DevOps and its Applications

CS457

Assignment-1

Jenkins

Under the Guidance of – Dr Uma S

Submitted By:

1. Aqtar parveez (18BCS010)
2. G K Bharath Bhushan (18BCS026)
3. K V S Chaitanya (18BCS045)
4. Neha T (18BCS060)
5. Rahul S (18BCS075)
6. Varun Awati (18BCS108)
7. Sushanth B Patil (18BCS102)
8. Trishul K S (18BCS104)

Setting up CI/CD Jenkins pipeline for Kubernetes

Tools and Technologies used :

- Github
- Docker and Docker hub
- Jenkins
- Kubernetes Cluster

Prerequisites:

- NodeJS v8+
- 2 AWS Ec2 Ubuntu instances of size t2.medium and 15 GB of volumes attached.
- Install Docker and kubernetes on AWS Instances.

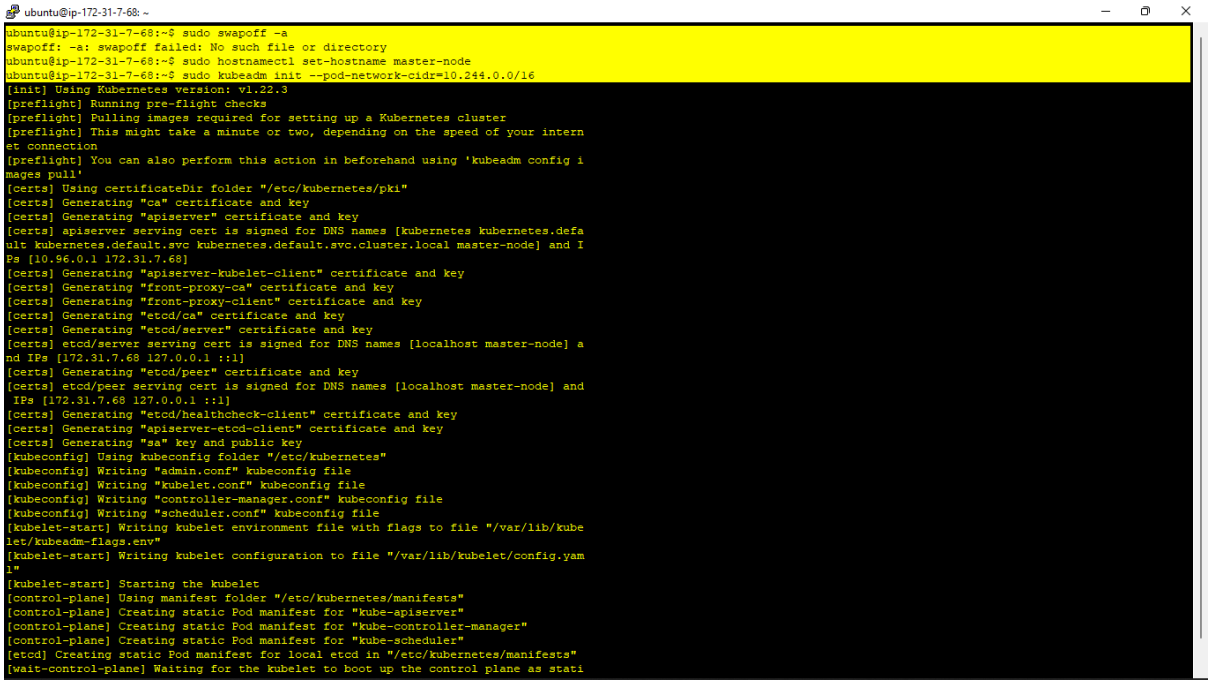
Step - 1 : Setting Up kubernetes Cluster

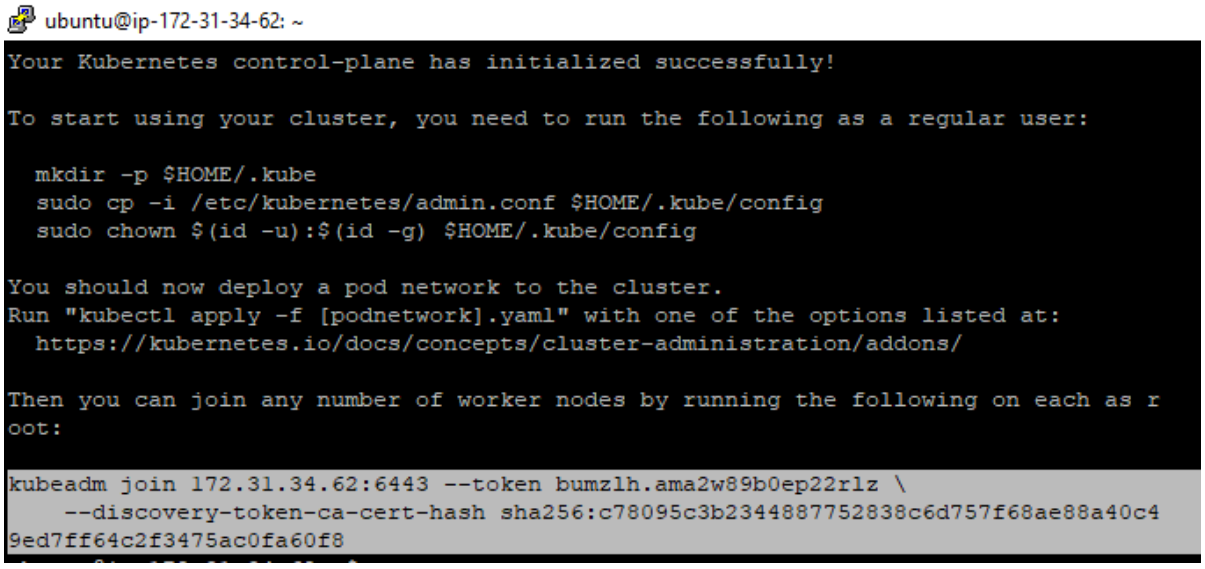
We have used the kubeadm tool to set up the kubernetes cluster. Setting up a kubernetes cluster containing 2 nodes master and worker.

1. Install docker and add docker daemon after that enable and start the docker on both the nodes

2. Install Kubernetes (Kubeadm ,Kubelet and Kubectl) on both the nodes

3. Initialize the master node using kubeadm. Output of this command would be a key , through which worker nodes can join the Kubernetes cluster. Copy the token and save it somewhere.

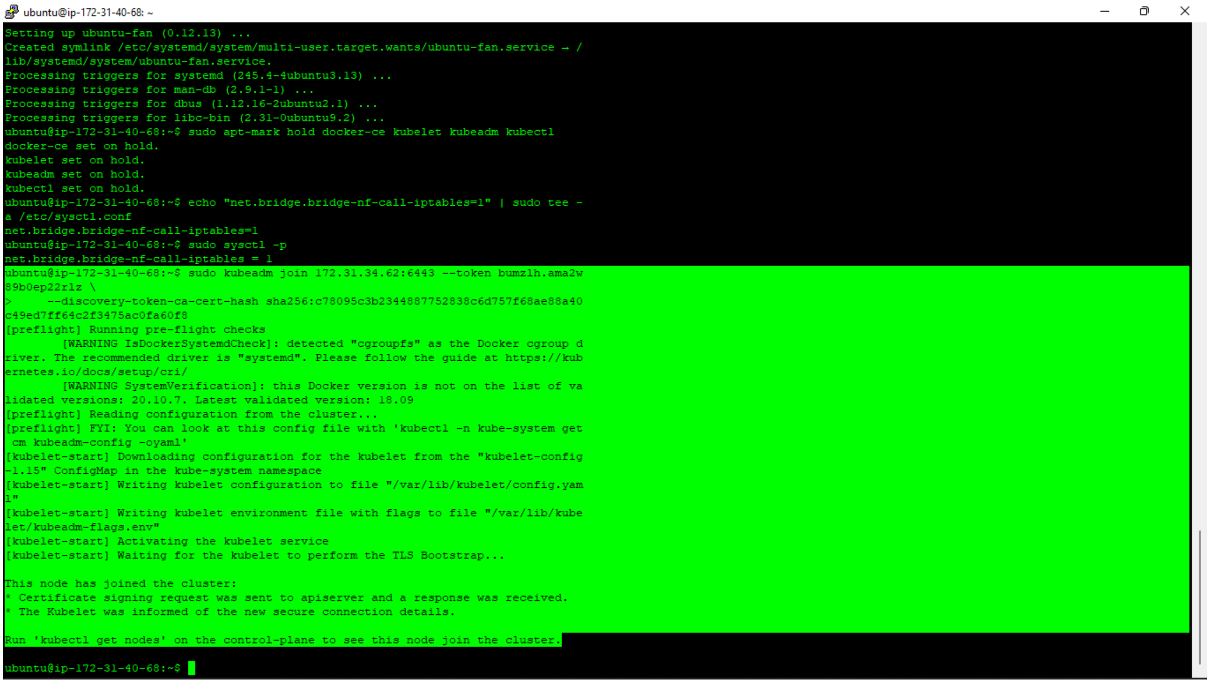




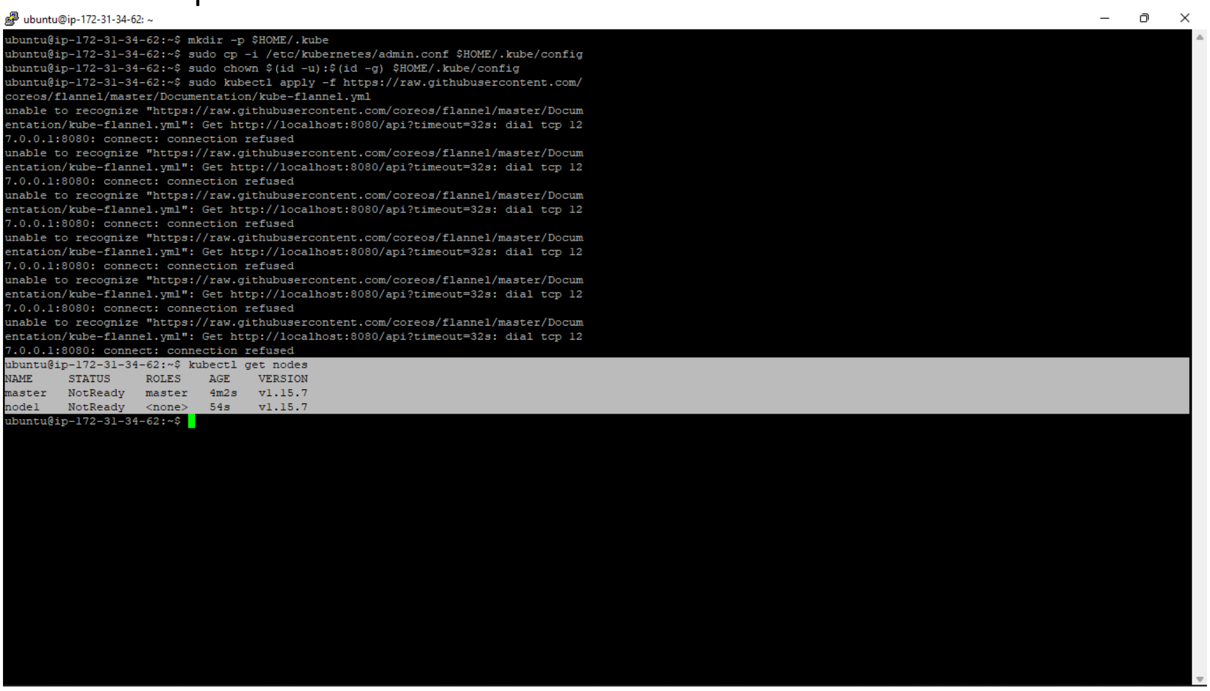
Using the token copied earlier run this command on worker node :

kubeadm join 172.31.34.62:6443 --token bumzlh.ama2w89b0ep22rlz \ --discovery-token-ca-cert-hash sha256:c78095c3b2344887752838c6d757f68ae88a40c49ed7ff64c2f3475ac0fa60f8

Get the following output:

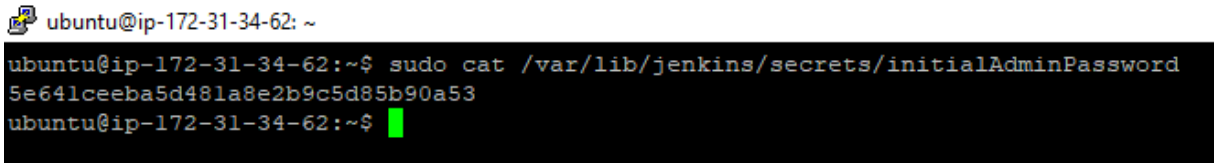


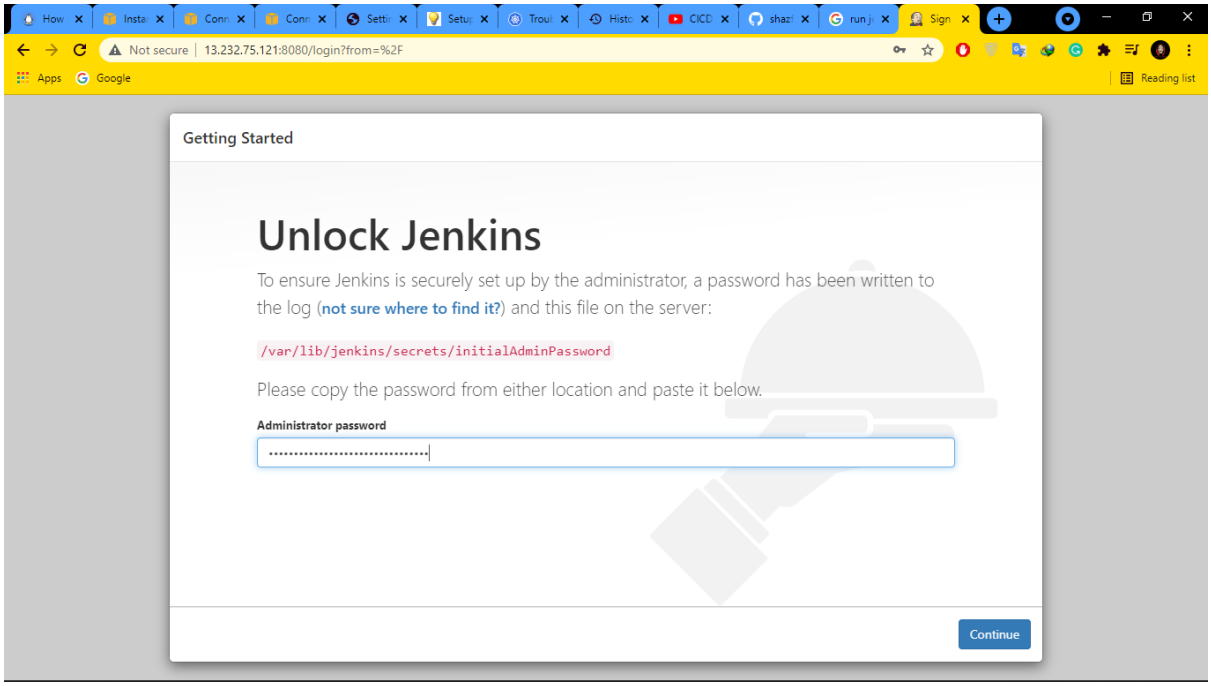
When we run kubectl get nodes on master node Now there would be 2 nodes one master and one newly joined worker node newly joined node's role name would be <none> to label it as worker Using the token copied earlier



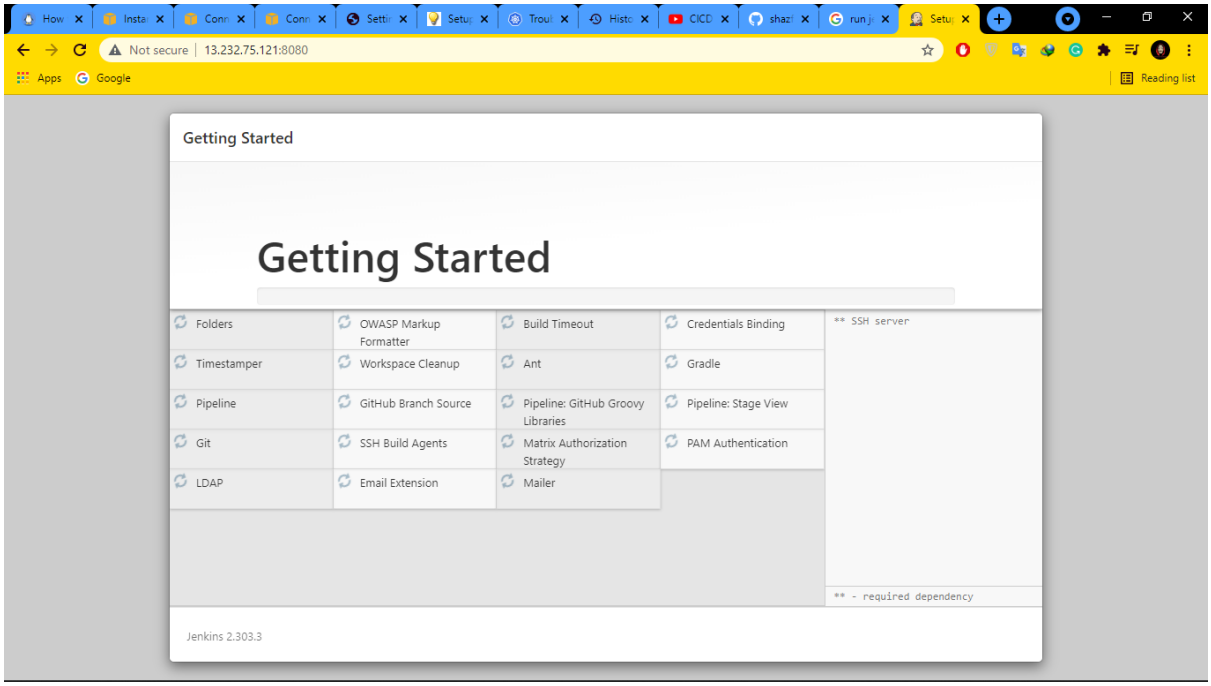
Step - 2 : Setting up jenkins On aws ec2 ubuntu

1. Install Java and Jenkins. Run the jenkins file, the first time it asks for an admin password. Run cat /var/lib/jenkins/secrets/initialAdminPassword and copy the password and paste it.



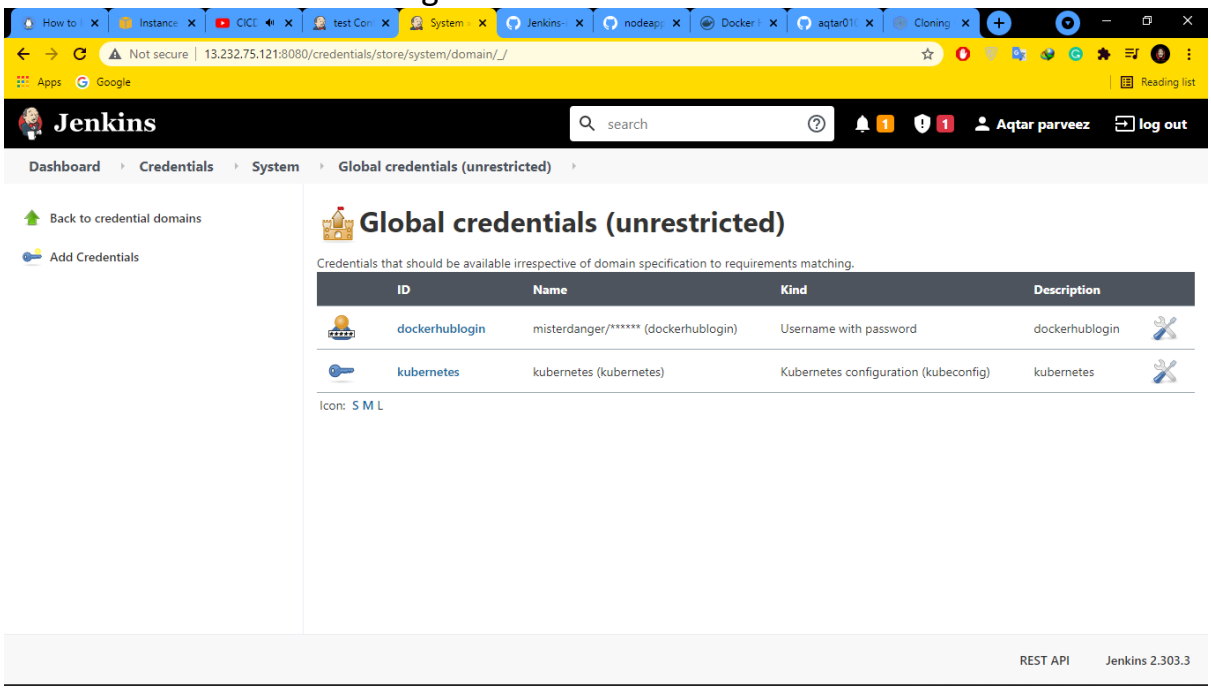


Enter the credentials and select the default plugin pack and wait for it to install. After installing start using Jenkins.



Go to Manage Jenkins -> Manage Plugins -> Available and install plugins for Nodejs, Docker, Kubernetes, Github, Kubernetes CLI.

Configure Docker Hub and Kubernetes Custer credentials. Go to Manage Jenkins -> Manage credentials and add a credential for docker hub with your username and password. Create a credential id (which will be used later) and description. Similarly add Kubernetes cluster by adding the config file i.e. home->.kube->config and save it.



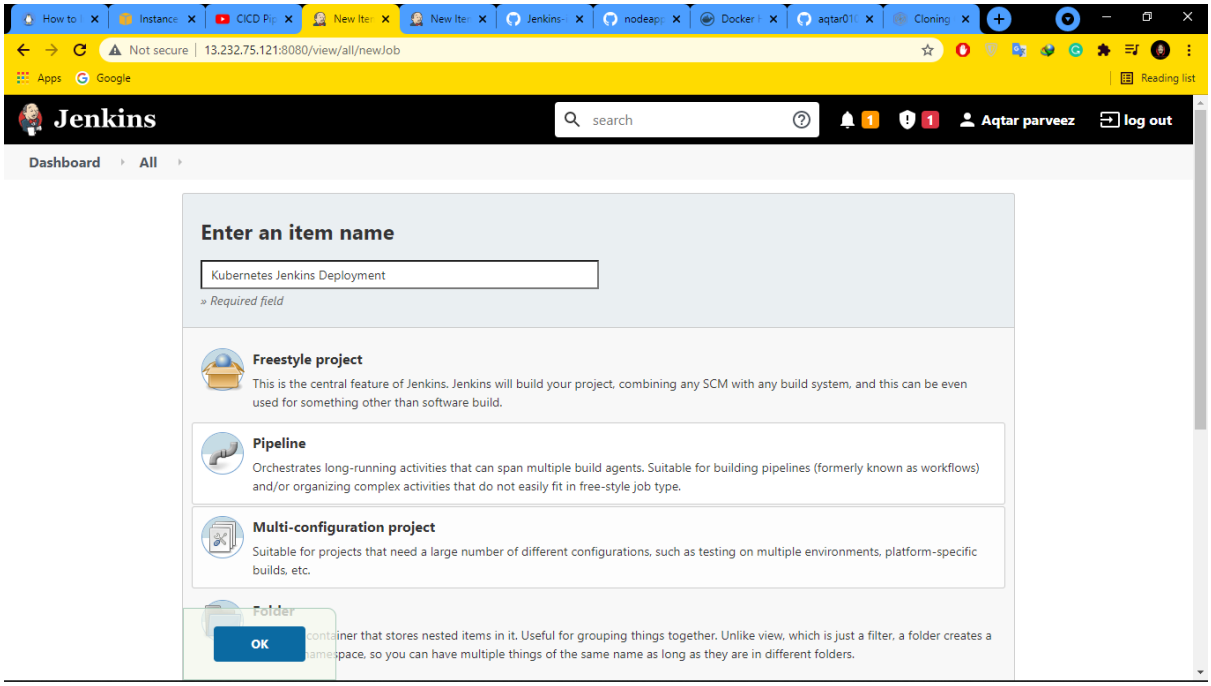
Install docker on Jenkins server and add current ubuntu user and Jenkins to docker group

Step - 3: Setting up code

We made a simple nodejs application, code is on github: [Github Link]

Step - 4: Building the pipeline

1. Create a jenkins pipeline.



Add pipeline code to create stages which include:

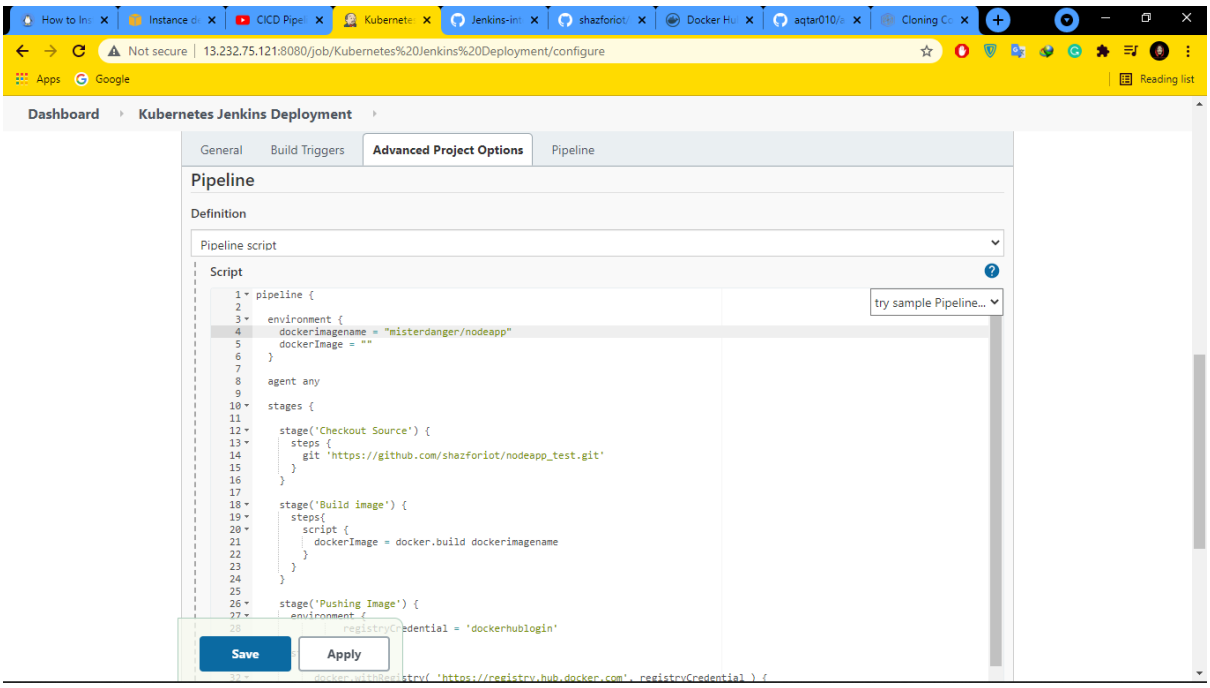
A. Create Enviornment & Git clone

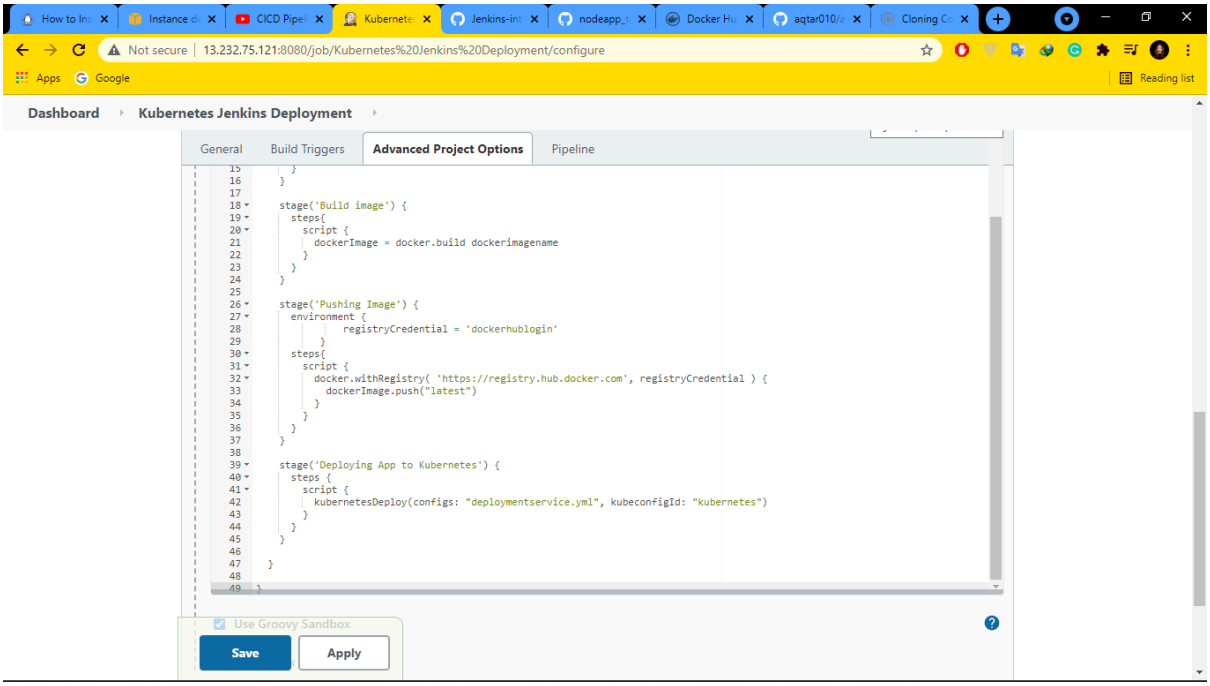
B. Docker Build

C. Docker login

D. Push image to docker hub

F. Copy deploymentservice.yml file to kubernetes master



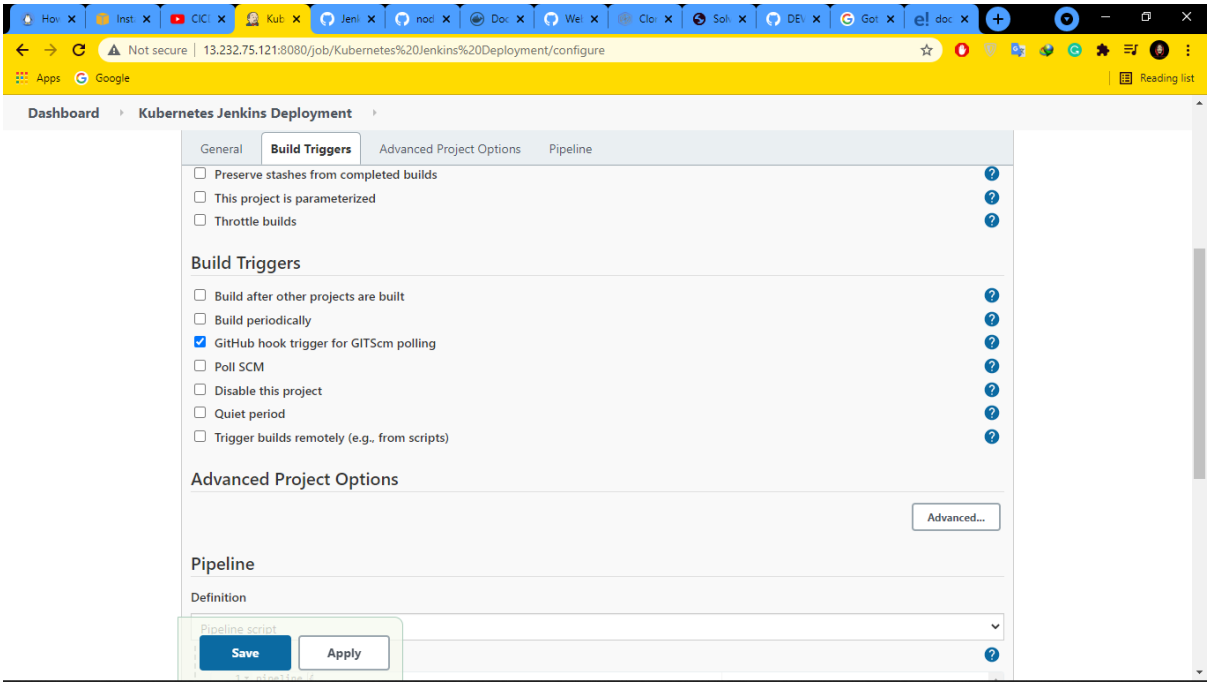


The Pipeline Code:

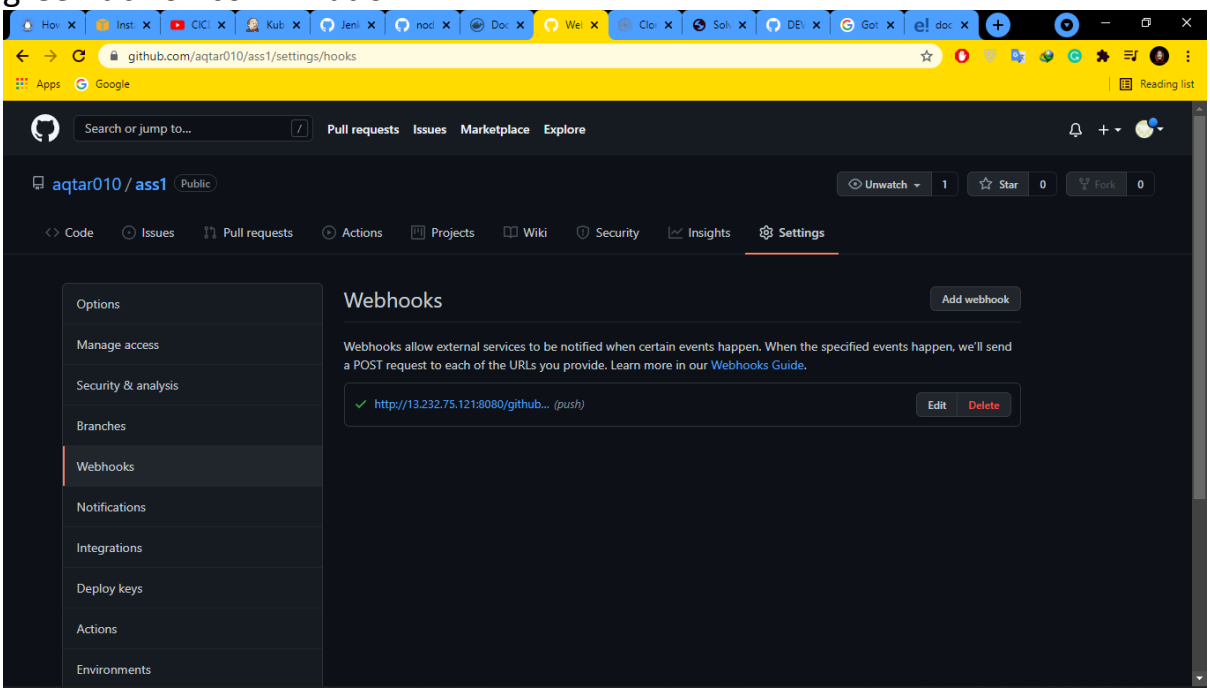
```
pipeline {


  environment {
    dockerimagename = "misterdanger/nodeapp"
    dockerImage = ""
  }
  agent any
  stages {
    stage('Checkout Source') {
      steps {
        git 'https://github.com/aqtar010/ass1.git'
      }
    }
    stage('Build image') {
      steps{
        script {
          dockerImage = docker.build dockerimagename
        }
      }
    }
    stage('Pushing Image') {
```

```
      environment {
                registryCredential = 'dockerhublogin'
           }
      steps{
        script {
          docker.withRegistry( 'https://registry.hub.docker.com',
registryCredential ) {
            dockerImage.push("latest")
          }
        }
      }
    }
    stage('Deploying App to Kubernetes') {
      steps {
        script {
          kubernetesDeploy(configs: "deploymentservice.yml", kubeconfigId:
"kubernetes")
        }
      }
    }
  }
}
```

Select the github hook trigger option and save.



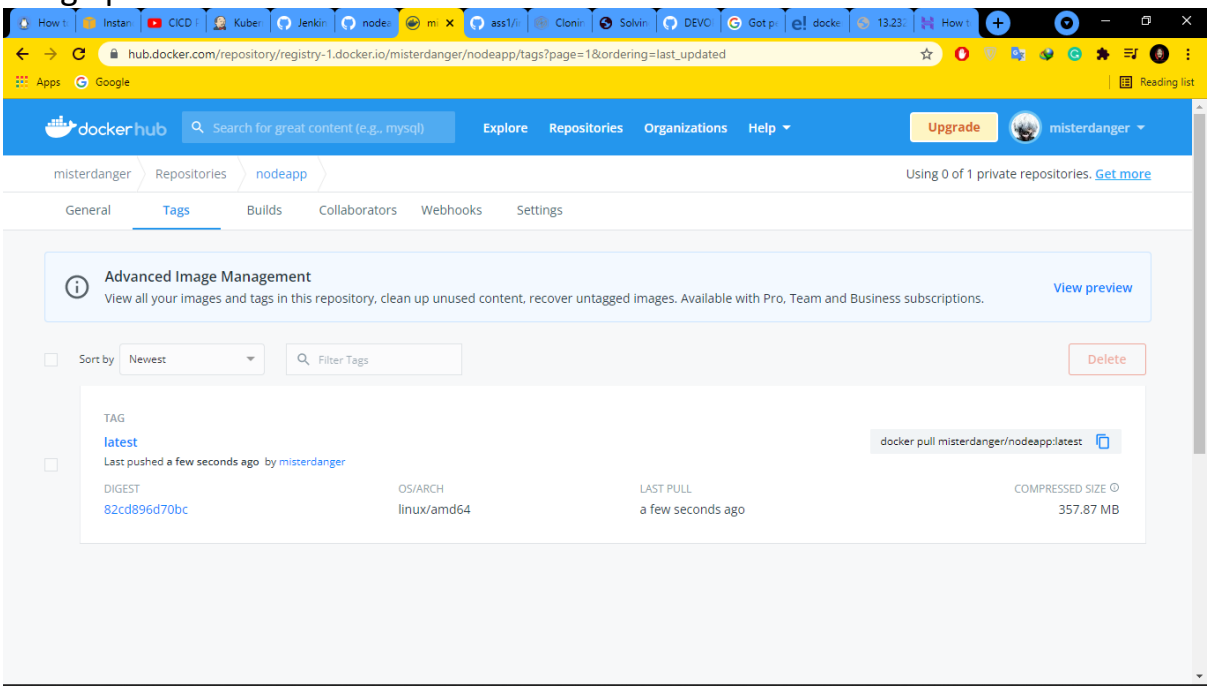
Go to github->your repo and add a webhook and enter the jenkins ip and port no. and add the webhook . It will ping your server and show a green tick on confirmation.



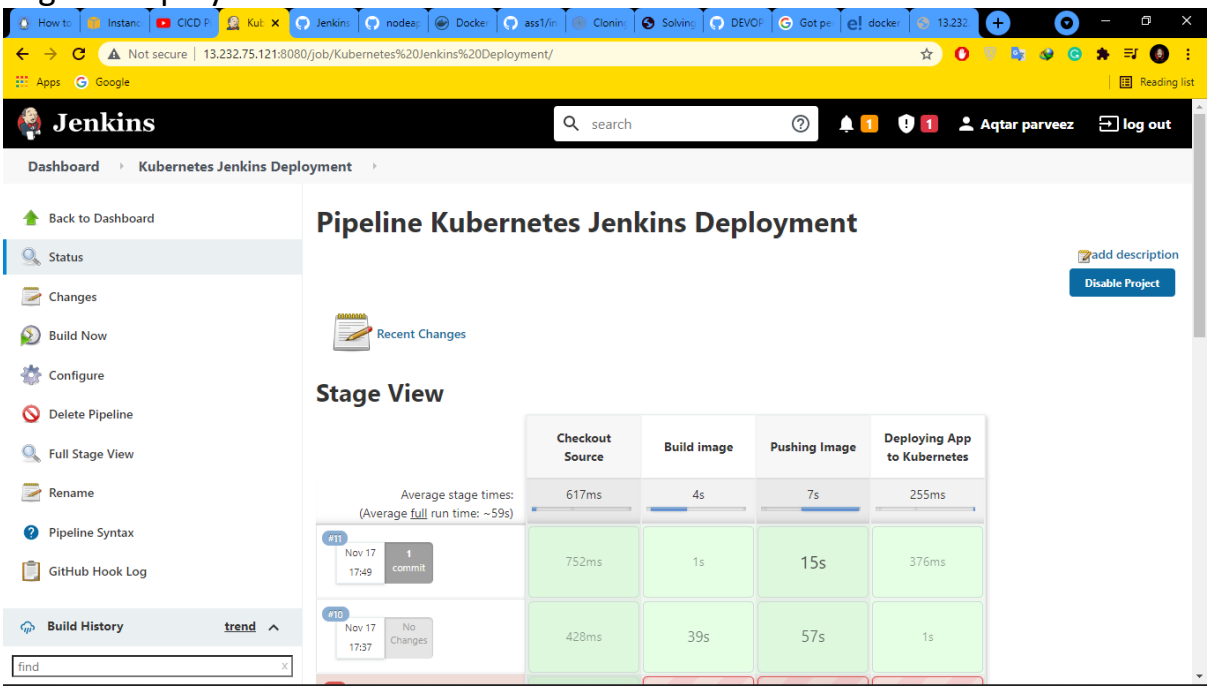
Build The Job on Jenkins

Output:

Image pushed on to docker Hub:



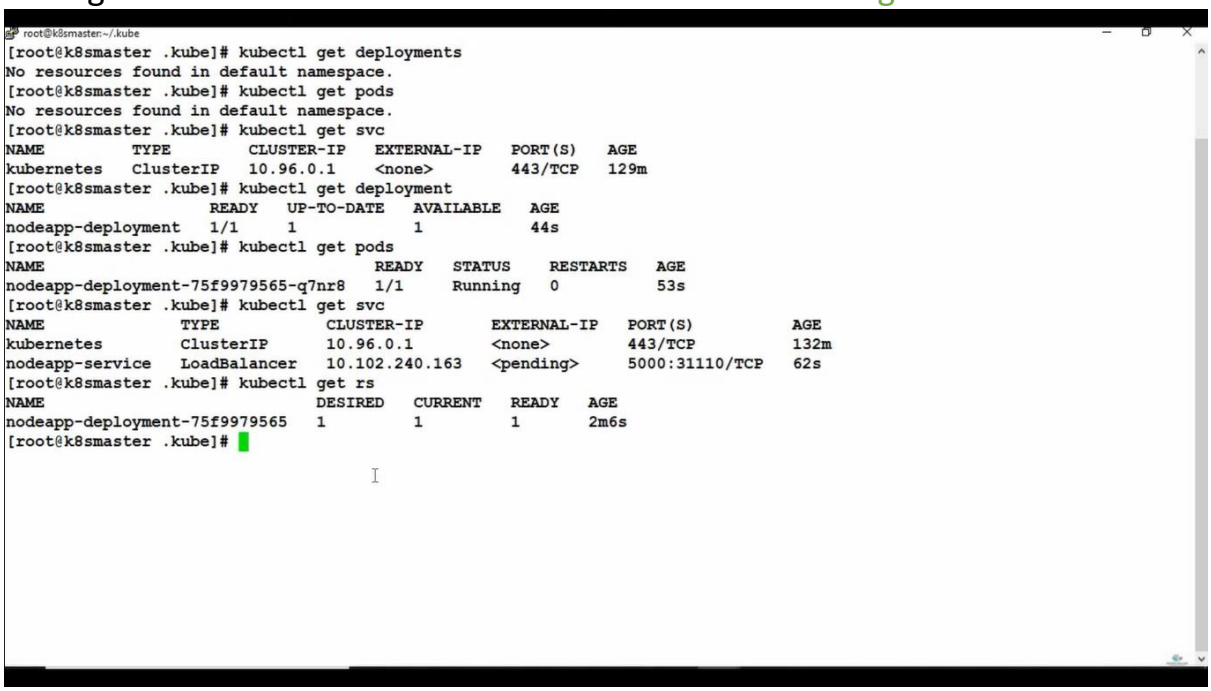
Logs of Deployment on Jenkins:



Checking the deployments on kubernetes server

1. To get list of deployments on server run kubectl get deployments

2. To get list of services on kubernetes server run kubectl get svc



node-js service in running on Nodeport, so Deployment is Successful

From the terminal output, node-js-service is running on port number 31110, so

open tcp custom 31110 on kubernetes service instance