

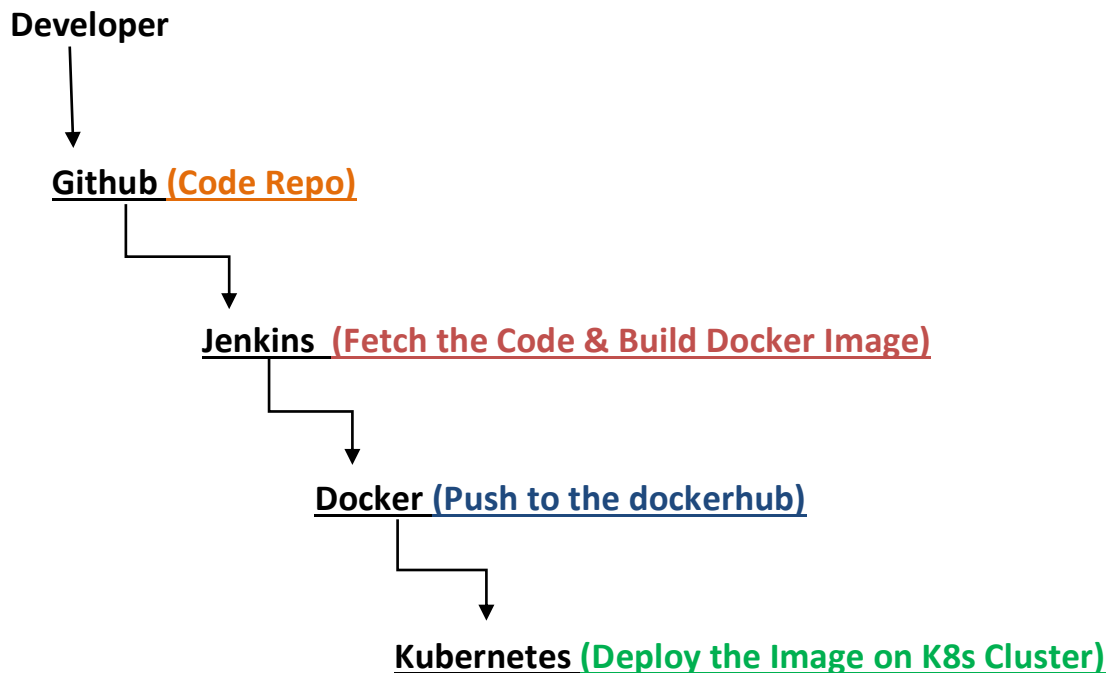
# K8s CI/CD Pipeline Project

In this project we're deploying amazon clone without backend services on the K8s Cluster by Using Github, Docker and Jenkins Pipeline Script.

Clone This Repo: -

<https://github.com/Hussain147/k8s-deploy-pipeline.git>

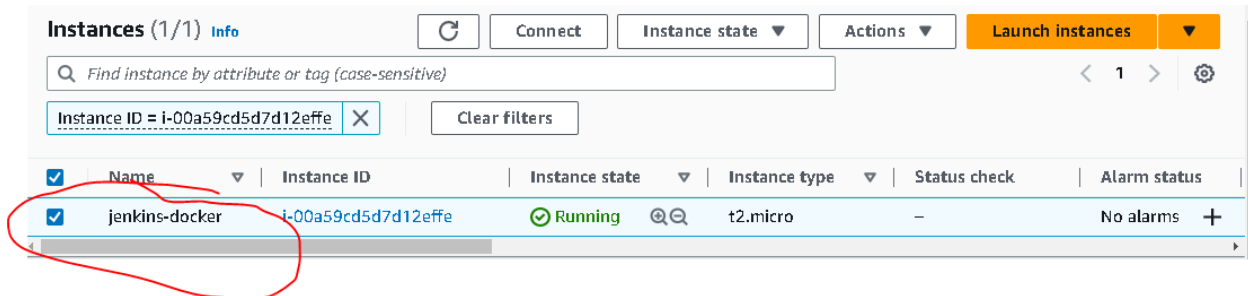
Project Architecture :-



Whenever the developer push the code to the Github repo, Jenkins will fetch the code & Build the Docker Image by using Docker Plugin & once the build completes Jenkins will push the image to the docker hub. And once it is pushed successfully, K8s Cluster will pull the image from docker hub and deploy the image on the worker nodes by using DeploymentService.Yaml file.

# Now, Let's Get Started

Launch a **Ubuntu Instance** with Security Group allowing port **8080 & 80**



Connect it through ssh:-

```
Hussain@DESKTOP-572PBGQ MINGW64 ~/OneDrive/Desktop
$ ssh -i horizon.pem ubuntu@54.89.30.221
```

install Java, Jenkins & Docker in it.

<https://www.jenkins.io/doc/book/installing/linux/#debianubuntu>

## Debian/Ubuntu

On Debian and Debian-based distributions like Ubuntu you can install Jenkins through **apt**.

### Long Term Support release

A **LTS (Long-Term Support) release** is chosen every 12 weeks from the stream of regular releases as the stable release for that time period. It can be installed from the **debian-stable apt repository**.

```
curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo tee \
/usr/share/keyrings/jenkins-keyring.asc > /dev/null
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
/etc/apt/sources.list.d/jenkins.list > /dev/null
sudo apt-get update
sudo apt-get install jenkins
```

Once Jenkins Installed, Check the status

**systemctl status Jenkins**

```

root@ip-172-31-53-40:~# systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/lib/systemd/system/jenkins.service; enabled; vendor prese
   Active: active (running) since Wed 2023-03-15 06:35:55 UTC; 9s ago
     Main PID: 6113 (java)
       Tasks: 44 (limit: 1143)
      Memory: 323.5M
         CPU: 43.201s
    CGroup: /system.slice/jenkins.service
            └─6113 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java

```

Now, install Docker :-

<https://docs.docker.com/engine/install/ubuntu/>

Once Installed, Check the status by :- docker ps

```

root@ip-172-31-53-40:~# sudo docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
root@ip-172-31-53-40:~# |

```

Now, add Jenkins user in docker group

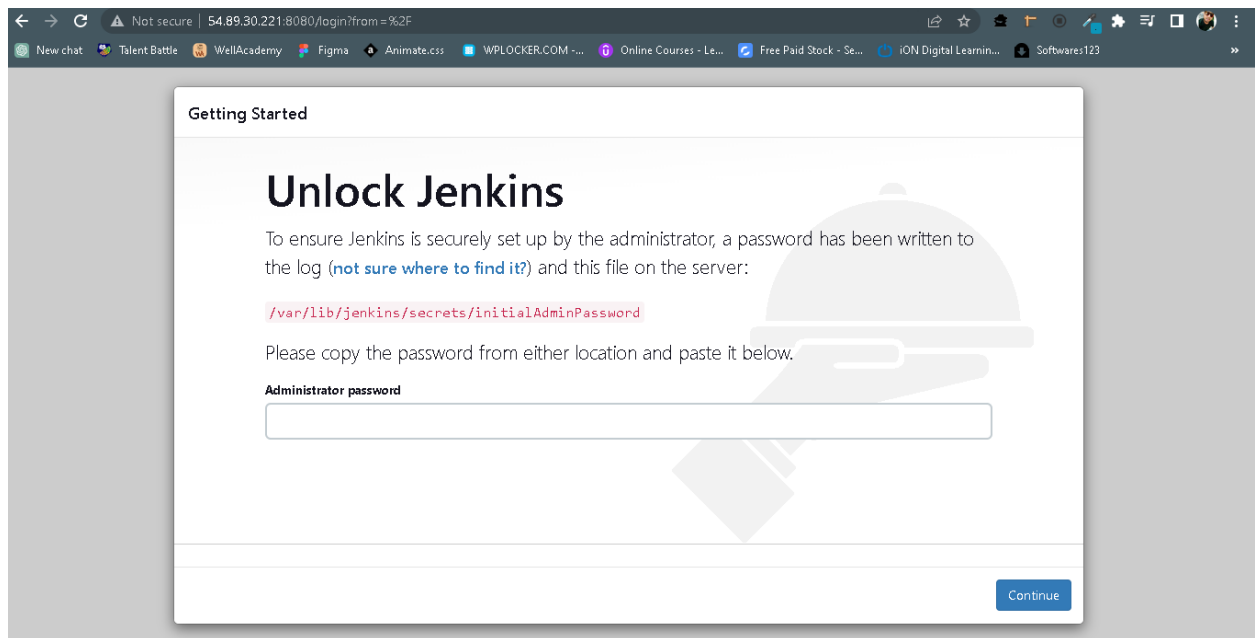
**usermod -aG docker Jenkins**

```

root@ip-172-31-53-40:~# usermod -aG docker jenkins
root@ip-172-31-53-40:~# |

```

Now, Copy th Public ip of Jenkins Instance & paste it in URL bar with port 8080  
to access Jenkins dashboard

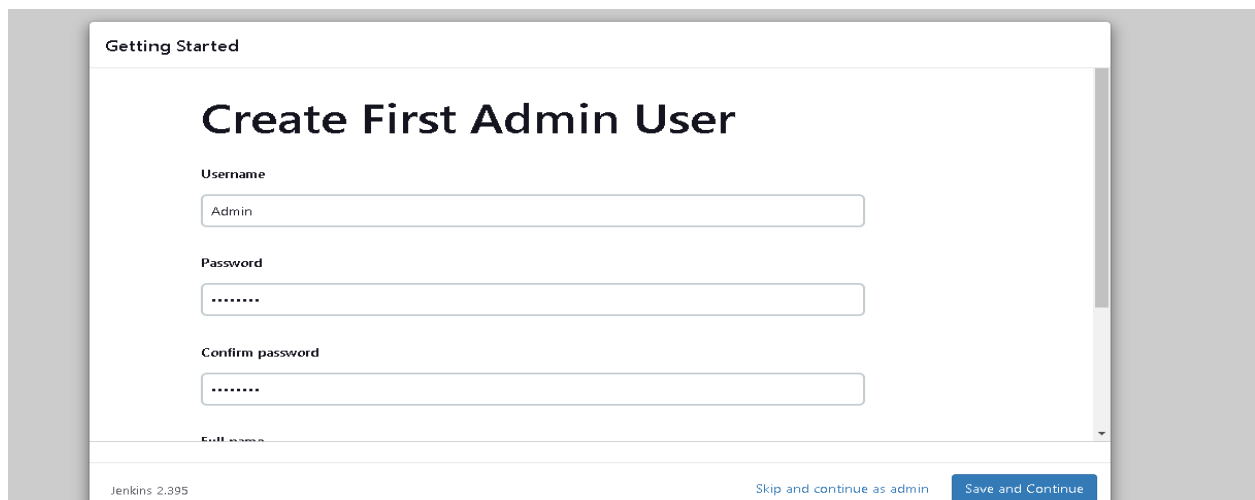


Copy the `/var/lib/jenkins/secrets/initialAdminPassword` and paste it in the Jenkins server in ssh. You'll get the initial password.

```
root@ip-172-31-53-40:~# cat /var/lib/jenkins/secrets/initialAdminPassword
664249d4f8fc4b82ae5557fef927c220
root@ip-172-31-53-40:~# |
```

Copy that initial password and paste it in the browser

Click Continue > Click Suggested Plugins > Setup the Credentials.



Click Save and Continue. > Start using Jenkins

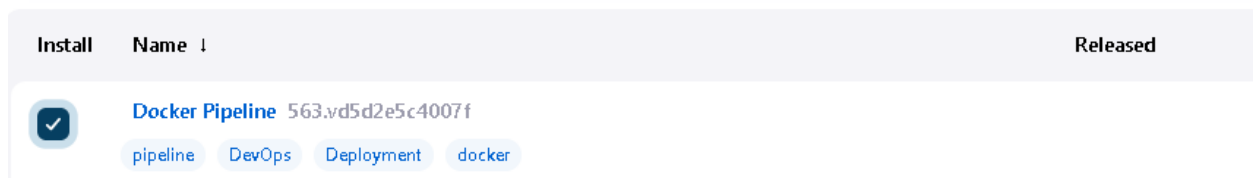
**Now Install Plugins :-**

**Go to Manage Jenkins > Manage Plugins > Available Plugins**

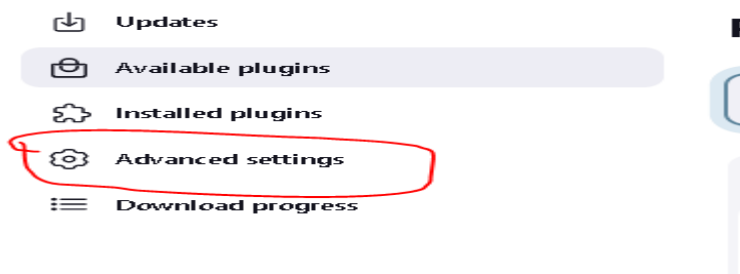
**Install these plugins:-**

**Docker pipeline**

**Kubernetes Continuous deploy**



**Click on Advanced Settings to upload/download our own plugin:-**



**Click on Choose file in Deploy Plugin or give the URL of the file**

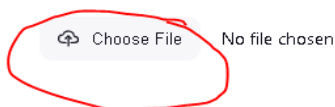
**<https://github.com/Hussain147/k8s-deploy-pipeline/blob/main/kubernetes-cd.hpi>**

**Click on View Raw to download the file**

#### Deploy Plugin

You can select a plugin file from your local system or provide a URL to install a plugin from outside the configured update site(s).

File



**Now Launch an ubuntu Based Instance with t2 medium type for Kubernetes Master Node.(Give SSH Permission & Allow All Traffic In SG)**

---

<input type="checkbox"/>	K8s Master	i-02ec53a3bbbe6c7f8	 Running		t2.medium	 2/2 checks passed
--------------------------	------------	---------------------	---	---	-----------	---

---

**Now Launch an Ubuntu Based Instance with t2 micro type for Kubernetes Worker Node.(Give SSH Permission & Allow All Traffic In SG)**

---

<input type="checkbox"/>	K8s Worker	i-0e9e9cce4675efcad	 Running		t2.micro	 2/2 checks passed
--------------------------	------------	---------------------	---	---	----------	---

---

**Connect it and install Kubeadm as per the steps from the given link:-**

<https://github.com/Hussain147/k8s-deploy-pipeline/blob/main/Kubeadm%20Installation.pdf>

Once installed Validate by **kubect! get pods in Master node**

```
ubuntu@ip-172-31-25-255:~$ kubect! get pods
No resources found in default namespace.
```

**Now, copy the kube config file and note it down somewhere**

```
ubuntu@ip-172-31-25-255:~$ cat ~/.kube/config
```

**<Copy all the content>**

**Now, Go to Manage Jenkins > Manage Credentials > System /Jenkins > Global Credentials > Add Credentials**

The screenshot shows the Jenkins 'New credentials' page. The breadcrumb trail is 'Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted)'. The form has the following fields and annotations:

- Kind:** A dropdown menu with 'Username with password' selected. A red circle highlights the label and the dropdown.
- Username:** A text input field containing 'kubehussain'. A red circle highlights the label and the input field.
- Treat username as secret:** An unchecked checkbox with a red circle around it.
- Password:** A text input field with masked characters '.....'. A red circle highlights the label and the input field.
- ID:** A text input field containing 'Dockercreds'. A red circle highlights the label and the input field.
- Description:** A text input field containing 'Dockercreds'. A red circle highlights the label and the input field.
- Create:** A blue button at the bottom. A red arrow points to it.

Kind : Username with password

Username : < your dockerhub username>

Password : <Your Dockerhub password>

Id: Dockercreds

Desc : Dockercreds



➤ Click **Create**

It will look like this,

## Global credentials (unrestricted)

[+ Add Credentials](#)

Credentials that should be available irrespective of domain specification to requirements matching.


ID	Name	Kind	Description
 Dockercreds	kubehussain/***** (Dockercreds)	Username with password	Dockercreds 


Now Again click on Add Credentials >

## New credentials


Kind


Kubernetes configuration (kubeconfig)

ID  kubernet

Description  kubernet

Kubeconfig

 Enter directly

Content 

```
preferences: {}
users:
- name: minikube
  user:
    client-certificate: /home/ubuntu/.minikube/profiles/minikube/client.crt
    client-key: /home/ubuntu/.minikube/profiles/minikube/client.key
```

Create

Kind : Kubernetes Configuration

Id : kubernet, Desc : kubernet

Kubeconfig : <paste the config file here>





Click Create

Now, You'll see like this

## Global credentials (unrestricted)

[+ Add Credentials](#)

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
 Dockercreds	kubehussain/***** (Dockercreds)	Username with password	Dockercreds 
 kubernet	kubernet (kubernet)	Kubernetes configuration (kubeconfig)	kubernet 



Dashboard >

+ New Item

People

Build History

Manage Jenkins

My Views

Now, Come to Dashboard > Click On New Item >

Item Name : **K8s-webapp-pipeline**

Select **Pipeline**

Click **Ok**

Enter an item name

K8s-webapp-pipeline

» Required field



**Freestyle project**

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build for something other than software build.



**Pipeline**

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipeline: and/or organizing complex activities that do not easily fit in free-style job type.



**Multi-configuration project**

Suitable for projects that need a large number of different configurations, such as testing on multiple builds, etc.



**Folder**

Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is a namespace, so you can have multiple things of the same name as long as they are in different namespaces.

OK

Multibranch Pipeline

Github Project : <https://github.com/Hussain147/k8s-deploy-pipeline.git>

☐ Discard old builds ?

☐ Do not allow concurrent builds

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

☒ GitHub project

Project url ?

Advanced ▾

☐ Pipeline speed/durability override ?

☐ Preserve stashes from completed builds ?

☐ This project is parameterized ?

☐ Throttle builds ?

Poll SCM : \* \* \* \* \*

Build Triggers

☐ Build after other projects are built ?

☐ Build periodically ?

☐ GitHub hook trigger for GITScm polling ?

☒ Poll SCM ?

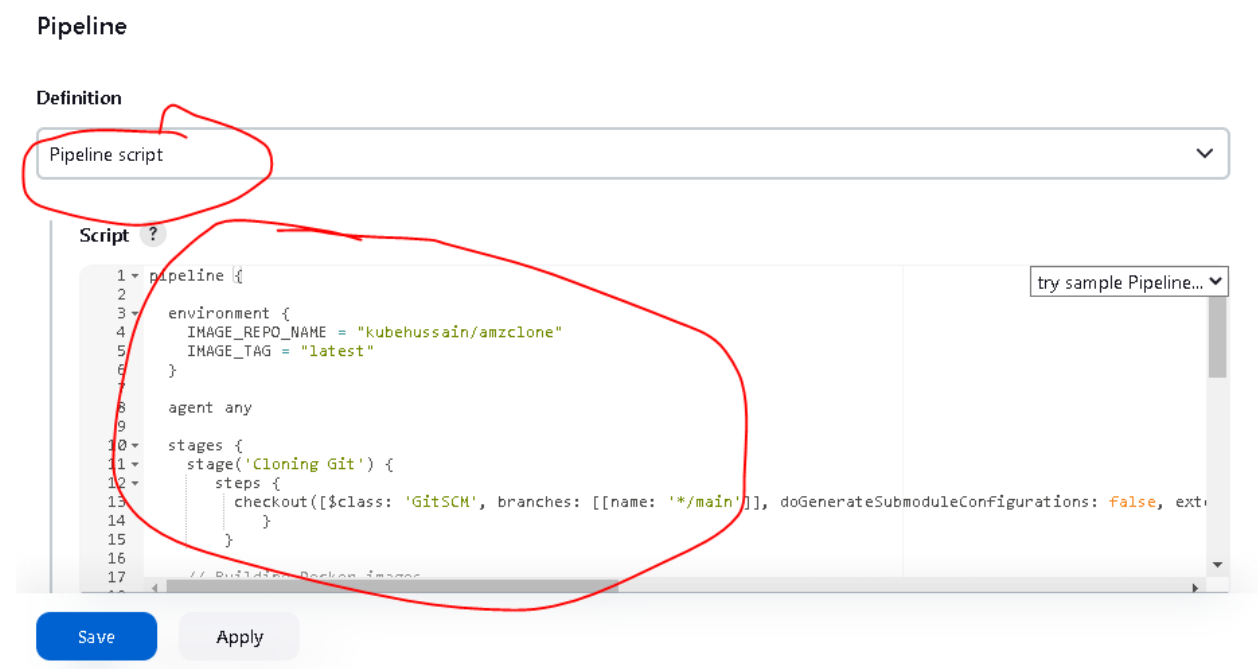
Schedule ?

⚠ Do you really mean "every minute" when you say "\* \* \* \* \*"? Perhaps you meant "H \* \* \* \* \*" to poll once per hour

Would last have run at Friday, March 17, 2023 at 10:37:26 AM Coordinated Universal Time; would next run at Friday, March 17, 2023 at 10:37:26 AM Coordinated Universal Time.

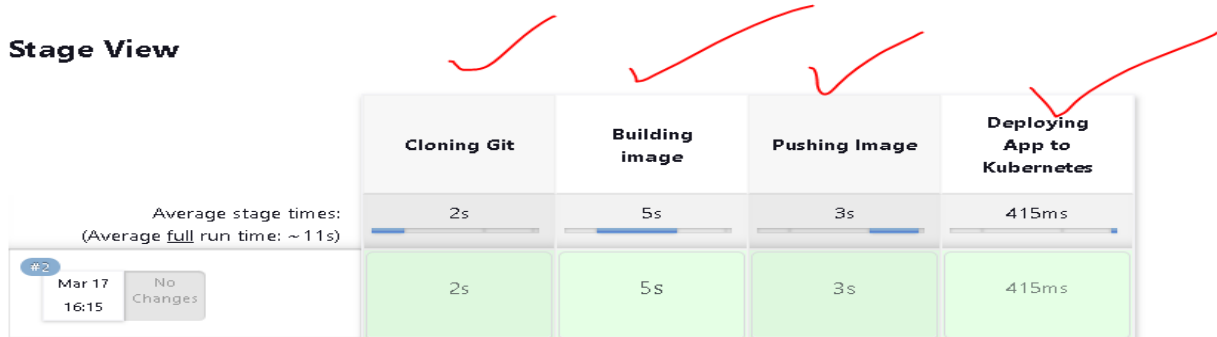
In Pipeline, Copy the script from the below link & paste(also read the Script for better understanding)

<https://github.com/Hussain147/k8s-deploy-pipeline/blob/main/Jenkinsfile>



See, Our all stages got succeed.

### Stage View



Our Amazon webapp clone has been successfully deployed on Kubernetes worker Node with replicas.

Now, Check in the master node,

### kubectl get deployment

```
ubuntu@ip-172-31-90-196:~$ kubectl get deployment
```

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
amazonclone-deployment	2/2	2	2	3h33m

### kubectl get pods

```
ubuntu@ip-172-31-90-196:~$ kubectl get pods
```

NAME	READY	STATUS	RESTARTS	AGE
amazonclone-deployment-79ccfb9f59-qpm5l	1/1	Running	0	3h34m
amazonclone-deployment-79ccfb9f59-wznkj	1/1	Running	0	3h34m

### kubectl get svc

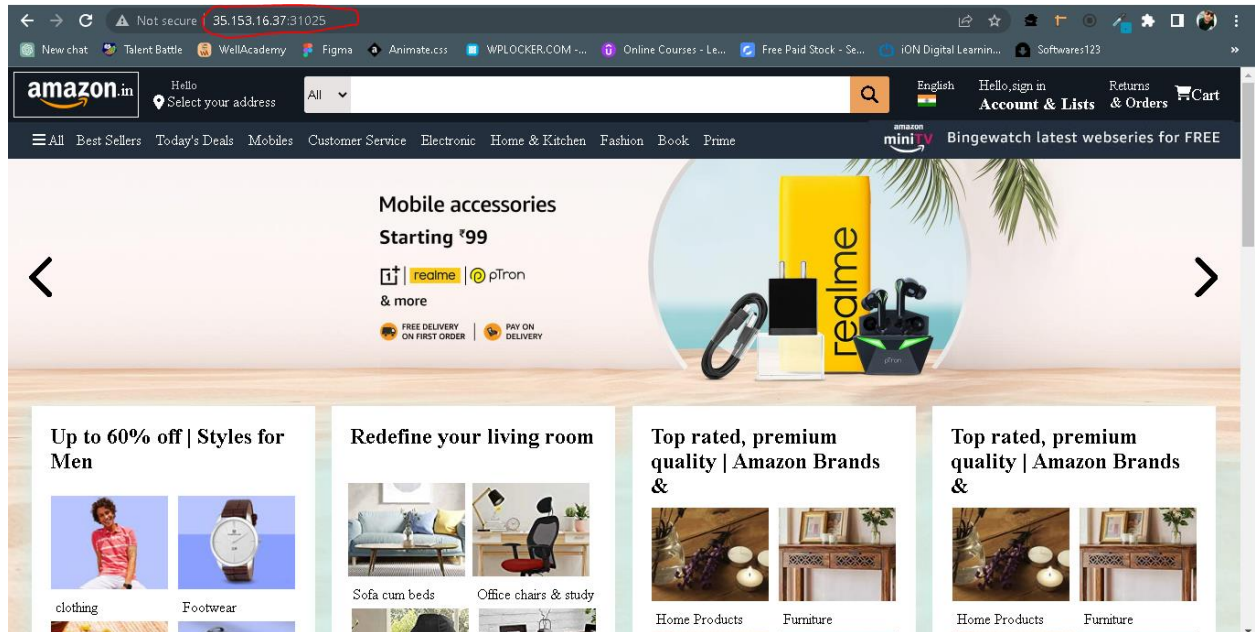
```
ubuntu@ip-172-31-90-196:~$ kubectl get svc
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
kubernetes	ClusterIP	10.96.0.1	<none>	443/TCP	4h23m
webapp-service	LoadBalancer	10.101.184.244	<pending>	80:31025/TCP	3h34m

See, our port 80 which is mapping to port 31025.

Now, Copy the worker node public ip with the port 31025

```
35.153.16.37:31025
```



**Congratulations...**

**You have deployed a Amazon Clone without backend on Kubernetes Cluster...**

**Now, If you do any changes to the github repo & push to central then CICD process will begin automatically.**