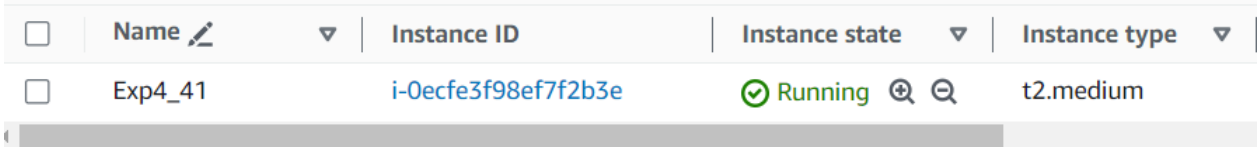
Name: Aditya Kirtane **Div: D15C** **Roll No: 26** **Academic Year: 2024-25**

Experiment No: 4

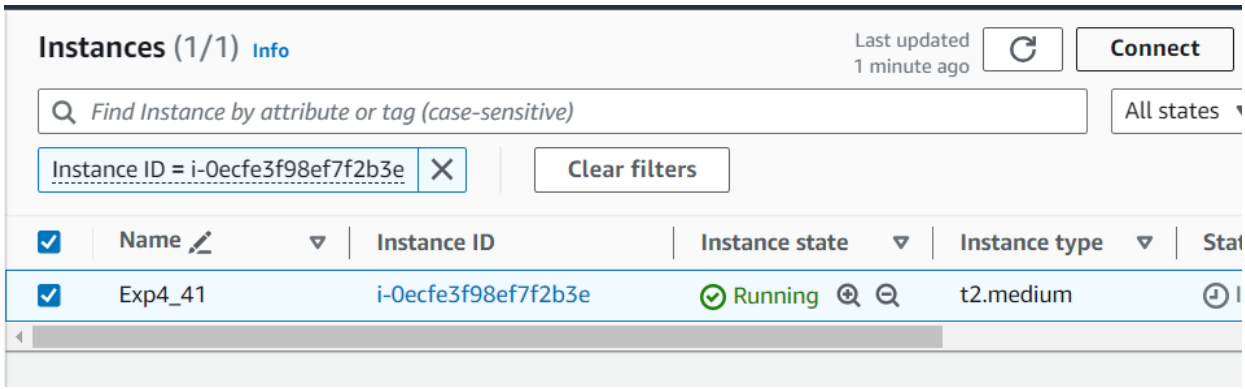
AIM: To install Kubectl and execute Kubectl commands to manage the Kubernetes cluster and deploy Your First Kubernetes Application.

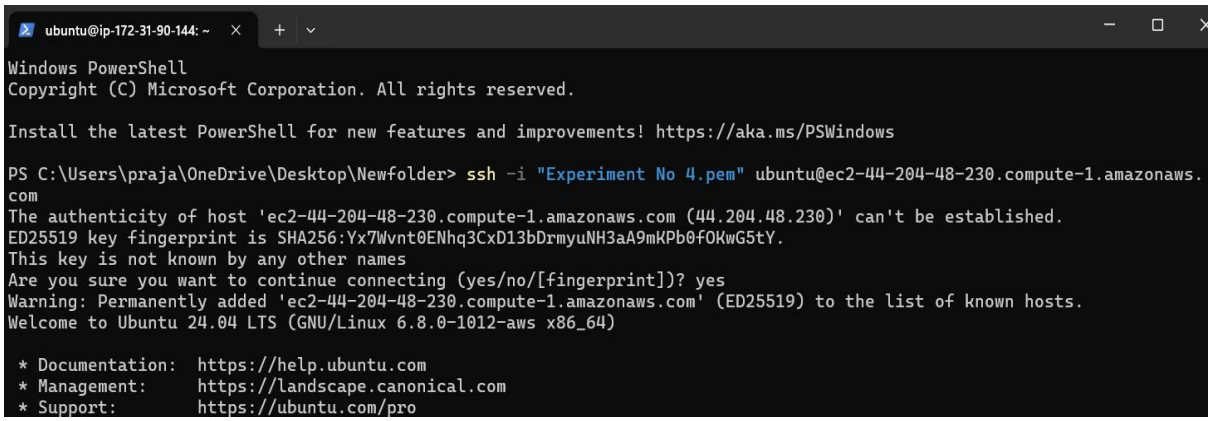
STEPS:

Step 1: Access your AWS Academy or personal account and initiate the launch of a new EC2 instance. Select **Ubuntu as the Amazon Machine Image (AMI)** and choose **t2.medium** for the instance type. Generate an RSA key with a .pem file extension, and relocate the downloaded key to the designated folder. **Note:** Since at least 2 CPUs are needed, ensure you select t2.medium



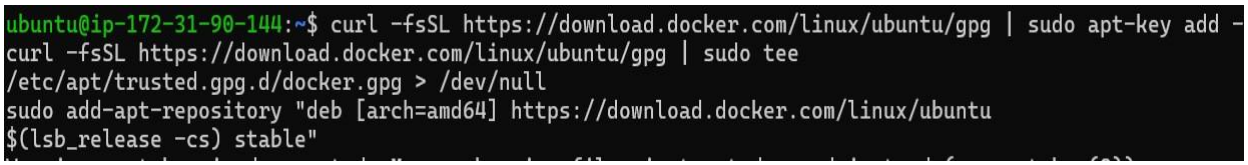
Step 2: Once the instance is created, click on the option to connect to the instance and go to the SSH client section.Now open the folder in the terminal where our .pem key is stored and paste the Example command (starting with ssh -i …..) in the terminal.





Step 3: Execute the following commands to **install and set up Docker.**

a) **curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -**
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee /etc/apt/trusted.gpg.d/docker.gpg > /dev/null
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"



b) sudo apt-get update
sudo apt-get install -y docker-ce

```
ubuntu@ip-172-31-90-144:~$ sudo apt-get update
sudo apt-get install -y docker-ce
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu noble InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored
/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

```
Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-90-144:~$
```

c) sudo mkdir -p /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
EOF

```
ubuntu@ip-172-31-90-144:~$ sudo mkdir -p /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
ubuntu@ip-172-31-90-144:~$
```

d) **sudo systemctl enable docker**
sudo systemctl daemon-reload
sudo systemctl restart docker

```
ubuntu@ip-172-31-90-144:~$ sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker
Synchronizing state of docker.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable docker
ubuntu@ip-172-31-90-144:~$
```

Step 4: Execute the following command to **install Kubernetes**.

a) **sudo rm -f /etc/apt/sources.list.d/kubernetes.list**

```
ubuntu@ip-172-31-90-144:~$ sudo rm -f /etc/apt/sources.list.d/kubernetes.list
```

b) **sudo apt-get update**

```
ubuntu@ip-172-31-90-144:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:5 https://download.docker.com/linux/ubuntu noble InRelease
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored
ection in apt-key(8) for details.
```

c) **sudo apt-get install -y apt-transport-https ca-certificates curl gpg**

```
ubuntu@ip-172-31-90-144:~$ sudo apt-get install -y apt-transport-https ca-certificates curl gpg
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ca-certificates is already the newest version (20240203).
ca-certificates set to manually installed.
gpg is already the newest version (2.4.4-2ubuntu17).
gpg set to manually installed.
The following NEW packages will be installed:
```

```
Fetched 904 kB in 0s (29.7 MB/s)
Selecting previously unselected package apt-transport-https.
(Reading database ... 68007 files and directories currently installed.)
Preparing to unpack .../apt-transport-https_2.7.14build2_all.deb ...
Unpacking apt-transport-https (2.7.14build2) ...
Preparing to unpack .../curl_8.5.0-2ubuntu10.4_amd64.deb ...
Unpacking curl (8.5.0-2ubuntu10.4) over (8.5.0-2ubuntu10.1) ...
Preparing to unpack .../libcurl4t64_8.5.0-2ubuntu10.4_amd64.deb ...
Unpacking libcurl4t64:amd64 (8.5.0-2ubuntu10.4) over (8.5.0-2ubuntu10.1) ...
Preparing to unpack .../libcurl3t64-gnutls_8.5.0-2ubuntu10.4_amd64.deb ...
Unpacking libcurl3t64-gnutls:amd64 (8.5.0-2ubuntu10.4) over (8.5.0-2ubuntu10.1) ...
Setting up apt-transport-https (2.7.14build2) ...
Setting up libcurl4t64:amd64 (8.5.0-2ubuntu10.4) ...
Setting up libcurl3t64-gnutls:amd64 (8.5.0-2ubuntu10.4) ...
Setting up curl (8.5.0-2ubuntu10.4) ...
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.2) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Running kernel seems to be up-to-date.

Restarting services...
 systemctl restart packagekit.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
```

d) sudo mkdir -p -m 755 /etc/apt/keyrings

```
ubuntu@ip-172-31-90-144:~$ sudo mkdir -p -m 755 /etc/apt/keyrings
```

e) curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg

```
ubuntu@ip-172-31-90-144:~$ curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key
ng.gpg
File '/etc/apt/keyrings/kubernetes-apt-keyring.gpg' exists. Overwrite? (y/N) y
```

f) echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list

```
ubuntu@ip-172-31-90-144:~$ echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.
c/apt/sources.list.d/kubernetes.list
deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /
```

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

```
ubuntu@ip-172-31-90-144:~$ sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 https://download.docker.com/linux/ubuntu noble InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  InRelease [1186 B]
Hit:6 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:7 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  Packages [4865 B]
Fetched 6051 B in 1s (9987 B/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.
ection in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  conntrack cri-tools kubernetes-cni
The following NEW packages will be installed:
  conntrack cri-tools kubeadm kubectl kubelet kubernetes-cni
0 upgraded, 6 newly installed, 0 to remove and 140 not upgraded.
Need to get 87.4 MB of archives.
After this operation, 314 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 conntrack amd64 1:1.4.8-1ubuntu1 [37.9 kB]
Get:2 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  cri-tools 1.31.1-1.1 [15.7 MB]
Get:3 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  kubeadm 1.31.1-1.1 [11.4 MB]
Get:4 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  kubectl 1.31.1-1.1 [11.2 MB]
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  kubernetes-cni 1.5.1-1.1 [33.9 MB]
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  kubelet 1.31.1-1.1 [15.2 MB]
Fetched 87.4 MB in 1s (70.6 MB/s)
Selecting previously unselected package conntrack.
```

```
Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
kubelet set on hold.
kubeadm set on hold.
kubectl set on hold.
```

h) sudo systemctl enable --now kubelet

```
ubuntu@ip-172-31-90-144:~$ sudo systemctl enable --now kubelet
ubuntu@ip-172-31-90-144:~$
```

i) sudo kubeadm init –pod-network-cidr=10.244.0.0/16

```
ubuntu@ip-172-31-90-144:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
W0928 04:27:54.973362    7321 checks.go:1080] [preflight] WARNING: Couldn't create
 new CRI runtime service: validate service connection: validate CRI v1 runtime API
e = Unimplemented desc = unknown service runtime.v1.RuntimeService
        [WARNING FileExisting-socat]: socat not found in system path
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your intern
[preflight] You can also perform this action beforehand using 'kubeadm config imag
error execution phase preflight: [preflight] Some fatal errors occurred:
failed to create new CRI runtime service: validate service connection: validate CR
: rpc error: code = Unimplemented desc = unknown service runtime.v1.RuntimeService
 with `--ignore-preflight-errors=...`
To see the stack trace of this error execute with --v=5 or higher
```

Here we got an error so to resolve this we use NEXT FOLLOWING COMMAND

j) sudo apt-get install -y containerd

```
ubuntu@ip-172-31-90-144:~$ sudo apt-get install -y containerd
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer re
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-co
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  runc
The following packages will be REMOVED:
  containerd.io docker-ce
The following NEW packages will be installed:
  containerd runc
0 upgraded, 2 newly installed, 2 to remove and 140 not upgraded.
Need to get 47.2 MB of archives.
After this operation, 53.1 MB disk space will be freed.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main
Fetched 47.2 MB in 1s (72.8 MB/s)
(Reading database ... 68068 files and directories currently installed.)
Removing docker-ce (5:27.3.1-1~ubuntu.24.04~noble) ...
Removing containerd.io (1.7.22-1) ...
Selecting previously unselected package runc.
(Reading database ... 68048 files and directories currently installed.)
Preparing to unpack .../runc_1.1.12-0ubuntu3.1_amd64.deb ...
Unpacking runc (1.1.12-0ubuntu3.1) ...
Selecting previously unselected package containerd.
Preparing to unpack .../containerd_1.7.12-0ubuntu4.1_amd64.deb ...
Unpacking containerd (1.7.12-0ubuntu4.1) ...
Setting up runc (1.1.12-0ubuntu3.1) ...
Setting up containerd (1.7.12-0ubuntu4.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.
```

k) sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml

```
ubuntu@ip-172-31-90-144:~$ sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml
disabled_plugins = []
imports = []
oom_score = 0
plugin_dir = ""
required_plugins = []
root = "/var/lib/containerd"
state = "/run/containerd"
temp = ""
version = 2

[cgroup]
  path = ""

[debug]
  address = ""
  format = ""
  gid = 0
  level = ""
  uid = 0

[grpc]
  address = "/run/containerd/containerd.sock"
  gid = 0
  max_recv_message_size = 16777216
  max_send_message_size = 16777216
  tcp_address = ""
  tcp_tls_ca = ""
  tcp_tls_cert = ""
  tcp_tls_key = ""
  uid = 0

[metrics]
  address = ""
  grpc_histogram = false

[plugins]
```

l) sudo systemctl restart containerd
sudo systemctl enable containerd

```
ubuntu@ip-172-31-90-144:~$ sudo systemctl restart containerd
sudo systemctl enable containerd
ubuntu@ip-172-31-90-144:~$
```

m) sudo systemctl status containerd

```
ubuntu@ip-172-31-90-144:~$ sudo systemctl status containerd
● containerd.service - containerd container runtime
     Loaded: loaded (/usr/lib/systemd/system/containerd.service; enabled; preset: enabled)
     Active: active (running) since Sat 2024-09-28 04:36:39 UTC; 2min 15s ago
       Docs: https://containerd.io
   Main PID: 8081 (containerd)
      Tasks: 8
     Memory: 13.5M (peak: 14.1M)
        CPU: 321ms
     CGroup: /system.slice/containerd.service
             └─8081 /usr/bin/containerd

Sep 28 04:36:39 ip-172-31-90-144 containerd[8081]: time="2024-09-28T04:36:39.260667478Z" level=info msg=
Sep 28 04:36:39 ip-172-31-90-144 containerd[8081]: time="2024-09-28T04:36:39.260698020Z" level=info msg=
Sep 28 04:36:39 ip-172-31-90-144 containerd[8081]: time="2024-09-28T04:36:39.260771896Z" level=info msg=
Sep 28 04:36:39 ip-172-31-90-144 containerd[8081]: time="2024-09-28T04:36:39.260798416Z" level=info msg=
Sep 28 04:36:39 ip-172-31-90-144 containerd[8081]: time="2024-09-28T04:36:39.260839996Z" level=info msg=
Sep 28 04:36:39 ip-172-31-90-144 containerd[8081]: time="2024-09-28T04:36:39.260855673Z" level=info msg=
Sep 28 04:36:39 ip-172-31-90-144 containerd[8081]: time="2024-09-28T04:36:39.260863302Z" level=info msg=
Sep 28 04:36:39 ip-172-31-90-144 containerd[8081]: time="2024-09-28T04:36:39.260869817Z" level=info msg=
Sep 28 04:36:39 ip-172-31-90-144 containerd[8081]: time="2024-09-28T04:36:39.261264494Z" level=info msg=
Sep 28 04:36:39 ip-172-31-90-144 systemd[1]: Started containerd.service - containerd container runtime.
lines 1-21/21 (END)
```

n) sudo apt-get install -y socat

```
ubuntu@ip-172-31-90-144:~$ sudo apt-get install -y socat
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer requ
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-comp
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  socat
0 upgraded, 1 newly installed, 0 to remove and 140 not upgraded.
Need to get 374 kB of archives.
After this operation, 1649 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 soca
Fetched 374 kB in 0s (16.3 MB/s)
Selecting previously unselected package socat.
(Reading database ... 68112 files and directories currently installed.)
Preparing to unpack .../socat_1.8.0.0-4build3_amd64.deb ...
Unpacking socat (1.8.0.0-4build3) ...
Setting up socat (1.8.0.0-4build3) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-90-144:~$
```

Step 5: Initialize the **Kubecluster**.

sudo kubeadm init --pod-network-cidr=10.244.0.0/16

```
ubuntu@ip-172-31-90-144:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet
[preflight] You can also perform this action beforehand using 'kubeadm config images
W0928 04:46:10.270292    8652 checks.go:846] detected that the sandbox image "regist
used by kubeadm.It is recommended to use "registry.k8s.io/pause:3.10" as the CRI san
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [ip-172-31-90-144 kubernetes
.local] and IPs [10.96.0.1 172.31.90.144]
[certs] Generating "apiserver-kubelet-client" certificate and key
[certs] Generating "front-proxy-ca" certificate and key
[certs] Generating "front-proxy-client" certificate and key
[certs] Generating "etcd/ca" certificate and key
[certs] Generating "etcd/server" certificate and key
[certs] etcd/server serving cert is signed for DNS names [ip-172-31-90-144 localhost
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-90-144 localhost]
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
[certs] Generating "sa" key and public key
[kubeconfig] Using kubeconfig folder "/etc/kubernetes"
[kubeconfig] Writing "admin.conf" kubeconfig file
[kubeconfig] Writing "super-admin.conf" kubeconfig file
[kubeconfig] Writing "kubelet.conf" kubeconfig file
[kubeconfig] Writing "controller-manager.conf" kubeconfig file
[kubeconfig] Writing "scheduler.conf" kubeconfig file
[etcd] Creating static Pod manifest for local etcd in "/etc/kubernetes/manifests"
[control-plane] Using manifest folder "/etc/kubernetes/manifests"
[control-plane] Creating static Pod manifest for "kube-apiserver"
[control-plane] Creating static Pod manifest for "kube-controller-manager"
[control-plane] Creating static Pod manifest for "kube-scheduler"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubele
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Starting the kubelet
```

```
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 172.31.90.144:6443 --token 75sp2s.4moocvobfpkimn9q \
        --discovery-token-ca-cert-hash sha256:42855a37f6c6446da3abb79740e05317a4aadc79d228cf7df4339bd90fd6f19d
ubuntu@ip-172-31-90-144:~$
```

Step 6: Copy the mkdir and chown commands from the top and execute them.
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config

```
ubuntu@ip-172-31-90-144:~$ mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
ubuntu@ip-172-31-90-144:~$
```

Step 7: Add a common networking plugin called **flannel** as mentioned in the code. Flannel is a simple and widely used networking plugin for Kubernetes that facilitates pod-to-pod communication across a cluster.
kubectl apply -f
https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml

```
ubuntu@ip-172-31-90-144:~$ kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
namespace/kube-flannel created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
serviceaccount/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created
ubuntu@ip-172-31-90-144:~$
```

Step 8: With the cluster now running, we can deploy our **Nginx server** onto it. Run the following command to apply the deployment configuration and set up the Nginx deployment.
kubectl apply -f https://k8s.io/examples/application/deployment.yaml

```
ubuntu@ip-172-31-90-144:~$ kubectl apply -f https://k8s.io/examples/application/deployment.yaml
deployment.apps/nginx-deployment created
ubuntu@ip-172-31-90-144:~$
```

kubectl get pods

```
ubuntu@ip-172-31-90-144:~$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-d556bf558-6r7bm    0/1     Pending   0          76s
nginx-deployment-d556bf558-bv4vz    0/1     Pending   0          76s
ubuntu@ip-172-31-90-144:~$
```

```
ubuntu@ip-172-31-90-144:~$ POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")
ubuntu@ip-172-31-90-144:~$
```

POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")
kubectl port-forward $POD_NAME 8080:80

```
ubuntu@ip-172-31-90-144:~$ POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")
ubuntu@ip-172-31-90-144:~$
```

```
ubuntu@ip-172-31-90-144:~$ kubectl port-forward $POD_NAME 8080:80
error: unable to forward port because pod is not running. Current status=Pending
ubuntu@ip-172-31-90-144:~$
```

Note: If you encounter an error where the pod status shows as "**Pending**," follow these steps. Run the commands below to resolve the issue, and then re-run the previous two commands.

kubectl taint nodes --all
node-role.kubernetes.io/control-plane-node/ip-172-31-20-171 untainted

```
ubuntu@ip-172-31-90-144:~$ kubectl taint nodes --all node-role.kubernetes.io/control-plane-
node/ip-172-31-90-144 untainted
ubuntu@ip-172-31-90-144:~$
```

kubectl get nodes

```
ubuntu@ip-172-31-90-144:~$ kubectl get nodes
NAME               STATUS   ROLES           AGE   VERSION
ip-172-31-90-144   Ready    control-plane   40m   v1.31.1
ubuntu@ip-172-31-90-144:~$
```

kubectl get pods

```
ubuntu@ip-172-31-90-144:~$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-d556bf558-6r7bm   1/1     Running   0          27m
nginx-deployment-d556bf558-bv4vz   1/1     Running   0          27m
ubuntu@ip-172-31-90-144:~$
```

Name: Shivam.R.Prajapati Div: D15C Roll No: 41 Academic Year: 2024-25

POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")
kubectl port-forward $POD_NAME 8080:80

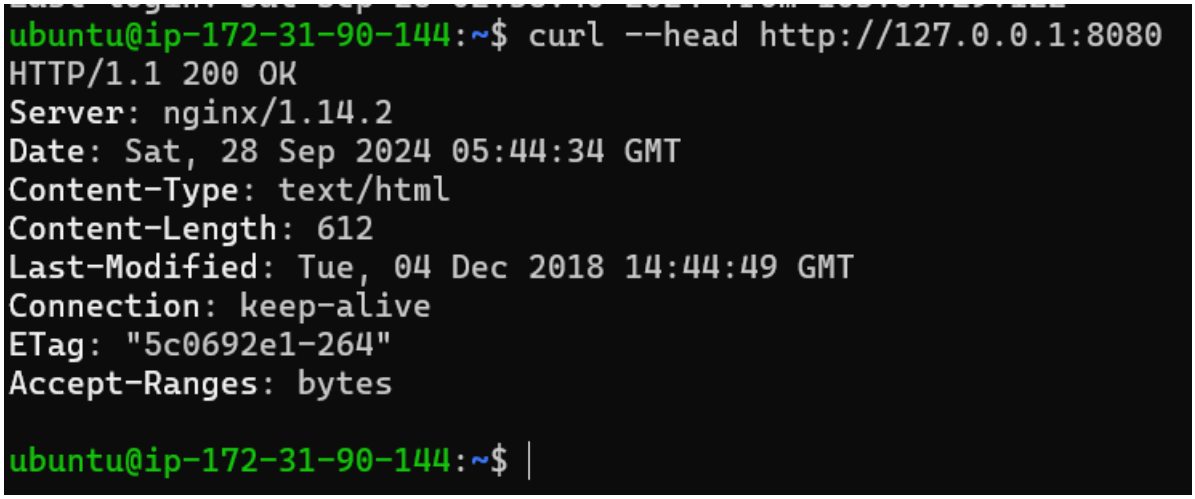
```
ubuntu@ip-172-31-90-144:~$ POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")
ubuntu@ip-172-31-90-144:~$
```

```
ubuntu@ip-172-31-90-144:~$ POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")
kubectl port-forward $POD_NAME 8080:80
Forwarding from 127.0.0.1:8080 -> 80
Forwarding from [::1]:8080 -> 80
```

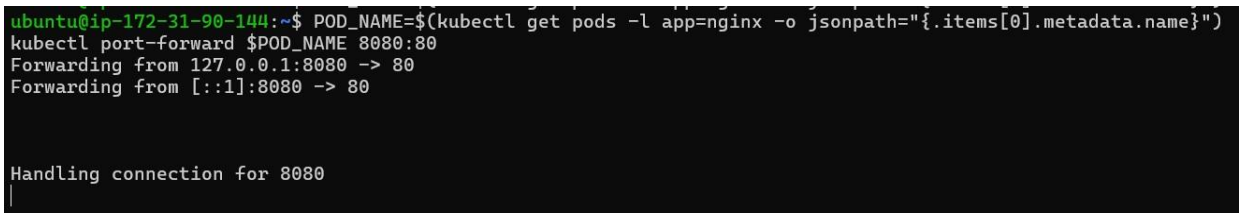
Step 9 : Check if your deployment is working. Open a new terminal and connect to your EC2 instance using SSH. Then, run the curl command to see if the Nginx server is up and running.

```
ubuntu@ip-172-31-90-144: ~
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\praja\OneDrive\Desktop\Newfolder> ssh -i "Experiment No 4.pem" ubuntu@ec2-44-204-48-230.compute-1.amazonaws
com
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1012-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Sat Sep 28 05:42:45 UTC 2024

  System load:  0.08              Processes:             153
  Usage of /:   55.6% of 6.71GB   Users logged in:       1
  Memory usage: 19%               IPv4 address for enX0: 172.31.90.144
  Swap usage:   0%

 * Ubuntu Pro delivers the most comprehensive open source security and
   compliance features.

   https://ubuntu.com/aws/pro

Expanded Security Maintenance for Applications is not enabled.

142 updates can be applied immediately.
38 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable
```

curl --head http://127.0.0.1:8080



After handling connecton



If you get a **200 OK** response and see the Nginx server name, your deployment was successful.

CONCLUSION:

In this experiment, we successfully installed Kubernetes on an EC2 instance and deployed an Nginx server using various Kubectl commands. During the process, we faced two main problems. The first issue was that the Kubernetes pod was stuck in a pending state. We fixed this by removing the control-plane taint with the command **kubectl taint nodes --all**, which allowed the pod to schedule correctly. The second problem was a missing containerd runtime, which we resolved by installing and starting the containerd service. To make sure our Kubernetes setup had enough resources, we used a t2.medium EC2 instance with 2 virtual CPUs. Overall, these steps led to a successful deployment of the Nginx server, showing how effective Kubernetes can be in a cloud environment.