Vivekanand Education Society's

Institute of Technology

An Autonomous Institute Affiliated to University of Mumbai,, Approved by AICTE & Recognized by Govt. of Maharashtra

Hashu Advani Memorial Complex, Collector Colony, Chembur East, Mumbai - 400074.

Department of Information Technology

A.Y. 2024-25

Advance DevOps Lab
Experiment 04

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

Roll No.	44
Name	GANESH SANJAY PANDHRE
Class	D15B
Subject	Advance DevOps Lab
LO Mapped	LO1: To understand the fundamentals of Cloud Computing and be fully proficient with Cloud based DevOps solution deployment options to meet your business requirements. LO2: To deploy single and multiple container applications and manage application deployments with rollouts in Kubernetes
Grade:	

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

Theory :

kubectl is the command-line interface (CLI) tool that allows users to interact with a Kubernetes cluster. As a central component of Kubernetes, **kubectl** provides the functionality needed to manage applications, inspect cluster resources, and perform administrative tasks through simple commands executed in a terminal.

Importance of Kubectl in Kubernetes Management

kubectl is essential for effective Kubernetes management for several reasons:

1. User-Friendly Interface: **kubectl** offers a command-line interface that simplifies complex operations, making it accessible for developers and administrators.

2. Resource Management: Users can create, update, and delete Kubernetes resources such as pods, deployments, services, and namespaces with straightforward commands.

3. Deployment and Scaling: **kubectl** facilitates the deployment of containerized applications and allows users to easily scale them up or down based on current demands.

4. Monitoring and Troubleshooting: The tool enables users to monitor the health and status of applications running in the cluster. It provides commands to view logs, describe resources, and check the current state of pods and services, which aids in troubleshooting issues.

5. Configuration Management: **kubectl** supports YAML configuration files that define the desired state of applications and resources, allowing users to apply changes consistently and repeatedly across different environments.

Key Features of Kubectl

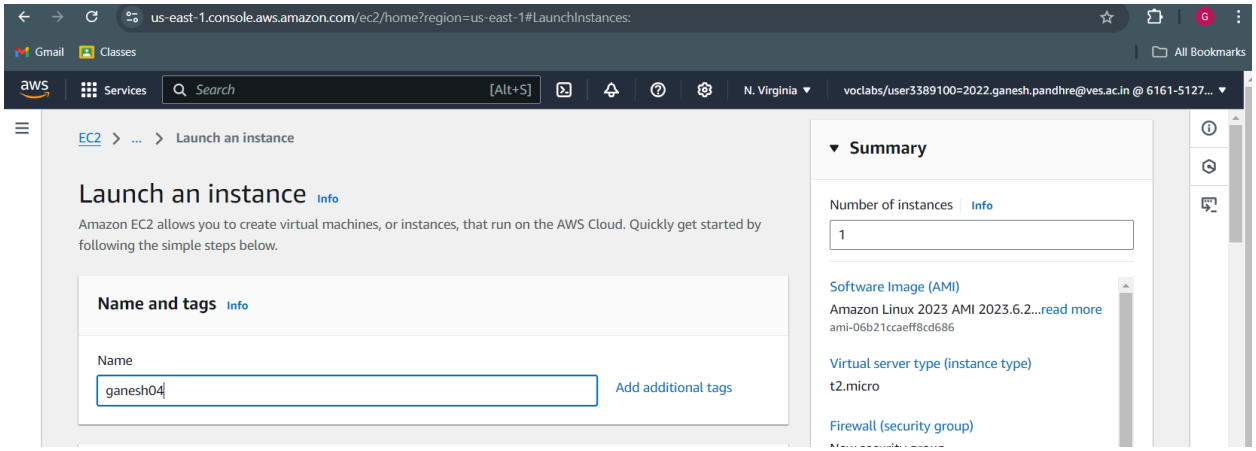
- Resource Discovery: **kubectl** can list all resources in a Kubernetes cluster, providing an overview of what is running and its current status.

- Detailed Resource Descriptions: The tool can display detailed information about specific resources, including configuration, current state, events, and resource utilization.

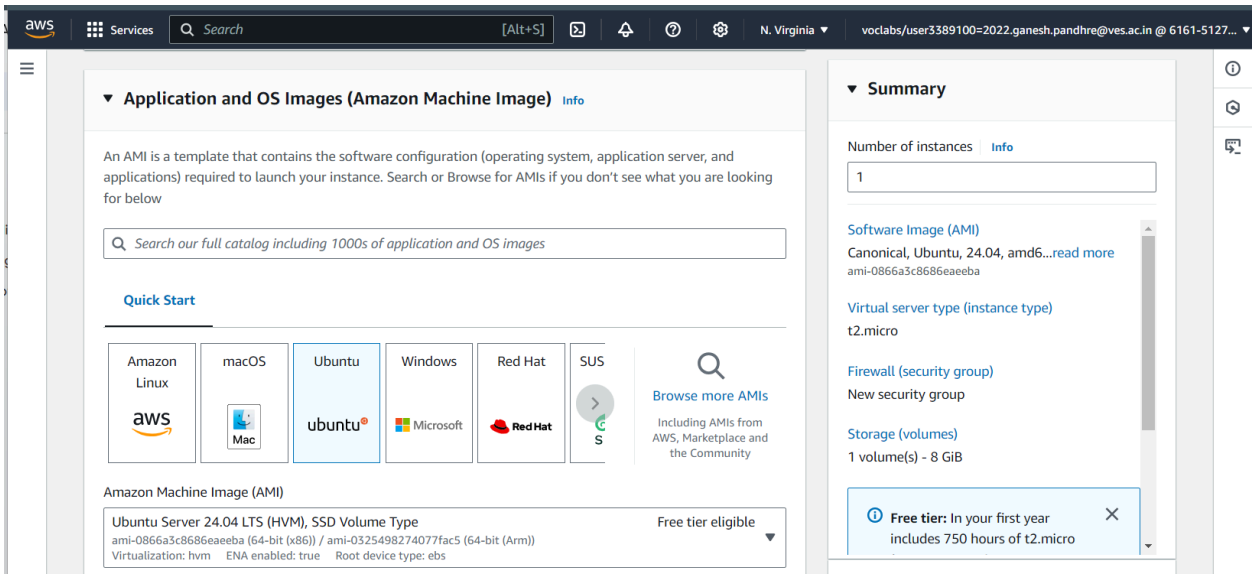
- Access to Container Logs: Users can view the logs generated by application containers, helping diagnose issues and understand application behavior.

Namespace Management: **kubectl** allows for the management of namespaces, which help organize resources and provide isolation within a cluster.

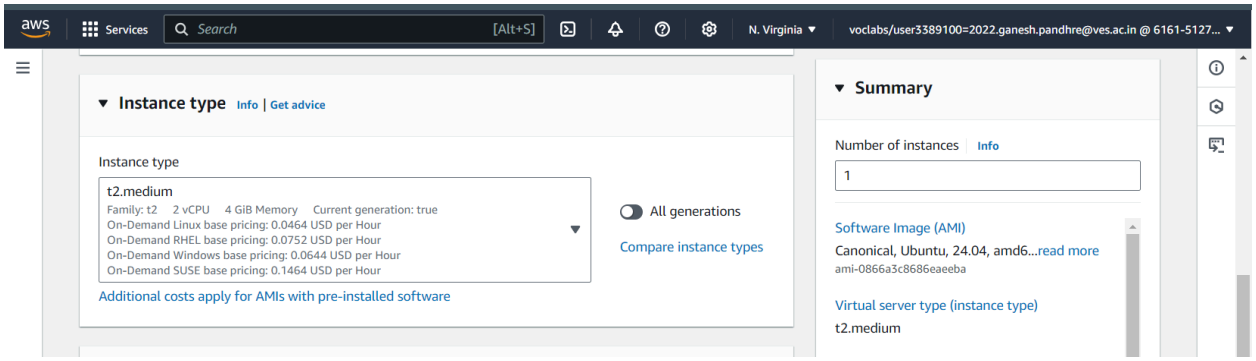
Launch an EC2 Instance



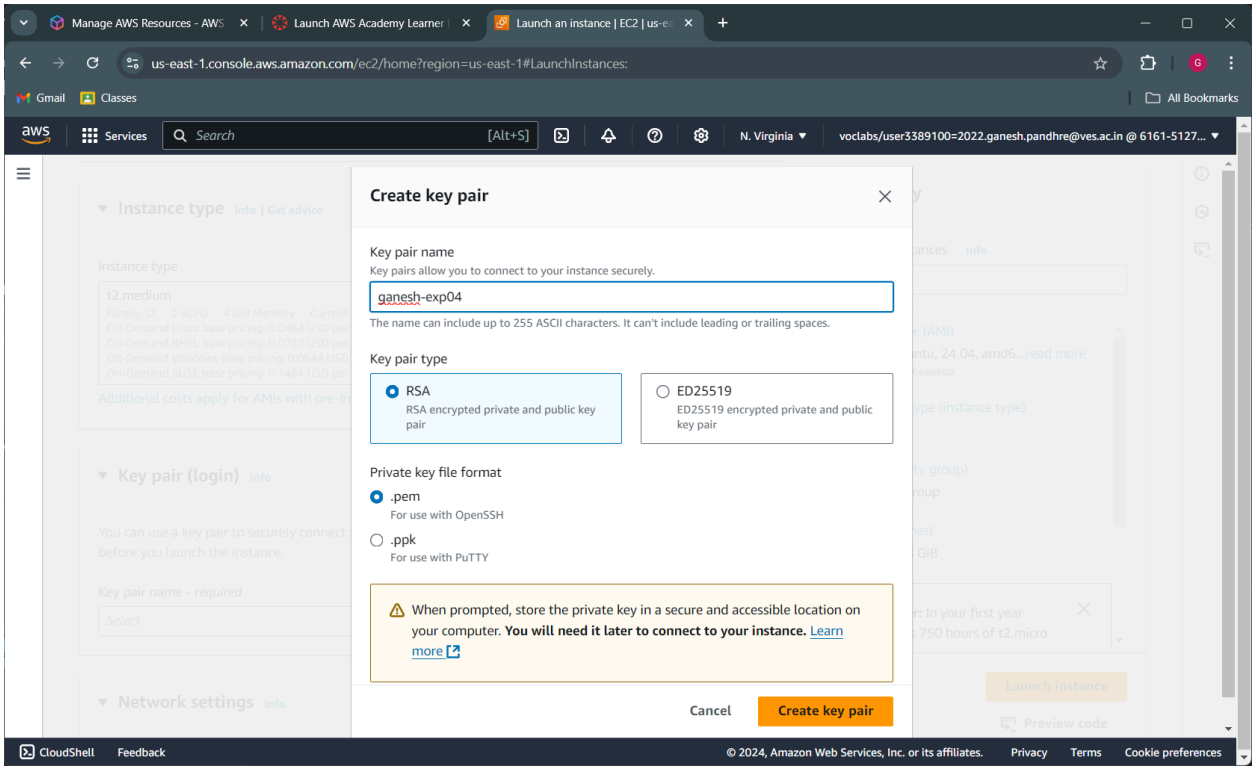
Choose Ubuntu Server 20.04 LTS (HVM), SSD Volume Type as your AMI.

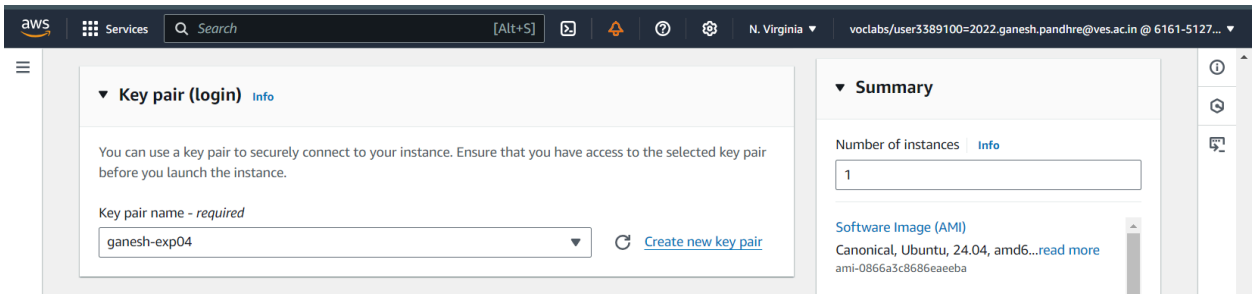


Select t2.medium as the instance type (2 CPUs).

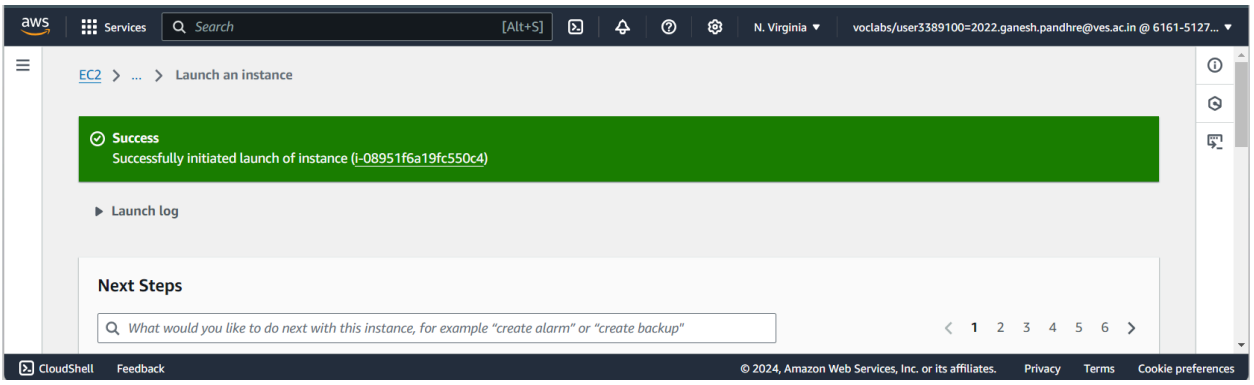


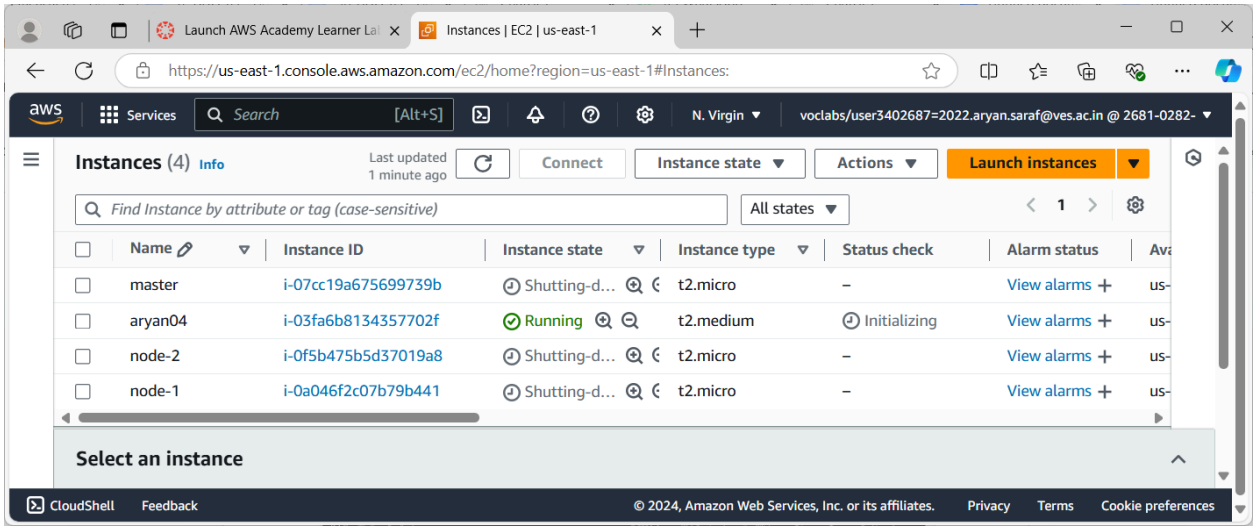
Select Create a new key pair, name it (e.g., ganesh-exp04), and click Download Key Pair. This will download a .pem file to your computer.



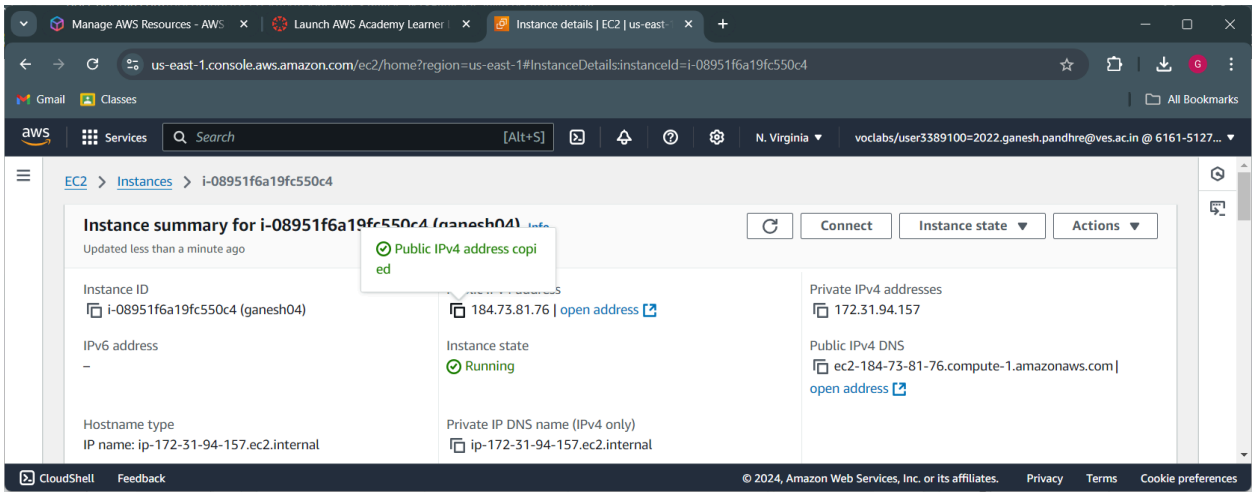


Launch the EC2 instance.

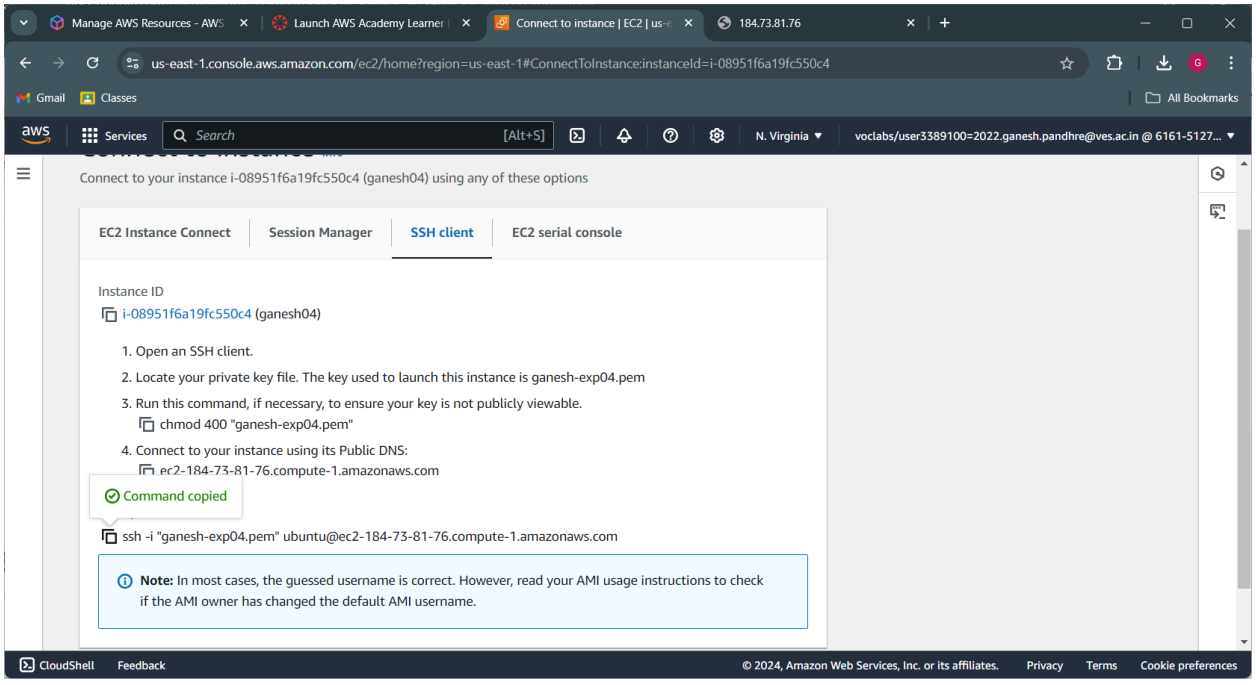




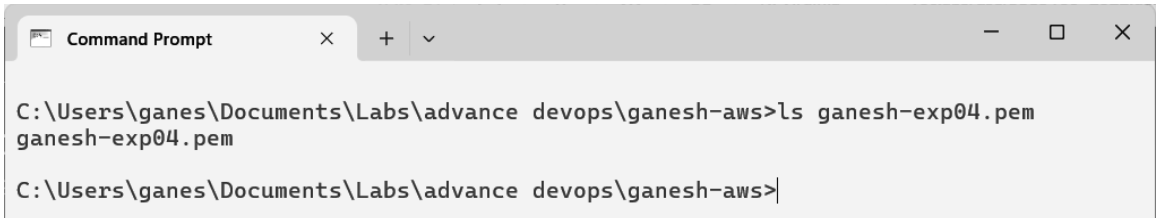
Click on the instance id of the newly created ec2 instance and copy the public url of it.



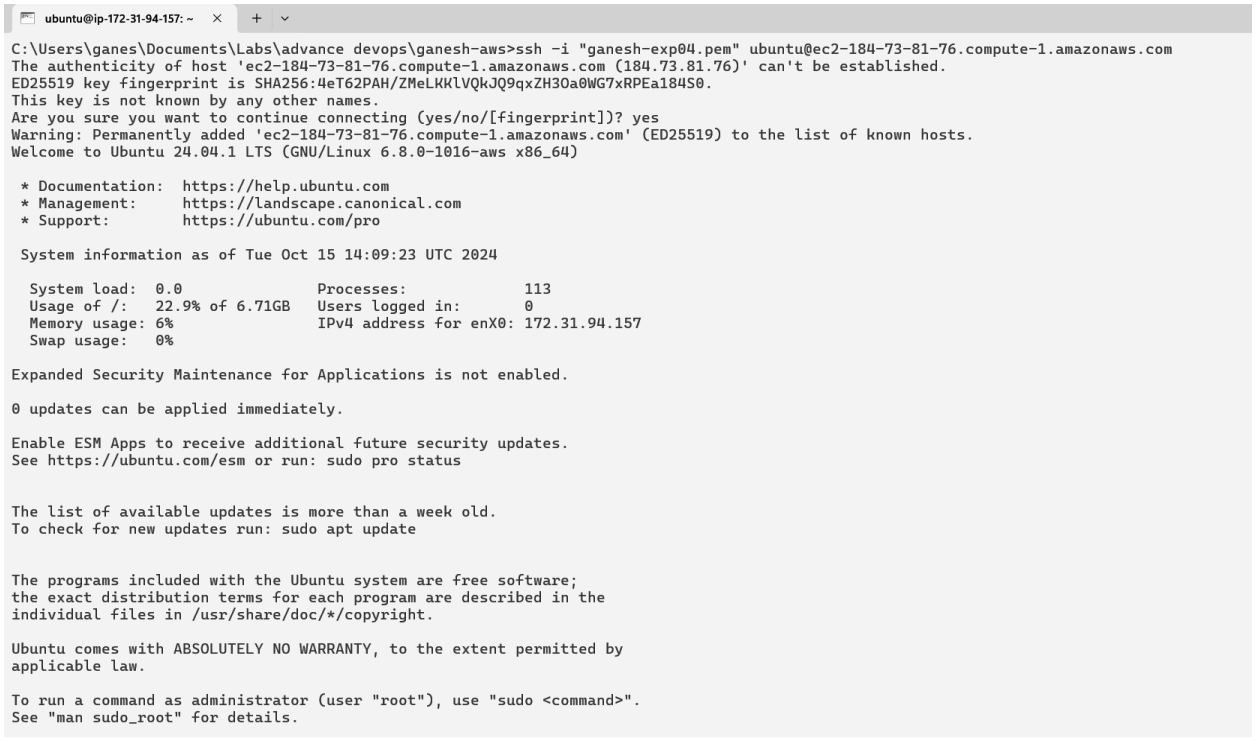
Click on connect and copy the command as shown



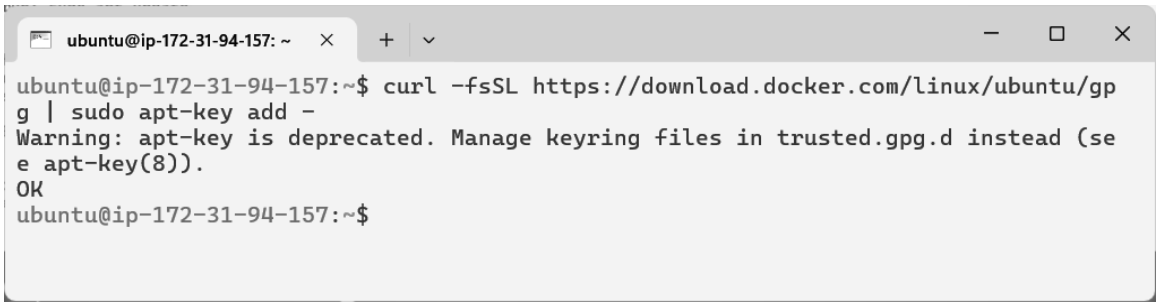
If you are using Windows, you might need a terminal like **Git Bash** or **PuTTY**.
Use the cd command to navigate to the folder where your downloaded key is located.



Run the following command, replacing the placeholder with your actual EC2 public DNS:
ssh -i "ganesh-exp04.pem" ubuntu@ec2-54-92-219-25.compute-1.amazonaws.com



To install Docker, Run the Following Commands:
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -



curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/trusted.gpg.d/docker.gpg

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"

```
ubuntu@ip-172-31-94-157: ~
ubuntu@ip-172-31-94-157:~$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu
$(lsb_release -cs) stable"
Repository: 'deb [arch=amd64] https://download.docker.com/linux/ubuntu noble stable'
Description:
Archive for codename: noble components: stable
More info: https://download.docker.com/linux/ubuntu
Adding repository.
Press [ENTER] to continue or Ctrl-c to cancel.
Adding deb entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Adding disabled deb-src entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 https://download.docker.com/linux/ubuntu noble InRelease [48.8 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8328 B]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [592 kB]
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [144 kB]
Get:16 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [15.3 kB]
Get:17 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 c-n-f Metadata [10.2 kB]
Get:18 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [697 kB]
Get:19 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe Translation-en [206 kB]
Get:20 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [45.0 kB]
Get:21 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 c-n-f Metadata [19.6 kB]
Get:22 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [385 kB]
Get:23 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted Translation-en [74.4 kB]
Get:24 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packages [14.8 kB]
Get:25 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse Translation-en [3820 B]
Get:26 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [212 B]
Get:27 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 c-n-f Metadata [552 B]
```

sudo apt-get update

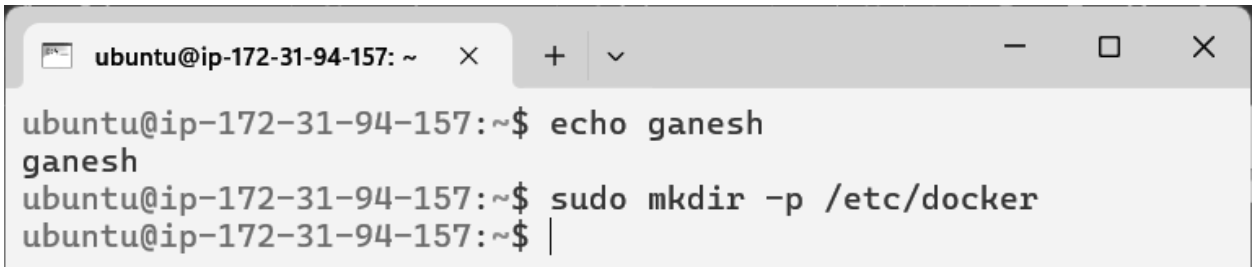
```
ubuntu@ip-172-31-94-157: ~
ubuntu@ip-172-31-94-157:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu noble InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
ubuntu@ip-172-31-94-157:~$
```

sudo apt-get install -y docker-ce

```
ubuntu@ip-172-31-94-157: ~
ubuntu@ip-172-31-94-157:~$ sudo apt-get install -y docker-ce
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done

The following additional packages will be installed:
  containerd.io docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0
  pigz slirp4netns
Suggested packages:
  aufs-tools cgroupfs-mount | cgroup-lite
The following NEW packages will be installed:
  containerd.io docker-buildx-plugin docker-ce docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7
  libslirp0 pigz slirp4netns
0 upgraded, 10 newly installed, 0 to remove and 19 not upgraded.
Need to get 123 MB of archives.
After this operation, 442 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 pigz amd64 2.8-1 [65.6 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libltdl7 amd64 2.4.7-7build1 [40.3 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libslirp0 amd64 4.7.0-1ubuntu3 [63.8 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 slirp4netns amd64 1.2.1-1build2 [34.9 kB]
Get:5 https://download.docker.com/linux/ubuntu noble/stable amd64 containerd.io amd64 1.7.22-1 [29.5 MB]
Get:6 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-buildx-plugin amd64 0.17.1-1~ubuntu.24.04~noble
[30.3 MB]
Get:7 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-ce-cli amd64 5:27.3.1-1~ubuntu.24.04~noble [15.0
 MB]
Get:8 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-ce amd64 5:27.3.1-1~ubuntu.24.04~noble [25.6 MB]
Get:9 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-ce-rootless-extras amd64 5:27.3.1-1~ubuntu.24.04
~noble [9588 kB]
Get:10 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-compose-plugin amd64 2.29.7-1~ubuntu.24.04~nobl
e [12.7 MB]
Fetched 123 MB in 1s (82.8 MB/s)
debconf: unable to initialize frontend: Dialog
debconf: (Dialog frontend requires a screen at least 13 lines tall and 31 columns wide.)
```

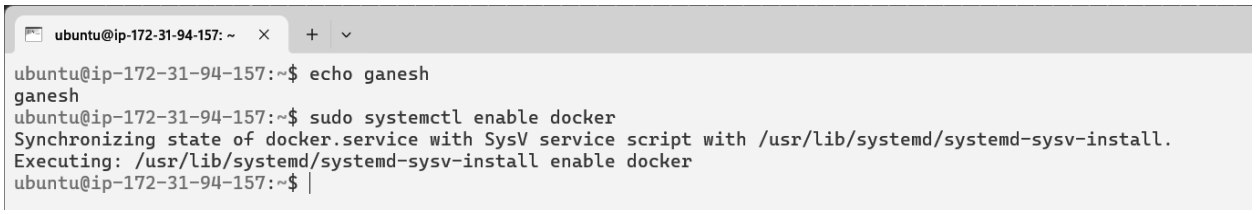
Configure Docker

sudo mkdir -p /etc/docker

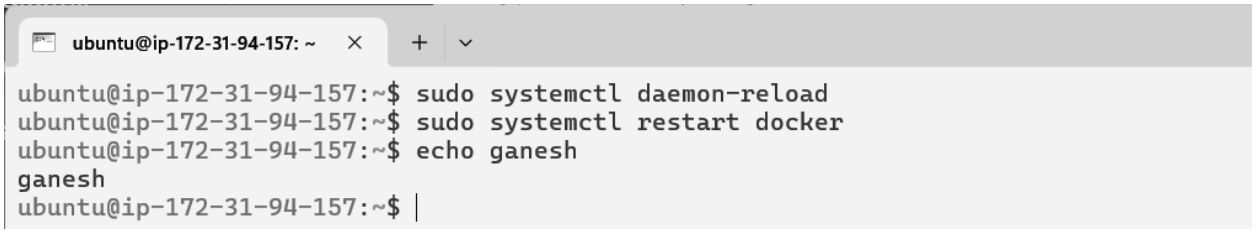


cat <<EOF | sudo tee /etc/docker/daemon.json
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
EOF

sudo systemctl enable docker

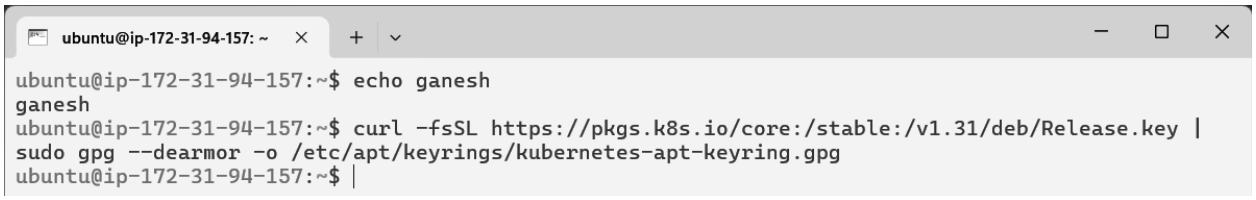


sudo systemctl daemon-reload
sudo systemctl restart docker



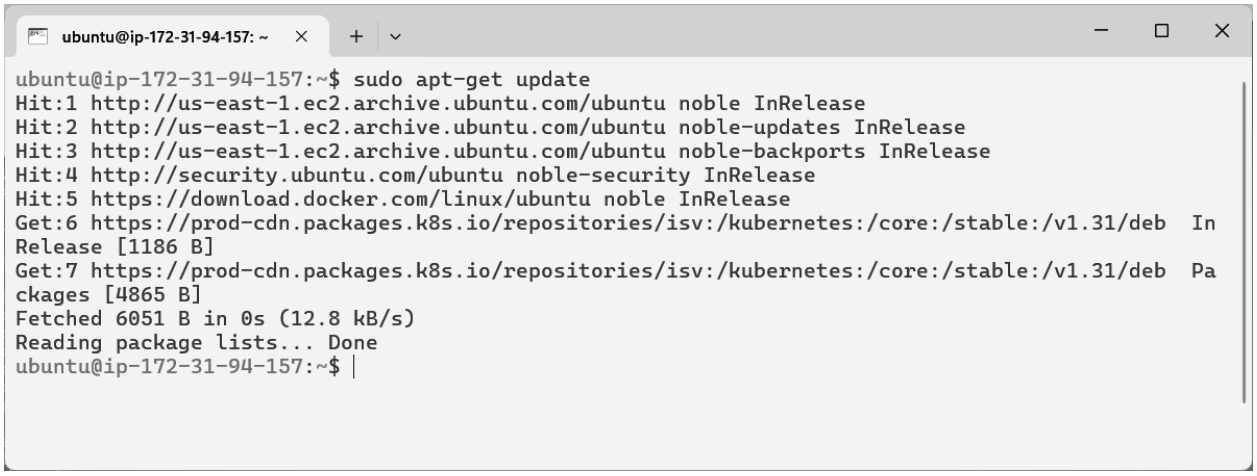
To Install Kubernetes, Add the Kubernetes Repository

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

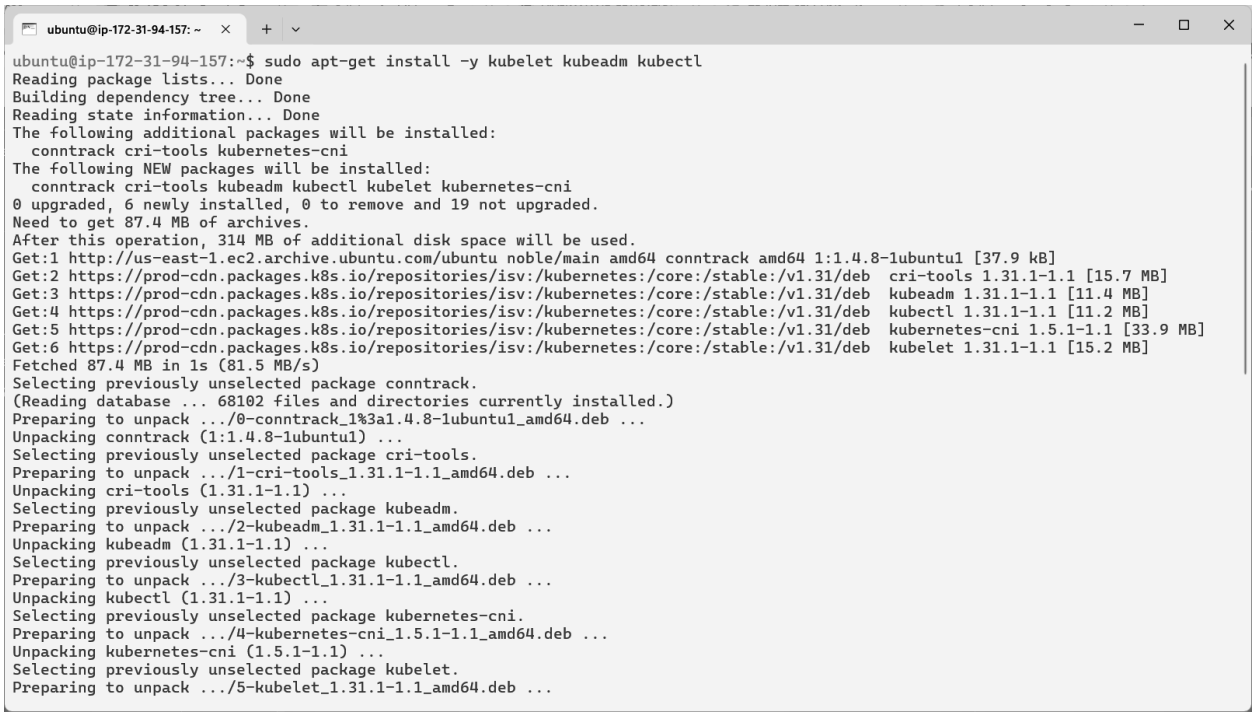


echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list

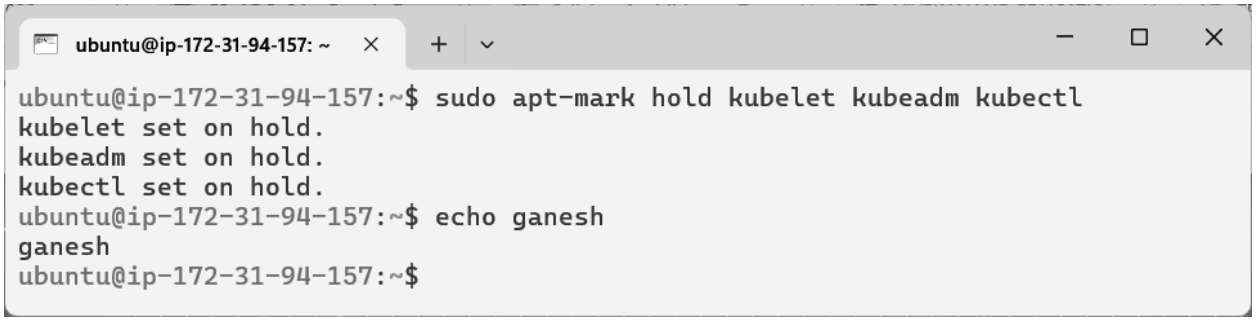
sudo apt-get update



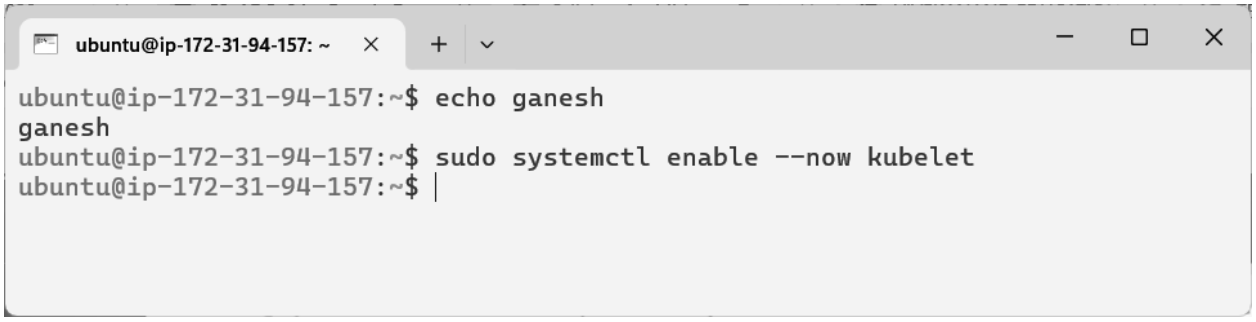
sudo apt-get install -y kubelet kubeadm kubectl



sudo apt-mark hold kubelet kubeadm kubectl



Enable and Start Kubelet:
sudo systemctl enable --now kubelet



To Initialize the Kubernetes Cluster, Run the Command

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

If you encounter errors, run the following commands to fix containerd issues:

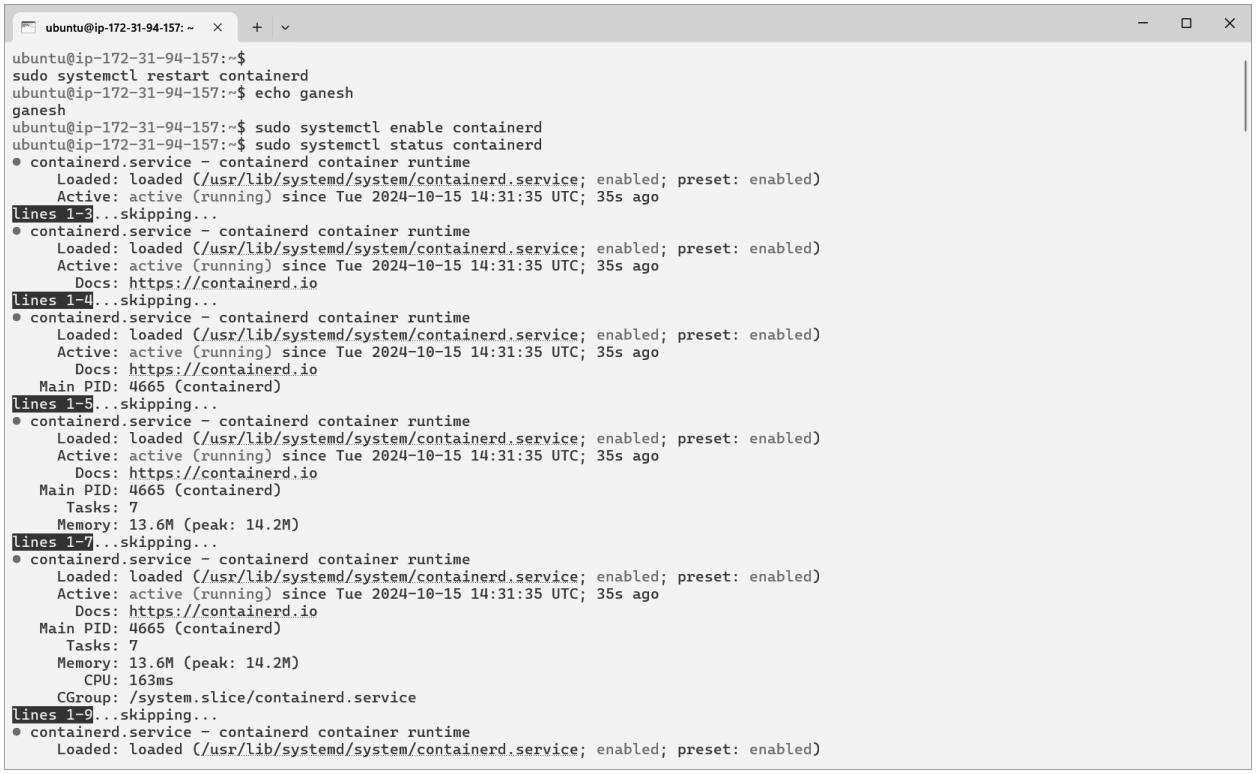
sudo apt-get install -y containerd

```
ubuntu@ip-172-31-94-157: ~
ubuntu@ip-172-31-94-157:~$ echo ganesh
ganesh
ubuntu@ip-172-31-94-157:~$ 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 required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz
  slirp4netns
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 19 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 amd64 runc amd64 1.1.12-0ubuntu3.1 [8599 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 containerd amd64 1.7.12-0ubuntu4.1 [38.6 MB]
Fetched 47.2 MB in 1s (39.5 MB/s)
(Reading database ... 68159 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 ... 68139 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.
```

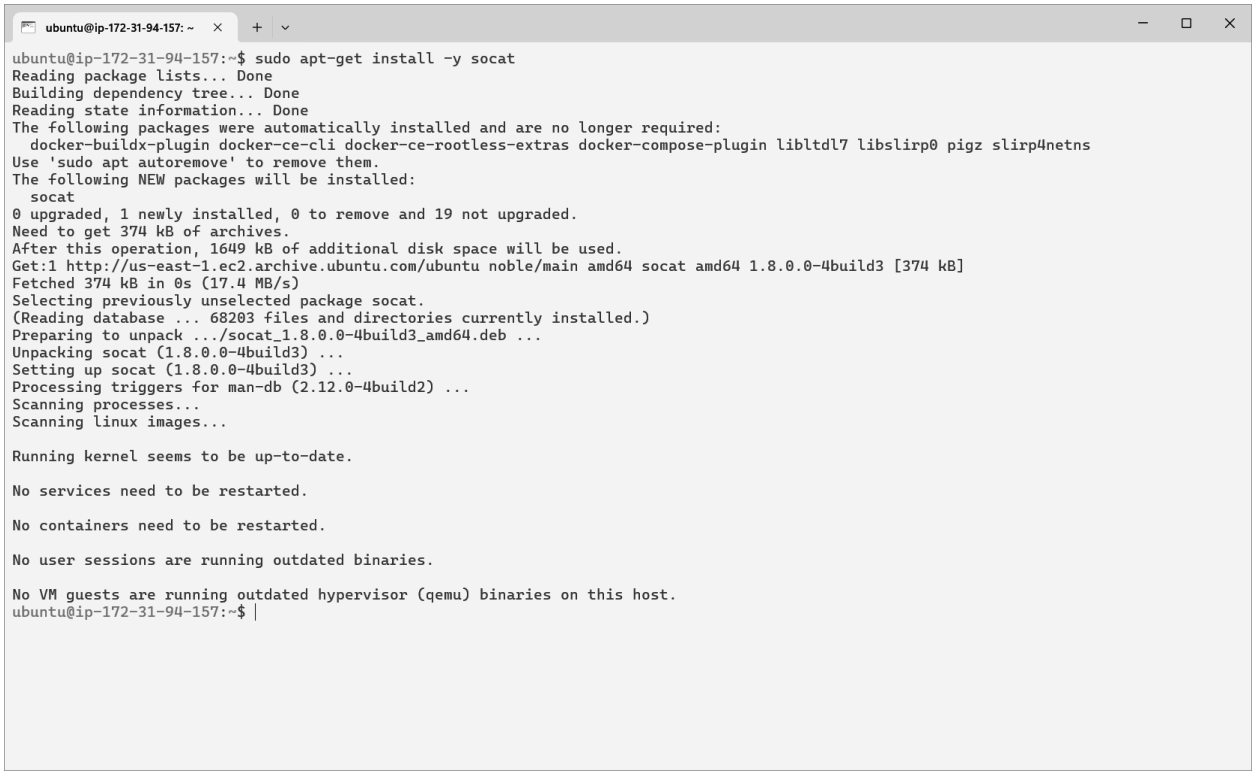
sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml

```
ubuntu@ip-172-31-94-157: ~
ubuntu@ip-172-31-94-157:~$ sudo mkdir -p /etc/containerd
ubuntu@ip-172-31-94-157:~$ echo ganesh
ganesh
ubuntu@ip-172-31-94-157:~$ 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]
```

sudo systemctl restart containerd
sudo systemctl enable containerd
sudo systemctl status containerd

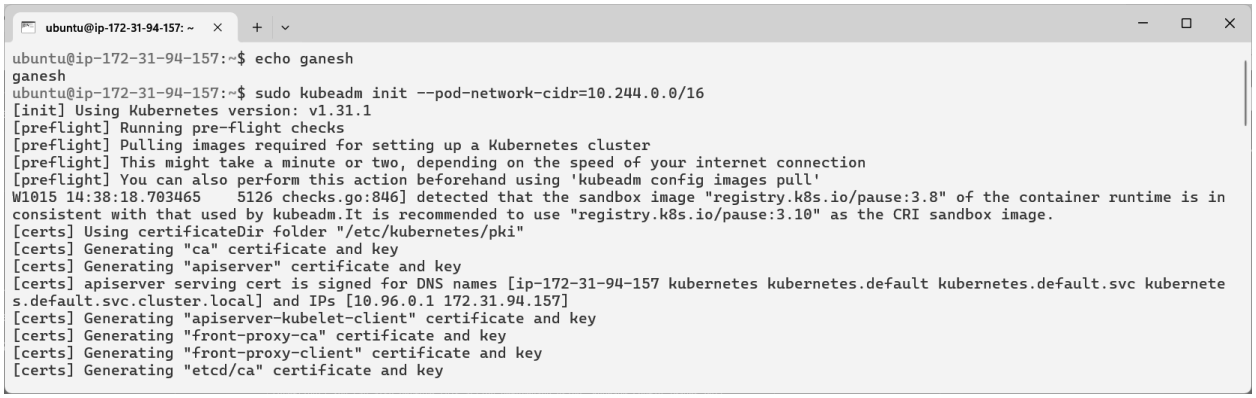


sudo apt-get install -y socat



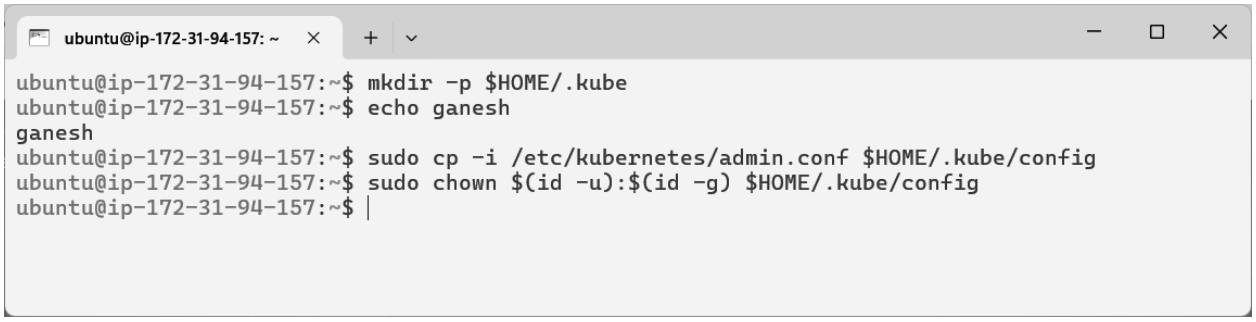
Re-run the Init Command:

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

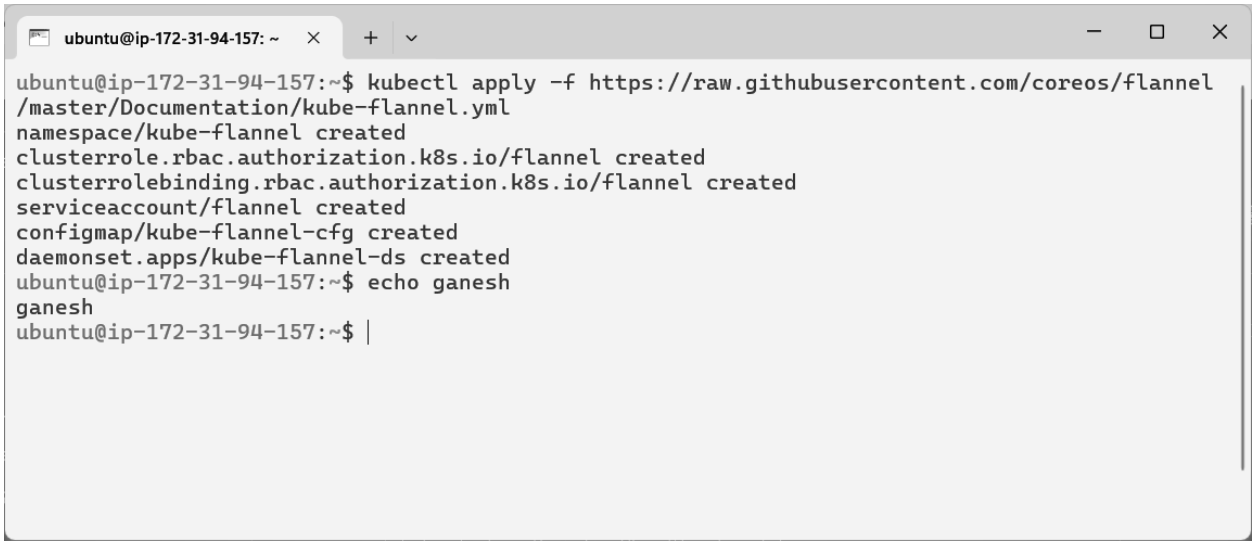


To Configure kubectl, **Set Up kubeconfig**

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



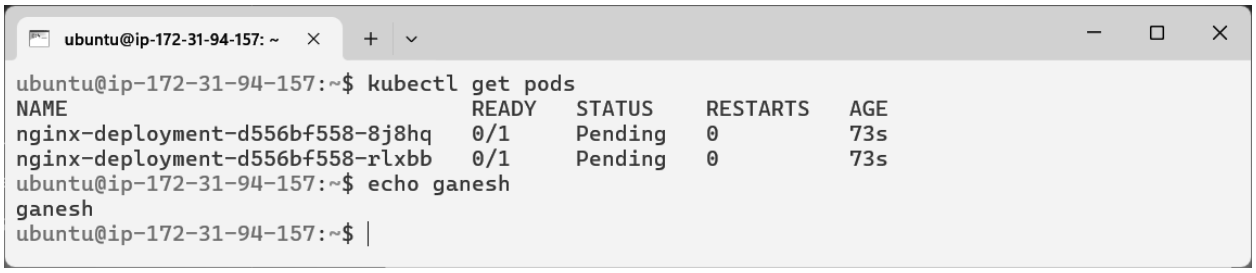
Install Flannel (a networking plugin):
kubectl apply -f
https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml



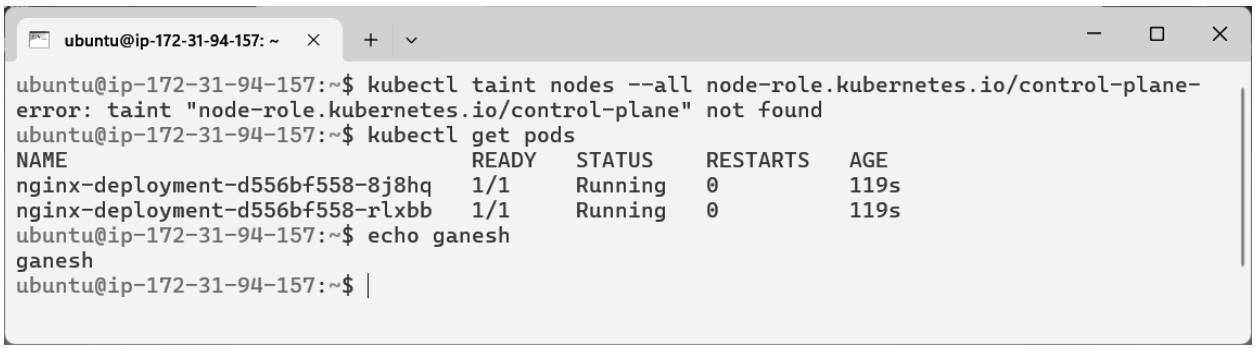
To Deploy Nginx Server, **Create a Deployment:**

kubectl apply -f https://k8s.io/examples/application/deployment.yaml

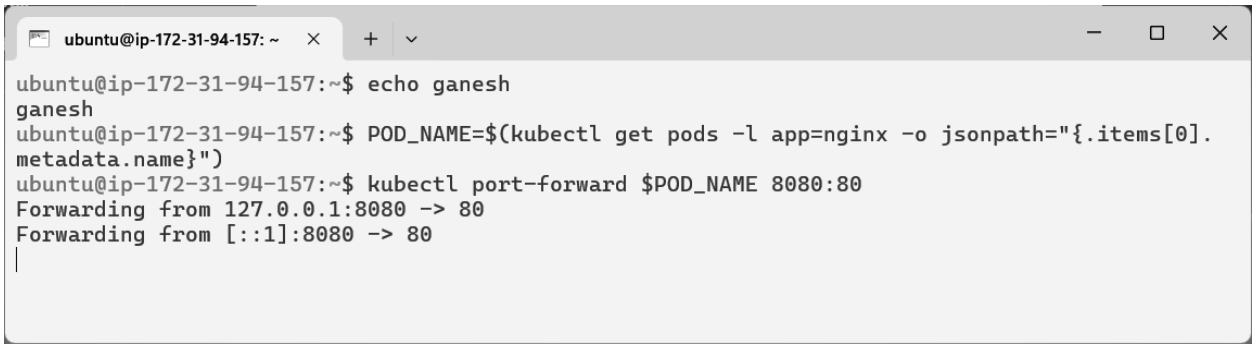
Check Pods:
kubectl get pods



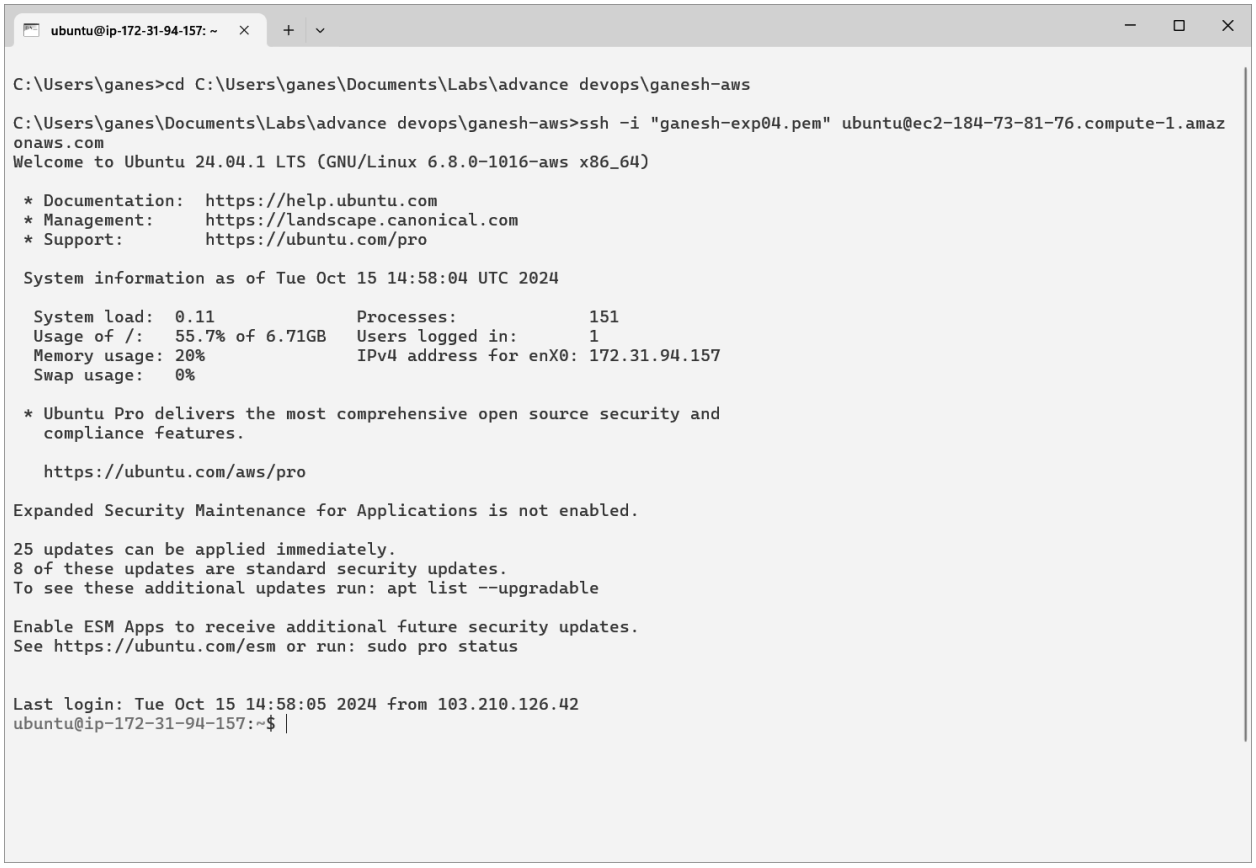
If the pod status is pending, you might need to remove the control-plane taint:
kubectl taint nodes --all node-role.kubernetes.io/control-plane-



Port Forward to Access Nginx: Find the Pod name
POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")
kubectl port-forward $POD_NAME 8080:80



Open a New Terminal and SSH back into your EC2 instance.



Use Curl to Check Nginx:
curl --head http://127.0.0.1:8080

If you see `200 OK`, your Nginx server is successfully running.

Conclusion :

Understanding **kubectl** is crucial for anyone working with Kubernetes, as it serves as the primary interface for managing applications and resources. Through `kubectl`, users can effectively deploy, monitor, and troubleshoot applications, ensuring that they run smoothly in a Kubernetes environment.