Name: Kriti
College: Lovely Professional University
Week5 Assignment
Email: kritijulia@gmail.com

Here I made two virtual machine on azure and install docker, kubelet, kubeadm and kubectl on both vm worknode and masternode

Create Virtual Machines (1 Master + 1 Workers)

Use the Azure Portal

1. Go to **https://portal.azure.com**
2. Click on **Virtual Machines** > **Create**
3. Select the following:
    - **Image**: Ubuntu Server 22.04 LTS
    - **Size**: Standard B1s (1 vCPU, 1 GB RAM) – cheapest for testing
    - **Authentication**: Use **SSH public key**
    - **Open ports**: Allow SSH (port 22)

Install Docker on All Nodes

```
sudo apt update
sudo apt install -y docker.io
sudo systemctl start docker
sudo systemctl enable docker
```

Update and install prerequisites

```
sudo apt-get update
sudo apt-get install -y apt-transport-https ca-certificates curl
```

Add the Kubernetes signing key

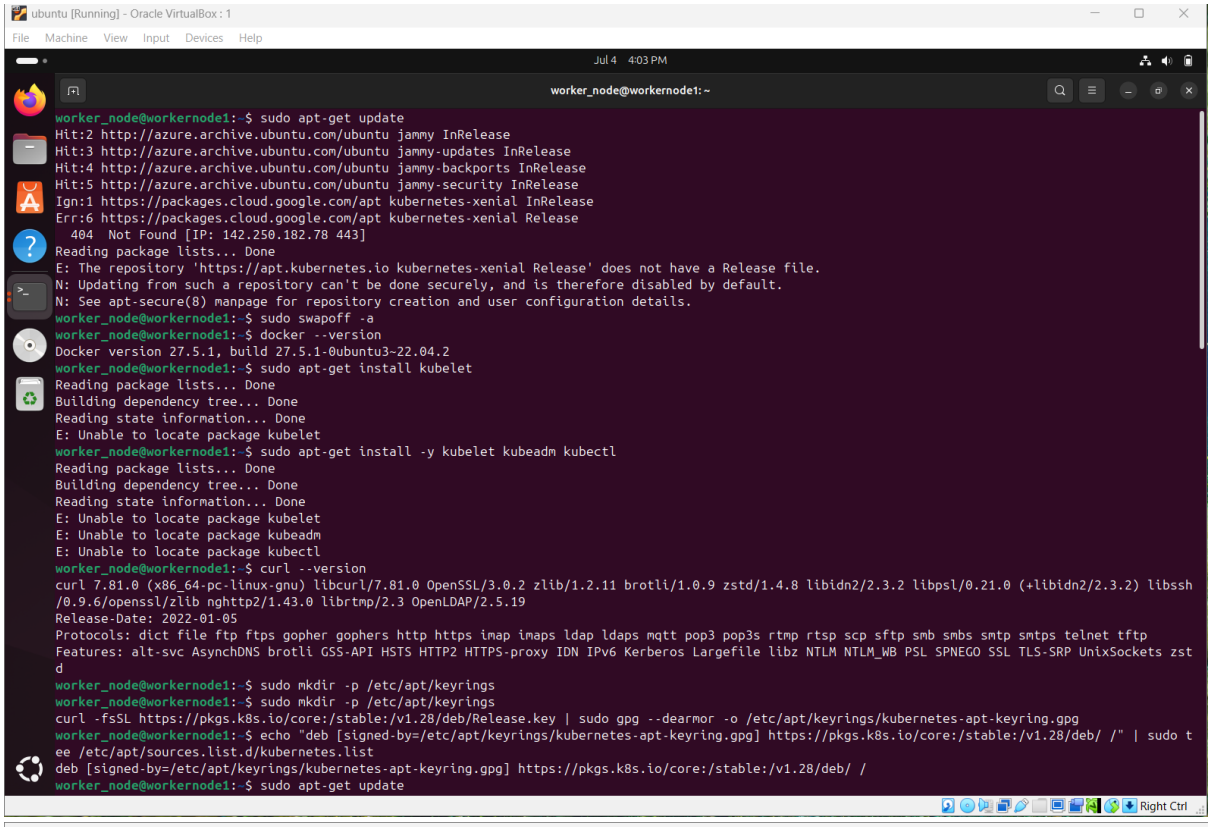
```
sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.28/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
```

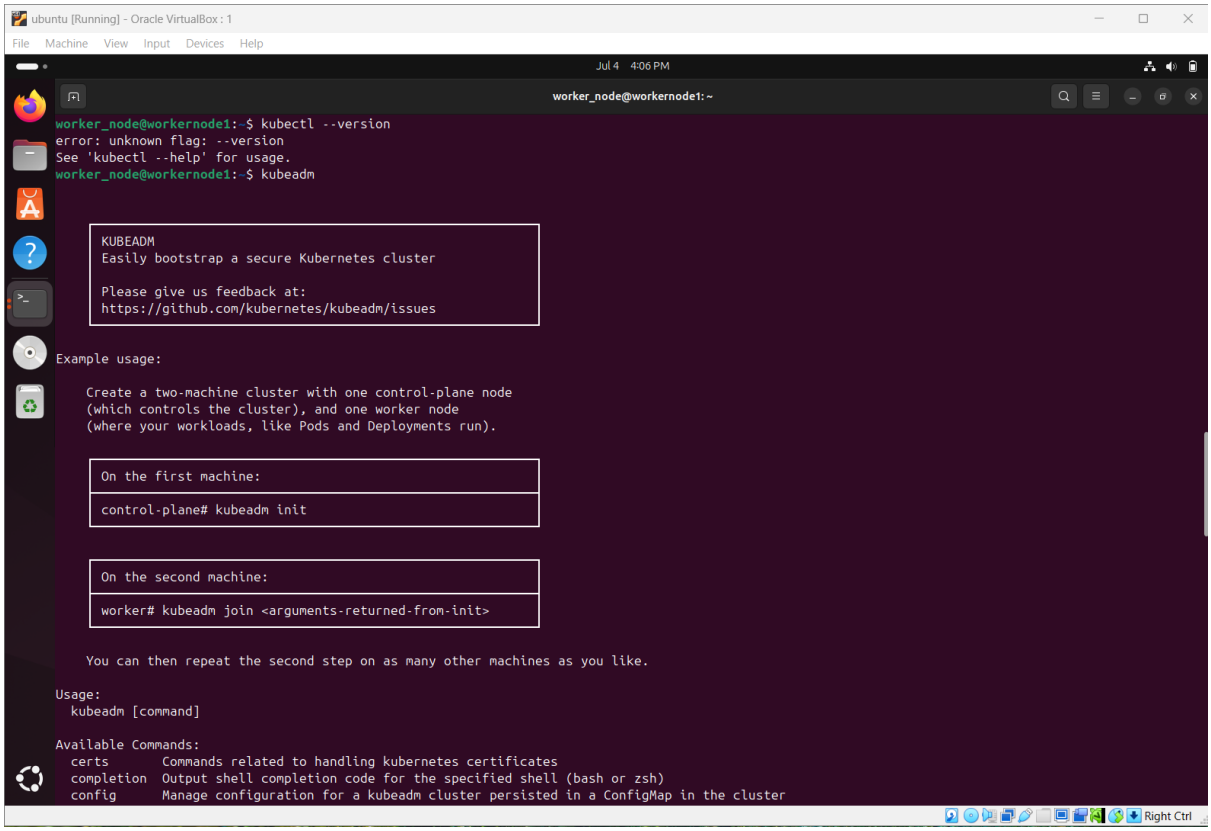
Add the Kubernetes APT repository

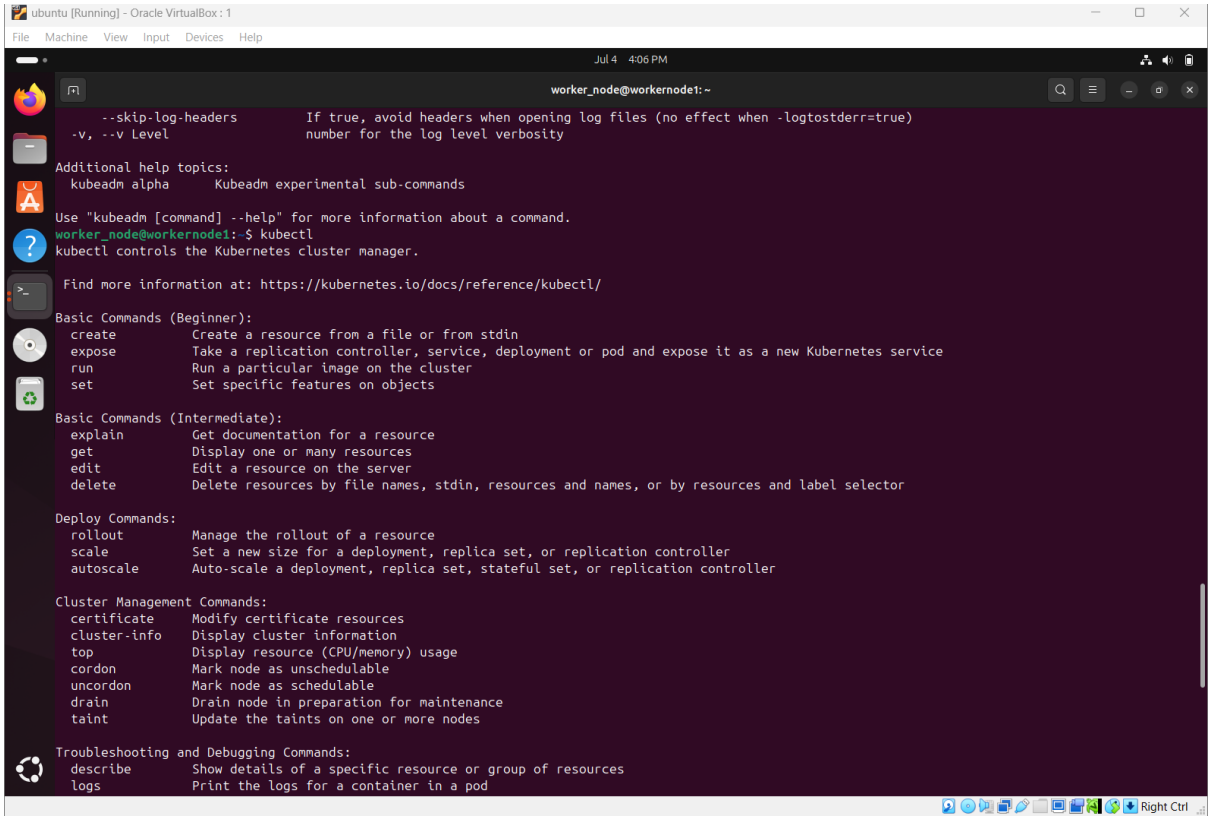
```
echo "deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.28/deb/ /" | sudo tee /etc/apt/sources.list.d/kubernetes.list
```

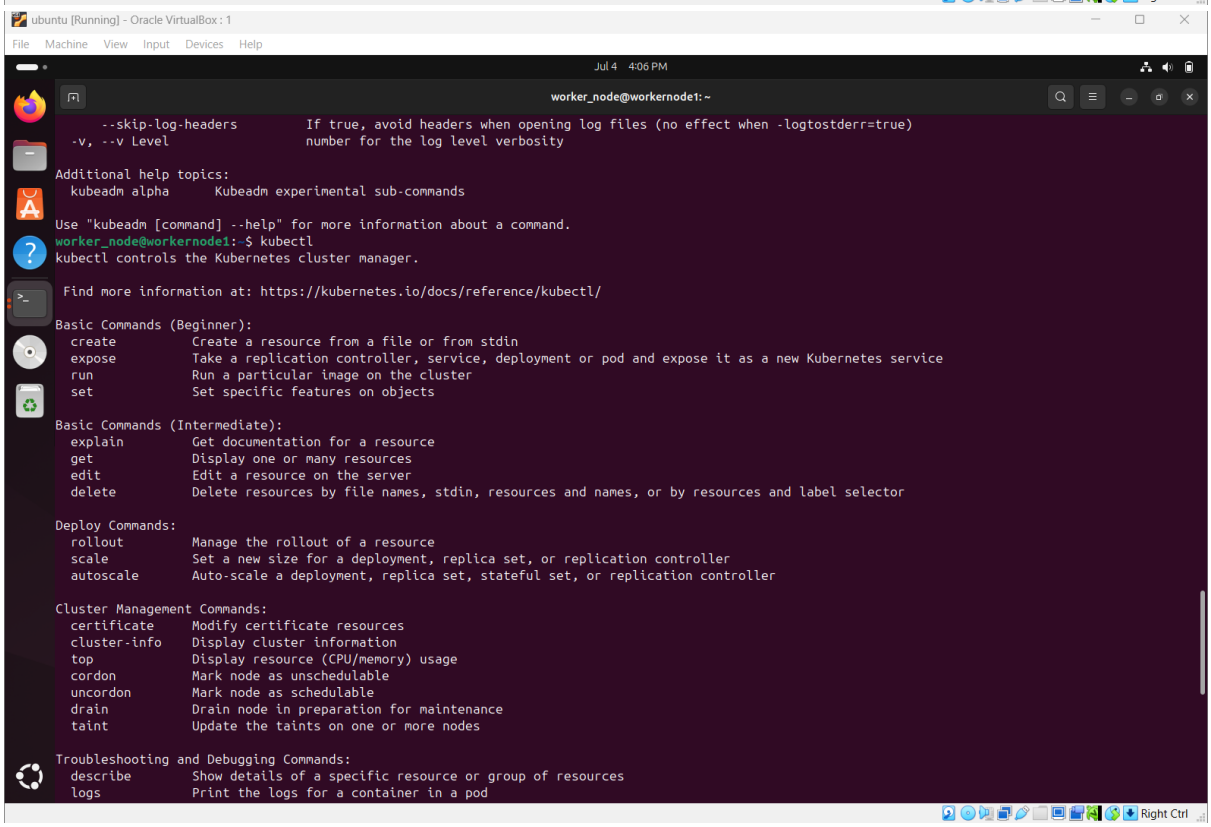
Update package list and install

```
sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl
```


ubuntu [Running] - Oracle VirtualBox : 1
File Machine View Input Devices Help
Jul 4 4:03 PM
worker_node@workernode1: ~
worker_node@workernode1:~$ sudo apt-get update
Hit:2 http://azure.archive.ubuntu.com/ubuntu jammy InRelease
Hit:3 http://azure.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:4 http://azure.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:5 http://azure.archive.ubuntu.com/ubuntu jammy-security InRelease
Ign:1 https://packages.cloud.google.com/apt kubernetes-xenial InRelease
Err:6 https://packages.cloud.google.com/apt kubernetes-xenial Release
404 Not Found [IP: 142.250.182.78 443]
Reading package lists... Done
E: The repository 'https://apt.kubernetes.io kubernetes-xenial Release' does not have a Release file.
N: Updating from such a repository can't be done securely, and is therefore disabled by default.
N: See apt-secure(8) manpage for repository creation and user configuration details.
worker_node@workernode1:~$ sudo swapoff -a
worker_node@workernode1:~$ docker --version
Docker version 27.5.1, build 27.5.1-0ubuntu3~22.04.2
worker_node@workernode1:~$ sudo apt-get install kubelet
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
E: Unable to locate package kubelet
worker_node@workernode1:~$ sudo apt-get install -y kubelet kubeadm kubectl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
E: Unable to locate package kubelet
E: Unable to locate package kubeadm
E: Unable to locate package kubectl
worker_node@workernode1:~$ curl --version
curl 7.81.0 (x86_64-pc-linux-gnu) libcurl/7.81.0 OpenSSL/3.0.2 zlib/1.2.11 brotli/1.0.9 zstd/1.4.8 libidn2/2.3.2 libpsl/0.21.0 (+libidn2/2.3.2) libssh
/0.9.6/openssl/zlib nghttp2/1.43.0 librtmp/2.3 OpenLDAP/2.5.19
Release-Date: 2022-01-05
Protocols: dict file ftp ftps gopher gophers http https imap imaps ldap ldaps mqtt pop3 pop3s rtmp rtsp scp sftp smb smbs smtp smtps telnet tftp
Features: alt-svc AsynchDNS brotli GSS-API HSTS HTTP2 HTTPS-proxy IDN IPv6 Kerberos Largefile libz NTLM NTLM_WB PSL SPNEGO SSL TLS-SRP UnixSockets zst
d
worker_node@workernode1:~$ sudo mkdir -p /etc/apt/keyrings
worker_node@workernode1:~$ sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.28/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
worker_node@workernode1:~$ echo "deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.28/deb/ /" | sudo t
ee /etc/apt/sources.list.d/kubernetes.list
deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.28/deb/ /
worker_node@workernode1:~$ sudo apt-get update
Right Ctrl


ubuntu [Running] - Oracle VirtualBox : 1
File Machine View Input Devices Help
Jul 4 4:06 PM
worker_node@workernode1: ~
worker_node@workernode1:~$ kubectl --version
error: unknown flag: --version
See 'kubectl --help' for usage.
worker_node@workernode1:~$ kubeadm
KUBEADM
Easily bootstrap a secure Kubernetes cluster
Please give us feedback at:
https://github.com/kubernetes/kubeadm/issues
Example usage:
Create a two-machine cluster with one control-plane node
(which controls the cluster), and one worker node
(where your workloads, like Pods and Deployments run).
On the first machine:
control-plane# kubeadm init
On the second machine:
worker# kubeadm join <arguments-returned-from-init>
You can then repeat the second step on as many other machines as you like.
Usage:
kubeadm [command]
Available Commands:
certs Commands related to handling kubernetes certificates
completion Output shell completion code for the specified shell (bash or zsh)
config Manage configuration for a kubeadm cluster persisted in a ConfigMap in the cluster
Right Ctrl


ubuntu [Running] - Oracle VirtualBox : 1
File Machine View Input Devices Help
Jul 4 4:06 PM
worker_node@workernode1: ~
--skip-log-headers If true, avoid headers when opening log files (no effect when -logtostderr=true)
-v, --v Level number for the log level verbosity
Additional help topics:
kubeadm alpha Kubeadm experimental sub-commands
Use "kubeadm [command] --help" for more information about a command.
worker_node@workernode1:~$ kubectl
kubectl controls the Kubernetes cluster manager.
Find more information at: https://kubernetes.io/docs/reference/kubectl/
Basic Commands (Beginner):
create Create a resource from a file or from stdin
expose Take a replication controller, service, deployment or pod and expose it as a new Kubernetes service
run Run a particular image on the cluster
set Set specific features on objects
Basic Commands (Intermediate):
explain Get documentation for a resource
get Display one or many resources
edit Edit a resource on the server
delete Delete resources by file names, stdin, resources and names, or by resources and label selector
Deploy Commands:
rollout Manage the rollout of a resource
scale Set a new size for a deployment, replica set, or replication controller
autoscale Auto-scale a deployment, replica set, stateful set, or replication controller
Cluster Management Commands:
certificate Modify certificate resources
cluster-info Display cluster information
top Display resource (CPU/memory) usage
cordon Mark node as unschedulable
uncordon Mark node as schedulable
drain Drain node in preparation for maintenance
taint Update the taints on one or more nodes
Troubleshooting and Debugging Commands:
describe Show details of a specific resource or group of resources
logs Print the logs for a container in a pod
Right Ctrl


ubuntu [Running] - Oracle VirtualBox : 1
File Machine View Input Devices Help
Jul 4 4:06 PM
worker_node@workernode1: ~
--skip-log-headers If true, avoid headers when opening log files (no effect when -logtostderr=true)
-v, --v Level number for the log level verbosity
Additional help topics:
kubeadm alpha Kubeadm experimental sub-commands
Use "kubeadm [command] --help" for more information about a command.
worker_node@workernode1:~$ kubectl
kubectl controls the Kubernetes cluster manager.
Find more information at: https://kubernetes.io/docs/reference/kubectl/
Basic Commands (Beginner):
create Create a resource from a file or from stdin
expose Take a replication controller, service, deployment or pod and expose it as a new Kubernetes service
run Run a particular image on the cluster
set Set specific features on objects
Basic Commands (Intermediate):
explain Get documentation for a resource
get Display one or many resources
edit Edit a resource on the server
delete Delete resources by file names, stdin, resources and names, or by resources and label selector
Deploy Commands:
rollout Manage the rollout of a resource
scale Set a new size for a deployment, replica set, or replication controller
autoscale Auto-scale a deployment, replica set, stateful set, or replication controller
Cluster Management Commands:
certificate Modify certificate resources
cluster-info Display cluster information
top Display resource (CPU/memory) usage
cordon Mark node as unschedulable
uncordon Mark node as schedulable
drain Drain node in preparation for maintenance
taint Update the taints on one or more nodes
Troubleshooting and Debugging Commands:
describe Show details of a specific resource or group of resources
logs Print the logs for a container in a pod
Right Ctrl

Q1. Create a Kubernetes cluster using minikube

Step1: Start Minikube Cluster

minikube start --driver=docker

Step 2: Check Cluster Status

kubectl get nodes

kubectl cluster-info

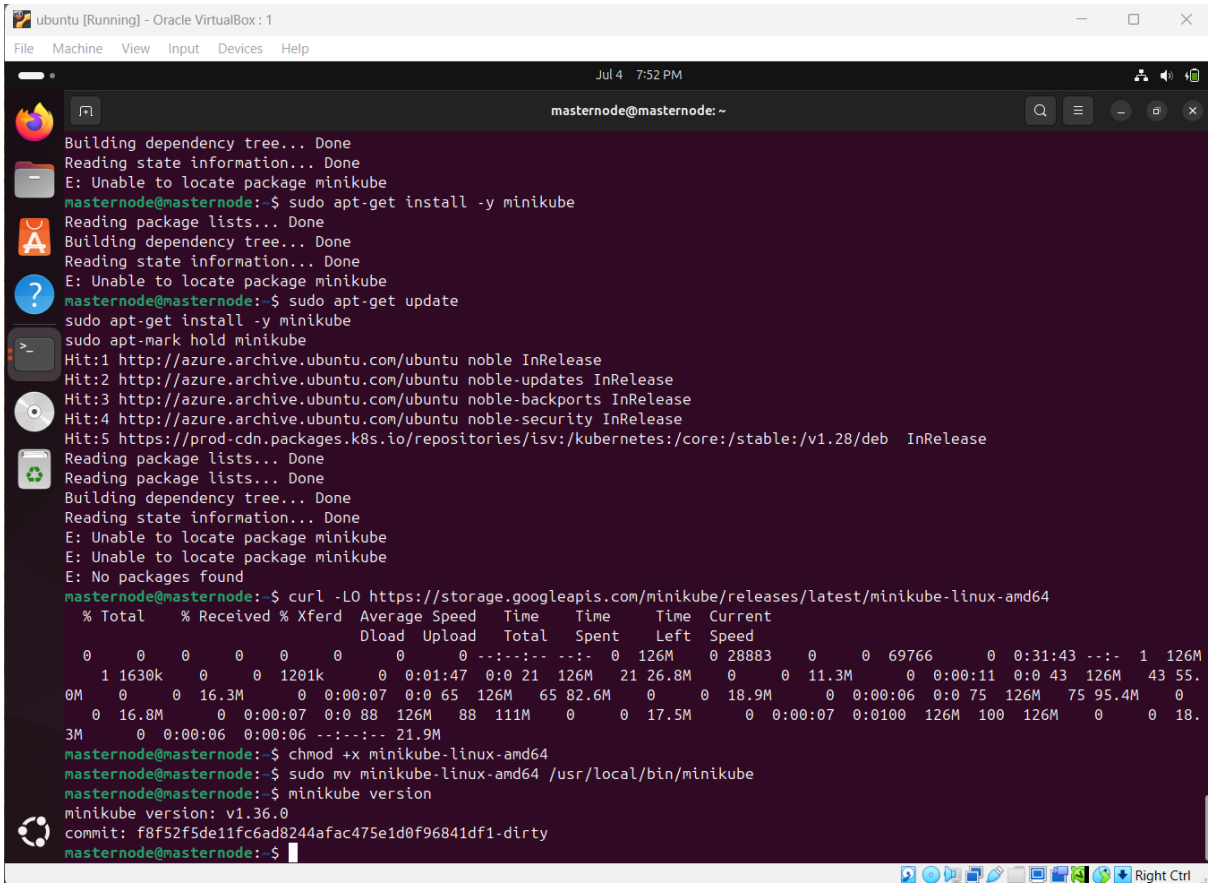
Step 3: **Test the Cluster** by deploying a test Nginx pod to verify everything is working.

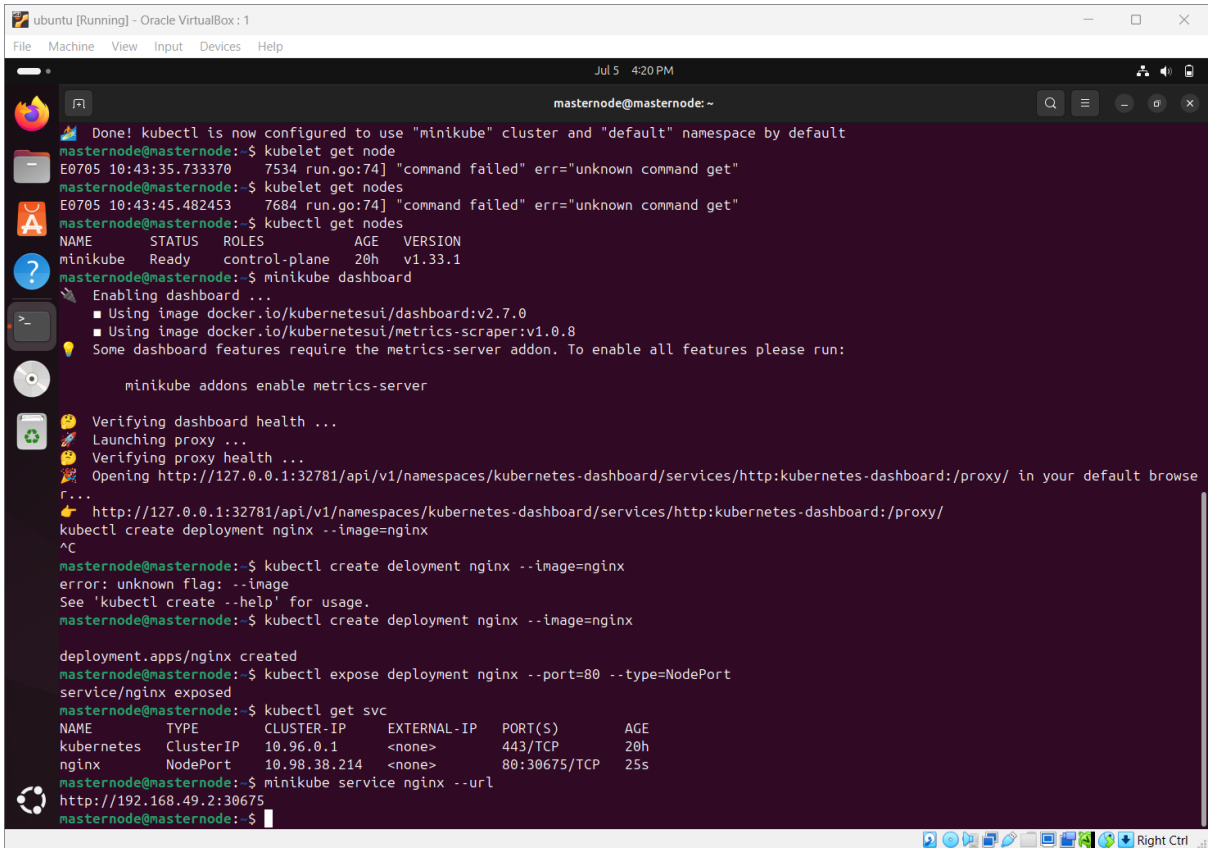
kubectl create deployment nginx --image=nginx

kubectl expose deployment nginx --port=80 --type=NodePort

kubectl get svc

minikube service nginx –url





Q2. Create a Kubernetes cluster using kubeadm

Step 1: Initialize the Cluster a link is generate for the worker node using that link connect the workernode to master node

```
sudo kubeadm init --pod-network-cidr=192.168.0.0/16
```

Step 2: Configure kubectl for User

```
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

step 3: test

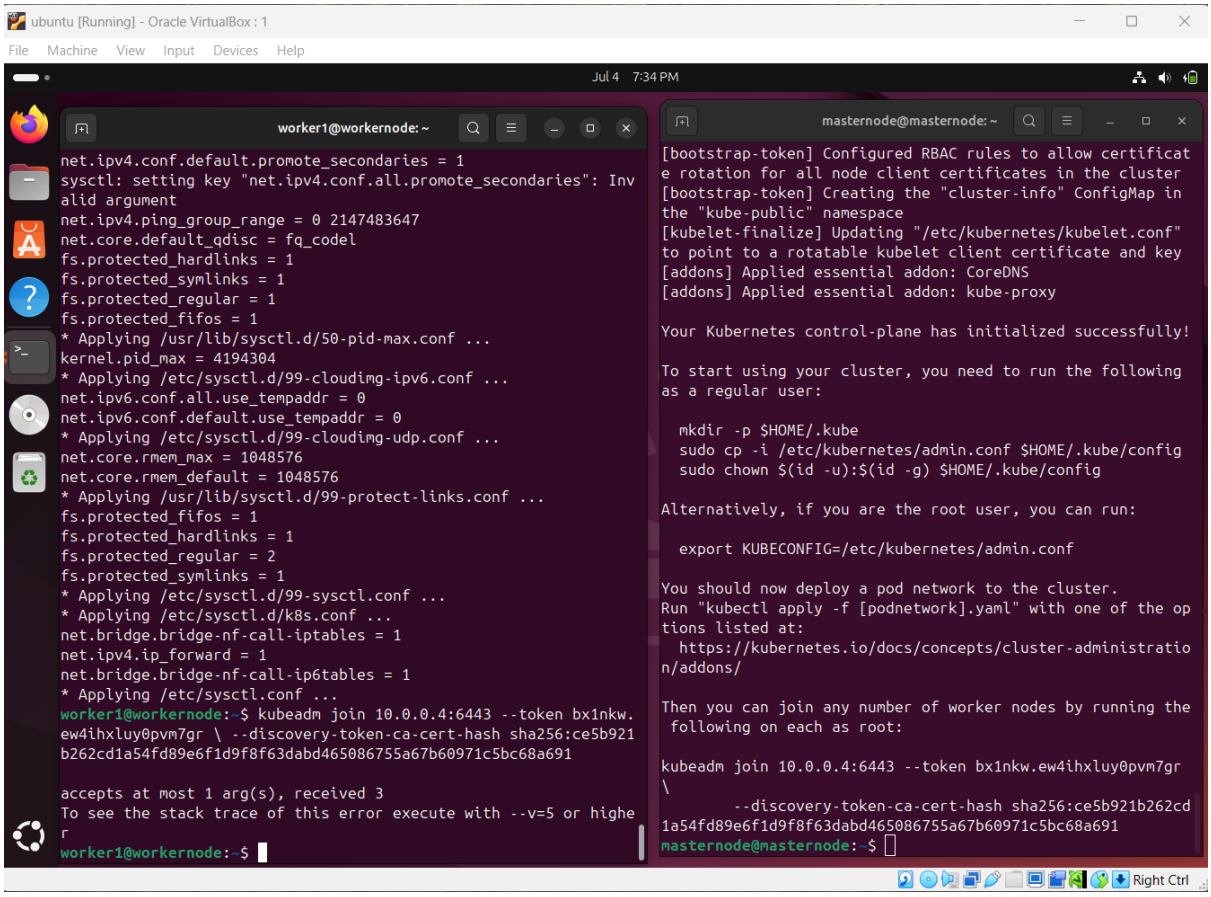
```
kubectl get nodes
```

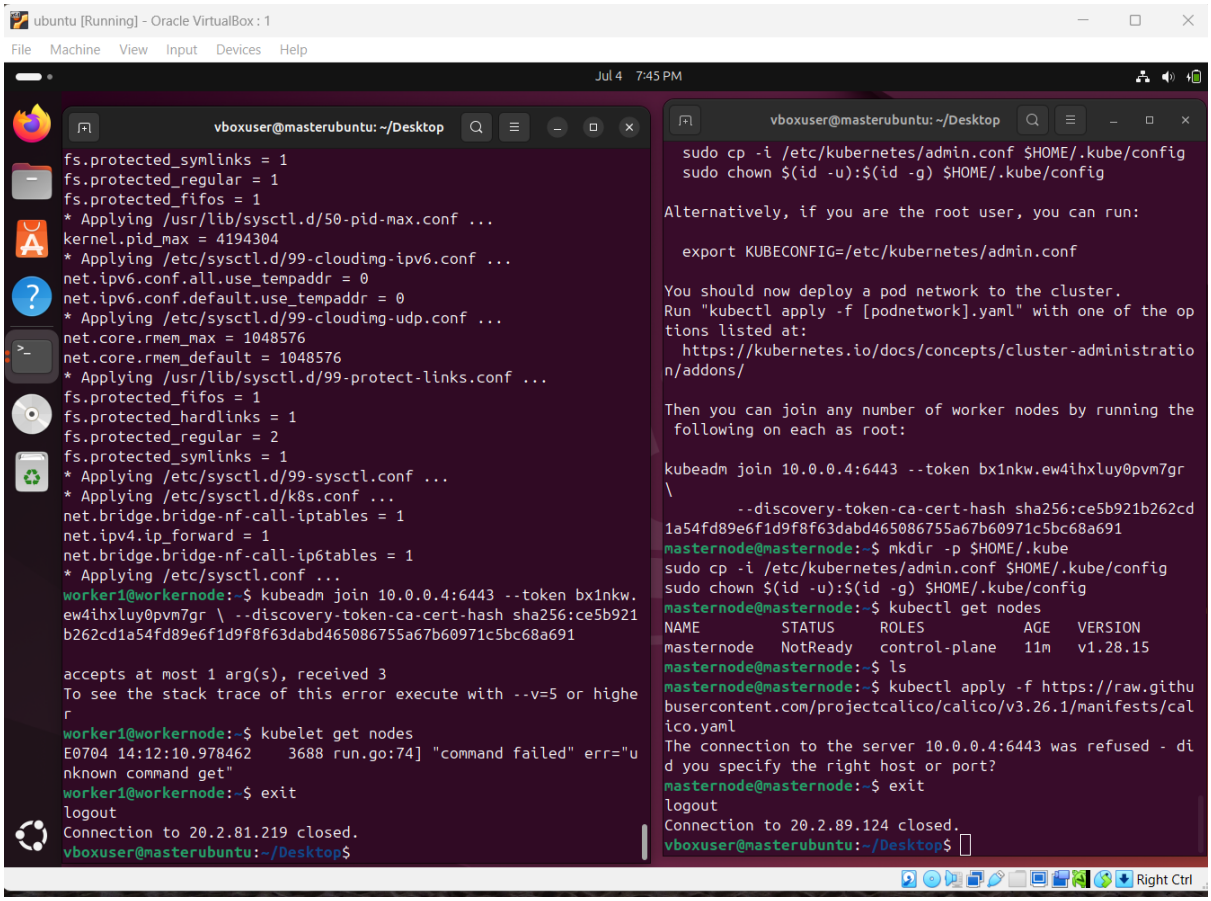
step 4: Install CNI Plugin (Calico)

```
kubectl apply -f https://raw.githubusercontent.com/projectcalico/calico/v3.27.0/manifests/calico.yaml
```

step5: Check status:

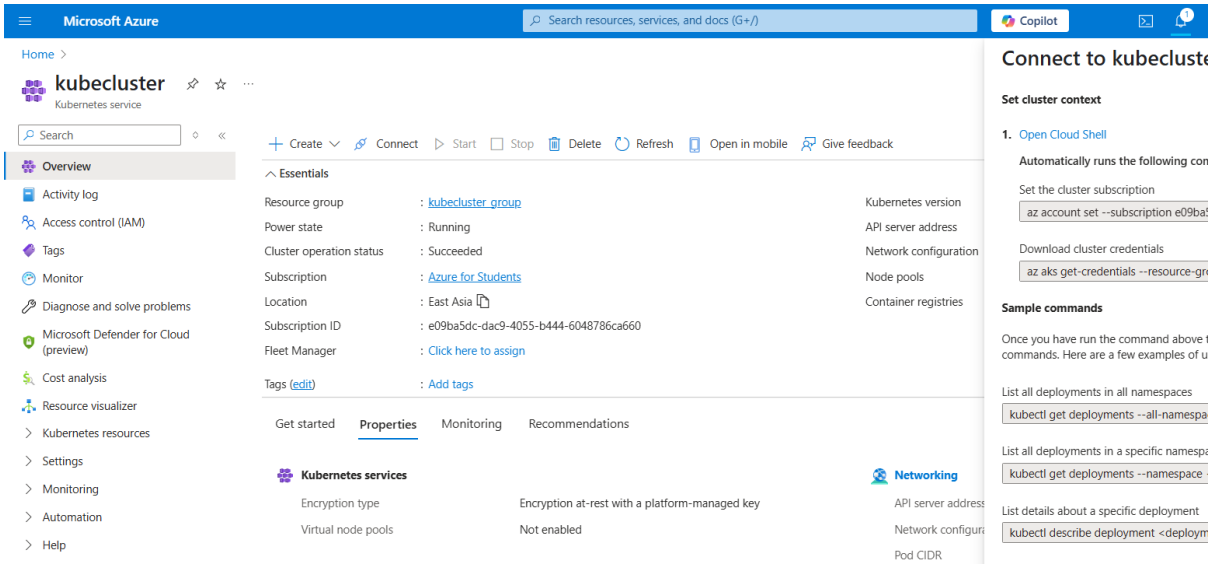
```
kubectl get pods -n kube-system
```


ubuntu [Running] - Oracle VirtualBox : 1
File Machine View Input Devices Help
Jul 4 7:34 PM
worker1@workernode: ~
net.ipv4.conf.default.promote_secondaries = 1
sysctl: setting key "net.ipv4.conf.all.promote_secondaries": Invalid argument
net.ipv4.ping_group_range = 0 2147483647
net.core.default_qdisc = fq_codel
fs.protected_hardlinks = 1
fs.protected_symlinks = 1
fs.protected_regular = 1
fs.protected_fifos = 1
* Applying /usr/lib/sysctl.d/50-pid-max.conf ...
kernel.pid_max = 4194304
* Applying /etc/sysctl.d/99-cloudimg-ipv6.conf ...
net.ipv6.conf.all.use_tempaddr = 0
net.ipv6.conf.default.use_tempaddr = 0
* Applying /etc/sysctl.d/99-cloudimg-udp.conf ...
net.core.rmem_max = 1048576
net.core.rmem_default = 1048576
* Applying /usr/lib/sysctl.d/99-protect-links.conf ...
fs.protected_fifos = 1
fs.protected_hardlinks = 1
fs.protected_regular = 2
fs.protected_symlinks = 1
* Applying /etc/sysctl.d/99-sysctl.conf ...
* Applying /etc/sysctl.d/k8s.conf ...
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip_forward = 1
net.bridge.bridge-nf-call-ip6tables = 1
* Applying /etc/sysctl.conf ...
worker1@workernode:~$ kubeadm join 10.0.0.4:6443 --token bx1nkw.ew4ihxluy0pvm7gr \ --discovery-token-ca-cert-hash sha256:ce5b921b262cd1a54fd89e6f1d9f8f63dabd465086755a67b60971c5bc68a691
accepts at most 1 arg(s), received 3
To see the stack trace of this error execute with --v=5 or higher
worker1@workernode:~$
masternode@masternode: ~
[bootstrap-token] Configured RBAC rules to allow certificate rotation for all node client certificates in the cluster
[bootstrap-token] Creating the "cluster-info" ConfigMap in the "kube-public" namespace
[kubelet-finalize] Updating "/etc/kubernetes/kubelet.conf" to point to a rotatable kubelet client certificate and key
[addons] Applied essential addon: CoreDNS
[addons] Applied essential addon: kube-proxy
Your Kubernetes control-plane has initialized successfully!
To start using your cluster, you need to run the following as a regular user:
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
Alternatively, if you are the root user, you can run:
export KUBECONFIG=/etc/kubernetes/admin.conf
You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
https://kubernetes.io/docs/concepts/cluster-administration/addons/
Then you can join any number of worker nodes by running the following on each as root:
kubeadm join 10.0.0.4:6443 --token bx1nkw.ew4ihxluy0pvm7gr \
--discovery-token-ca-cert-hash sha256:ce5b921b262cd1a54fd89e6f1d9f8f63dabd465086755a67b60971c5bc68a691
masternode@masternode:~$
Right Ctrl


ubuntu [Running] - Oracle VirtualBox : 1
File Machine View Input Devices Help
Jul 4 7:45 PM
vboxuser@masterubuntu: ~/Desktop
fs.protected_symlinks = 1
fs.protected_regular = 1
fs.protected_fifos = 1
* Applying /usr/lib/sysctl.d/50-pid-max.conf ...
kernel.pid_max = 4194304
* Applying /etc/sysctl.d/99-cloudimg-ipv6.conf ...
net.ipv6.conf.all.use_tempaddr = 0
net.ipv6.conf.default.use_tempaddr = 0
* Applying /etc/sysctl.d/99-cloudimg-udp.conf ...
net.core.rmem_max = 1048576
net.core.rmem_default = 1048576
* Applying /usr/lib/sysctl.d/99-protect-links.conf ...
fs.protected_fifos = 1
fs.protected_hardlinks = 1
fs.protected_regular = 2
fs.protected_symlinks = 1
* Applying /etc/sysctl.d/99-sysctl.conf ...
* Applying /etc/sysctl.d/k8s.conf ...
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip_forward = 1
net.bridge.bridge-nf-call-ip6tables = 1
* Applying /etc/sysctl.conf ...
worker1@workernode:~$ kubeadm join 10.0.0.4:6443 --token bx1nkw.ew4ihxluy0pvm7gr \ --discovery-token-ca-cert-hash sha256:ce5b921b262cd1a54fd89e6f1d9f8f63dabd465086755a67b60971c5bc68a691
accepts at most 1 arg(s), received 3
To see the stack trace of this error execute with --v=5 or higher
worker1@workernode:~$ kubelet get nodes
E0704 14:12:10.978462 3688 run.go:74] "command failed" err="unknown command get"
worker1@workernode:~$ exit
logout
Connection to 20.2.81.219 closed.
vboxuser@masterubuntu:~/Desktop$
vboxuser@masterubuntu: ~/Desktop
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
Alternatively, if you are the root user, you can run:
export KUBECONFIG=/etc/kubernetes/admin.conf
You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
https://kubernetes.io/docs/concepts/cluster-administration/addons/
Then you can join any number of worker nodes by running the following on each as root:
kubeadm join 10.0.0.4:6443 --token bx1nkw.ew4ihxluy0pvm7gr \
--discovery-token-ca-cert-hash sha256:ce5b921b262cd1a54fd89e6f1d9f8f63dabd465086755a67b60971c5bc68a691
masternode@masternode:~$ mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
masternode@masternode:~$ kubectl get nodes
NAME STATUS ROLES AGE VERSION
masternode NotReady control-plane 11m v1.28.15
masternode@masternode:~$ ls
masternode@masternode:~$ kubectl apply -f https://raw.githubusercontent.com/projectcalico/calico/v3.26.1/manifests/calico.yaml
The connection to the server 10.0.0.4:6443 was refused - did you specify the right host or port?
masternode@masternode:~$ exit
logout
Connection to 20.2.89.124 closed.
vboxuser@masterubuntu:~/Desktop$
Right Ctrl

Q3. Deploy an AKS cluster using the portal. Access the dashboard and create roles for multiple users

Step1: Deploy an AKS cluster using the portal



Step 2: Access AKS Cluster via Cloud Shell

After deployment:

1. Click **"Go to Resource"**.
2. On the AKS cluster page, click **“Connect”**.
3. Open **Azure Cloud Shell (Bash)** in top menu or install az CLI locally.

   az login

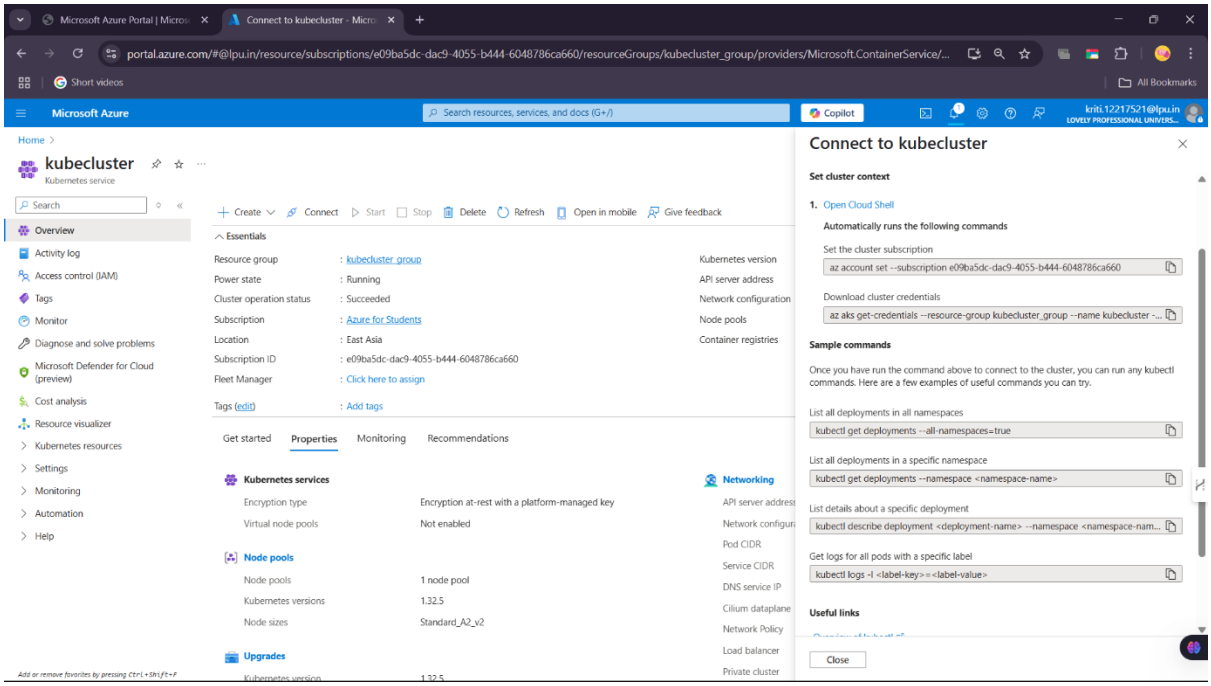
   az account set --subscription "<your-subscription-name>"

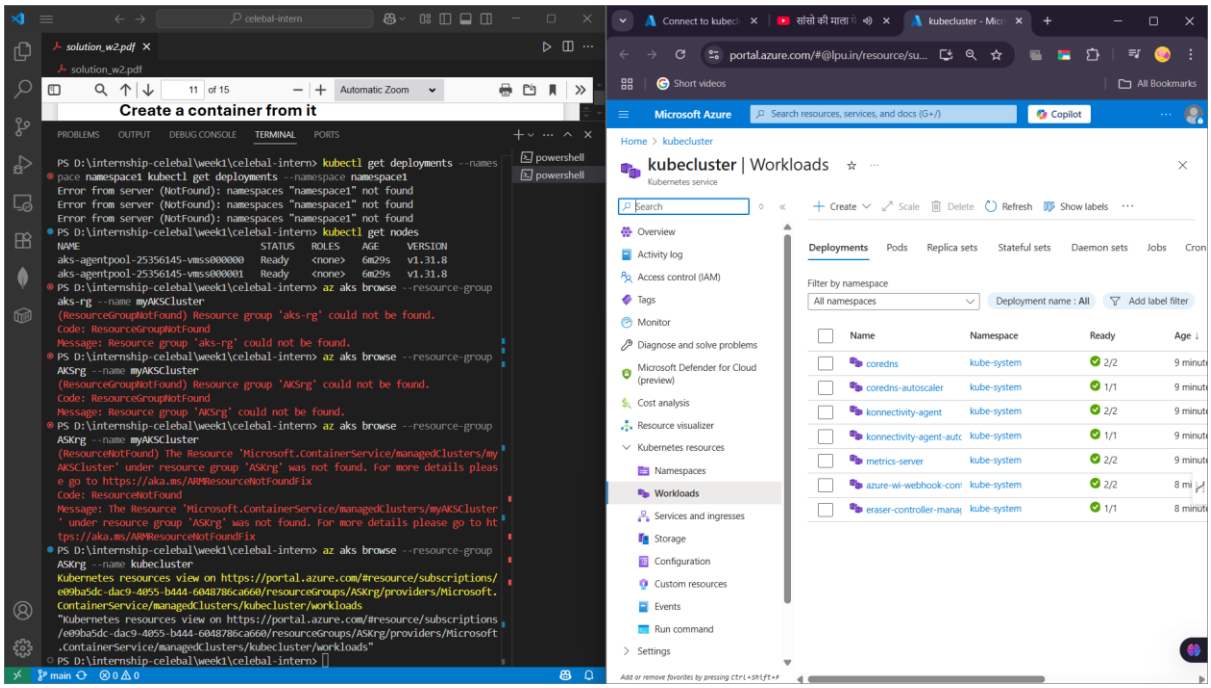
   az aks get-credentials --resource-group aks-rg --name myAKSCluster

   kubectl get nodes

Step 3: Access Kubernetes Dashboard

az aks browse --resource-group kubeclustergroup --name myxyz





Step 4: Create Roles for Multiple Users (RBAC)

```
# role.yaml

apiVersion: rbac.authorization.k8s.io/v1

kind: Role

metadata:

 namespace: default

 name: pod-reader
```

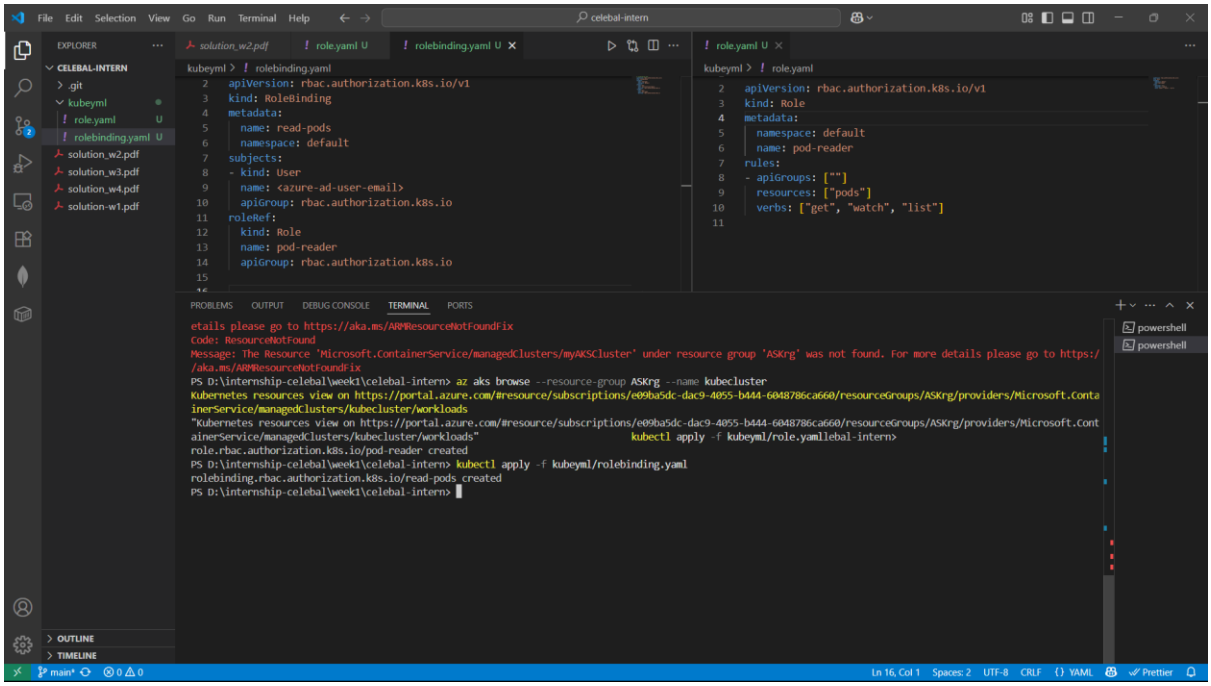
```
rules:
- apiGroups: [""]
  resources: ["pods"]
  verbs: ["get", "watch", "list"]
```

And bind it to a user:

```
# rolebinding.yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: read-pods
  namespace: default
subjects:
- kind: User
  name: <azure-ad-user-email>
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: Role
  name: pod-reader
  apiGroup: rbac.authorization.k8s.io
```

Apply with:

```
kubectl apply -f role.yaml
kubectl apply -f rolebinding.yaml
```



Q4. Deploy a microservice application on AKS cluster and access it using public internet

I'll deploy a simple **multi-container microservice** (e.g., frontend + backend) and expose it using a **LoadBalancer** service.

Step 1: Create a Sample Microservice App (YAML files)

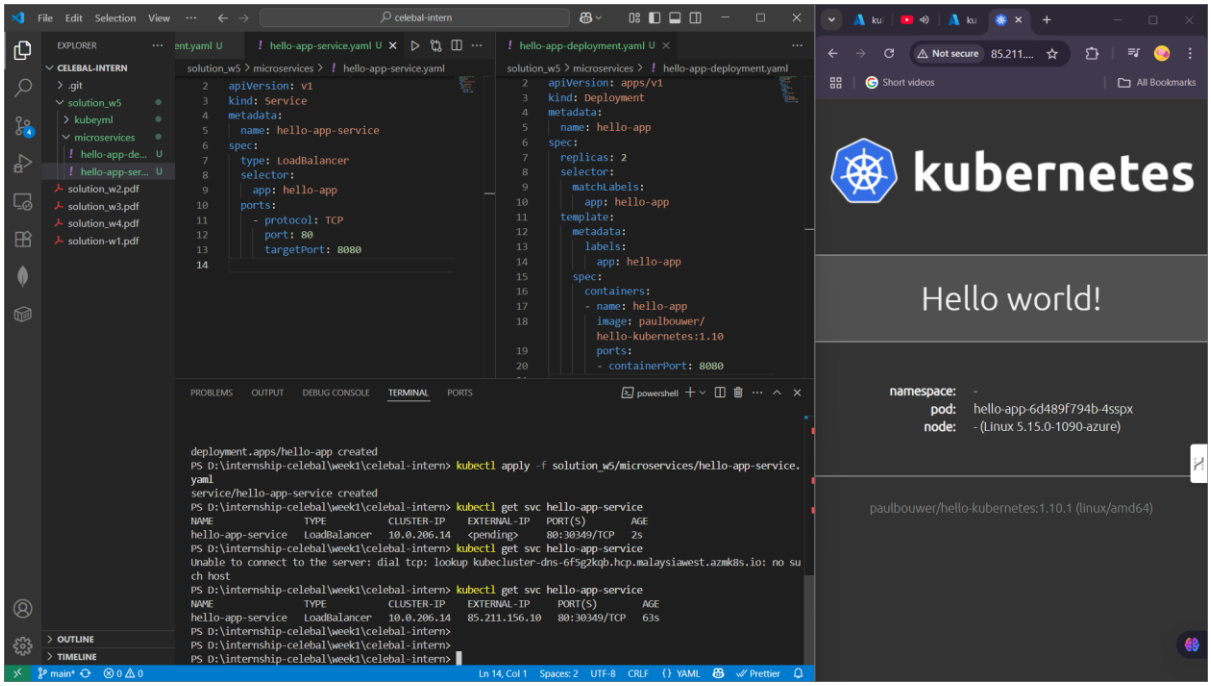
Step 2: Deploy to AKS

az aks get-credentials --resource-group aks-rg --name myAKSCluster

kubectl apply -f hello-app-deployment.yaml

kubectl apply -f hello-app-service.yaml

Step 3: Check the LoadBalancer External IP



Q5. Expose services in the cluster with node port, cluster IP, load balancer

Step 1: Create Deployment

kubectl create deployment nginx --image=nginx

Step 2: Expose as ClusterIP (default)

This is the **default service type**, only accessible **within the cluster**.

kubectl expose deployment nginx --port=80 --target-port=80 --name=nginx-clusterip

Check: kubectl get svc nginx-clusterip

Step 3: Expose as NodePort

Makes the app available on each node's IP and a high port (e.g., 30000-32767).

kubectl expose deployment nginx --port=80 --target-port=80 --type=NodePort --name=nginx-nodeport

Check: kubectl get svc nginx-nodeport

Example output:

nginx-nodeport NodePort 10.0.x.x <none> 80:31123/TCP 1m

kubectl get nodes -o wide

Use the node's external IP + node port, e.g.:

👉 http://<NodePublicIP>:31123

⚠️ On AKS, **NodePort access requires NSG/Firewall to allow that port**, which can be tricky. That's why LoadBalancer is preferred.

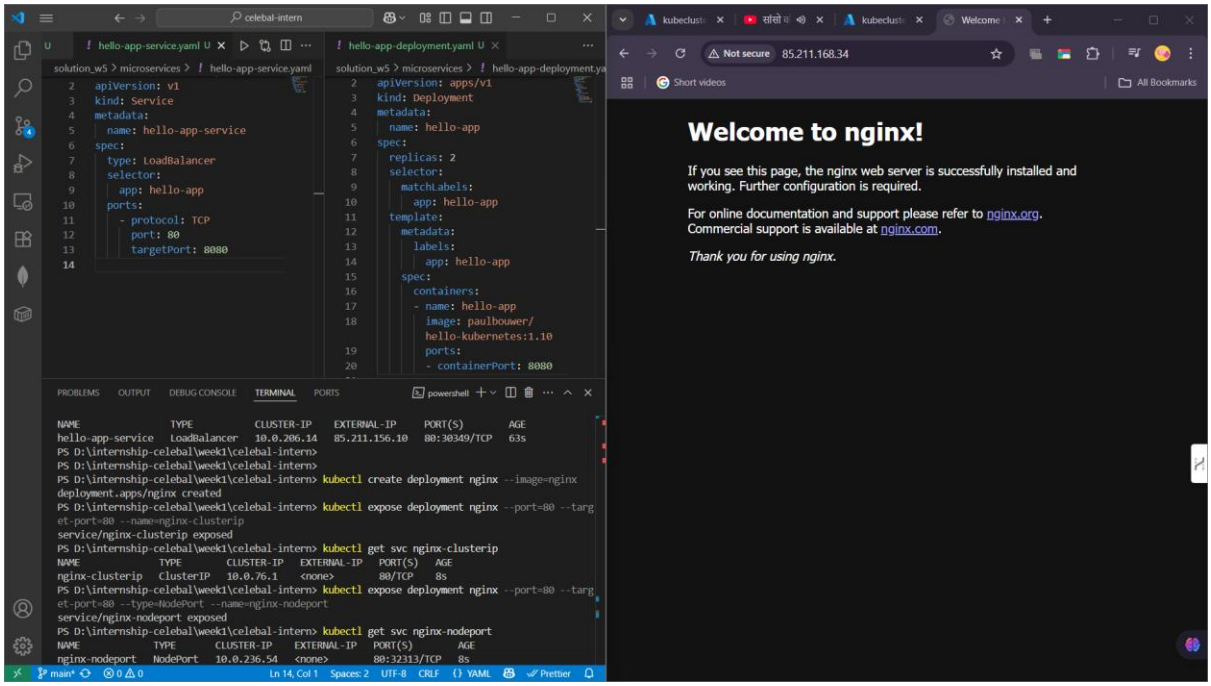
Step 4: Expose as LoadBalancer

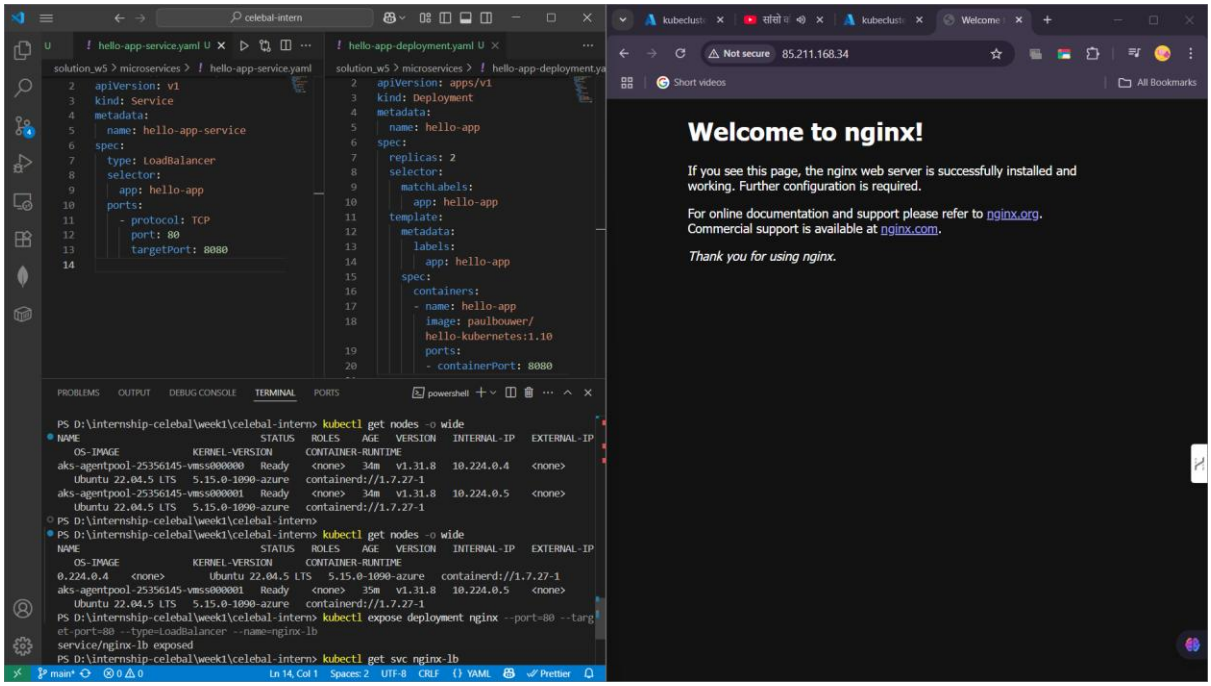
```
kubectl expose deployment nginx --port=80 --target-port=80 --
type=LoadBalancer --name=nginx-lb

kubectl get svc nginx-lb
```

Wait until the EXTERNAL-IP appears and Access:
👉 http://<EXTERNAL-IP>

Final: Clean Up

```
kubectl delete svc nginx-clusterip nginx-nodeport nginx-lb

kubectl delete deployment nginx
```


hello-app-service.yaml
hello-app-deployment.yaml
apiVersion: v1
kind: Service
metadata:
name: hello-app-service
spec:
type: LoadBalancer
selector:
app: hello-app
ports:
- protocol: TCP
port: 80
targetPort: 8080
apiVersion: apps/v1
kind: Deployment
metadata:
name: hello-app
spec:
replicas: 2
selector:
matchLabels:
app: hello-app
template:
metadata:
labels:
app: hello-app
spec:
containers:
- name: hello-app
image: paulbouwer/
hello-kubernetes:1.10
ports:
- containerPort: 8080
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
hello-app-service LoadBalancer 10.0.206.14 85.211.156.10 80:30349/TCP 63s
PS D:\internship-celebal\week1\celebal-intern>
PS D:\internship-celebal\week1\celebal-intern>
PS D:\internship-celebal\week1\celebal-intern> kubectl create deployment nginx --image=nginx
deployment.apps/nginx created
PS D:\internship-celebal\week1\celebal-intern> kubectl expose deployment nginx --port=80 --target-port=80 --name=nginx-clusterip
service/nginx-clusterip exposed
PS D:\internship-celebal\week1\celebal-intern> kubectl get svc nginx-clusterip
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
nginx-clusterip ClusterIP 10.0.76.1 <none> 80/TCP 8s
PS D:\internship-celebal\week1\celebal-intern> kubectl expose deployment nginx --port=80 --target-port=80 --type=NodePort --name=nginx-nodeport
service/nginx-nodeport exposed
PS D:\internship-celebal\week1\celebal-intern> kubectl get svc nginx-nodeport
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
nginx-nodeport NodePort 10.0.236.54 <none> 80:32313/TCP 8s
Not secure 85.211.168.34
Welcome to nginx!
If you see this page, the nginx web server is successfully installed and working. Further configuration is required.
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.
Thank you for using nginx.


hello-app-service.yaml
hello-app-deployment.yaml
apiVersion: v1
kind: Service
metadata:
name: hello-app-service
spec:
type: LoadBalancer
selector:
app: hello-app
ports:
- protocol: TCP
port: 80
targetPort: 8080
apiVersion: apps/v1
kind: Deployment
metadata:
name: hello-app
spec:
replicas: 2
selector:
matchLabels:
app: hello-app
template:
metadata:
labels:
app: hello-app
spec:
containers:
- name: hello-app
image: paulbouwer/
hello-kubernetes:1.10
ports:
- containerPort: 8080
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\internship-celebal\week1\celebal-intern> kubectl get nodes -o wide
NAME STATUS ROLES AGE VERSION INTERNAL-IP EXTERNAL-IP OS-IMAGE KERNEL-VERSION CONTAINER-RUNTIME
aks-agentpool-25356145-vmss000000 Ready <none> 34m v1.31.8 10.224.0.4 <none> Ubuntu 22.04.5 LTS 5.15.0-1090-azure containerd://1.7.27-1
aks-agentpool-25356145-vmss000001 Ready <none> 34m v1.31.8 10.224.0.5 <none> Ubuntu 22.04.5 LTS 5.15.0-1090-azure containerd://1.7.27-1
PS D:\internship-celebal\week1\celebal-intern>
PS D:\internship-celebal\week1\celebal-intern> kubectl get nodes -o wide
NAME STATUS ROLES AGE VERSION INTERNAL-IP EXTERNAL-IP OS-IMAGE KERNEL-VERSION CONTAINER-RUNTIME
0.224.0.4 <none> Ubuntu 22.04.5 LTS 5.15.0-1090-azure containerd://1.7.27-1
aks-agentpool-25356145-vmss000001 Ready <none> 35m v1.31.8 10.224.0.5 <none> Ubuntu 22.04.5 LTS 5.15.0-1090-azure containerd://1.7.27-1
PS D:\internship-celebal\week1\celebal-intern> kubectl expose deployment nginx --port=80 --target-port=80 --type=LoadBalancer --name=nginx-lb
service/nginx-lb exposed
PS D:\internship-celebal\week1\celebal-intern> kubectl get svc nginx-lb
Not secure 85.211.168.34
Welcome to nginx!
If you see this page, the nginx web server is successfully installed and working. Further configuration is required.
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.
Thank you for using nginx.