Create single t2.medium machine in aws

sudo su

**########Now install docker###############**

sudo apt update && apt -y install docker.io

**###install Kubectl###**

curl -LO https://storage.googleapis.com/kubernetes-release/release/$(curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt)/bin/linux/amd64/kubectl && chmod +x ./kubectl && sudo mv ./kubectl /usr/local/bin/kubectl

**###install Minikube####**

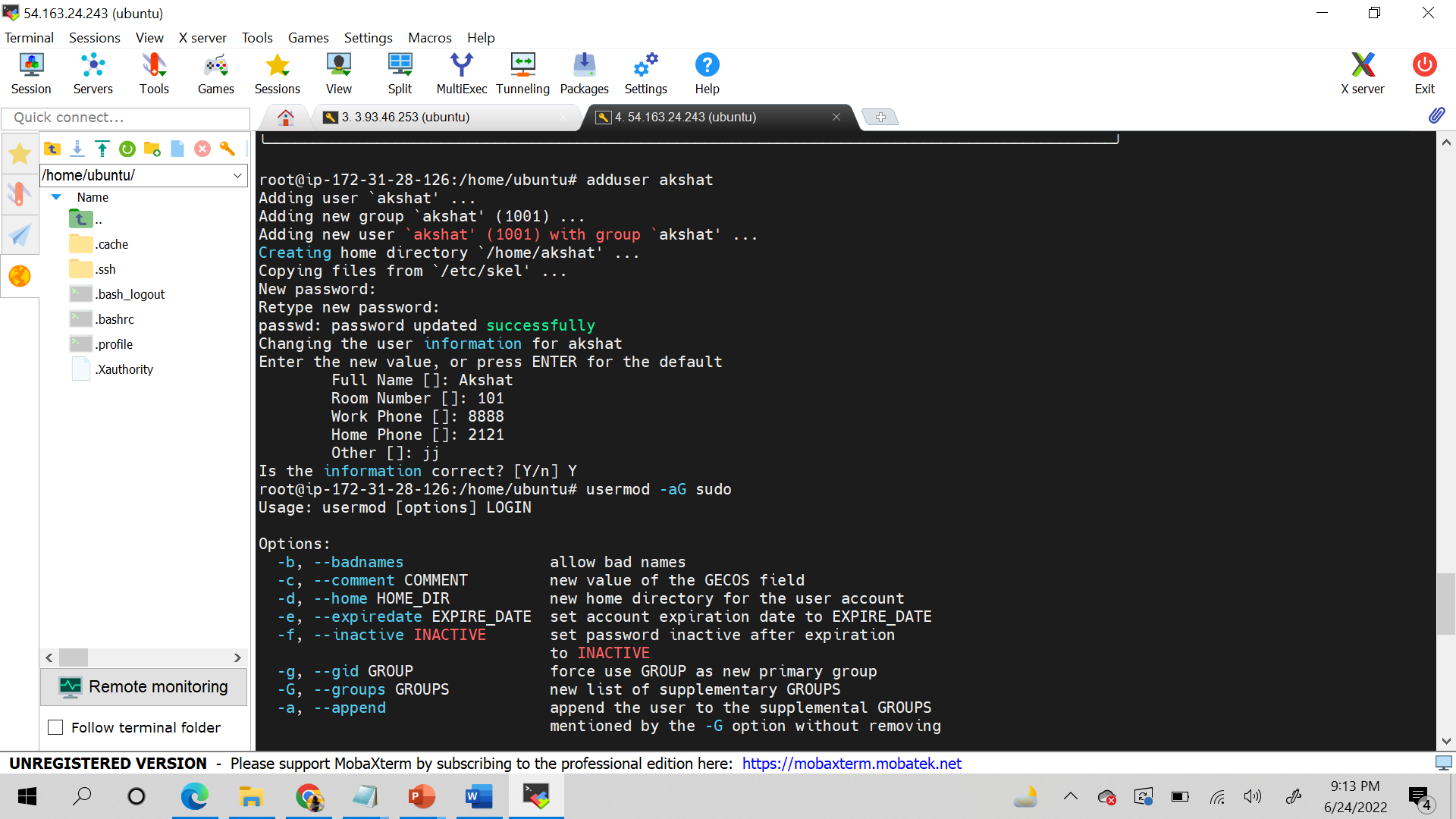
curl -Lo minikube https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64 && chmod +x minikube && sudo mv minikube /usr/local/bin/

**##Start minikube####**

apt install conntrack

#Create user and add it to docker

adduser akshat



usermod -aG sudo akshat

su - akshat

sudo groupadd docker

sudo usermod -aG docker $USER && newgrp docker

minikube start --vm-driver=docker

minikube status

vi pod1.yml \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* press i \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

kind: Pod

apiVersion: v1

metadata:

name: testpod

annotations:

description: Our first testing pod

spec:

containers:

- name: c00

image: ubuntu

command: ["/bin/bash", "-c", "while true; do echo Test Message; sleep 5 ; done"]

restartPolicy: Never # Defaults to Always

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Press esc and then :wq \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

kubectl apply -f pod1.yml

Now ,  
kubectl get pods

(you will find a pod with the name testpod)

kubectl get pods -o wide

kubectl describe pod testpod

kubectl logs -f testpod (to see what is running in container)

kubectl logs -f testpod -c c00 (c00 is the container name as mentioned in the pod1.yml file )

kubectl get nodes

kubectl exec testpod -it -c c00 -- /bin/bash (to enter inside the container)

\*\*After your enter container \*\*\*

Ps

Ps -ef

exit

\*\*\*\*Delete a pod\*\*\*\*\*\*\*\*\*\*\*\*\*

kubectl delete pod testpod

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**\*\*\*\*\*\*\*\*MULTI CONTAINER POD ENVIRONMENT (If you want to create multiple container in a single pod)\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*vi testpod3.yml \*\*\***

kind: Pod

apiVersion: v1

metadata:

name: testpod3

spec:

containers:

- name: c00

image: ubuntu

command: ["/bin/bash", "-c", "while true; do echo My first message; sleep 5 ; done"]

- name: c01

image: ubuntu

command: ["/bin/bash", "-c", "while true; do echo Hello-Devops; sleep 5 ; done"]

\*\*then escape and :wq\*\*

kubectl apply -f testpod3.yml

Now ,  
kubectl get pods

(you will find a pod with the name testpod)

kubectl get pods -o wide

kubectl describe pod testpod3

kubectl logs -f testpod3 c00 (to see what is running in container)

kubectl logs -f testpod3 c01

kubectl get nodes

kubectl exec testpod3 -it -c c00 -- /bin/bash (to enter inside the container)

\*\*After your enter container \*\*\*

Ps

Ps -ef

exit

\*\*\*\*Delete a pod\*\*\*\*\*\*\*\*\*\*\*\*\*

kubectl delete pod testpod3

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

LAB 3: POD ENVIRONMENT VARIABLES

vi pod3.yml

press i

kind: Pod

apiVersion: v1

metadata:

name: environments1

spec:

containers:

- name: c00

image: ubuntu

command: ["/bin/bash", "-c", "while true; do echo Hello-kubernetes; sleep 5 ; done"]

env:

- name: state

value: production

\*\*\*\*\*\*\*\*\*\*\*now press escape and press :wq \*\*\*\*\*\*\*\*\*\*\*\*

kubectl apply -f testpod2.yml

Now ,  
kubectl get pods

(you will find all the pods in your node)

kubectl get pods -o wide

kubectl describe pod environments1

kubectl logs -f environments c00 (to see what is running in container)

kubectl get nodes

kubectl exec environments -it -c c00 -- /bin/bash (to enter inside the container)

\*\*After your enter container \*\*\*

Env

Echo $MYNAME

exit

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**LAB 5\*\*\*\*\*\*\*EXAMPLE OF LABELS\*\*\*\*\*\*\*\*\*\***

Vi pod7.yml

kind: Pod

apiVersion: v1

metadata:

name: testlabel

labels:

env: development

class: pods

spec:

containers:

- name: c00

image: ubuntu

command: ["/bin/bash", "-c", "while true; do echo Welcome to the training lets learn labels in pod; sleep 5 ; done"]

**\*\*Press escape and :wq \*\***

kubectl apply -f pod7.yml

kubectl get pods

kubectl get pods -o wide

kubectl get pods --show-labels

kubectl label pods testlabel env=production (manual command to add a label to existing pod)

kubectl get pods –show —- labels

kubectl get pods -l env=development

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**LAB 6 NODE SELECTOR EXAMPLE (if you want to create pods in a node with specific hardware)\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**vi pod8.yml**

kind: Pod

apiVersion: v1

metadata:

name: nodelabels

labels:

env: development

spec:

containers:

- name: c00

image: ubuntu

command: ["/bin/bash", "-c", "while true; do echo Hello-Kubernetes; sleep 5 ; done"]

nodeSelector:

hardware: t2-medium

\*\*\*\*\*\*\*\*\*\*\*\*\*Escape and :wq\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

kubectl apply -f pod8.yml

kubectl get pods

o/p : It will not be ready because in pod8.yml we have mentioned to create pod in label t2.medium but none of the nodes as label has t2.medium

kubectl describe pod nodelabels

o/p : you will see node selector as hardware as t2.micro but in events it will show as non ready

Now we will put the label in the node,

kubectl get nodes

kubectl label nodes ip-<<name of the node>> hardware=t2-medium

kubectl describe pod nodelabels

kubectl get pods

**\*\*\*\*\*\*\*LAB7: EXAMPLE OF REPLICATION CONTROLLER\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**vi myrc.yml**

kind: ReplicationController

apiVersion: v1

metadata:

name: myreplica

spec:

replicas: 2

selector:

myname: akshat

template:

metadata:

name: testpod6

labels:

myname: Akshat

spec:

containers:

- name: c00

image: ubuntu

command: ["/bin/bash", "-c", "while true; do echo Hello-Akshat; sleep 5 ; done"]

kubectl apply -f myrc.yml

kubectl get rc

kubectl describe rc myreplica

kubectl get pods

kubectl delete pod <<pod name>>

kubectl get pods

<<again you will see 2 pods as replication is enabled>>

Kubectl get rc

Kubectl describe rc myreplica

Kubectl get pods –show-labels

Kubectl scale –replicas=8 rc -l myname=Akshat (scale up)

Kubectl get rc

Kubectl get pods

Kubectl scale –replicas=1 rc -l myname=Akshat (scale down)

Kubectl get pods

Kubectl get rc

Kubectl get pods –show-labels

Kubectl delete -f myrc.yml

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**LAB 8: EXAMPLE OF REPLICA SET\*\*\*\*\*\*\*\*\*\*\*\***

kind: ReplicaSet

apiVersion: apps/v1

metadata:

name: myrs

spec:

replicas: 2

selector:

matchExpressions: # these must match the labels

- {key: myname, operator: In, values: [Akshat, Akshat, Akshu]}

- {key: env, operator: NotIn, values: [production]}

template:

metadata:

name: testpod7

labels:

myname: Akshat

spec:

containers:

- name: c00

image: ubuntu

command: ["/bin/bash", "-c", "while true; do echo Happy new year; sleep 5 ; done"]