Kubernetes Task

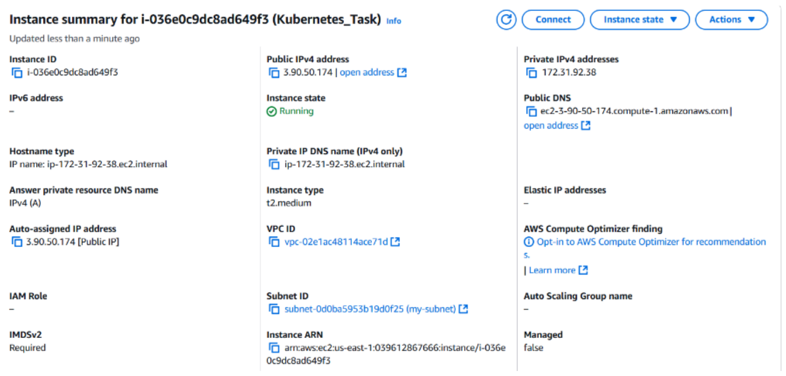
TASKS

Work Flow:

- Create an EC2 instance with the help of AWS Management Console with linux OS of required configuration and ensure that where the instance type should be t2.medium.
- Now, Connect an EC2 instance with an help of Windows Terminal or Gitbash or Vbox.
- To connect an EC2 instance the command is:
  - ssh -i “**key_file**” ec2-user@“**Public_IP_address**”

Key_file --- Key file of the instance with the extension .pem

Public_IP_address --- Public IP address of the instance.


Instance summary for i-036e0c9dc8ad649f3 (Kubernetes_Task) Info

Updated less than a minute ago

Instance ID
i-036e0c9dc8ad649f3

IPv6 address
–

Hostname type
IP name: ip-172-31-92-38.ec2.internal

Answer private resource DNS name
IPv4 (A)

Auto-assigned IP address
3.90.50.174 [Public IP]

IAM Role
–

IMDSv2
Required

Public IPv4 address
3.90.50.174 | open address

Instance state
Running

Private IP DNS name (IPv4 only)
ip-172-31-92-38.ec2.internal

Instance type
t2.medium

VPC ID
vpc-02e1ac48114ace71d

Subnet ID
subnet-0d0ba5953b19d0f25 (my-subnet)

Instance ARN
arn:aws:ec2:us-east-1:039612867666:instance/i-036e0c9dc8ad649f3

Private IPv4 addresses
172.31.92.38

Public DNS
ec2-3-90-50-174.compute-1.amazonaws.com | open address

Elastic IP addresses
–

AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. | Learn more

Auto Scaling Group name
–

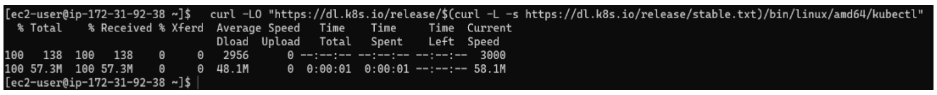
Managed
false

1. **Setup minikube at your local and explore creating namespaces (Go through official documentation).**

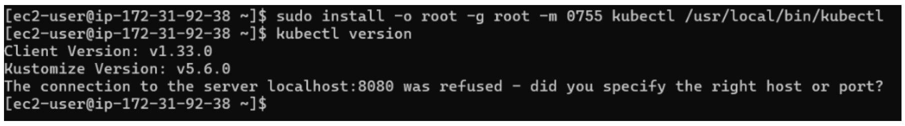
Step 1: Install an Kubernetes & Minikube in an EC2 instance

Install An Kubernetes :

- ✓ To install an kubernetes in linux machine go to an official website by using below link.
- ✓ Link : https://kubernetes.io/docs/tasks/tools/install-kubectl/
- ✓ Now you can see the instructions given in the official page to install an kubernetes, follow all the steps to install.
- ✓ And also where the command as given below to install an Kubernetes from an official Page.
  - curl -LO "https://dl.k8s.io/release/**$(**curl -L -s https://dl.k8s.io/release/stable.txt**)**/bin/linux/amd64/kubectl"


```
[ec2-user@ip-172-31-92-38 ~]$   curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   138  100   138    0     0   2956      0 --:--:-- --:--:-- --:--:--  3000
100 57.3M  100 57.3M    0     0  48.1M      0  0:00:01  0:00:01 --:--:-- 58.1M
[ec2-user@ip-172-31-92-38 ~]$
```

- sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl



```
[ec2-user@ip-172-31-92-38 ~]$ sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl
[ec2-user@ip-172-31-92-38 ~]$ kubectl version
Client Version: v1.33.0
Kustomize Version: v5.6.0
The connection to the server localhost:8080 was refused - did you specify the right host or port?
[ec2-user@ip-172-31-92-38 ~]$
```

Install An Minikube :

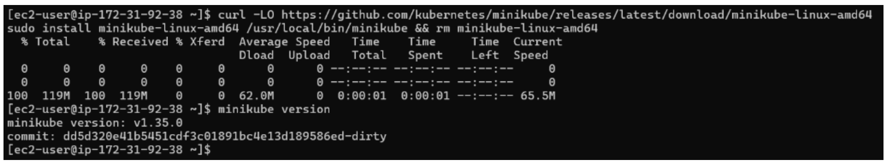
✓ To install an Minukube in linux machine go to an official website by using below link.

✓ Link : https://minikube.sigs.k8s.io/docs/start/

✓ Now you can see the instructions given in the official page to install an kubernetes, follow all the steps to install.

✓ And also where the command as given below to install an Minikube from an official Page.

- curl -LO https://github.com/kubernetes/minikube/releases/latest/download/minikube-linux-amd64
- sudo install minikube-linux-amd64 /usr/local/bin/minikube && rm minikube-linux-amd64



```
[ec2-user@ip-172-31-92-38 ~]$ curl -LO https://github.com/kubernetes/minikube/releases/latest/download/minikube-linux-amd64
sudo install minikube-linux-amd64 /usr/local/bin/minikube && rm minikube-linux-amd64
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100  119M  100  119M    0     0  62.0M      0  0:00:01  0:00:01 --:--:-- 65.5M
[ec2-user@ip-172-31-92-38 ~]$ minikube version
minikube version: v1.35.0
commit: dd5d320e41b5451cdf3c01891bc4e13d189586ed-dirty
[ec2-user@ip-172-31-92-38 ~]$
```

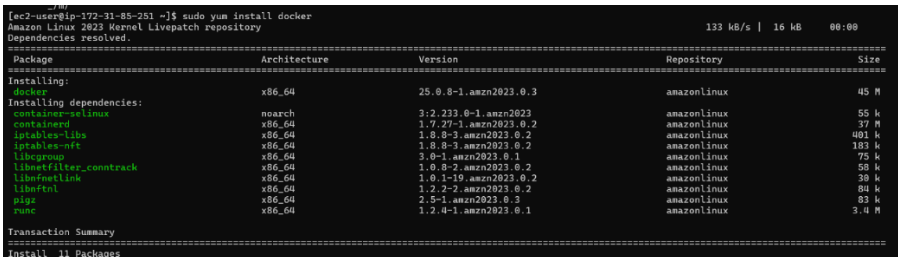
Install An Docker:

✓ Before starting an minikube, we have to install an docker then only where the minikube will works, otherwise it will not work.

✓ Follow the below steps to install an docker in your instance.

✓ To install an docker in linux machine, the command is:

- **sudo yum install docker**



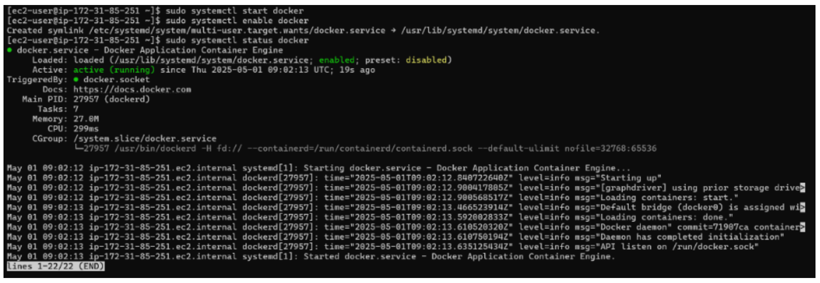
```
[ec2-user@ip-172-31-85-251 ~]$ sudo yum install docker
Amazon Linux 2023 Kernel Livepatch repository                              133 kB/s |  16 kB     00:00
Dependencies resolved.
==========================================================================================================
 Package                  Architecture     Version                          Repository              Size
==========================================================================================================
Installing:
 docker                   x86_64           25.0.8-1.amzn2023.0.3            amazonlinux             45 M
Installing dependencies:
 container-selinux        noarch           3:2.233.0-1.amzn2023             amazonlinux             55 k
 containerd               x86_64           1.7.27-1.amzn2023.0.2            amazonlinux             37 M
 iptables-libs            x86_64           1.8.8-3.amzn2023.0.2             amazonlinux            401 k
 iptables-nft             x86_64           1.8.8-3.amzn2023.0.2             amazonlinux            183 k
 libcgroup                x86_64           3.0-1.amzn2023.0.1               amazonlinux             75 k
 libnetfilter_conntrack   x86_64           1.0.8-2.amzn2023.0.2             amazonlinux             58 k
 libnfnetlink             x86_64           1.0.1-19.amzn2023.0.2            amazonlinux             30 k
 libnftnl                 x86_64           1.2.2-2.amzn2023.0.2             amazonlinux             84 k
 pigz                     x86_64           2.5-1.amzn2023.0.3               amazonlinux             83 k
 runc                     x86_64           1.2.4-1.amzn2023.0.1             amazonlinux            3.4 M

Transaction Summary
==========================================================================================================
Install  11 Packages
```

✓ To start and enable an docker service, The command is:

- **sudo systemctl start docker**
- **sudo systemctl enable docker**

✓ To check the status of the docker service, The command is:

- **sudo systemctl status docker**



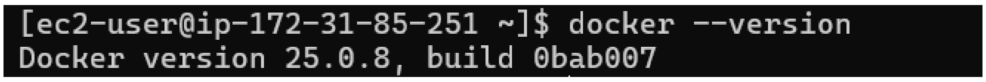
```
[ec2-user@ip-172-31-85-251 ~]$ sudo systemctl start docker
[ec2-user@ip-172-31-85-251 ~]$ sudo systemctl enable docker
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
[ec2-user@ip-172-31-85-251 ~]$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: disabled)
     Active: active (running) since Thu 2025-05-01 09:02:13 UTC; 19s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 27957 (dockerd)
      Tasks: 7
     Memory: 27.0M
        CPU: 299ms
     CGroup: /system.slice/docker.service
             └─27957 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock --default-ulimit nofile=32768:65536

May 01 09:02:12 ip-172-31-85-251.ec2.internal systemd[1]: Starting docker.service - Docker Application Container Engine...
May 01 09:02:12 ip-172-31-85-251.ec2.internal dockerd[27957]: time="2025-05-01T09:02:12.840722640Z" level=info msg="Starting up"
May 01 09:02:12 ip-172-31-85-251.ec2.internal dockerd[27957]: time="2025-05-01T09:02:12.900417805Z" level=info msg="[graphdriver] using prior storage drive>
May 01 09:02:12 ip-172-31-85-251.ec2.internal dockerd[27957]: time="2025-05-01T09:02:12.900568517Z" level=info msg="Loading containers: start."
May 01 09:02:13 ip-172-31-85-251.ec2.internal dockerd[27957]: time="2025-05-01T09:02:13.466523914Z" level=info msg="Default bridge (docker0) is assigned wi>
May 01 09:02:13 ip-172-31-85-251.ec2.internal dockerd[27957]: time="2025-05-01T09:02:13.592002833Z" level=info msg="Loading containers: done."
May 01 09:02:13 ip-172-31-85-251.ec2.internal dockerd[27957]: time="2025-05-01T09:02:13.610520320Z" level=info msg="Docker daemon" commit=71907ca container>
May 01 09:02:13 ip-172-31-85-251.ec2.internal dockerd[27957]: time="2025-05-01T09:02:13.610750194Z" level=info msg="Daemon has completed initialization"
May 01 09:02:13 ip-172-31-85-251.ec2.internal dockerd[27957]: time="2025-05-01T09:02:13.635125434Z" level=info msg="API listen on /run/docker.sock"
May 01 09:02:13 ip-172-31-85-251.ec2.internal systemd[1]: Started docker.service - Docker Application Container Engine.
lines 1-22/22 (END)
```

✓ To add an ec2-user to docker group, the command is:

- **sudo usermod -aG docker ec2-user**

✓ To check an version of the docker and to verify an installation, the command is:

- **docker --version**

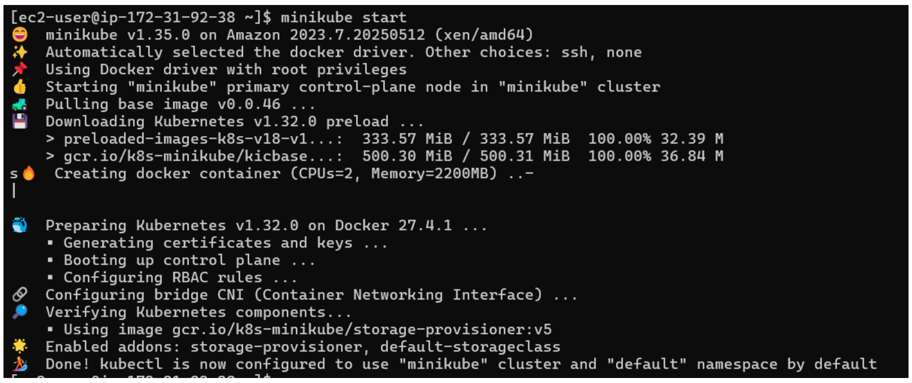


```
[ec2-user@ip-172-31-85-251 ~]$ docker --version
Docker version 25.0.8, build 0bab007
```

Step 2: Now start an minikube.

✓ Once the Kubernetes and minikube is installed now we are going to start an minikube,To start an minikube, The command is:

- **minikube start**

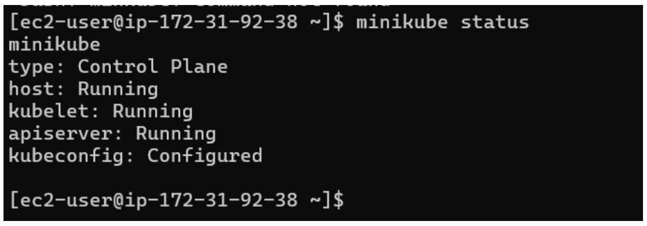


```
[ec2-user@ip-172-31-92-38 ~]$ minikube start
😄  minikube v1.35.0 on Amazon 2023.7.20250512 (xen/amd64)
✨  Automatically selected the docker driver. Other choices: ssh, none
📌  Using Docker driver with root privileges
👍  Starting "minikube" primary control-plane node in "minikube" cluster
🚜  Pulling base image v0.0.46 ...
💾  Downloading Kubernetes v1.32.0 preload ...
    > preloaded-images-k8s-v18-v1...:  333.57 MiB / 333.57 MiB  100.00% 32.39 M
    > gcr.io/k8s-minikube/kicbase...:  500.30 MiB / 500.31 MiB  100.00% 36.84 M
s🔥  Creating docker container (CPUs=2, Memory=2200MB) ..-
|

🐳  Preparing Kubernetes v1.32.0 on Docker 27.4.1 ...
    ▪ Generating certificates and keys ...
    ▪ Booting up control plane ...
    ▪ Configuring RBAC rules ...
🔗  Configuring bridge CNI (Container Networking Interface) ...
🔎  Verifying Kubernetes components...
    ▪ Using image gcr.io/k8s-minikube/storage-provisioner:v5
🌟  Enabled addons: storage-provisioner, default-storageclass
🏄  Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
```

✓ The above command will Create a local Kubernetes cluster using a VM or Docker container and Set up kubectl context to interact with the local cluster.

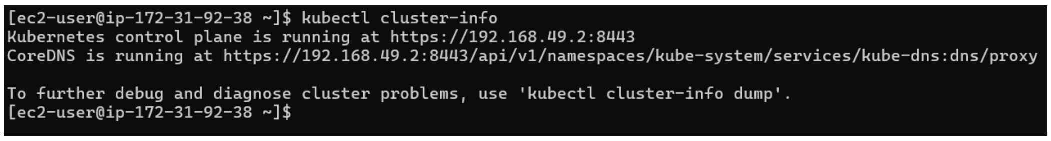
✓ To check the status of the minikube, The command is:

- **minikube status**



```
[ec2-user@ip-172-31-92-38 ~]$ minikube status
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured

[ec2-user@ip-172-31-92-38 ~]$
```

- ✓ To check if the Kubernetes is running or not, the command is:
  - **kubectl cluster-info**



```
[ec2-user@ip-172-31-92-38 ~]$ kubectl cluster-info
Kubernetes control plane is running at https://192.168.49.2:8443
CoreDNS is running at https://192.168.49.2:8443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
[ec2-user@ip-172-31-92-38 ~]$
```

- ✓ You should see info about the Kubernetes control plane.

Step 3: Understand Kubernetes Namespaces.

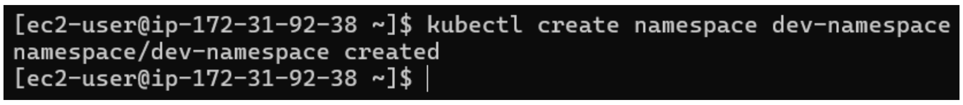
- ✓ Namespaces are a way to separate groups of resources within a single Kubernetes cluster. The names of resources must be different within a namespace, but they can be the same between namespaces.

- ✓ In Kubernetes, *namespaces* provides a mechanism for isolating groups of resources within a single cluster. Names of resources need to be unique within a namespace, but not across namespaces.

- ✓ **Benefits of Namespaces**
  - Allowing teams or projects to existing in their own virtual clusters.
  - Role-based access controls (RBAC)
  - Simple method for separating containerized application
  - Resource quotas

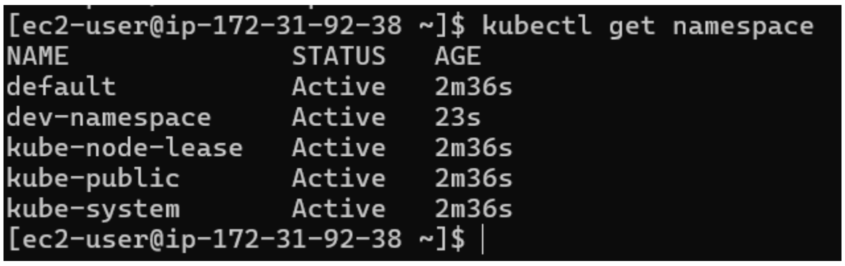
Step 4: Create a Namespace.

- ✓ To create an namespace using kubernetes , the command is :
  - **kubectl create namespace dev-namespace**



```
[ec2-user@ip-172-31-92-38 ~]$ kubectl create namespace dev-namespace
namespace/dev-namespace created
[ec2-user@ip-172-31-92-38 ~]$
```

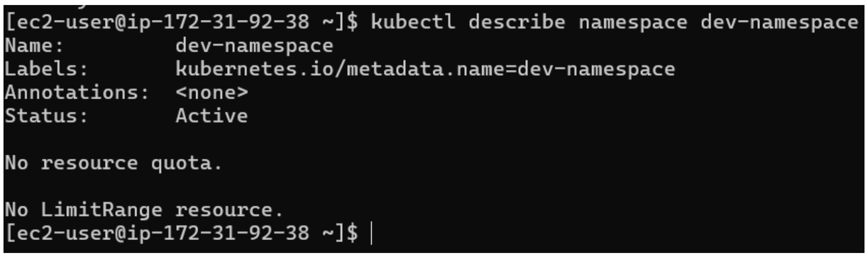
- ✓ To List one or more namespaces using kubernetes, the command is:
  - **kubectl get namespace**



```
[ec2-user@ip-172-31-92-38 ~]$ kubectl get namespace
NAME              STATUS   AGE
default           Active   2m36s
dev-namespace     Active   23s
kube-node-lease   Active   2m36s
kube-public       Active   2m36s
kube-system       Active   2m36s
[ec2-user@ip-172-31-92-38 ~]$
```

✓ To display the detailed state of particular namespaces , the command is :

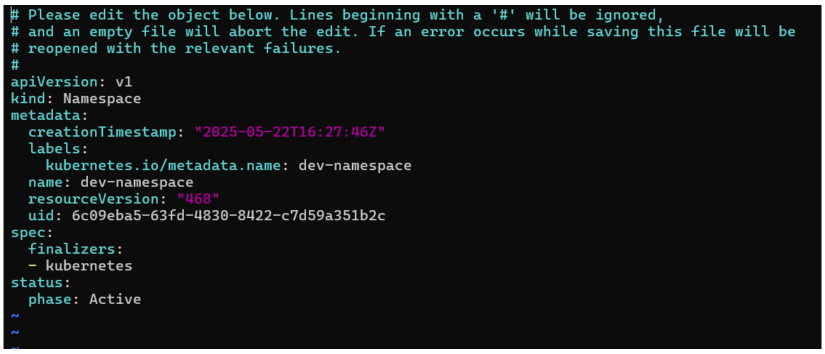
- **kubectl describe namespace dev-namespace**



```
[ec2-user@ip-172-31-92-38 ~]$ kubectl describe namespace dev-namespace
Name:         dev-namespace
Labels:       kubernetes.io/metadata.name=dev-namespace
Annotations:  <none>
Status:       Active

No resource quota.

No LimitRange resource.
[ec2-user@ip-172-31-92-38 ~]$ 
```

✓ To edit and update the definition of a namespace, The command is:

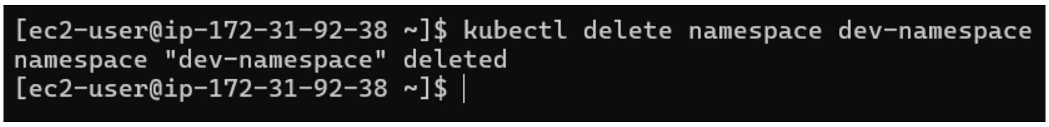
- **kubectl edit namespace dev-namespace**



```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: Namespace
metadata:
  creationTimestamp: "2025-05-22T16:27:46Z"
  labels:
    kubernetes.io/metadata.name: dev-namespace
  name: dev-namespace
  resourceVersion: "468"
  uid: 6c09eba5-63fd-4830-8422-c7d59a351b2c
spec:
  finalizers:
  - kubernetes
status:
  phase: Active
~
~
```

✓ To delete an particular namespace, the command is:

- **kubectl delete namespace dev-namespace**



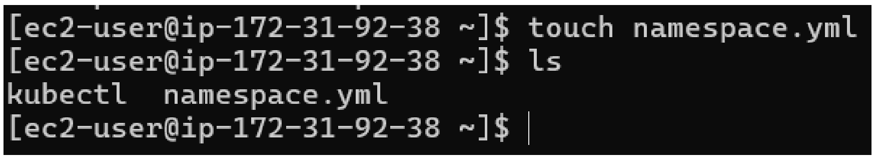
```
[ec2-user@ip-172-31-92-38 ~]$ kubectl delete namespace dev-namespace
namespace "dev-namespace" deleted
[ec2-user@ip-172-31-92-38 ~]$ 
```

Step 5: Define namespaces in YAML files.

✓ You can also define namespaces in YAML files using the namespace field.

✓ To create one YAML file for creation of namespace, the command:

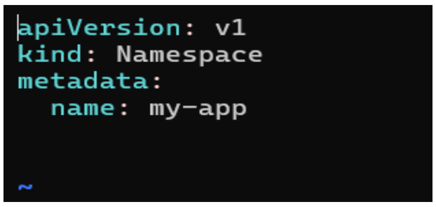
- **touch namespace.yml**



```
[ec2-user@ip-172-31-92-38 ~]$ touch namespace.yml
[ec2-user@ip-172-31-92-38 ~]$ ls
kubectl  namespace.yml
[ec2-user@ip-172-31-92-38 ~]$ 
```

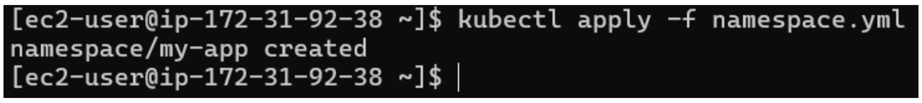
✓ To open and write an code to create an namespace, the command:

- **vi namespace.yml**



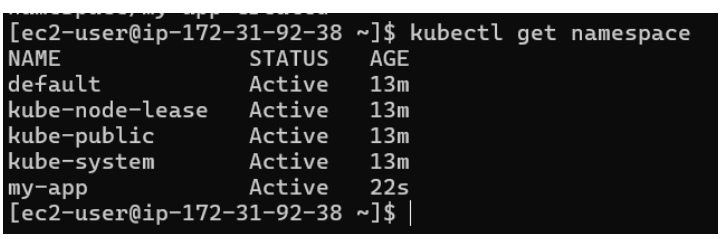
```
apiVersion: v1
kind: Namespace
metadata:
  name: my-app

~
```

- ✓ To apply that yaml file to create an namespace using kubernetes, The command is:
  - **Kubectl apply -f namespace.yml**



```
[ec2-user@ip-172-31-92-38 ~]$ kubectl apply -f namespace.yml
namespace/my-app created
[ec2-user@ip-172-31-92-38 ~]$ |
```

- ✓ Now to check the namespace where created or not using the below command:
  - **kubectl get namespaces**



```
[ec2-user@ip-172-31-92-38 ~]$ kubectl get namespace
NAME              STATUS   AGE
default           Active   13m
kube-node-lease   Active   13m
kube-public       Active   13m
kube-system       Active   13m
my-app            Active   22s
[ec2-user@ip-172-31-92-38 ~]$ |
```

- ✓ Here you can see that where the “my-app” is name of the namespace was created.

************** **TASK COMPLETED** **************