**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" —-# make sure region is correct

}

resource "aws\_instance" "K8-M" {

ami = "ami-0da7657fe73215c0c"

instance\_type = "t2.medium"

key\_name = "Aksith-Don't delete" —#pass your key

tags = {

Name = "Kmaster" #instance 2

}

}

resource "aws\_instance" "K8-S1" {

ami = "ami-0da7657fe73215c0c"

instance\_type = "t2.micro"

key\_name = "Aksith-Don't delete"

tags = {

Name = "Kslave1" #Instance3

}

}

resource "aws\_instance" "K8-S2" {

ami = "ami-0da7657fe73215c0c"

instance\_type = "t2.micro"

key\_name = "Aksith-Don't delete"

tags = {

Name = "Kslave2" #instance4

}

}

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

**Go to host and just add Kmaster**

**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 Kmaster

hosts: Kmaster

become: true

tasks:

- name: Executing script on Kmaster

script: k-master.sh

**RUN PLAYBOOK :** ansible-playbook <nameoffile>

**Jenkins\_terraform\_ansible.sh:**

sudo apt update

sudo apt-get install openjdk-11-jdk -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-11-jdk -y

sudo apt install docker.io -y

**Once ansible is done!**

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('7ae8b9a4-000a-4515-a9f7-4950cbe8696b')

}

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**

**On Master and Slave**

sudo apt-get install docker.io -y

sudo apt install -y curl apt-transport-https ca-certificates software-properties-common

curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -

sudo add-apt-repository "deb http://apt.kubernetes.io/ kubernetes-xenial main"

sudo swapoff -a

sudo apt update

sudo apt install -y kubelet kubeadm kubectl

sudo kubeadm init

**Paste the token in slave:**

Sudo <token>

**On master**

mkdir -p $HOME/.kube

sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

sudo chown $(id -u):$(id -g) $HOME/.kube/config

kubectl apply -f <https://github.com/weaveworks/weave/releases/download/v2.8.1/weave-daemonset-k8s.yaml>

Kubectl get nodes

**Go and run the Job by pushing the Master branch!**