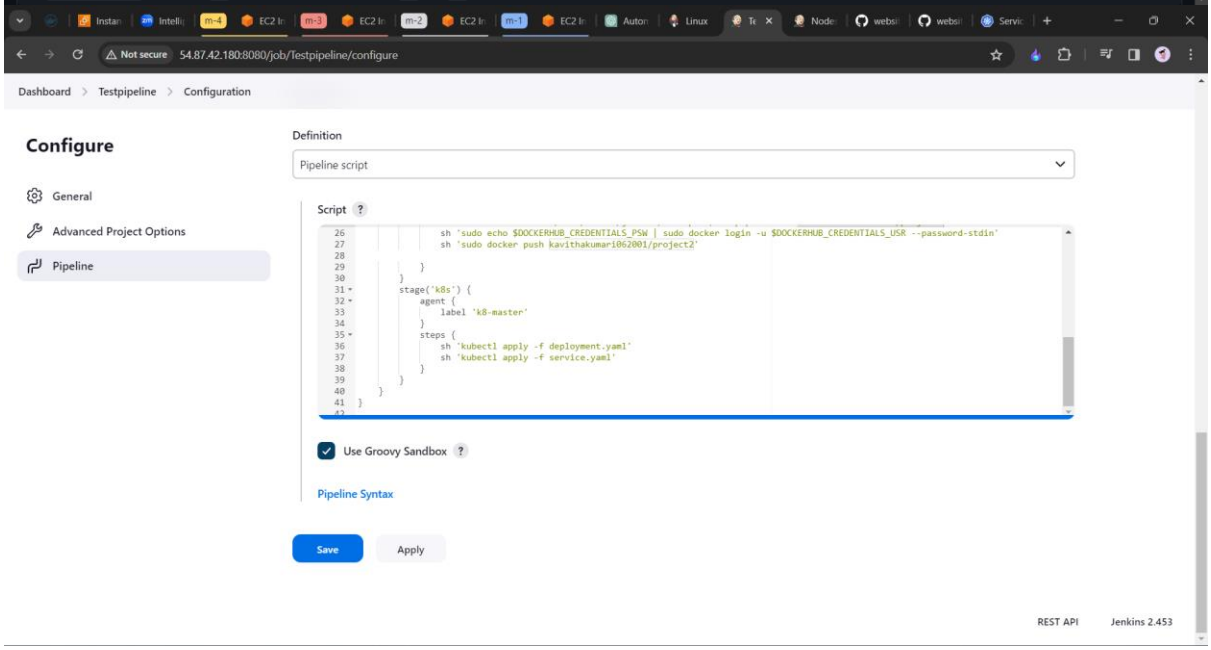
```
+ sudo echo ****
+ sudo docker login -u kavithakumari062001 --password-stdin
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
[Pipeline] sh
+ sudo docker push kavithakumari062001/project2
Using default tag: latest
The push refers to repository [docker.io/kavithakumari062001/project2]
5e807553f311: Preparing
5e43e67dce9f: Preparing
cdd2aaee2975: Preparing
a2d6ae7d60c7: Preparing
cdd2aaee2975: Mounted from ubuntu/apache2
5e43e67dce9f: Mounted from ubuntu/apache2
a2d6ae7d60c7: Mounted from ubuntu/apache2
5e807553f311: Pushed
latest: digest: sha256:35892c153b5eacecf4e3ce9999d3d843cc350f591c7c1adc3e2c048d57a7740f size: 1158
[Pipeline] }
[Pipeline] // node
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withCredentials
[Pipeline] End of Pipeline
Finished: SUCCESS
```

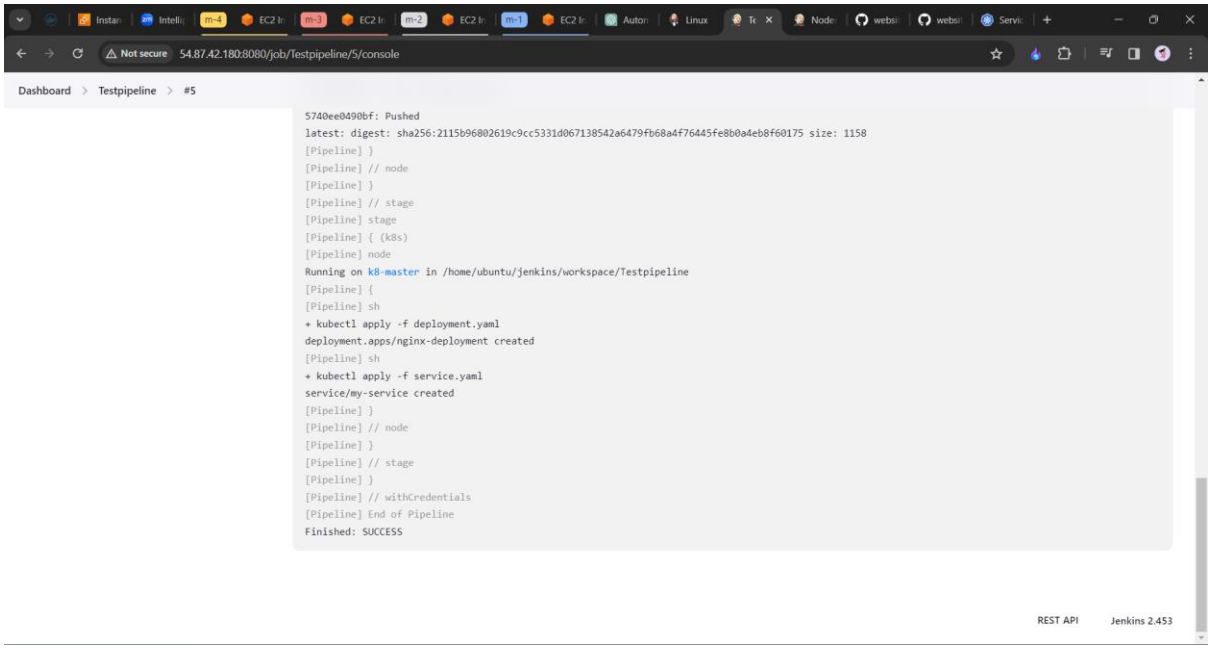

hub.docker.com
dockerhub
Explore
Repositories
Organizations
Search Docker Hub
ctrl+K
kavithakumari062001
Search by repository nam
All Content
Create repository
kavithakumari062001 / project2
Contains: Image • Last pushed: 1 minute ago
Security unknown
0
0
Public
kavithakumari062001 / dockerdemo-2023-december
Contains: Image • Last pushed: 3 months ago
Security unknown
0
2
Public
Create An Organization
Create and manage users and grant access to your repositories.


github.com/Origamini/website10-04-2024/new/master
Code
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
website10-04-2024 /
deployment.yaml
in master
Cancel changes
Commit changes...
Edit
Preview
Spaces
2
No wrap
apiVersion: apps/v1
kind: Deployment
metadata:
name: nginx-deployment
labels:
app: nginx
spec:
replicas: 2
selector:
matchLabels:
app: nginx
template:
metadata:
labels:
app: nginx
spec:
containers:
- name: nginx
image: kavithakumari062001/project2:latest
ports:
- containerPort: 80
Use Control + Shift + m to toggle the tab key moving focus. Alternatively, use esc then tab to move to the next interactive element on the page.

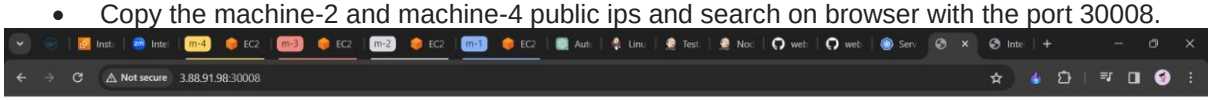




- Save and build the file.

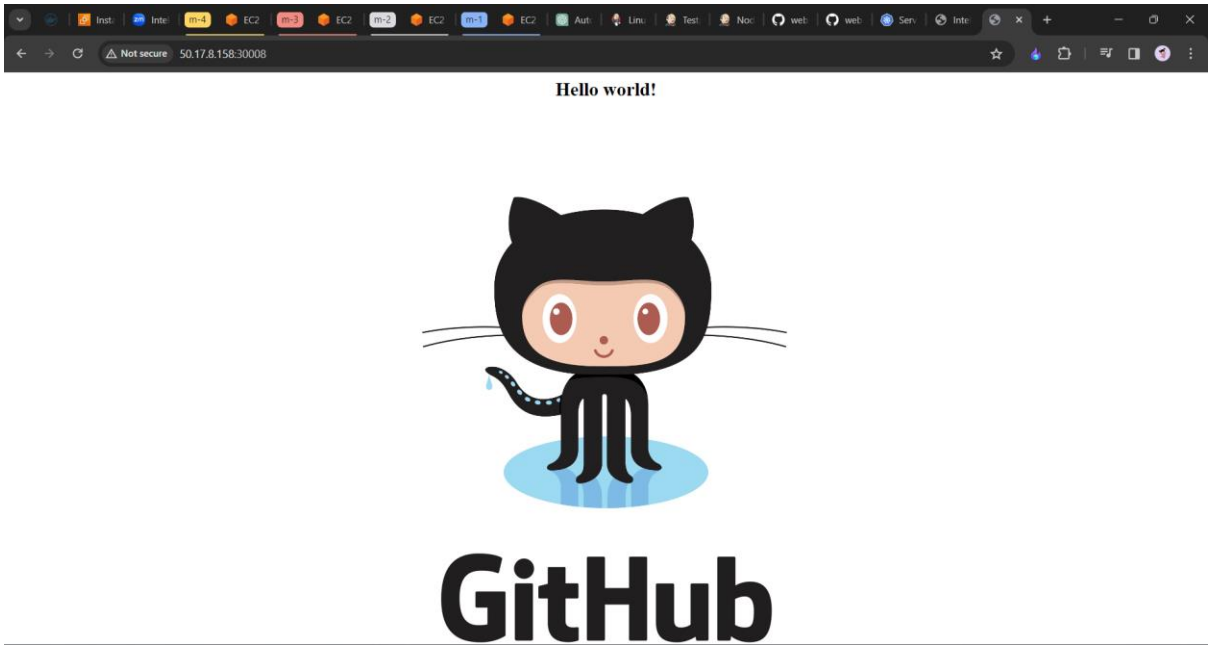


- Copy the machine-2 and machine-4 public ips and search on browser with the port 30008.

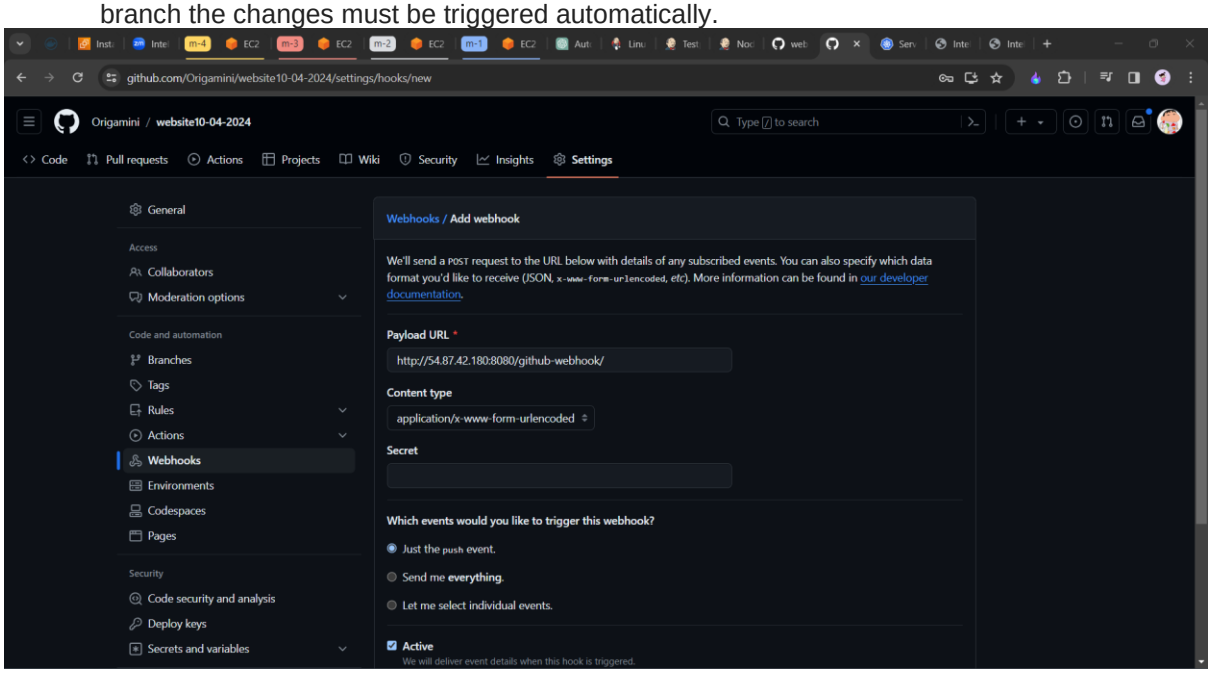


Hello world!

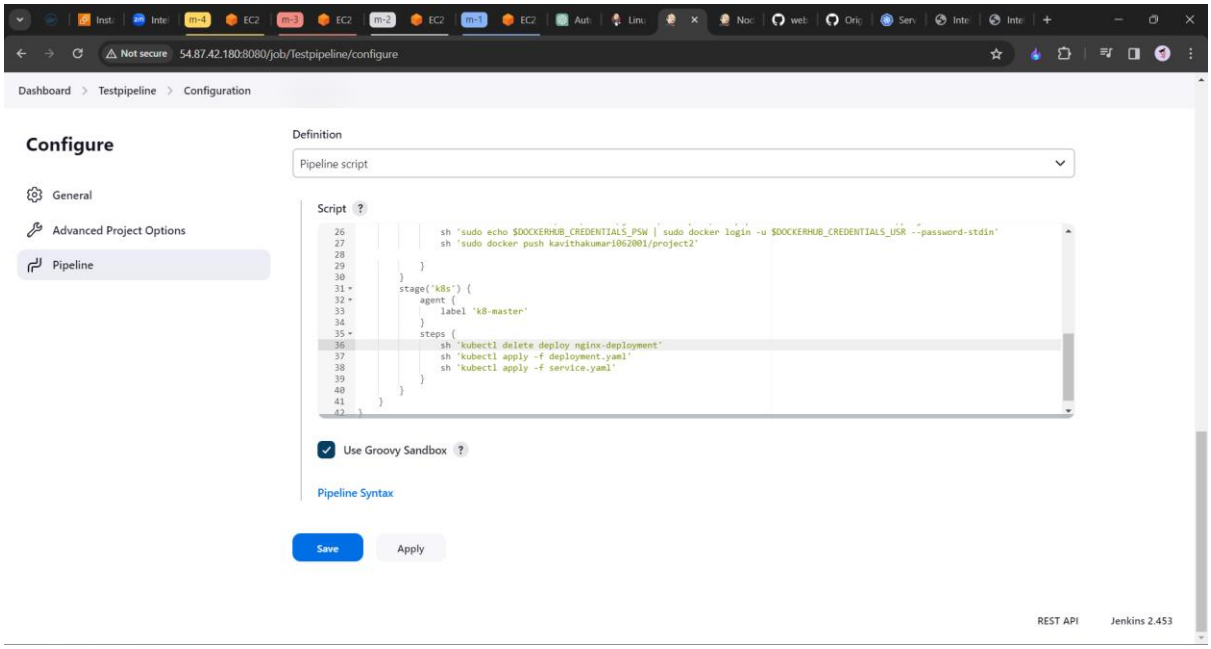
GitHub



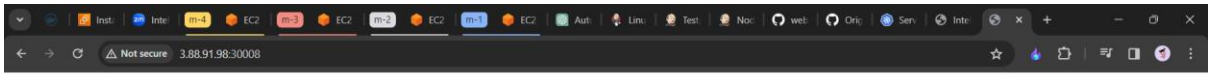
- Now we will automate the project so that when ever the changes are made to the master branch the changes must be triggered automatically.



- Click on add webhooks. Now we will try to make changes in the index.html

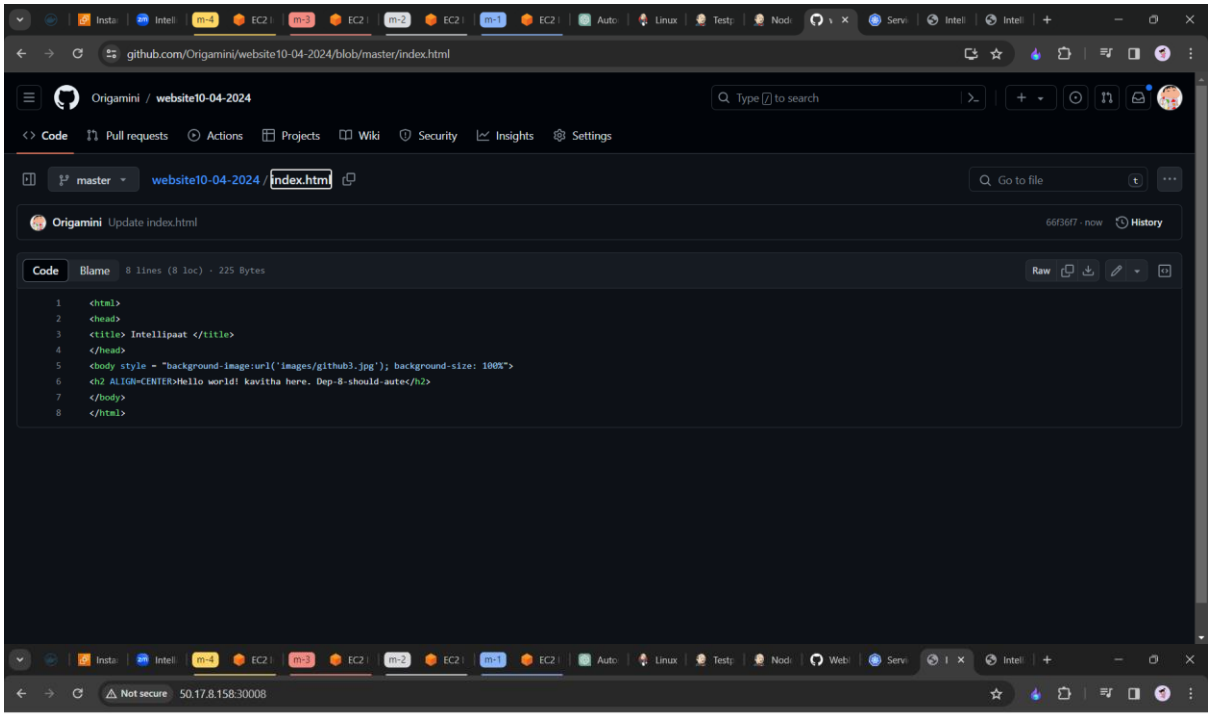


- Save and build now.



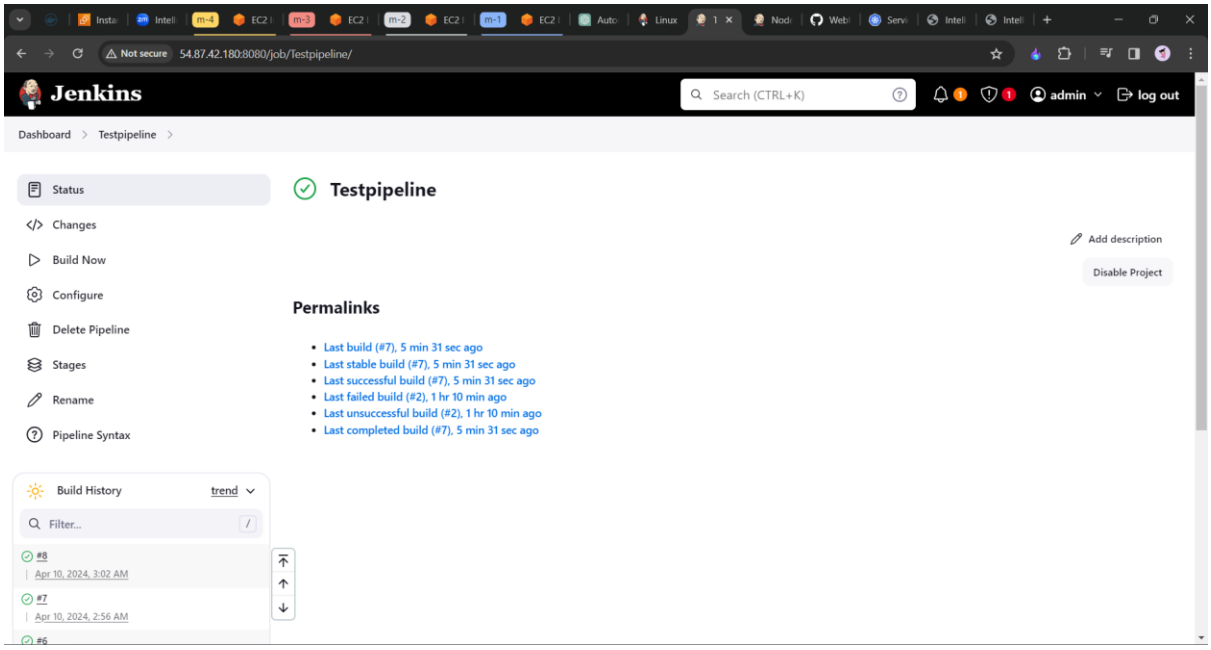
Hello world! kavitha here. Dep-7

- Now If I make changes in my repository. The pipeline will automatically be triggered



Hello world! kavitha here. Dep-8-should-aute

GitHub



Pipeline: -

```
pipeline {
    agent none
    environment {
        DOCKERHUB_CREDENTIALS = credentials('bbc5b51c-edf6-4f2d-afb4-7403a22787e9')
    }
    stages {
        stage('Hello') {
            steps {
                echo 'Hello World'
            }
        }
        stage('Git') {
            agent {
                label 'k8-master'
            }
            steps {
                git 'https://github.com/Origamini/website10-04-2024.git'
            }
        }
        stage('Docker') {
            agent {
                label 'k8-master'
            }
            steps {
                sh 'sudo docker build /home/ubuntu/jenkins/workspace/Testpipeline -t kavithakumari062001/project2'
                sh 'sudo echo $DOCKERHUB_CREDENTIALS_PSW | sudo docker login -u $DOCKERHUB_CREDENTIALS_USR --password-stdin'
                sh 'sudo docker push kavithakumari062001/project2'
```

```
            }
        }
        stage('k8s') {
            agent {
                label 'k8-master'
            }
            steps {
                sh 'kubectl delete deploy nginx-deployment'
                sh 'kubectl apply -f deployment.yaml'
                sh 'kubectl apply -f service.yaml'
            }
        }
    }
}
```

----------END----------