Machine 1: Jenkins_Terraform_Ansible
Machine2: Kmaster
Machine3: Kslave1
Machine4: Kslave2

Create 1 instance named “Jenkins_Terraform_Ansible >

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 -y

sudo nano main.tf

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

resource "aws_instance" "K8-M" {
  ami = ""
  instance_type = "t2.medium"
  key_name = ""
  tags = {
    Name = "Kmaster"
  }
}

resource "aws_instance" "K8-S1" {
  ami = ""
  instance_type = "t2.medium"
  key_name = ""
  tags = {
    Name = "Kslave1"
  }
```

```
}

resource "aws_instance" "K8-S2" {
  ami = ""
  instance_type = "t2.medium"
  key_name = ""
  tags = {
    Name = "Kslave2"
  }
}
```

Terraform init
Terraform plan
Terraform apply

You will get to see 3 ec2 instances running.

Install Ansible on the Jenkins_Terraform_Ansible

sudo apt update
sudo apt install software-properties-common
sudo add-apt-repository --yes --update ppa:ansible/ansible
sudo apt install ansible -y

Now create a cluster for KMaster
On Master:
Ssh-keygen
Sudo cat _____
Copy ssh key

Go on Kmaster
Cd .ssh
Sudo nano authorized_keys
Paste ssh key

Go on Master:
Cd /etc/ansible
Ls
Sudo nano hosts
[test]
Private Ip of Kmaster
Ansible -m ping all

Playbook Syntax:

Sudo nano playbook.yaml

```
---
- name: Installations on Master
  hosts: localhost
  become: true
  tasks:
  - name: Executing script on master
    script: Jenkins_terraform_ansible.sh

- name: Installations on test
  hosts: test
  become: true
  tasks:
  - name: Executing script on test
    script: K-master.sh
```

Jenkins_terraform_ansible.sh:

```
sudo apt update
sudo apt install openjdk-17-jre -y
sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
```

https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key

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 -y

sudo apt-get install docker.io -y

Kmaster.sh

sudo apt update

sudo apt install openjdk-17-jre -y

sudo apt install docker.io -y

RUN PLAYBOOK :

ansible-playbook play.yaml –synatax-check

ansible-playbook play.yaml --check

ansible-playbook play.yaml

Open the Github repository and fork it.

Clone this new repository > Go to folder > Create a Dockerfile

Dockerfile syntax

FROM ubuntu

RUN apt update

RUN apt-get install apache2 -y

```
ADD . /var/www/html

ENTRYPOINT apachectl -D FOREGROUND
```

Create deploy.yml file

```
apiVersion: apps/v1

kind: Deployment

metadata:

  name: custom-deployment

  labels:

    app: custom

spec:

  replicas: 2

  selector:

    matchLabels:

      app: custom

  template:

    metadata:

      labels:

        app: custom

    spec:

      containers:

      - name: custom

        image: docker6767/image

        ports:

        - containerPort: 80
```

Create SVC.yml

```
apiVersion: v1
kind: Service
metadata:
  name: my-custom-deployment
spec:
  type: NodePort
  ports:
    - targetPort: 80
      port: 80
      nodePort: 30008
  selector:
    app: custom
```

Git status

Git add .

Git commit -m “add dockerfile”

Git branch > you will get a master branch

Open Jenkins Dashboard and add K-Master machine.

Go to Credentials, Click on Global , Add credential with Dockerhub username and password and Save it. Yo will get a docker id which will be used at below-highlighted space.

Create a Pipeine job

```
pipeline{
   agent none
   environment {

DOCKERHUB_CREDENTIALS=credentials('cba30343-cfdb-4b74-9ddd-518485437254
')
   }


   stages{
      stage('Hello'){
         agent{
            label 'KMaster'
         }
         steps{
            echo 'Hello World'
         }
      }
      stage('git'){
         agent{
            label 'KMaster'
         }


         steps{


            git'https://github.com/Intellipaat-Training/Test.git'
```

```
    }
  }
  stage('docker') {
    agent {
      label 'KMaster'
    }

    steps {

      sh 'sudo docker build /home/ubuntu/jenkins/workspace/FinalProject -t docker6767/image'
      sh 'sudo echo $DOCKERHUB_CREDENTIALS_PSW | sudo docker login -u $DOCKERHUB_CREDENTIALS_USR --password-stdin'
      sh 'sudo docker push docker6767/image'

    }
  }
  stage('Kubernetes') {
    agent {
      label 'KMaster'
    }

    steps {

      sh 'kubectl create -f deploy.yml'
```

```
            sh 'kubectl create -f svc.yml'
        }
    }


  }
}
```

Install Kubernetes on KMaster. Launch remaining two machines(Kslave1 and Kslave2) and connect it as slaves

Go and run the Job by pushing the Master branch!