

# Devops Capstone Project2

## Create a host Machine

**Launch an instance**

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

**Name and tags**

Name: HostMachine

**Application and OS Images (Amazon Machine Image)**

Search your full catalog including 1000s of application and OS images

Amazon Linux, macOS, Ubuntu, Windows, Red Hat, SUSE Linux

Amazon Machine Image (AMI): Ubuntu Server 22.04 LTS (HVM), 550 Volume Type

**Instance type**

Instance type: t2.medium

**Key pair (login)**

Key pair name: NewKey09Sep2024

**Network settings**

Network: vpc-03316fc1240e158f8 | VPC

Subnet: No preference (Default subnet in any availability zone)

Auto-assign public IP: Enable

Firewall (security group): default

Storage (volumes): 1 volume(s) - 8 GiB

**Summary**

Number of instances: 1

Software Image (AMI): Canonical, Ubuntu, 22.04 LTS

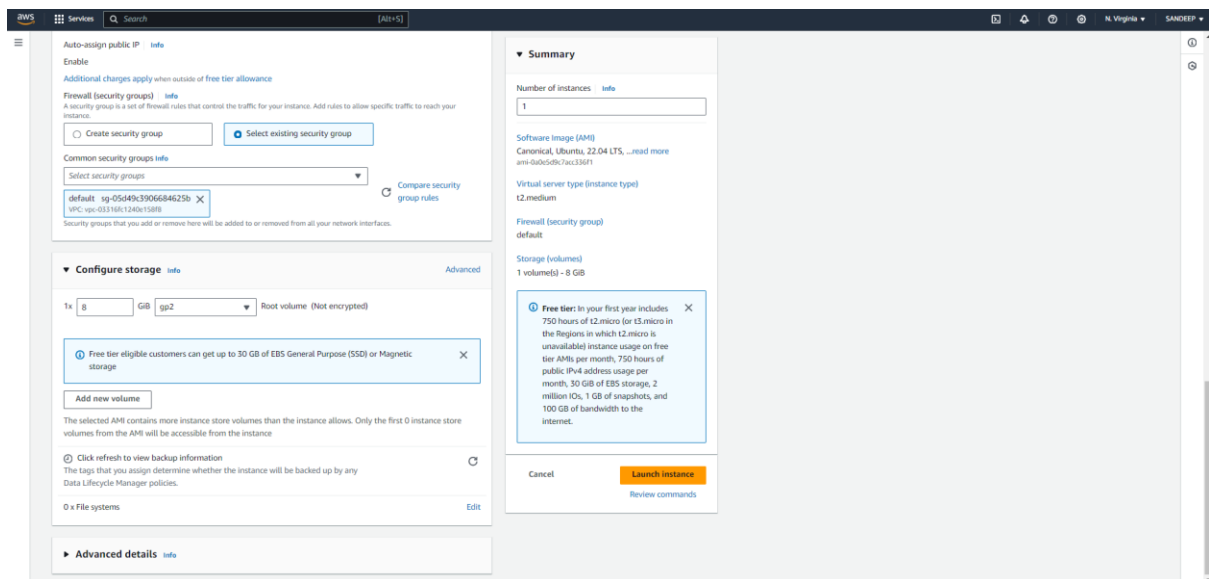
Virtual server type (instance type): t2.medium

Firewall (security group): default

Storage (volumes): 1 volume(s) - 8 GiB

**Free tier:** In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million I/Os, 1 GiB of snapshots, and 100 GiB of bandwidth to the internet.

**Launch instance**



Sudo apt-get update

Install Terraform

```
wget -O- https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o
/usr/share/keyrings/hashicorp-archive-keyring.gpg
```

```
echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg]
https://apt.releases.hashicorp.com $(lsb_release -cs) main" | sudo tee
/etc/apt/sources.list.d/hashicorp.list
```

```
sudo apt update && sudo apt install terraform
```

```
ubuntu@ip-172-31-91-170:~$ terraform --version
Terraform v1.9.5
on linux_amd64
ubuntu@ip-172-31-91-170:~$
```

```
i-0946d6e4167cbb958 (HOSTMachine)
PublicIPs: 18.206.56.70 PrivateIPs: 172.31.91.170
```

Sudo nano main.tf

```
provider "aws" {
  region = "us-east-1"
  access_key = "AKIA3FLD5QD2AMOVAPXG"
  secret_key = "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"
}

resource "aws_instance" "K8s-Slave1" {
  ami = "ami-0a0e5d9c7acc336f1"
```

```

instance_type = "t2.medium"
key_name = "Newkey09Sep2024"
tags = {
    Name = "m2-k8s-slave1"
}
}

resource "aws_instance" "K8s-master" {
    ami = "ami-0a0e5d9c7acc336f1"
    instance_type = "t2.medium"
    key_name = "Newkey09Sep2024"
    tags = {
        Name = "m3-k8s-master"
    }
}

resource "aws_instance" "K8s-Slave2" {
    ami = "ami-0a0e5d9c7acc336f1"
    instance_type = "t2.medium"
    key_name = "Newkey09Sep2024"
    tags = {
        Name = "m4-k8s-slave2"
    }
}

```

save and exit

- 5- terraform init
- 6-terraform plan
- 7- terraform apply
- 8- sudo apt install software-properties-common (install ansible on ubuntu)
- 9- sudo add-apt-repository --yes --update ppa:ansible/ansible
- 10- sudo apt install ansible

Three Slave Machine has been created through terraform script.

Now Go to Host Machine

Now connect m2-k8s-slave1 & m3-k8s-&master & m4-k8s-slave2  
run below command

- 1-sudo apt update

Now go to Sandeep-master

```
cd .ssh
ssh-keygen
enter
enter
enter
ls
authorized_keys id_rsa id_rsa.pub
```

2- sudo cat id\_rsa.pub  
it will display id code

3- Now in m2-k8s slave1  
cd .ssh  
sudo nano authorized\_keys  
Paste the id\_rsa.pub code

Now in m3-k8s-master  
cd .ssh  
sudo nano authorized\_keys  
Paste the id\_rsa.pub code

Now in m4-k8s-slave2  
cd .ssh  
sudo nano authorized\_keys  
Paste the id\_rsa.pub code

CTRL+s CTRL+ X

now in sandeep-master  
1-cd  
2-cd /etc/ansible  
3-ls  
4- sudo nano hosts

```
ubuntu@ip-172-31-91-170:~/.ssh$ cd /etc/ansible
ubuntu@ip-172-31-91-170:/etc/ansible$ ls
ansible.cfg  hosts  roles
ubuntu@ip-172-31-91-170:/etc/ansible$ |
```

i-0946d6e4167cbb958 (HOSTMachine)

PublicIPs: 18.206.56.70 PrivateIPs: 172.31.91.170

we will make two group one mater and other slaves  
paste all three private ip of m2 m3 & m4

```
aws | Services | Search

GNU nano 6.2

[master]
172.31.91.136
[slaves]
172.31.81.83
172.31.91.84
```

```
ubuntu@ip-172-31-91-170:/etc/ansible$ sudo nano hosts
ubuntu@ip-172-31-91-170:/etc/ansible$ ansible -m ping all
The authenticity of host '172.31.81.83 (172.31.81.83)' can't be established.
ED25519 key fingerprint is SHA256:Lj/U8oIfuzTW1LsD/TzpjLqbUyW/Ljh/3DPfx2Fm9LI.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? The authenticity of host '172.31.91.136 (172.31.91.136)' can't be established.
ED25519 key fingerprint is SHA256:Ph+mE5PoFeAGXeVfmdJ+0rxbeSZafSHTOIwF2J52bbk.
This key is not known by any other names
The authenticity of host '172.31.91.84 (172.31.91.84)' can't be established.
ED25519 key fingerprint is SHA256:LvpoYWuL2+NUkAQ+rn24029F/crX+hivXnVEMWyIv48.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
172.31.81.83 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "ping": "pong"
}
yes
172.31.91.136 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "ping": "pong"
}
yes
172.31.91.84 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "ping": "pong"
}
ubuntu@ip-172-31-91-170:/etc/ansible$ |

i-0946d6e4167cbb958 (HOSTMachine)
PublicIPs: 18.206.56.70 PrivateIPs: 172.31.91.170
```

All three machines connected

```
ansible-playbook play.yaml --syntax-check
ansible-playbook play.yaml --check
ansible-playbook play.yaml
```

```

ubuntu@ip-172-31-91-170:/etc/ansible$ sudo nano script1.sh
ubuntu@ip-172-31-91-170:/etc/ansible$ sudo nano script2.sh
ubuntu@ip-172-31-91-170:/etc/ansible$ sudo nano script3.sh
ubuntu@ip-172-31-91-170:/etc/ansible$ sudo nano play.yaml
ubuntu@ip-172-31-91-170:/etc/ansible$ ansible-playbook play.yaml --syntax -check

playbook: play.yaml
ubuntu@ip-172-31-91-170:/etc/ansible$ ansible-playbook play.yaml --check

```

i-0946d6e4167cbb958 (HOSTMachine)

PublicIPs: 18.206.56.70 PrivateIPs: 172.31.91.170

```

aws Services Q Search [Alt+S]
ubuntu@ip-172-31-91-170:/etc/ansible$ ansible-playbook play.yaml --syntax -check

playbook: play.yaml
ubuntu@ip-172-31-91-170:/etc/ansible$ ansible-playbook play.yaml --check

PLAY [install Jenkins and Java and docker on host] *****
TASK [Gathering Facts] *****
ok: [localhost]

TASK [running script to install tools on host] *****
skipping: [localhost]

PLAY [install K8s and Java on main] *****
TASK [Gathering Facts] *****
ok: [172.31.91.136]

TASK [running script to install tools on main] *****
skipping: [172.31.91.136]

PLAY [install K8s on nodes] *****
TASK [Gathering Facts] *****
ok: [172.31.91.84]
ok: [172.31.81.83]

TASK [running script to install tools on node] *****
skipping: [172.31.81.83]
skipping: [172.31.91.84]

PLAY RECAP *****
172.31.81.83      : ok=1    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
172.31.91.136   : ok=1    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
172.31.91.84    : ok=1    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
localhost       : ok=1    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0

```

```
ubuntu@ip-172-31-91-170:/etc/ansible$ ansible-playbook play.yaml

PLAY [install Jenkins and Java and docker on host] *****

TASK [Gathering Facts] *****
ok: [localhost]

TASK [running script to install tools on host] *****
changed: [localhost]

PLAY [install K8s and Java on main] *****

TASK [Gathering Facts] *****
ok: [172.31.91.136]

TASK [running script to install tools on main] *****
changed: [172.31.91.136]

PLAY [install K8s on nodes] *****

TASK [Gathering Facts] *****
ok: [172.31.91.84]
ok: [172.31.81.83]

TASK [running script to install tools on node] *****
changed: [172.31.91.84]
changed: [172.31.81.83]

PLAY RECAP *****
172.31.81.83      : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
172.31.91.136   : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
172.31.91.84    : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
localhost       : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

ubuntu@ip-172-31-91-170:/etc/ansible$ |
```

i-0946d6e4167cbb958 (HOSTMachine)

PublicIPs: 18.206.56.70 PrivateIPs: 172.31.91.170

Now on m3-K8-master – Run below commands

```
sudo kubeadm config images pull
sudo kubeadm init
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

```
Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

  mkdir -p $HOME/.kube
  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
  sudo chown $(id -u):$(id -g) $HOME/.kube/config

Alternatively, if you are the root user, you can run:

  export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each as root:

kubeadm join 172.31.91.136:6443 --token 2nzhbm.n0i7hb22etk9kpyb \
--discovery-token-ca-cert-hash sha256:442d8319bcd3cbf30e1b4b426372a3ed38168c0572d6d7762b03d29630118569
ubuntu@ip-172-31-91-136:~$ |
```

i-040f177639d06c3d5 (m3-k8s-master)

PublicIPs: 52.90.136.233 PrivateIPs: 172.31.91.136

Install the “Calico Network” to run the cluster using this command: `curl https://raw.githubusercontent.com/projectcalico/calico/v3.27.2/manifests/calico.yaml -O`

Run this command: `kubectl apply -f calico.yaml`

Join slave machine through below command

```
kubeadm join 172.31.91.136:6443 --token 2nzhbm.n0i7hb22etk9kpyb \
--discovery-token-ca-cert-hash
sha256:442d8319bcd3cbf30e1b4b426372a3ed38168c0572d6d7762b03d29630118569 --v=5
```

```
This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.

root@ip-172-31-81-83:/home/ubuntu# | |
```

i-0e54c3e9a1ce6158d (m2-k8s-slave1)  
PublicIPs: 18.208.167.49 PrivateIPs: 172.31.81.83

```
This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.

root@ip-172-31-91-84:/home/ubuntu# | |
```

i-0f6ea9918b6791df6 (m4-k8s-slave2)  
PublicIPs: 100.26.101.84 PrivateIPs: 172.31.91.84

```
ubuntu@ip-172-31-91-136:~$ kubectl get no
NAME                STATUS    ROLES    AGE     VERSION
ip-172-31-81-83     Ready    <none>   2m20s   v1.29.0
ip-172-31-91-136    Ready    control-plane  18m     v1.29.0
ip-172-31-91-84     Ready    <none>   2m36s   v1.29.0
ubuntu@ip-172-31-91-136:~$ | |
```

i-040f177639d06c3d5 (m3-k8s-master)  
PublicIPs: 52.90.136.233 PrivateIPs: 172.31.91.136

Paste IP of Host Machine on browser with 8080

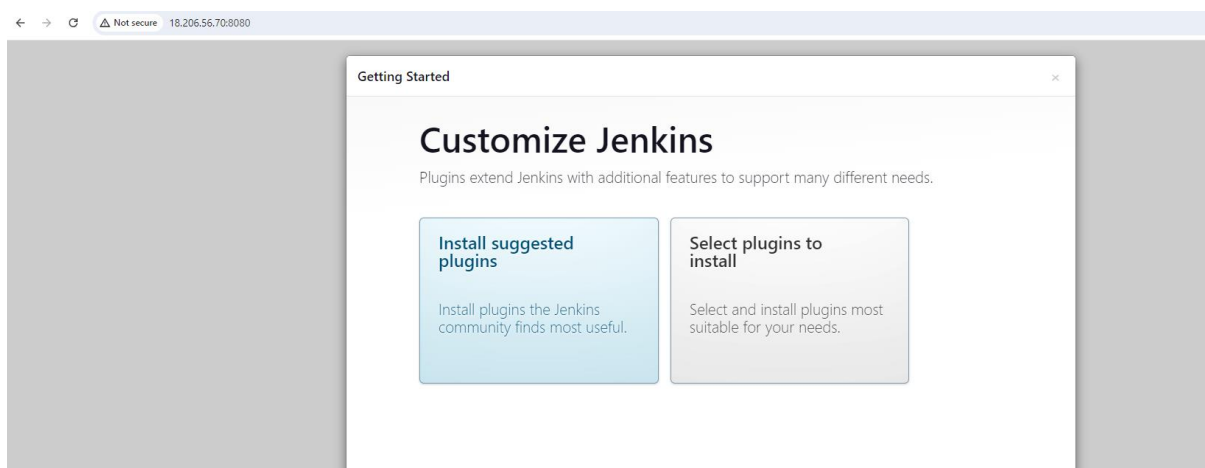
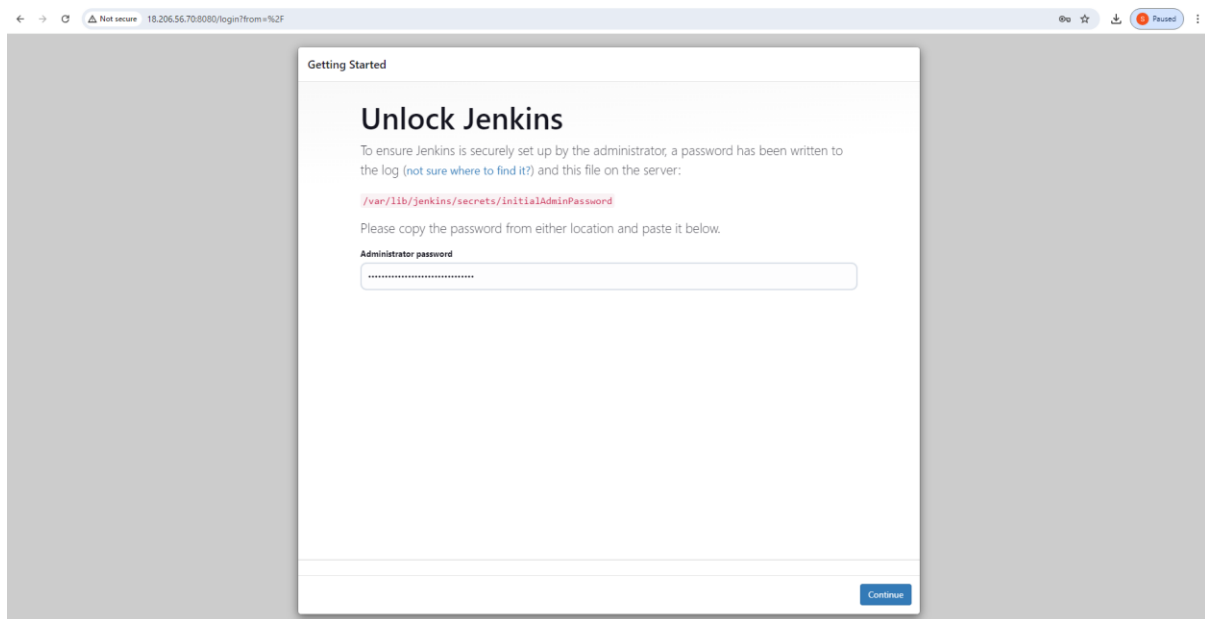


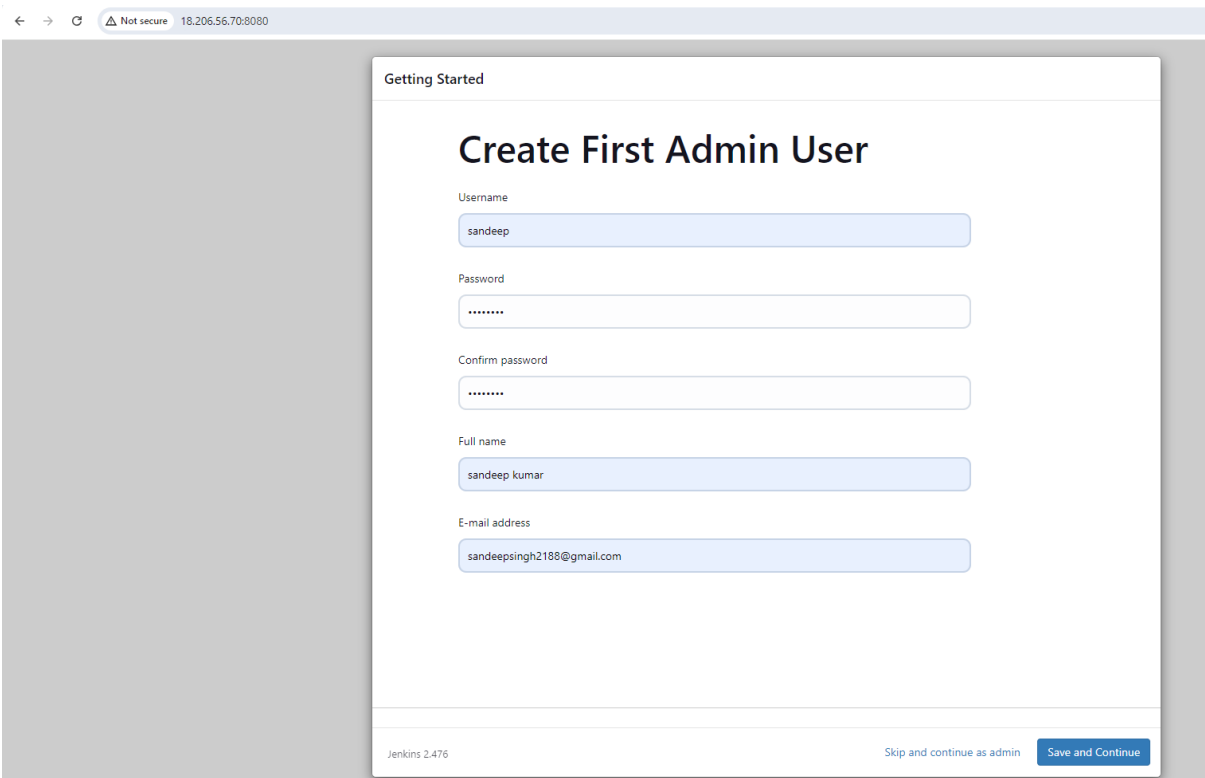
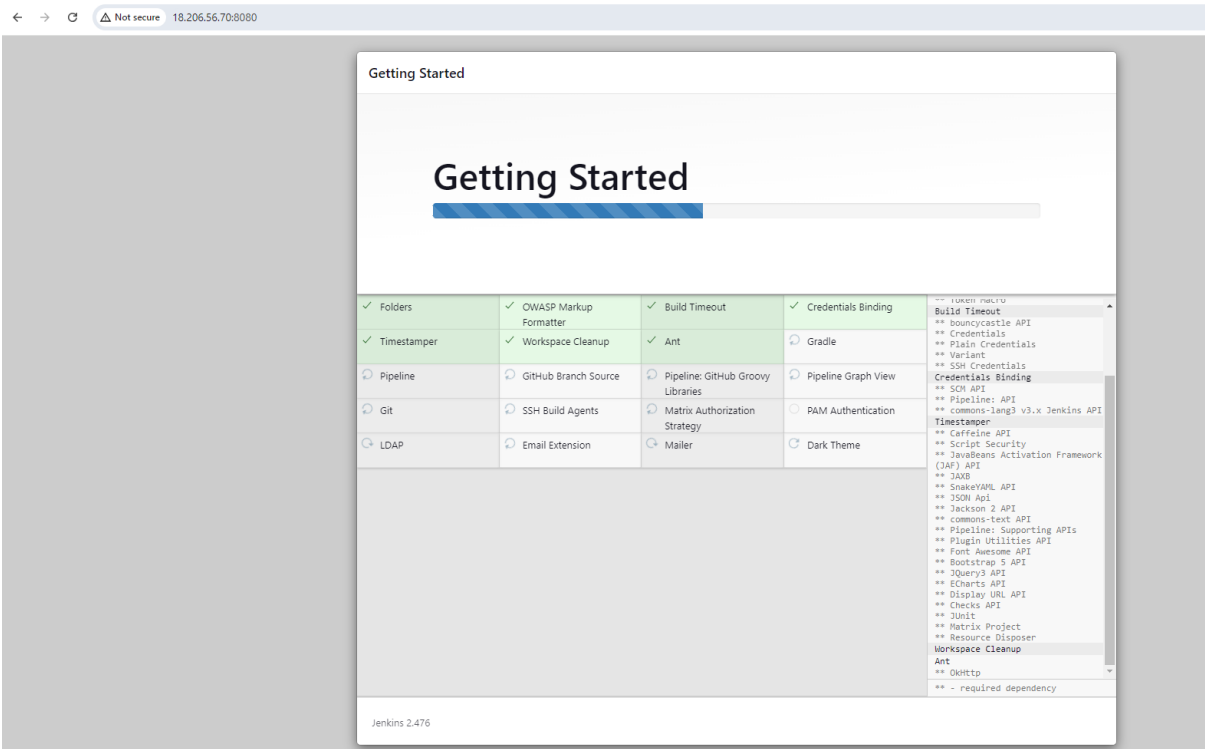
```
ubuntu@ip-172-31-91-170:~$ jenkins --version
2.476
ubuntu@ip-172-31-91-170:~$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
e4e942c48ccf4f50b4e0c0917690bb94
ubuntu@ip-172-31-91-170:~$ |
```

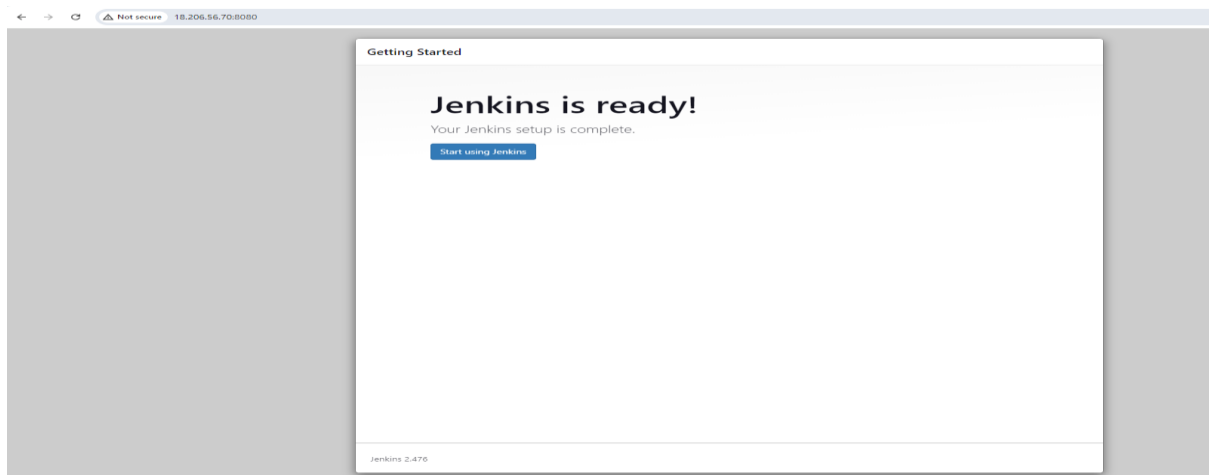
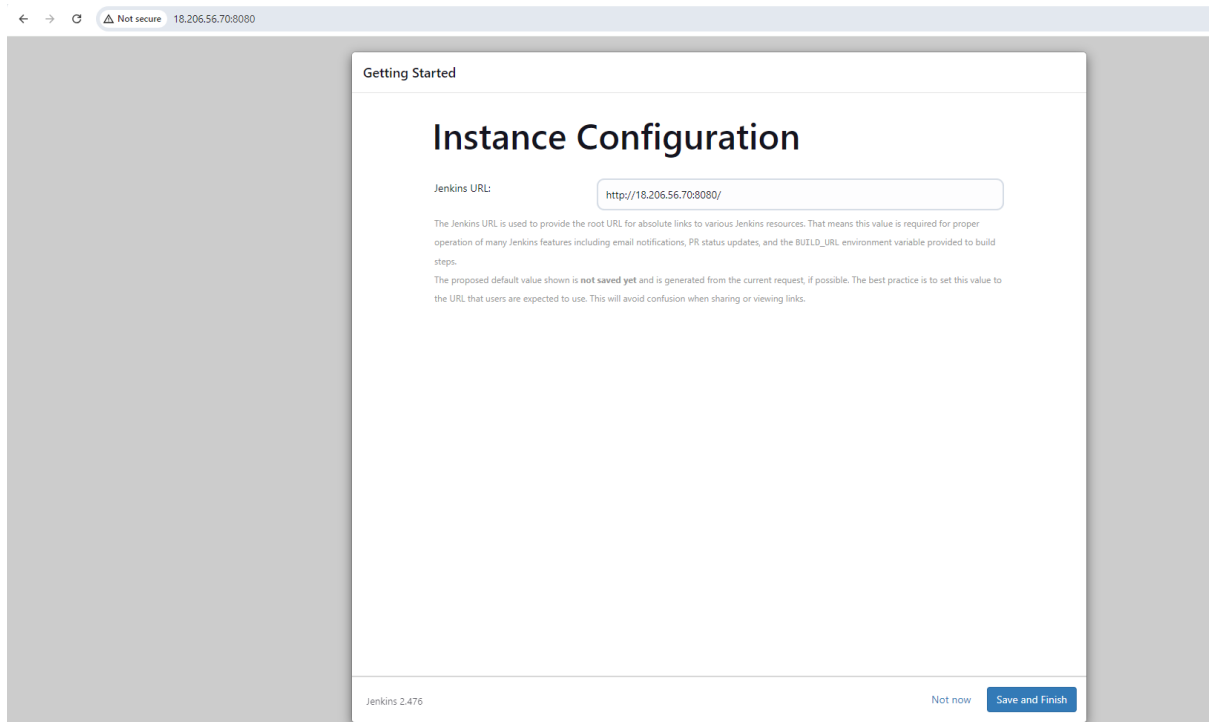
i-0946d6e4167cbb958 (HOSTMachine)

PublicIPs: 18.206.56.70 PrivateIPs: 172.31.91.170

Run on terminal and copy paste the password and again paste on Jenkins page







## Install SSH Agent



## Now Create a Node

← → ↻ Not secure 54.172.153.110:8080/computer/slave/configure

Jenkins

Search (CTRL+K)

🔒 sandeep kumar log out

Dashboard > Nodes > slave > Configure

🖨 Status  
🗑 Delete Agent  
⚙ **Configure**  
📋 Build History  
📊 Load Statistics  
📄 Script Console  
📄 Log  
📄 System Information  
🔌 Disconnect

Build Executor Status  
(0 of 1 executor busy)

Name ⓘ  
slave

Description ⓘ

Plain text [Preview](#)

Number of executors ⓘ  
1

Remote root directory ⓘ  
/home/ubuntu/jenkins

Labels ⓘ  
slave

Usage ⓘ  
Use this node as much as possible

Launch method ⓘ

Add credential with .pem key

← → ↻ Not secure 54.172.153.110:8080/computer/slave/configure

Jenkins

Search (CTRL+K)

🔒 sandeep kumar log out

Dashboard > Nodes > slave > Configure

Launch method ⓘ  
Launch agents via SSH

Host ⓘ  
172.31.91.136

Credentials ⓘ  
ubuntu (key)  
[+ Add](#)

Host Key Verification Strategy ⓘ  
Non verifying Verification Strategy ⓘ

Advanced Edited

Availability ⓘ  
Keep this agent online as much as possible ⓘ

Node Properties

☐ Disable deferred wipeout on this node ⓘ  
☐ Disk Space Monitoring Thresholds  
☐ Environment variables  
☐ Tool Locations

Save

← → ↻ Not secure 54.172.153.110:8080/computer/

Jenkins

Search (CTRL+K)

🔒 sandeep kumar log out

Dashboard > Nodes >

🖨 Nodes  
☁ Clouds

Build Queue  
No builds in the queue.

Build Executor Status  
built-in node = 1 agent (0 of 3 executors busy)

**Nodes** [+ New Node](#) [Configure Monitors](#)

S	Name ⓘ	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
🖨	<a href="#">Built-In Node</a>	Linux (amd64)	In sync	2.78 GiB	0 B	2.78 GiB	0ms
🖨	slave	Linux (amd64)	In sync	1.73 GiB	0 B	1.73 GiB	43ms
	Data obtained		6 min 18 sec	6 min 18 sec	6 min 18 sec	6 min 18 sec	6 min 18 sec

Icon: S M L

Legend

Create item

← → ↻ Not secure 54.172.153.110:8080/job/Testpipeline/configure

⌵ Search (CTRL+K) 🔒 🔴 sandeep kumar log out

Dashboard > Testpipeline > Configuration

Configure

General

Advanced Project Options

Pipeline

General

Enabled

Description

Testpipeline

Plain text [Preview](#)

☐ Discard old builds ?

☐ Do not allow concurrent builds

☐ Do not allow the pipeline to resume if the controller restarts

☐ GitHub project

☐ Pipeline speed/durability override ?

☐ Preserve stashes from completed builds ?

☐ This project is parameterized ?

☐ Throttle builds ?

Build Triggers

☐ Build after other projects are built ?

← → ↻ Not secure 54.172.153.110:8080/job/Testpipeline/configure

Dashboard > Testpipeline > Configuration

Configure

General

Advanced Project Options

Pipeline

General

☐ Build after other projects are built ?

☐ Build periodically ?

☒ GitHub hook trigger for GITScm polling ?

☐ Poll SCM ?

☐ Quiet period ?

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

Advanced Project Options

Advanced ▾

Pipeline

Definition

Pipeline script ▾

Script ?

```
1- pipeline {
2-   agent none
3-   environment {
4-     DOCKERHUB_CREDENTIALS = credentials('412ec374-fc79-4d99-9088-d66a856e1af6')
5-   }
6-   stages {
7-     stage('Hello') {
8-       steps {
9-         echo 'Hello World'
10-      }
11-    }
12-    stage('git') {
13-      agent {
14-        label 'slave'
15-      }
16-      steps {
17-        git 'https://github.com/Sandeepsingh2188/website.git'
18-      }
19-    }
20-  }
21-}
```

Save

Apply

← → ↻ Not secure 54.172.153.110:8080/me/my-views/view/all/

⌵ Search (CTRL+K) 🔒 🔴 sandeep kumar log out

Dashboard > sandeep kumar > My Views > All >

+ New Item

Build History

Project Relationship

Check File Fingerprint

Build Queue

No builds in the queue.

Build Executor Status

built-in node - 2 agents (0 of 5 executors busy)

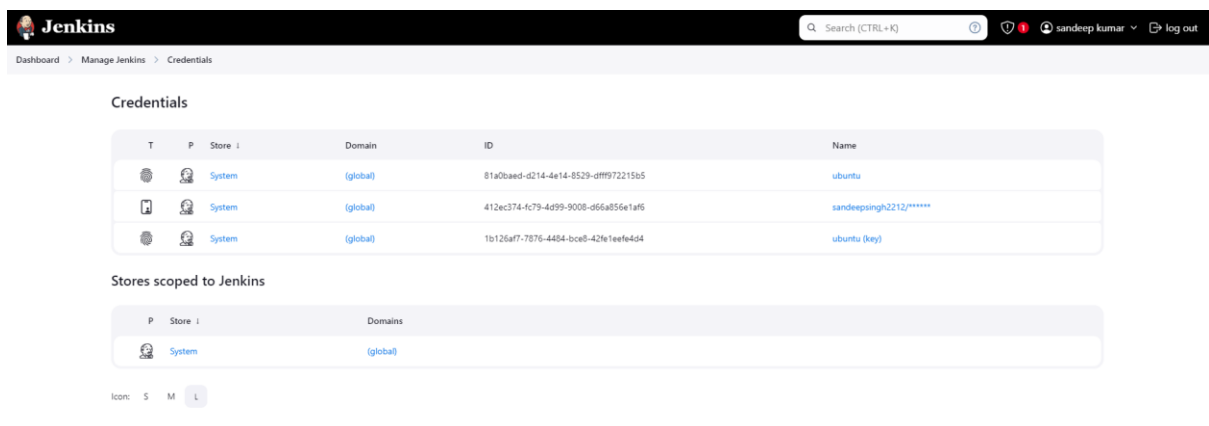
All +

S	W	Name	Last Success	Last Failure	Last Duration
		Testpipeline	17 hr #16	17 hr #13	18 sec

Icon: S M L

Add description

## Created docker hub credential

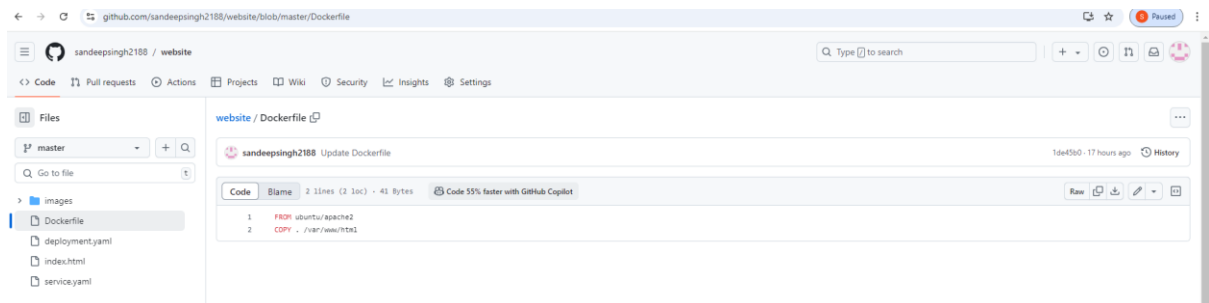


The screenshot shows the Jenkins 'Credentials' page. At the top, there's a search bar and a user profile for 'sandeep kumar'. Below the navigation bar, the 'Credentials' section displays a table of credentials. The table has columns for 'T' (Type), 'P' (Provider), 'Store' (System), 'Domain' (global), 'ID', and 'Name'. There are three entries: 'ubuntu', 'sandeepsingh2212/\*\*\*\*\*', and 'ubuntu (key)'. Below this, the 'Stores scoped to Jenkins' section shows a table with 'P' (System) and 'Domains' (global). At the bottom, there are tabs for 'Icons', 'S', 'M', and 'L'.

T	P	Store	Domain	ID	Name
🔑	System	System	global	81a0baed-d214-4e14-8529-dfff97221505	ubuntu
🔑	System	System	global	412ec374-4c79-4d99-9008-d66a8561af6	sandeepsingh2212/*****
🔑	System	System	global	1b125af7-7876-4484-bce8-42fe1ee4d4	ubuntu (key)

P	Store	Domains
🔑	System	global

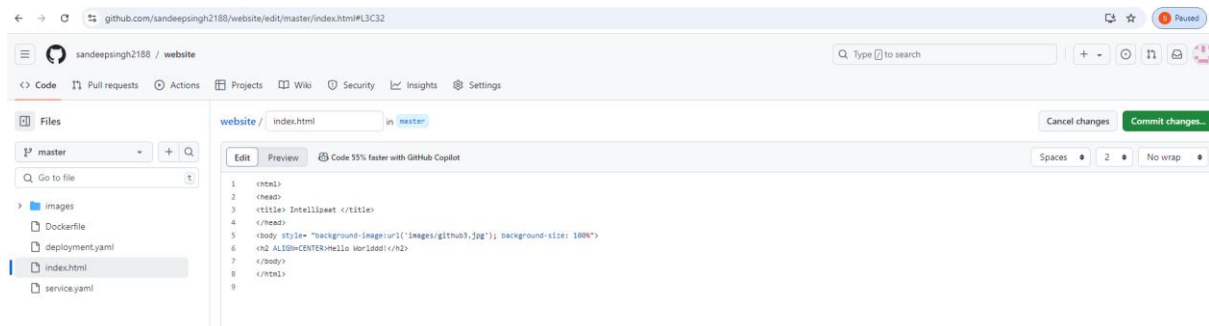
## Create dockerfile and committed



The screenshot shows a GitHub repository page for 'sandeepsingh2188 / website'. The 'Dockerfile' file is selected, showing its commit history and code. The code is as follows:

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

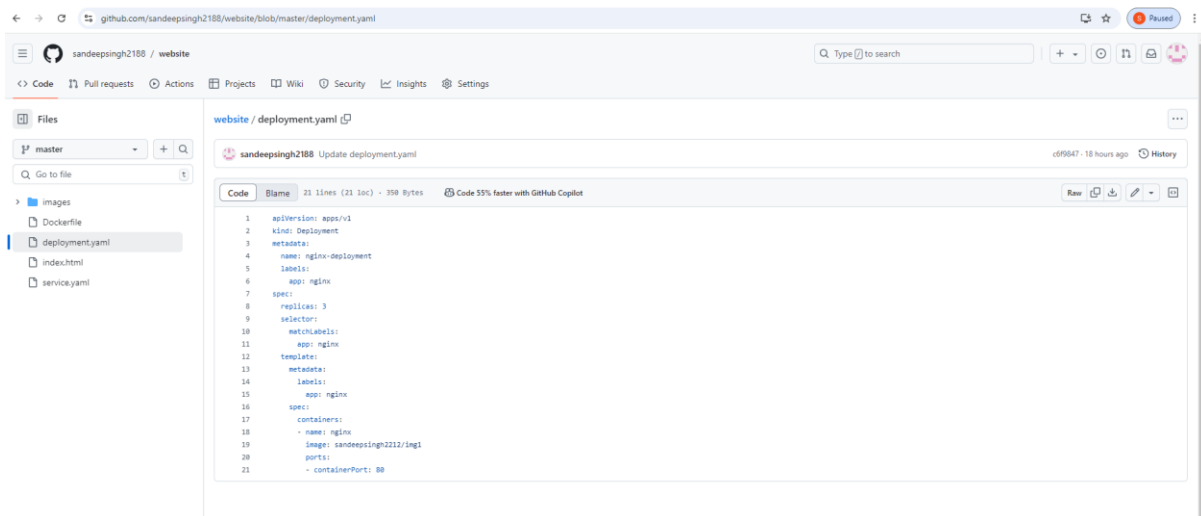
## Index.html



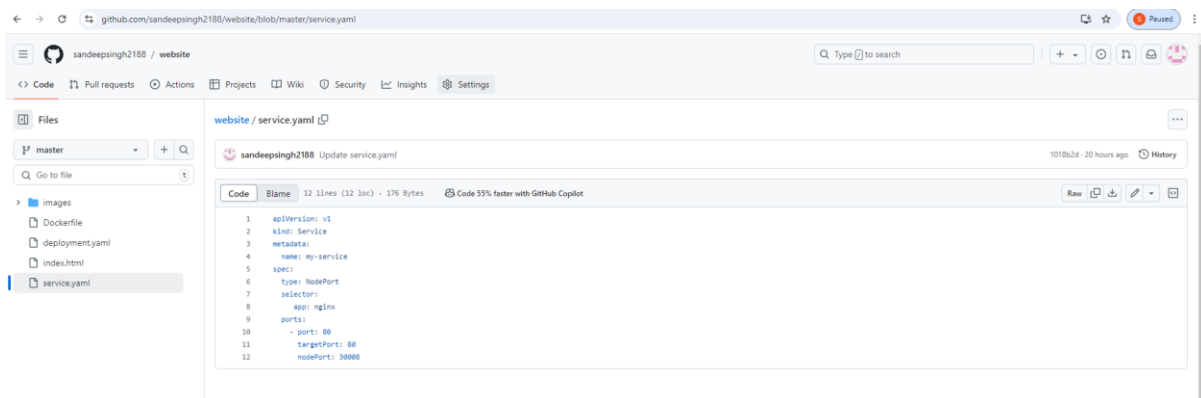
The screenshot shows the GitHub 'index.html' file being edited. The file content is as follows:

```
<html>
<head>
<title> Intellipaat </title>
</head>
<body style="background-image:url('images/github3.jpg'); background-size: 100%">
<h2 ALIGN=CENTER>Hello World!!</h2>
</body>
</html>
```

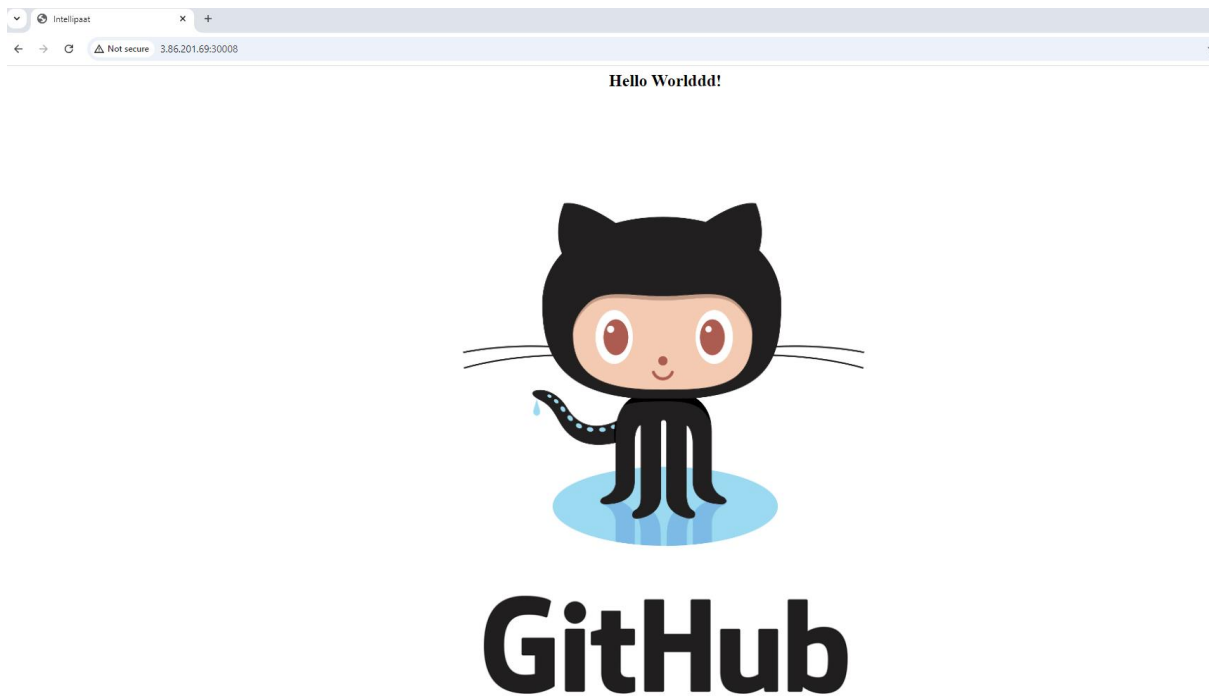
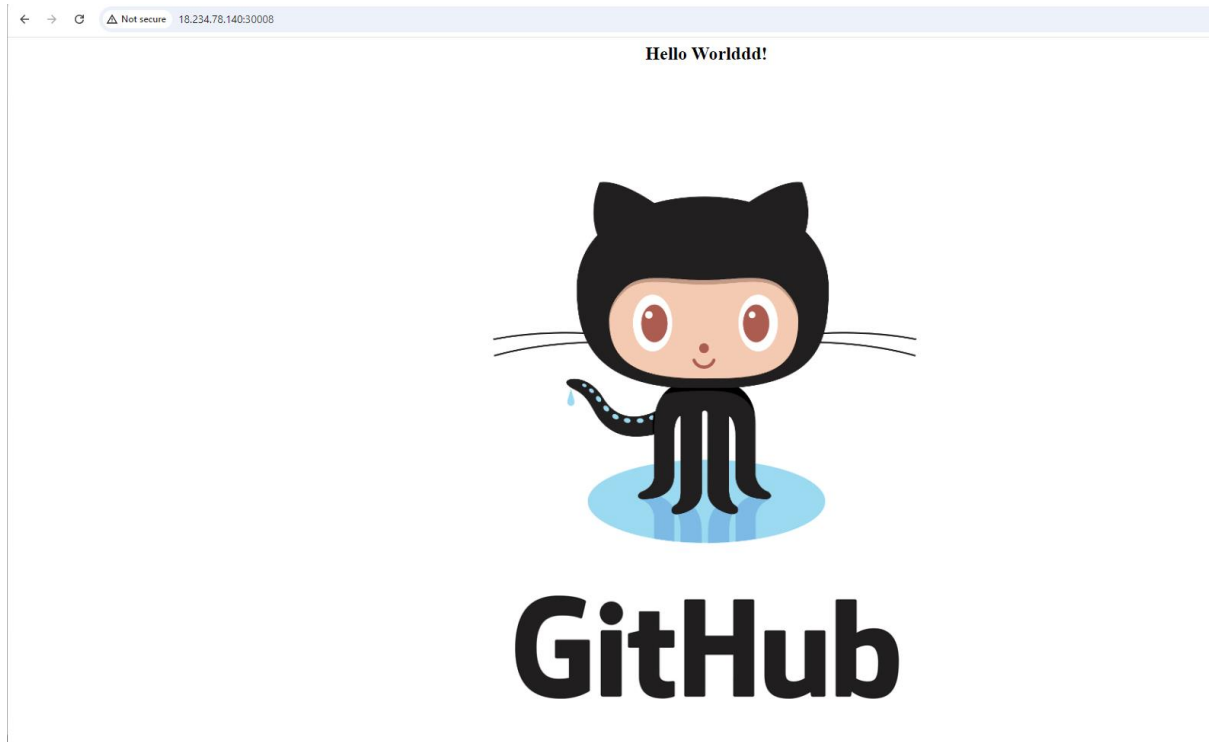
## Deployment.yaml



## Service.yaml



Copy Slave 1 and slave 2 public ip with 30008  
Output attached below



Project 2 Completed

By  
SANDEEP KUMAR  
9453743921