**********CICD pipeline**********

Implement Docker and Kubernetes into DevOps Workflow :::

CICD :: git/jenkins/docker/kubernetes

==> CICD Pipeline - to automate the build and deployment.

1. Developers Create the Source Code
2. Commit the Source_Code to Source_Code Repository(Github)
3. Jenkins_pipeline ---> for Monolith Application Architecture
- SCM_Checkout - Download the source_code to build server
- Application_Build - Process of compiling the source code and create artifacts(Binaries - *.war/*.jar)
- Deploy the artifacts to Target Server(QA/UAT/PROD)
3.1 Jenkins Pipeline ---> for Containerized - Micro-Service Based Application Architecture
- SCM_Checkout
- Application_Build using Maven
- Application Image Build using Docker
- Published to Container Registry(Dockerhub)
- Deploy the Container Image in the Target Server and run the Application using Container.

PROJECT WORK FLOW: Micro service based application
- Jenkins Pipeline ---> for Containerized - Micro-Service Based Application Architecture
-SCM_Checkout
- Application_Build using Maven
- Application Image Build using Docker
- Published to Container Registry(Dockerhub)
- Deploy the Container Image in the Target Server and run the Application using Container

- How to On-board any Application to DevOps CICD Process : "Checklist"

First need devops assessment:

- For that we must have create task and infrastructure list
- For build server which tool need ans all

Implement DevOps:
- Java web application

Create/What CICD pipeline:
- it is composed of various stages to automate build and deploy.

Pipeline stages:

- SCM_Checkout
- Application_Build using Maven
- Application Image Build using Docker
- Published to Container Registry(Dockerhub)
- Deploy the Container Image in the Target Server and run the Application using Container.

==> Till here we understand what task we have to perform

Now try understand/think what resources we need to fulfil this Deployment

Checklist:

1. **Infra-structure:**
   a) **Jenkins Master-node** --> Used to create CICD pipeline and schedule to run in slave node
      a.1 - Jenkins Slave Node(Build server) --> Perform application build and create artifacts. (*.war/.jar)
   b) Kubernetes Master ==> Deploy the application image
      -- K8s Worker node 1
      -- K8s Worker node 2

Total 5 VMs

Tool:

A) **Jenkins Master-node** --> idk, jenkins, git
   a.1 - Jenkins Slave Node --> git, jdk, mave, Docker engine
B) **Kubernetes Master: -->** All the k8s components to be installed.
   -- K8s Worker node 1
   -- K8s Worker node 2

CICD Pipeline:

- scm_checkout--> Build application artifacts & Unit testing ---> Build docker application image --> publish to docker registry --> Deploy to Kubernetes

- DevOps CICD Pipeline Trigger/Execution WorkFlow :::

Developers' Role :

1) In dev env they use IDE to create source code
2) Push source code to the github ---> "Configure Webhook for automation "

Thru DevOps Automation :

3. GITHUB Webhook to Trigger the CICD Pipeline
   --> that will perform pipeline job task
4. - SCM_Checkout
   - Application_Build using Maven
   , **- Application Image Build using Docker**
   - Published to Container Registry(Dockerhub)
   - Deploy the Container Image in the Kubernetes Cluster

5. Email Notification to Users(Devloper and devops eng for succes deploy) Through Jenkins Pipeline

---

Understanding Source Code Repository :::

- src
- pom.xml # Define the dependencies and plugins required to build java mvn application
- Dockerfile # Used to define the properties of Application and its dependencies to Build Application Image
- kubernetes manifest file - *.yaml
  # Used to Create Deployment Controller Object and NodePort Service

---

Implementation ::::

Jenkins_Master (VM)
Jenkins_Slave(Build_Server) (VM)

Kubernetes_Master (VM)
Kubernetes_WorkerNode1 (VM)
Kubernetes_WorkerNode2 (VM)

VMs

- Creating Jenkins **Master** and configure **slave**:

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

On Jmaster: Install jdk and jenkins ==> jdk,jenkins,git

```
1  ls
2  sudo apt update
3  sudo apt install fontconfig openjdk-17-jre
4  java -version
5  sudo wget -O /usr/share/keyrings/jenkins-keyring.asc   https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
6  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
7  sudo apt-get update
8  sudo apt-get install jenkins
9  jenkis
10  sudo systemctl enable jenkins
11  sudo systemctl start jenkins
12  sudo systemctl status jenkins
root@ip-172-31-88-33:~# git --version
git version 2.34.1
```

Jenkins slave install jdk: ==> git,jdk,maven,Docker

```
1  sudo apt update
2  sudo apt install fontconfig openjdk-17-jre
3  java -version
4  git --version
6  history
7  apt install maven -y
8  git -version
9  git --version
10  java -version
11  mvn -version
12  apt install docker.io
13  history
```

---

1. Infrastructure is ready Configure Jenkins Master to slave
2.Configure Jenkins Master to K8s Master node

1. 1 Configure Jenkins Slave to Master

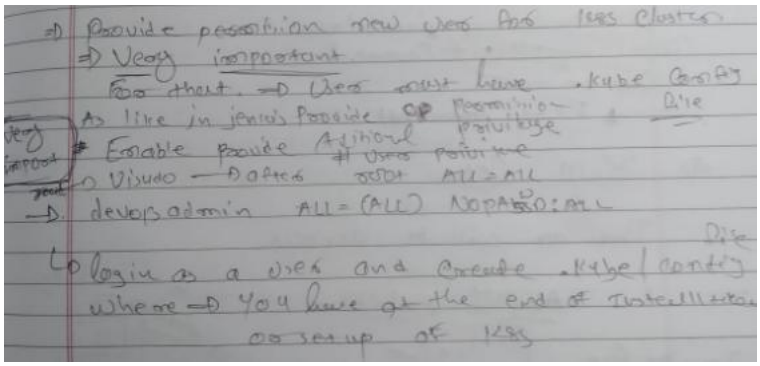
On slave node(Build server):
Create user
--$ useradd devopsadmin -s /bin/bash -m -d /home/devopsadmin
- $ su - devopsadmin

From new user
Create ssh key --> why?--> THis will provide to master node for connection
--$ ssh-keygen -t ecdsa -b 521

Create authorized_keys --> why?--> this key use for authontication purpose
--$ cat id_ecdsa.pub > authorized_keys

Change permission of keys --> why?--> It is read by grops that's don't want.
-- $ chmod 600 /home/devopsadmin/.ssh/*

TO access docker by user we have to add user to docker group why?--> That's how user will intrect with docker.
-- $ usermod -aG docker devopsadmin



====== Still here==> slave Node ready to connect with matser====
It wil connect with new user authorized key
==========

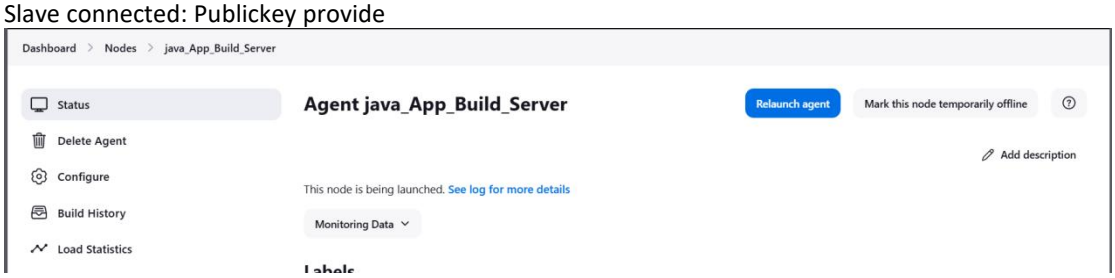
Login to jenkins Dashboard and attach jenkins slave to master node

==========

1. 1 Configure Jenkins slave to Master

```
16  cat /var/lib/jenkins/secrets/initialAdminPassword
    cd /var/lib/jenkins/plugins/
```

Create agent and provide devopsadmin user ssh key pattern to connect to slave:
Slave connected: Publickey provide



Done

:**Create pipeline job and write pipeline script:**

For scm checkout and application==> on slave

```
pipeline {
  agent { label 'slave1'}

  stages {
    stage('SCM_Checkout') {
      steps {
        echo 'Perfrom SCM Checkout'
        git 'https://github.com/maulik2311/devops-javamvn-webapp.git'
      }
    }
    stage('Application Build') {
      steps {
        echo 'Perfrom build Action'
        sh 'mvn clean package'
      }
    }
  }
}
```

```
Started by user Maulik DEVANI
[Pipeline] Start of Pipeline
[Pipeline] node
Running on java_App_Build_Server in /home/devopsadmin/workspace/Project1
[Pipeline] {
[Pipeline] stage
```

```
[1;34mINFO[m] The original artifact has been renamed to /home/devopsadmin/workspace/Project1/target/demo-1.0-
SNAPSHOT.war.original
[1;34mINFO[m] [1m------------------------------------------------------------------------[m
[1;34mINFO[m] [1;32mBUILD SUCCESS[m
[1;34mINFO[m] [1m------------------------------------------------------------------------[m
[1;34mINFO[m] Total time:  28.724 s
[1;34mINFO[m] Finished at: 2024-06-23T16:08:21Z
[1;34mINFO[m] [1m------------------------------------------------------------------------[m
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

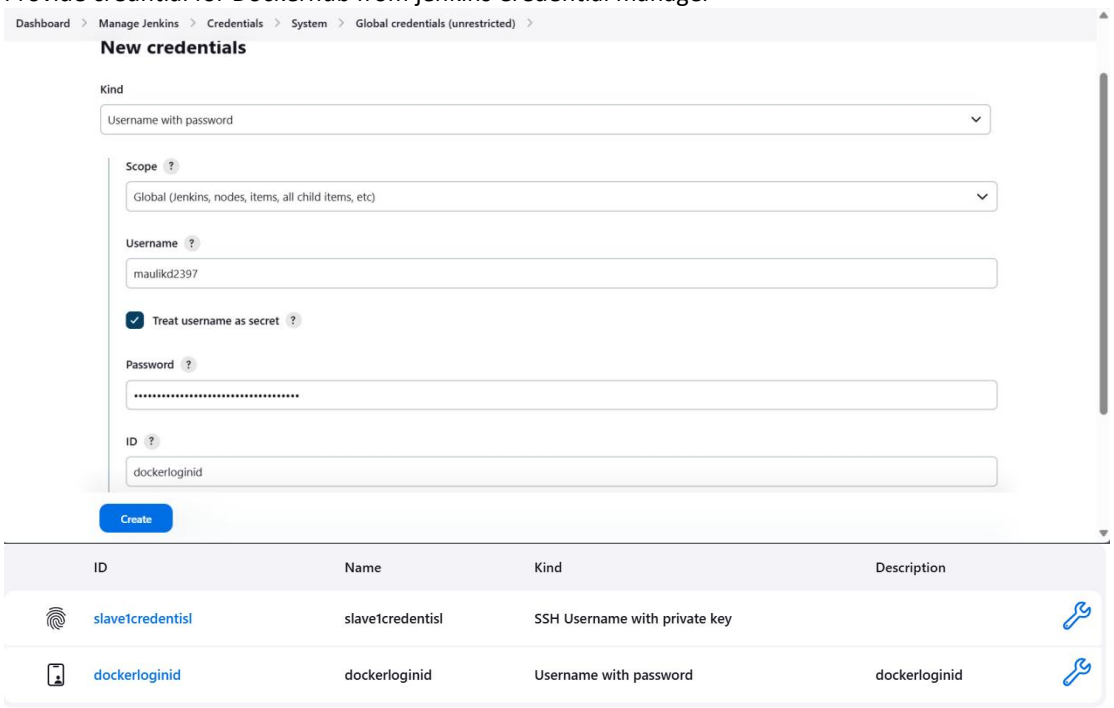
```
root@ip-172-31-81-26:/home/devopsadmin/workspace/Project1# ls
Dockerfile  bin  jenkinsfile  k8smvndeployment.yaml  mvnw  mvnw.cmd  pom.xml  src  target
```

==========Now Create Docker image with pipeline==========

```
pipeline {
  agent { label 'slave1'}

  stages {
    stage('SCM_Checkout') {
      steps {
        echo 'Perfrom SCM Checkout'
        git 'https://github.com/maulik2311/devops-javamvn-webapp.git'
      }
    }
    stage('Application Build') {
      steps {
        echo 'Perfrom build Action'
        sh 'mvn clean package'
      }
    }
    stage('Build Docker image') {
      steps {
        echo 'Building docker image'
        sh 'docker version'
        sh "docker build -t maulikd2397/maulik-app:${BUILD_NUMBER} ."
        sh 'docker image list'
        sh "docker tag maulikd2397/maulik-app:${BUILD_NUMBER} maulikd2397/maulik-app:latest "
      }
    }
  }
}
```

```
+ docker image list
REPOSITORY                TAG       IMAGE ID       CREATED                  SIZE
maulikd2397/maulik-app    4         43ac952b9185   Less than a second ago   376MB
tomcat                    8.0       ef6a7c98d192   5 years ago              356MB
[Pipeline] sh
+ docker tag maulikd2397/maulik-app:4 maulikd2397/maulik-app:latest
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

+==========Log2dockehub==========
Provide credntial for Dockerhub from jenkins Credential manager



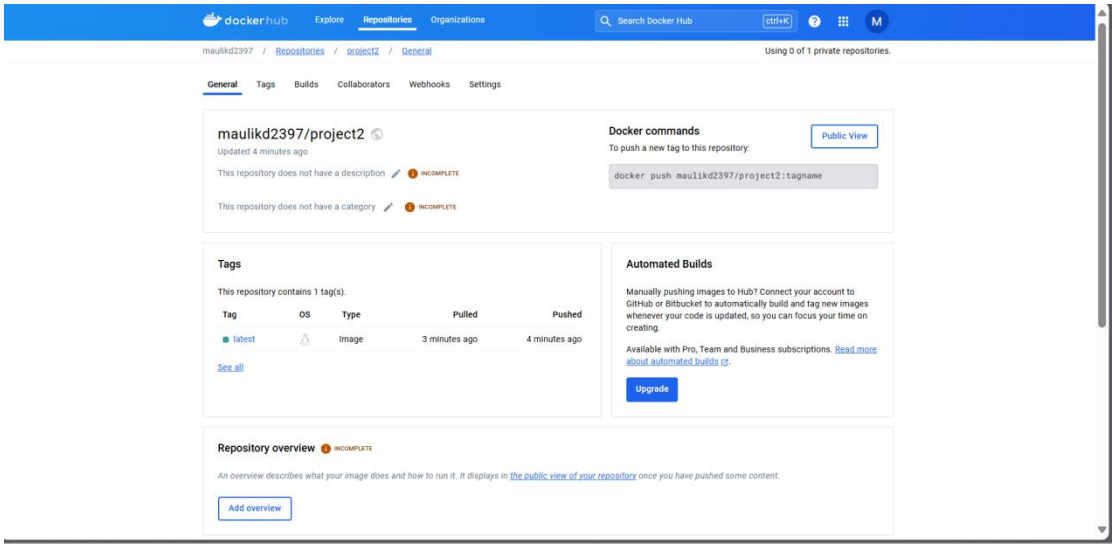
ID	Name	Kind	Description
slave1credentisl	slave1credentisl	SSH Username with private key	
dockerloginid	dockerloginid	Username with password	dockerloginid

Assign
19-06-2024
Continue 30-06

Push container(docker) image to container registry(dockerhub)

```
pipeline {
   agent {label 'slave1'}

   environment {
                        DOCKERHUB_CREDENTIALS=credentials('dockerloginid')
            }
   stages {
      stage('SCM_Chekout') {
         steps {
            git  'https://github.com/maulik2311/devops-javamvn-webapp.git'
         }
      }
      stage('Application_Build') {
         steps {
            sh 'mvn clean package'
         }
      }
      stage('Docker_Image_build') {
         steps {
            sh 'docker version'
            sh "docker build -t maulikd2397/project2:${BUILD_NUMBER} ."
            sh 'docker images'
            sh "docker tag maulikd2397/project2:${BUILD_NUMBER} maulikd2397/project2:latest "
         }
      }
      stage('Login2DockerHub') {

                                 steps {
                                          sh   'echo   $DOCKERHUB_CREDENTIALS_PSW   |   docker   login   -u
$DOCKERHUB_CREDENTIALS_USR --password-stdin'
                                 }
                        }
                        stage('Publish_to_Docker_Registry') {
                                 steps {
                                          sh "docker push maulikd2397/project2:latest"
                                 }
                        }
   }
}
```

**********Push to docker hub**********



Deploy To k8s-==>
21-06

Kubernetes:

To deploy the application manifest used to create and deploy the Service(Nodeport) and Object(Deployment) of k8s

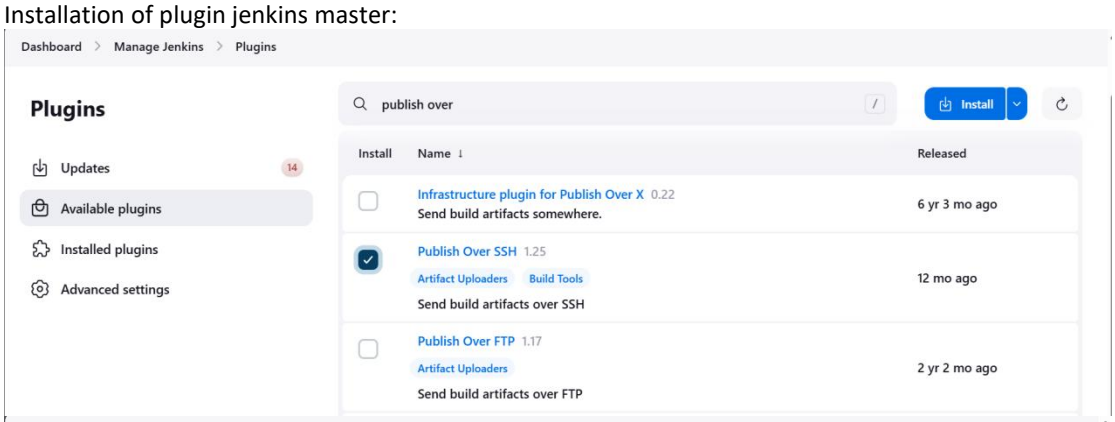
For now; Clone source code have manifets file as well to so copy that file jenkins slave node to k8s master:

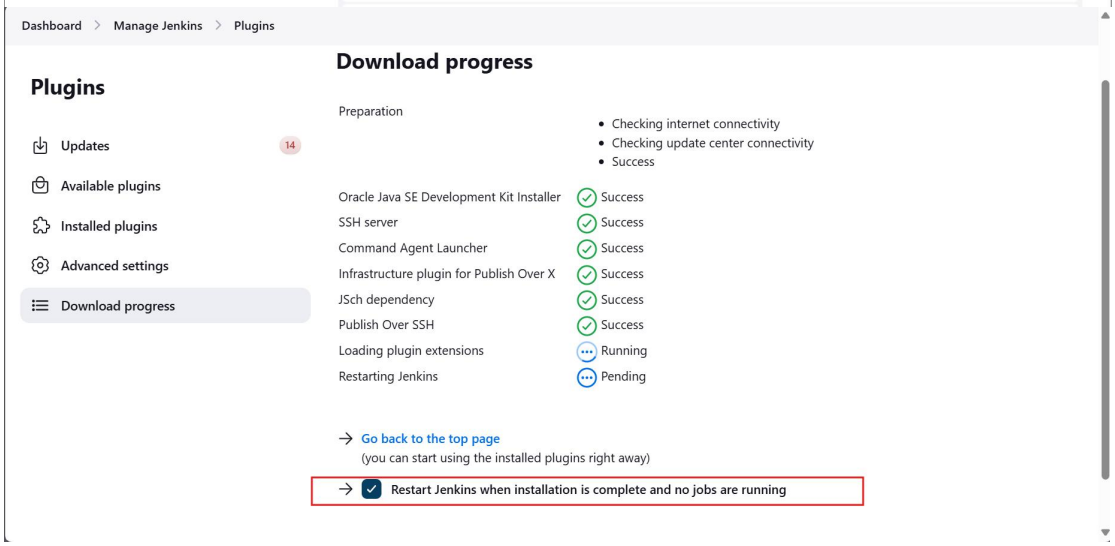
```
#####################
Day 9 - 21st June 2024
#####################

        Infra-Structure to Create CICD Pipeline - (VMs)

            Jenkins_Master                  --> Create Jenkins CICD Pipeline Projects and Schedule it to run in the Slave Node
                Jenkins_Slave(Build_Server) --> Src_Code Repo.
                                                Perform Application Build and Create Artifacts(Binaries - executables - *.war/*.jar)
                                                Build Application Image

            Kubernetes_Master               --> Schedule to Deploy the Application Images  # Target Environment
                Kubernetes_WorkerNode1
                Kubernetes_WorkerNode2



        kubectl create -f <file_name>.yaml          # This Command should be executed in Kubernetes_Master
```

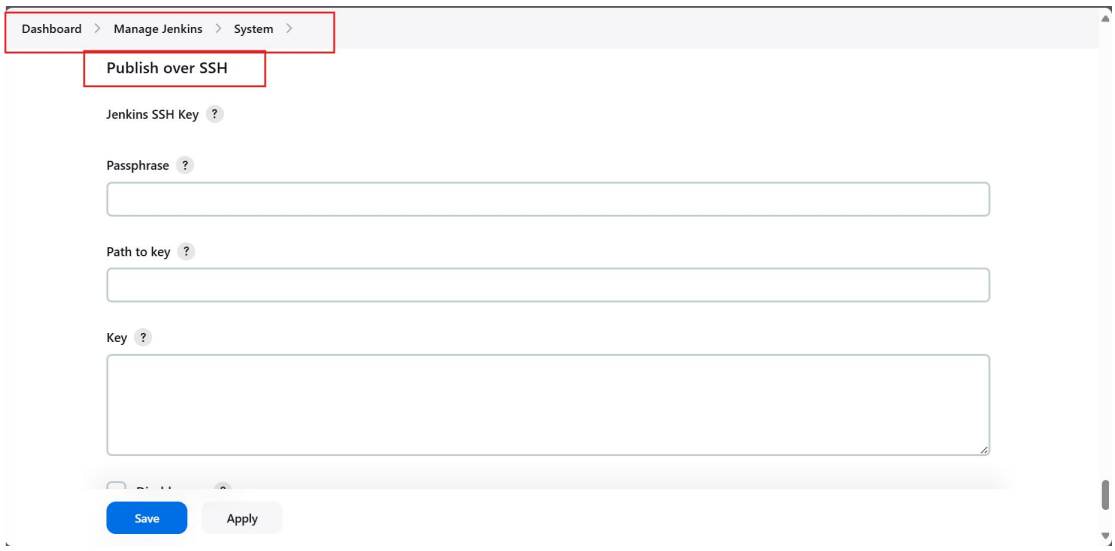
How to copy jenkins slave node file to k8s master?? how??
Using **Publish over SSH plugin**
It means its need key based authentication to attach 2 vms

Install publish over ssh plugin on jenkins
Using this plugin connect to k8s master and copy the menifest file

Installation of plugin jenkins master:







Plugin set

SSH connection Required?==
Host_name
User_name
Credential (Key)

Jenkins slave and master how its connected?
Through==> Add new user, provide privilege

Steps:
Creating user:
-$ useradd devopsadmin -s /bin/bash -m -d /home/devopsadmin

Login as new user
On k8s master:
Genrate ssh key:
-$ ssh-keygen -t ecdsa -b 521

```
devopsadmin@kmaster-node:~$ ssh-keygen -t ecdsa -b 521
Generating public/private ecdsa key pair.
Enter file in which to save the key (/home/devopsadmin/.ssh/id_ecdsa):
Created directory '/home/devopsadmin/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/devopsadmin/.ssh/id_ecdsa
Your public key has been saved in /home/devopsadmin/.ssh/id_ecdsa.pub
The key fingerprint is:
SHA256:hLV6qdKpyOEbBZK84Xw/yct1G/K3VK6y2BH8nUGrGoQ devopsadmin@kmaster-node
The key's randomart image is:
+---[ECDSA 521]---+
|                 |
|               o .|
|o o.    . o     .|
|o.o.   oo.  . . .|
| + .. .ES+   +   |
|  . .O.=.  o = o |
|  o .=*o = + +   |
| o +.o= *.B.     |
|  =.o . *=o.     |
+----[SHA256]-----+
devopsadmin@kmaster-node:~$ ls
devopsadmin@kmaster-node:~$ ll
total 24
drwxr-x--- 3 devopsadmin devopsadmin 4096 Jul  3 14:26 ./
drwxr-xr-x 4 root        root        4096 Jul  3 14:23 ../
-rw-r--r-- 1 devopsadmin devopsadmin  220 Jan  6  2022 .bash_logout
-rw-r--r-- 1 devopsadmin devopsadmin 3771 Jan  6  2022 .bashrc
-rw-r--r-- 1 devopsadmin devopsadmin  807 Jan  6  2022 .profile
drwx------ 2 devopsadmin devopsadmin 4096 Jul  3 14:26 .ssh/
devopsadmin@kmaster-node:~$
```

Create authorized_key from public key:

```
devopsadmin@kmaster-node:~/.ssh$ cat id_ecdsa.pub > authorized_key
devopsadmin@kmaster-node:~/.ssh$ ls
authorized_key  id_ecdsa  id_ecdsa.pub
```

Change the permission of aythorized_key
For only user

```
-rw-rw-r-- 1 devopsadmin devopsadmin  278 Jul  3 14:29 authorized_key
```

-$ chmod 600 /home/devopsadmin/.ssh/*

```
-rw------- 1 devopsadmin devopsadmin  278 Jul  3 14:29 authorized_key
```

Root user:
-$ usermod -aG docker devopsadmin

As a user not able to get nodes because not have permission to attach to cluster: Good with master

```
devopsadmin@kmaster-node:~$ kubectl get nodes
E0703 14:38:28.297083   13745 memcache.go:265] couldn't get current server API group list: Get "http://localhost:8080/api?timeout=32s": dial tcp 127.0.0.
1:8080: connect: connection refused
E0703 14:38:28.300623   13745 memcache.go:265] couldn't get current server API group list: Get "http://localhost:8080/api?timeout=32s": dial tcp 127.0.0.
1:8080: connect: connection refused
E0703 14:38:28.302248   13745 memcache.go:265] couldn't get current server API group list: Get "http://localhost:8080/api?timeout=32s": dial tcp 127.0.0.
1:8080: connect: connection refused
E0703 14:38:28.303236   13745 memcache.go:265] couldn't get current server API group list: Get "http://localhost:8080/api?timeout=32s": dial tcp 127.0.0.
1:8080: connect: connection refused
E0703 14:38:28.304784   13745 memcache.go:265] couldn't get current server API group list: Get "http://localhost:8080/api?timeout=32s": dial tcp 127.0.0.
1:8080: connect: connection refused
The connection to the server localhost:8080 was refused - did you specify the right host or port?
devopsadmin@kmaster-node:~$
```

```
The connection to the server localhost:8080 was refused - did you
devopsadmin@kmaster-node:~$ exit
logout
root@kmaster-node:~# kubectl get nodes
NAME           STATUS   ROLES           AGE   VERSION
kmaster-node   Ready    control-plane   14d   v1.29.6
worker-node1   Ready    <none>          14d   v1.29.6
worker-node2   Ready    <none>          14d   v1.29.6
```

To interact with k8s cluster, user must have .kube config file
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config

To run this user must have privilege to execute this command==> How to provide the privilege==> By editing visudo file ==>
Providing access to system component

Means:: root user have that permission but new user not have that much access
Provide privilege from visudo file
Accessing user to k8s cluster::
Visudo:

```
  GNU nano 6.2                                                    /etc/sudoers.tmp *
#Defaults:%sudo env_keep += "GREP_COLOR"

# While you shouldn't normally run git as root, you need to with etckeeper
#Defaults:%sudo env_keep += "GIT_AUTHOR_* GIT_COMMITTER_*"

# Per-user preferences; root won't have sensible values for them.
#Defaults:%sudo env_keep += "EMAIL DEBEMAIL DEBFULLNAME"

# "sudo scp" or "sudo rsync" should be able to use your SSH agent.
#Defaults:%sudo env_keep += "SSH_AGENT_PID SSH_AUTH_SOCK"

# Ditto for GPG agent
#Defaults:%sudo env_keep += "GPG_AGENT_INFO"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL
devopsadmin ALL=(ALL) NOPASSWD:ALL
# Members of the admin group may gain root privileges
%admin ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "@include" directives:

^G Help      ^O Write Out  ^W Where Is   ^K Cut       ^T Execute    ^C Location   M-U Undo      M-A Set Mark  M-] To Bracket
^X Exit      ^R Read File  ^\ Replace    ^U Paste     ^J Justify    ^/ Go To Line M-E Redo      M-6 Copy      ^Q Where Was
```

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

```
devopsadmin@kmaster-node:~$ mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
devopsadmin@kmaster-node:~$ ls -a
.  ..  .bash_history  .bash_logout  .bashrc  .kube  .profile  .ssh
devopsadmin@kmaster-node:~$ ls .kube/
config
devopsadmin@kmaster-node:~$
```

Interview quention:

How you give access to any Linux user to interact with kubernetes?
Ans: Home directory shuod be updated with config file of k8s.(.kube)

Set:

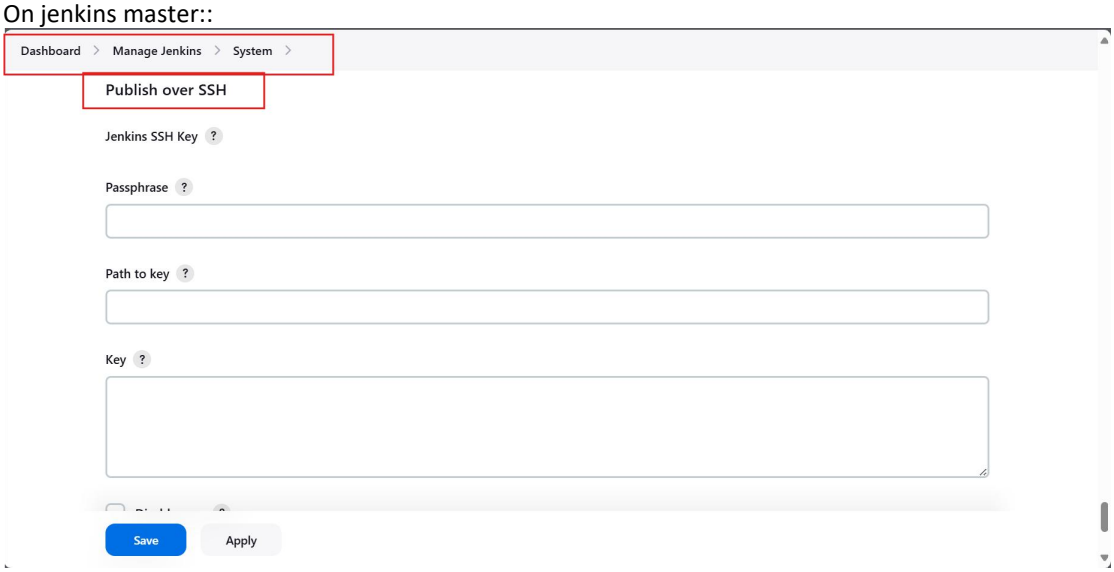
```
devopsadmin@kmaster-node:~$ kubectl get nodes
NAME           STATUS   ROLES           AGE   VERSION
kmaster-node   Ready    control-plane   14d   v1.29.6
worker-node1   Ready    <none>          14d   v1.29.6
worker-node2   Ready    <none>          14d   v1.29.6
devopsadmin@kmaster-node:~$
```

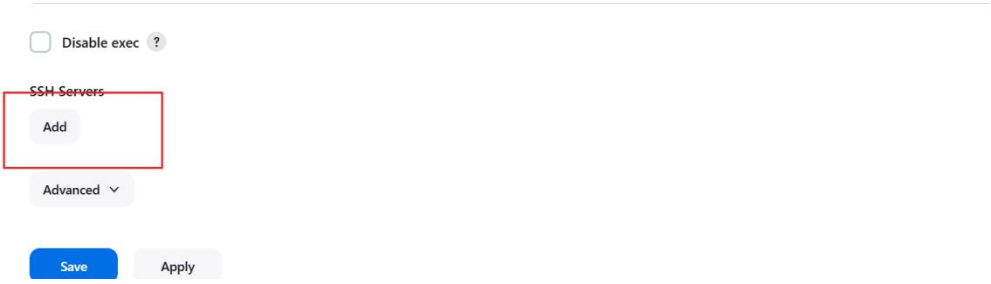
```
devopsadmin@kmaster-node:~/.ssh$ cat id_ecdsa
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAAArAAAABNlY2RzYS
1zaGEyLW5pc3RwNTIxAAAACG5pc3RwNTIxAAAAhQQBImi0RHEqMeUfAnCeM8x7/m+DMwsp
vzdUBPAMJYYC9edNEu5zgtgXB3MIwi/WJGN/2yNRKR4NG1l41LN+OfxsVb4BbG170cbmjS
00AN2WL2Xf5Px6p/JBEdwMtzj1qqM8+hsFnBqQ1i/RiAESoqHZBNguIo7YyRU5zRDPuJiK
Q2IIHdwAAAEYQ41JU0ONSVMAAAATZWNkc2Etc2hhMi1uaXN0cDUyMQAAAAhuaXN0cDUyMQ
AAAIUEASJotERxKjHlHwJwnjPMe/5vgzMLKb83VATwDCWGAvXnTRLuc4LYFwdzCMIv1iRj
f9sjUSkeDRtZeNSzfjn8bFW+AWxte9HG5o0tNADdli9l3+T8eqfyQRHcDLc49aqjPPobBZ
wakNYv0YgBEqKh2QTYLiKO2MkVOc0Qz7iYikNiCB3cAAAAQgGaCbR4K/qDtbtaV+IYA3QV
p3ezS9IMimio/qemjAzCwtVHp5nOA910kbsVORSV0QLRBMJeIoKILFFz05NLeYU6IQAAAB
hkZXZvcHNhZG1pbkBrbWFzdGVyLW5vZGUBAg==
-----END OPENSSH PRIVATE KEY-----
```

Till we have keys and set new user with required privilege

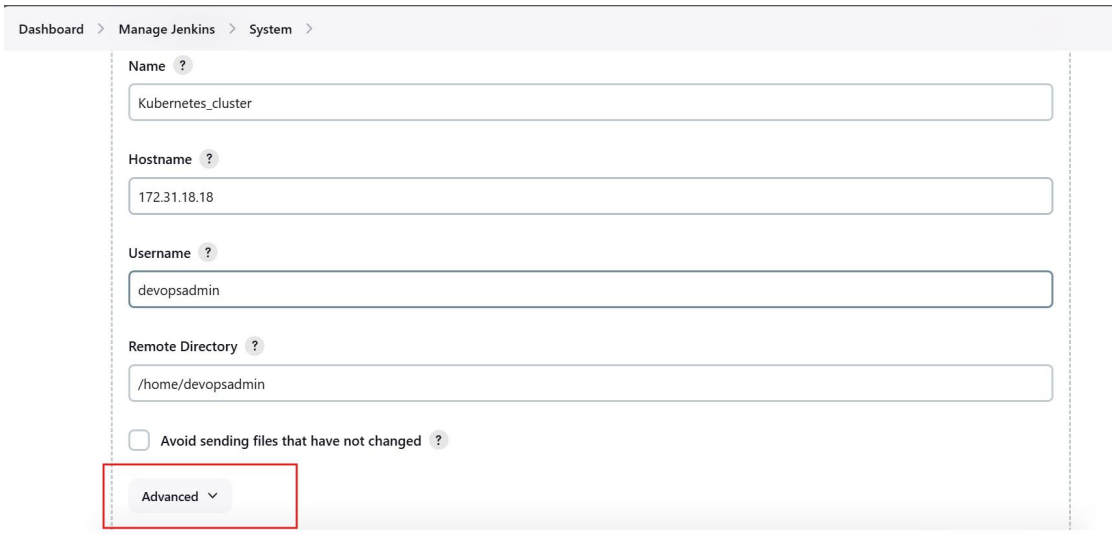
Next::

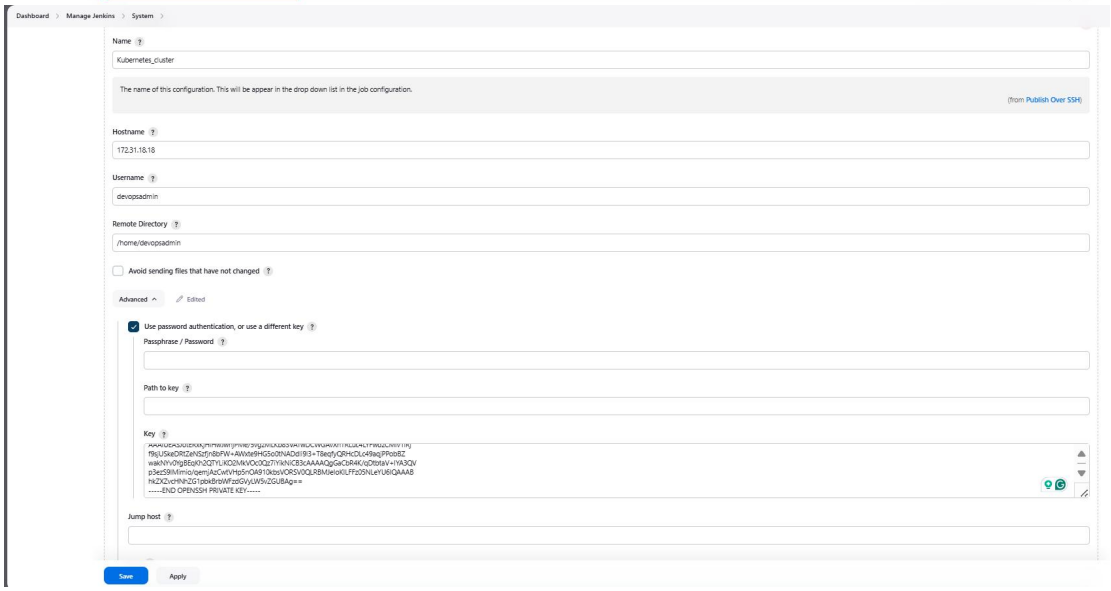
On jenkins master::

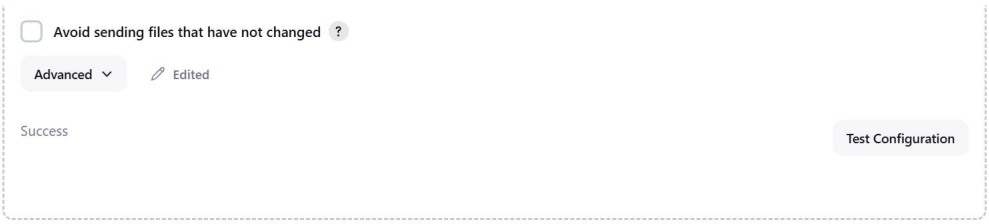




Name= anything
Hostname== Privateip k8s master
Username: Server username
Remote Directory== Path of execution

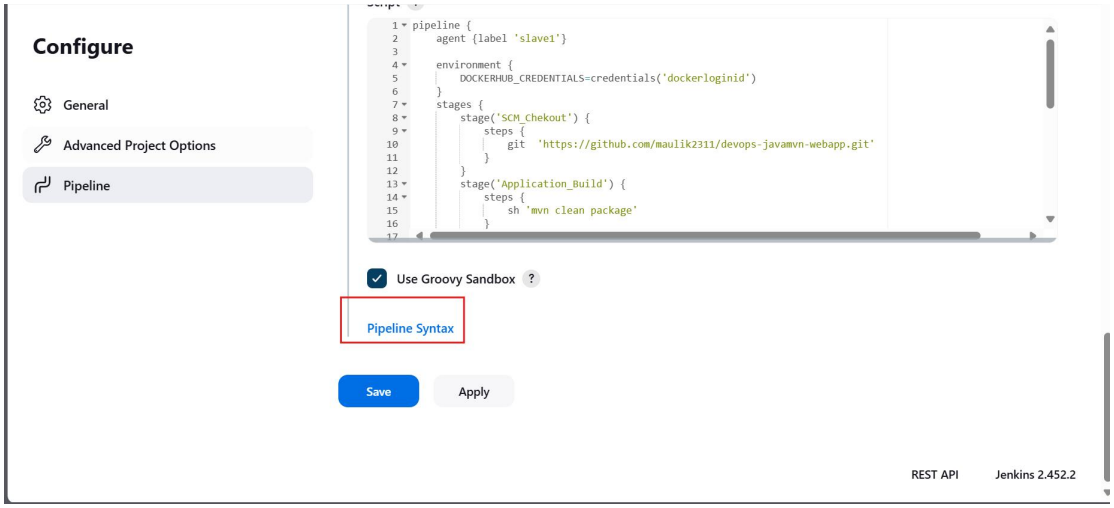




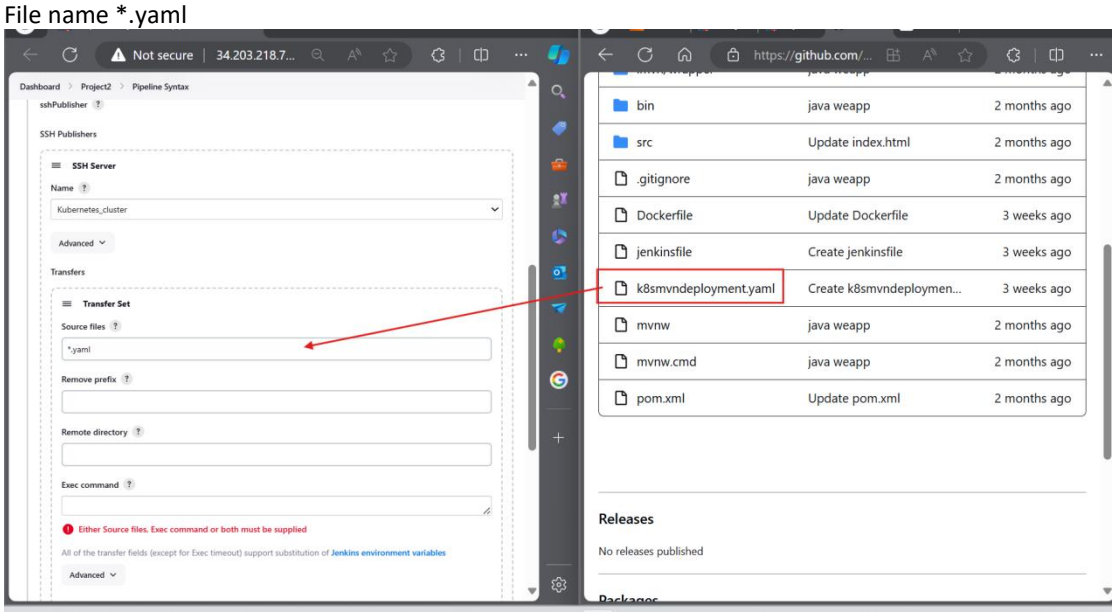


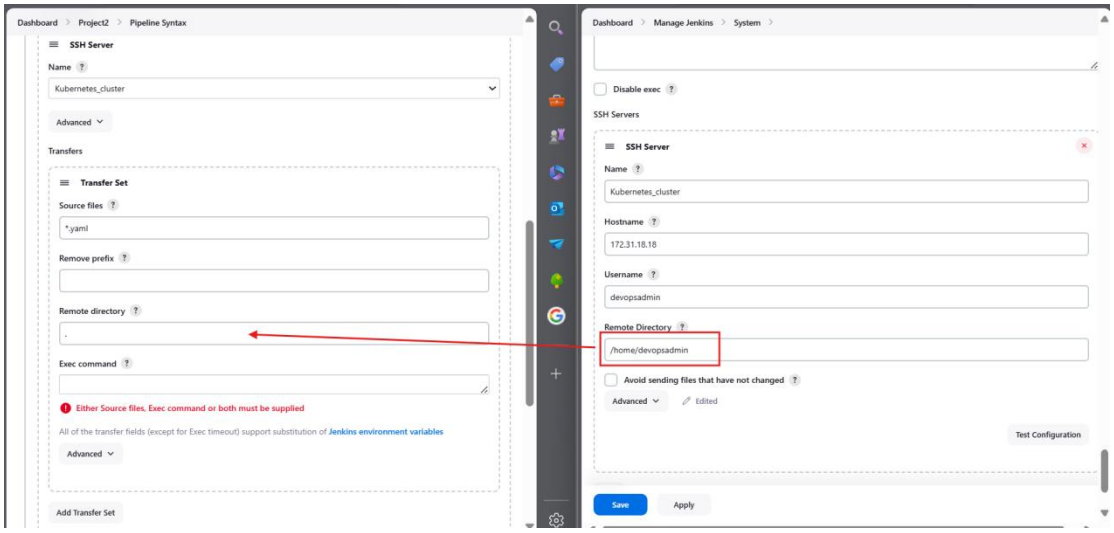
Now write a script for copy the artifacts and paste and deploy the object and service

It will take manifets file from slave node of jenkins and apply on k8s master

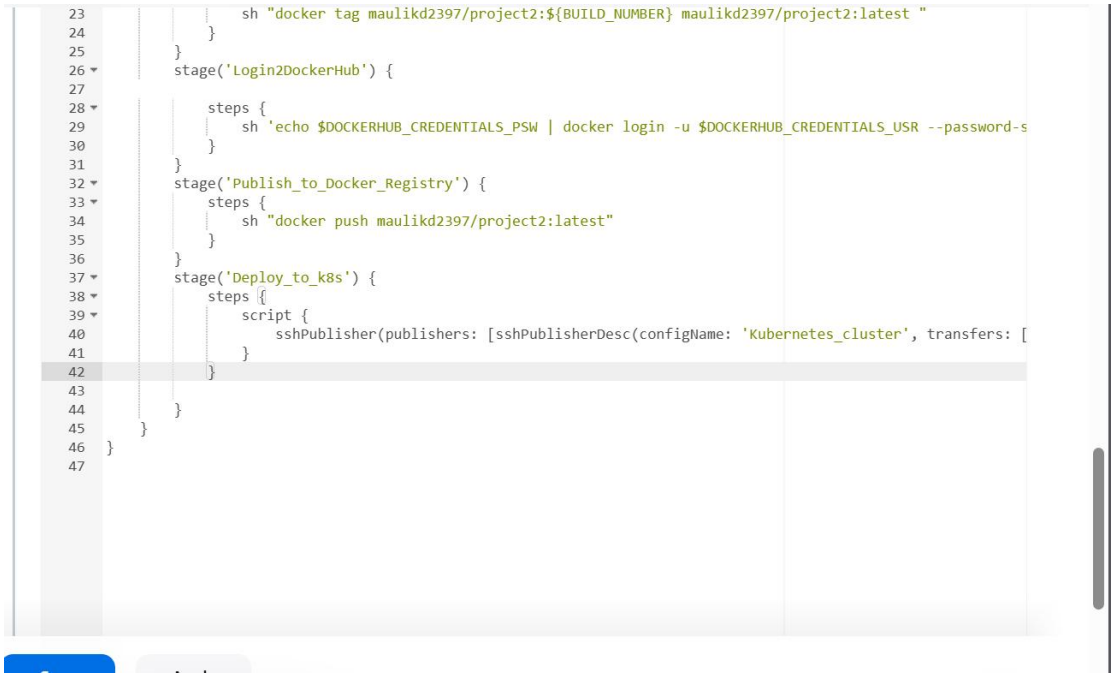


File name *.yaml





Provide command that need to execute while run the job
Create will try to create deploy new object that will head to error
Apply will update and if the deployment as exist, it will do nothing

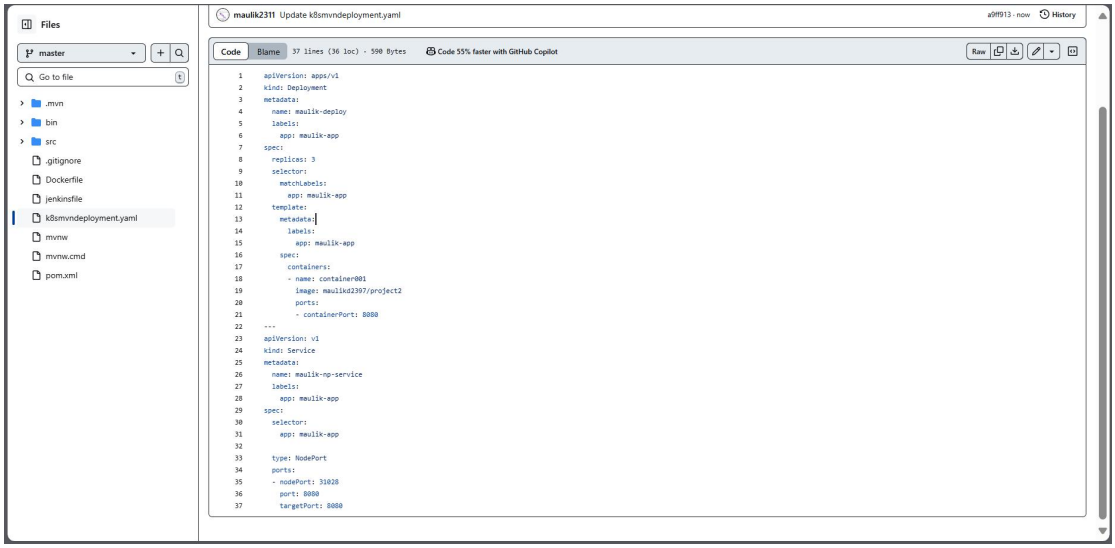


```
            sh "docker tag maulikd2397/project2:${BUILD_NUMBER} maulikd2397/project2:latest "
        }
    }
    stage('Login2DockerHub') {

        steps {
            sh 'echo $DOCKERHUB_CREDENTIALS_PSW | docker login -u $DOCKERHUB_CREDENTIALS_USR --password-s
        }
    }
    stage('Publish_to_Docker_Registry') {
        steps {
            sh "docker push maulikd2397/project2:latest"
        }
    }
    stage('Deploy_to_k8s') {
        steps {
            script {
                sshPublisher(publishers: [sshPublisherDesc(configName: 'Kubernetes_cluster', transfers: [
            }
        }

    }
  }
}
```

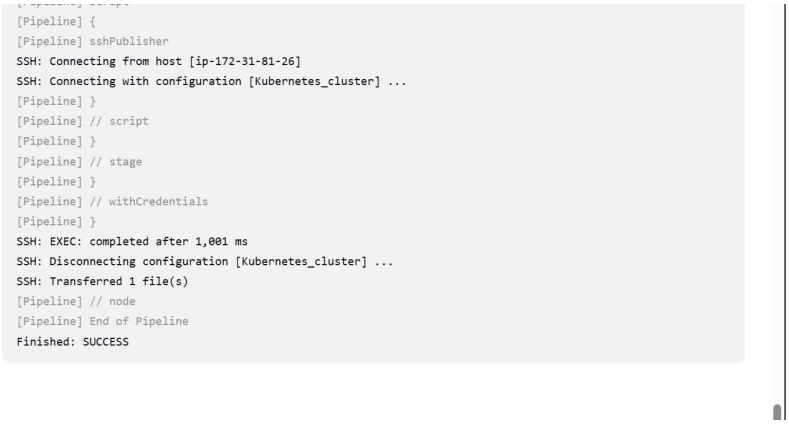
Entire jenkins pipeline script:

```
pipeline {
   agent {label 'slave1'}

   environment {
                        DOCKERHUB_CREDENTIALS=credentials('dockerloginid')
            }
   stages {
      stage('SCM_Chekout') {
         steps {
            git  'https://github.com/maulik2311/devops-javamvn-webapp.git'
         }
      }
      stage('Application_Build') {
         steps {
            sh 'mvn clean package'
         }
      }
      stage('Docker_Image_build') {
         steps {
            sh 'docker version'
            sh "docker build -t maulikd2397/project2:${BUILD_NUMBER} ."
            sh 'docker images'
            sh "docker tag maulikd2397/project2:${BUILD_NUMBER} maulikd2397/project2:latest "
         }
      }
      stage('Login2DockerHub') {

                              steps {
                                       sh  'echo  $DOCKERHUB_CREDENTIALS_PSW  |  docker  login  -u
$DOCKERHUB_CREDENTIALS_USR --password-stdin'
                              }
                        }
                        stage('Publish_to_Docker_Registry') {
                              steps {
                                       sh "docker push maulikd2397/project2:latest"
                              }
                        }
                        stage('Deploy_to_k8s') {
                              steps {
                                       script {
                                                sshPublisher(publishers:  [sshPublisherDesc(configName:
'Kubernetes_cluster', transfers: [sshTransfer(cleanRemote: false, excludes: '', execCommand: 'kubectl
apply  -f  k8smvndeployment.yaml',  execTimeout:  120000,  flatten:  false,  makeEmptyDirs:  false,
noDefaultExcludes: false, patternSeparator: '[, ]+', remoteDirectory: '.', remoteDirectorySDF: false,
removePrefix: '', sourceFiles: '*.yaml')], usePromotionTimestamp: false, useWorkspaceInPromotion:
false, verbose: false)])
                                       }
                              }

                        }
   }
}
```

Source file edited



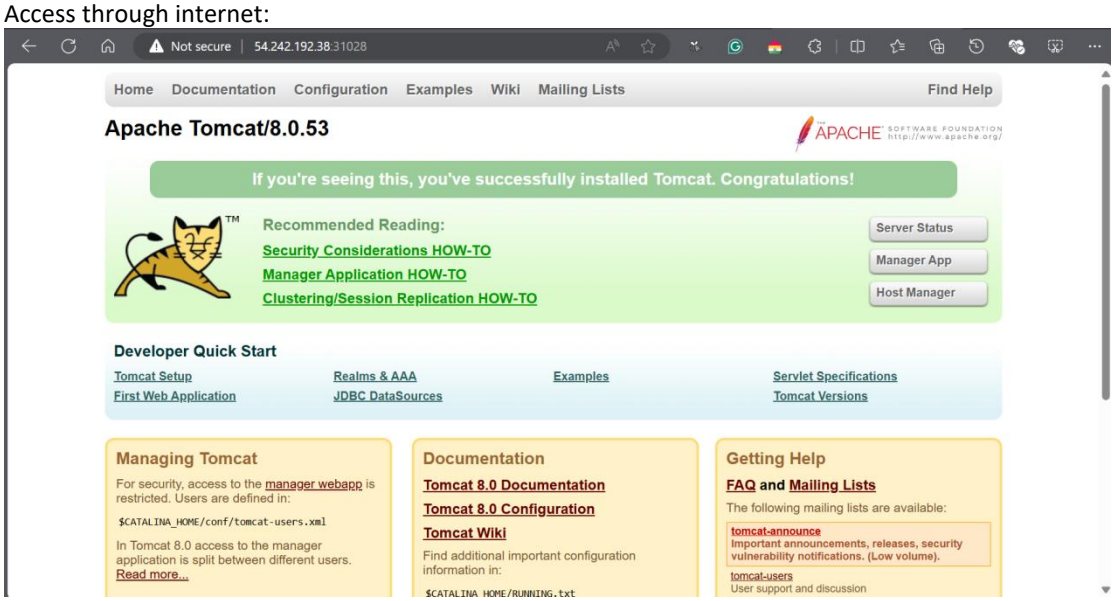
Job success:



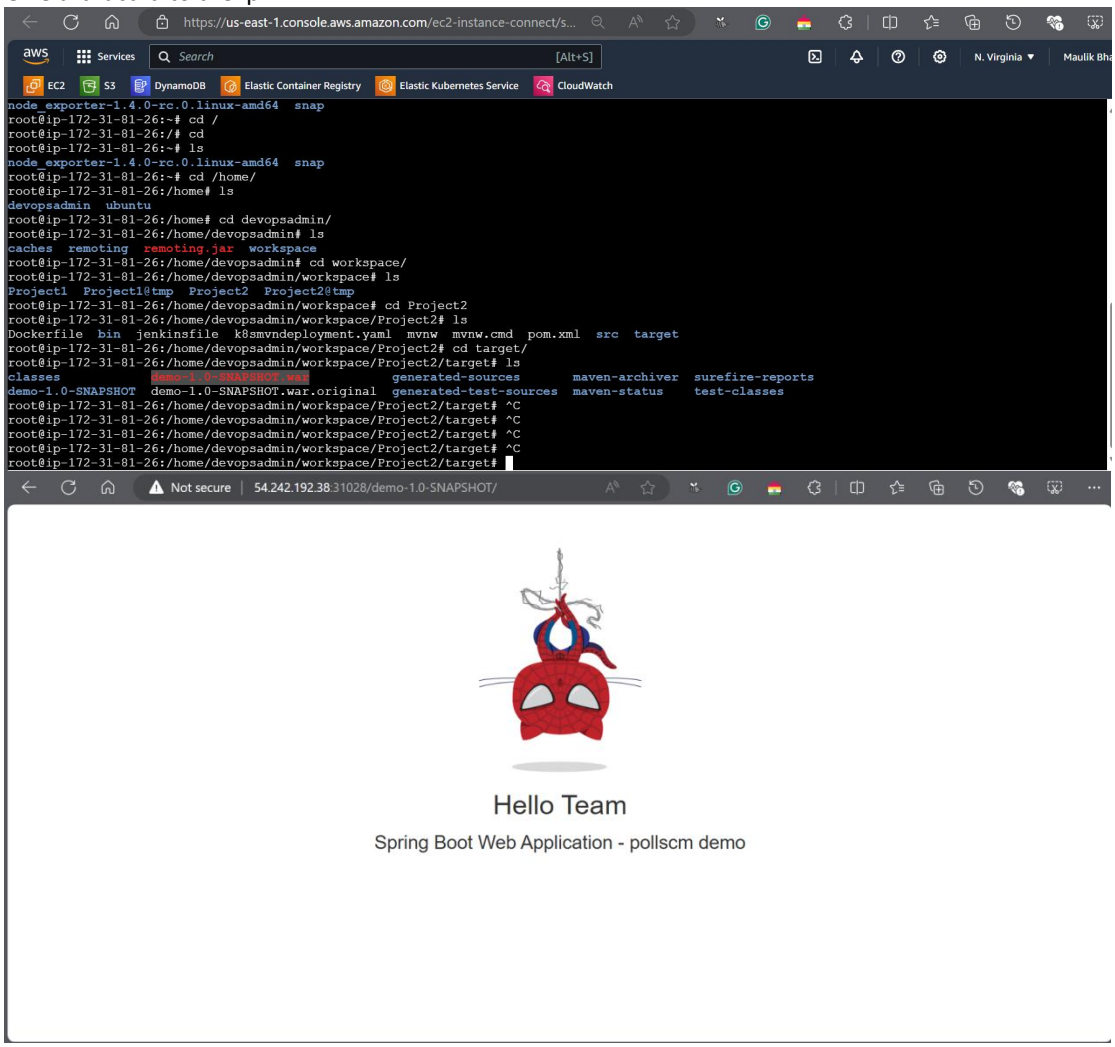
Get info from K8s master:;; Deployment took place along with services

```
root@kmaster-node:~# kubectl get nodes
NAME          STATUS   ROLES           AGE   VERSION
kmaster-node  Ready    control-plane   14d   v1.29.6
worker-node1  Ready    <none>          14d   v1.29.6
worker-node2  Ready    <none>          14d   v1.29.6
root@kmaster-node:~# kubectk get pods -o wide
Command 'kubectk' not found, did you mean:
  command 'kubectx' from snap kubectx (0.9.5)
  command 'kubectl' from snap kubectl (1.29.6)
See 'snap info <snapname>' for additional versions.
root@kmaster-node:~# kubectl get pods -o wide
NAME                          READY   STATUS    RESTARTS       AGE    IP            NODE          NOMINATED NODE   READINESS GATES
maulik-deploy-7844855d99-gc9sl   1/1     Running   0              103s   10.244.1.34   worker-node1  <none>           <none>
maulik-deploy-7844855d99-jkftw   1/1     Running   0              103s   10.244.2.36   worker-node2  <none>           <none>
maulik-deploy-7844855d99-mkvk8   1/1     Running   0              103s   10.244.1.35   worker-node1  <none>           <none>
newdeploy-5c474648bc-7wqjm       1/1     Running   1 (141m ago)   11d    10.244.1.33   worker-node1  <none>           <none>
newdeploy-5c474648bc-ss58n       1/1     Running   1 (141m ago)   11d    10.244.2.34   worker-node2  <none>           <none>
newdeploy-5c474648bc-v8mlg       1/1     Running   1 (141m ago)   11d    10.244.2.35   worker-node2  <none>           <none>
springboot-5f7bc787b-wbb6x       1/1     Running   1 (141m ago)   10d    10.244.1.32   worker-node1  <none>           <none>
root@kmaster-node:~# kubectl get deploy
NAME            READY   UP-TO-DATE   AVAILABLE   AGE
maulik-deploy   3/3     3            3           116s
newdeploy       3/3     3            3           11d
springboot      1/1     1            1           10d
root@kmaster-node:~# kubectl get sv
error: the server doesn't have a resource type "sv"
root@kmaster-node:~# kubectl get service
NAME                TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
kubernetes          ClusterIP   10.96.0.1        <none>        443/TCP          14d
maulik-np-service   NodePort    10.107.196.230   <none>        8080:31028/TCP   2m18s
newdeploy           NodePort    10.96.87.85      <none>        8080:30716/TCP   11d
springboot          NodePort    10.104.166.57    <none>        8080:32201/TCP   10d
root@kmaster-node:~#
```

Access through internet:



Give artifact id to the ip:



Access through both server = High availability