Lab Exercise 9- Managing Namespaces in Kubernetes

Name: Aditya Tomar

Sap: 500106015

Batch: B2

Step 1: Understand Namespaces

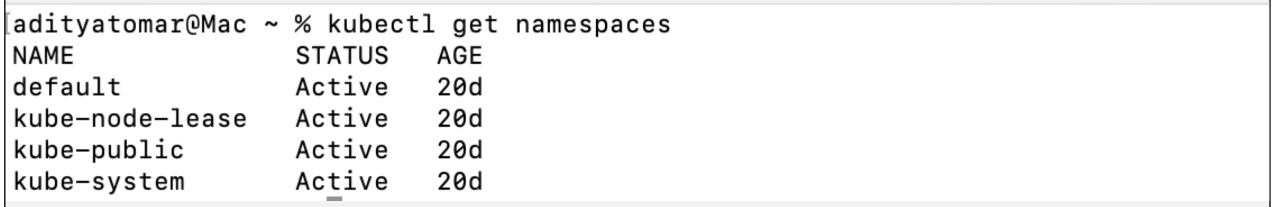
Namespaces provide a mechanism for scoping resources in a cluster. Namespaces can be used to:

- Create environments for different applications or teams.
- Apply policies like resource quotas or network policies on a per-namespace basis.
- Separate operational environments (like development and production).

Step 2: List Existing Namespaces

To list all the namespaces in your Kubernetes cluster:

```
kubectl get namespaces
```



```
adityatomar@Mac ~ % kubectl get namespaces
NAME              STATUS   AGE
default           Active   20d
kube-node-lease   Active   20d
kube-public       Active   20d
kube-system       Active   20d
```

You will typically see default namespaces like default, kube-system, and kube-public.

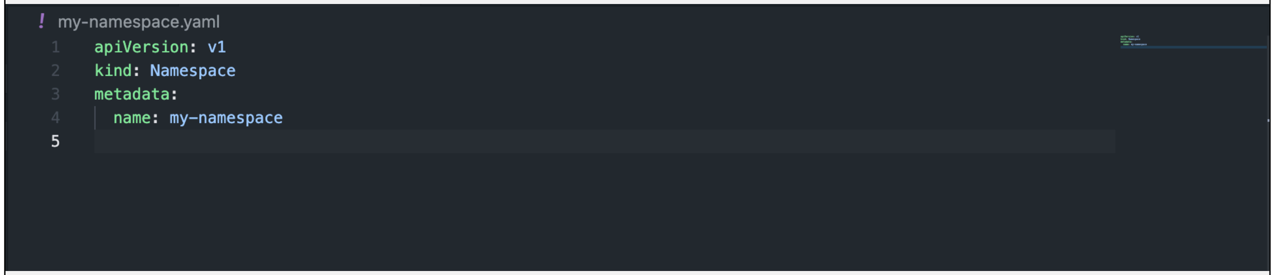
Step 3: Create a Namespace

You can create a namespace using a YAML file or directly with the kubectl command.

Using YAML File

Create a file named ***my-namespace.yaml*** with the following content:

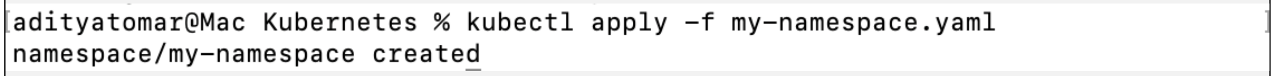
```
apiVersion: v1
kind: Namespace
metadata:
  name: my-namespace
```



```
my-namespace.yaml
apiVersion: v1
kind: Namespace
metadata:
  name: my-namespace
```

Apply this YAML to create the namespace:

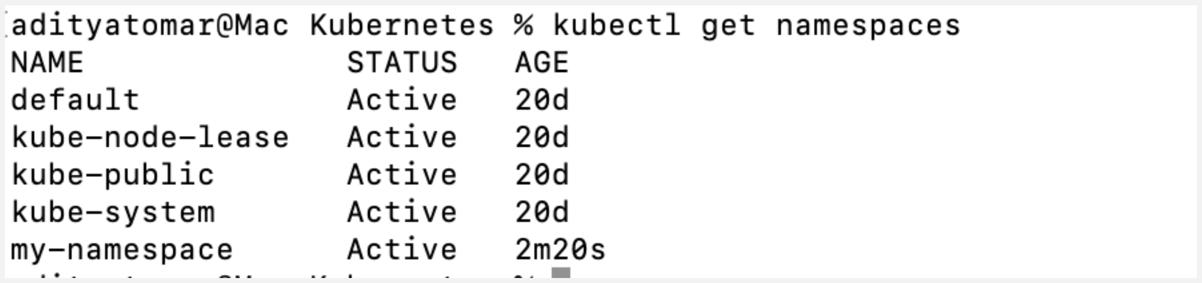
```
kubectl apply -f my-namespace.yaml
```



```
adityatomar@Mac Kubernetes % kubectl apply -f my-namespace.yaml
namespace/my-namespace created
```

Verify that the namespace is created:

```
kubectl get namespaces
```



```
adityatomar@Mac Kubernetes % kubectl get namespaces
NAME              STATUS   AGE
default           Active   20d
kube-node-lease   Active   20d
kube-public       Active   20d
kube-system       Active   20d
my-namespace      Active   2m20s
```

You should see my-namespace listed in the output.

Step 4: Deploy Resources in a Namespace

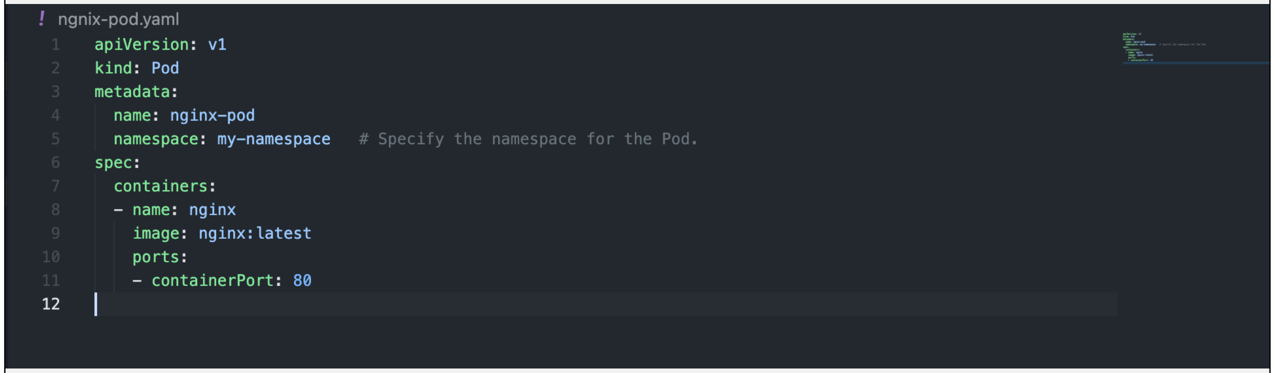
Create resources such as Pods, Services, or Deployments within the new namespace.

Deploy a Pod in the Namespace

Create a YAML file named ***nginx-pod.yaml*** with the following content:

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
  namespace: my-namespace   # Specify the namespace for the Pod.
spec:
```

```
containers:
- name: nginx
  image: nginx:latest
  ports:
  - containerPort: 80
```

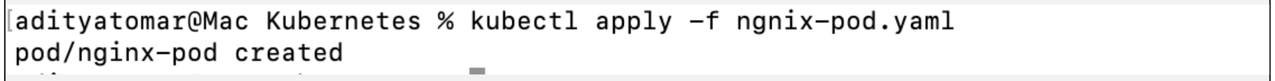


```
ngnix-pod.yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
  namespace: my-namespace   # Specify the namespace for the Pod.
spec:
  containers:
  - name: nginx
    image: nginx:latest
    ports:
    - containerPort: 80

```

Apply this YAML to create the Pod:

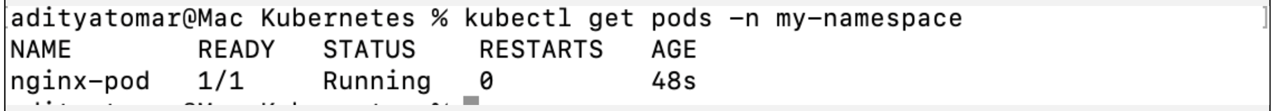
```
kubectl apply -f nginx-pod.yaml
```



```
adityatomar@Mac Kubernetes % kubectl apply -f ngnix-pod.yaml
pod/nginx-pod created
```

Check the status of the Pod within the namespace:

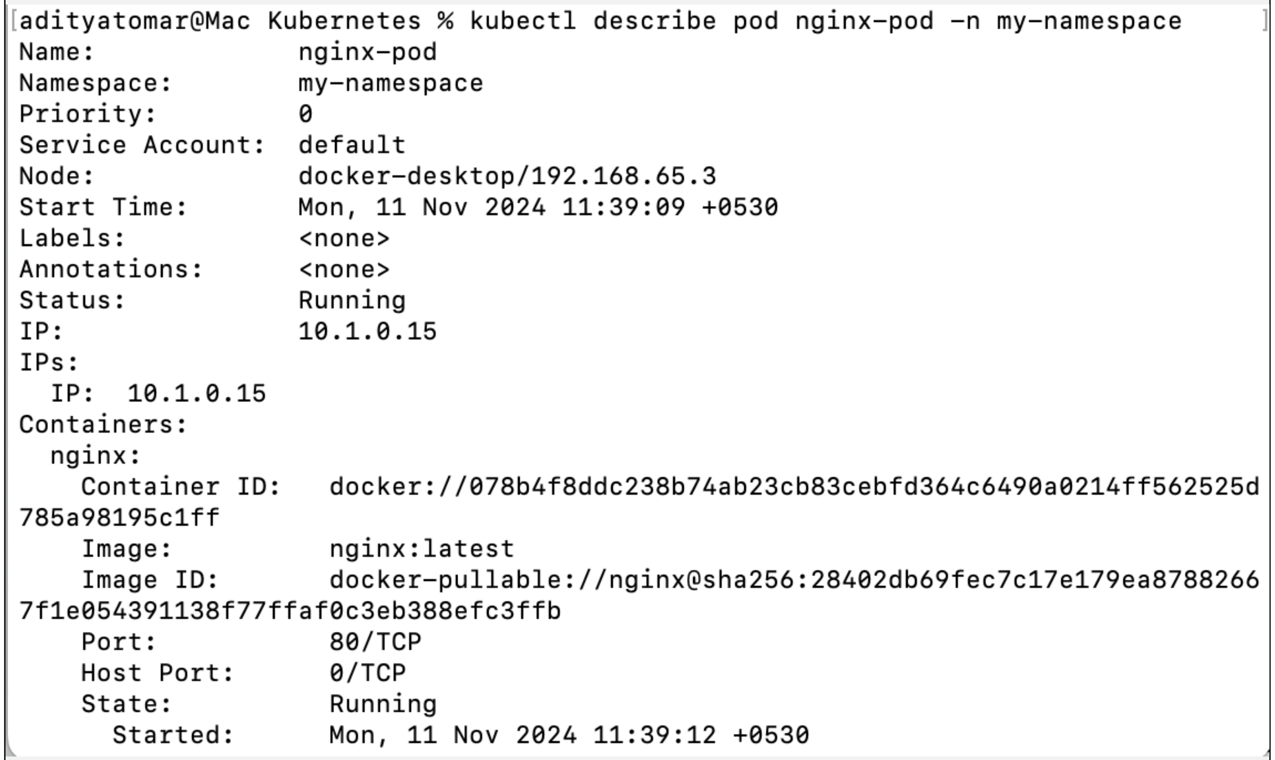
```
kubectl get pods -n my-namespace
```



```
adityatomar@Mac Kubernetes % kubectl get pods -n my-namespace
NAME        READY   STATUS    RESTARTS   AGE
nginx-pod   1/1     Running   0          48s
```

To describe the Pod and see detailed information:

```
kubectl describe pod nginx-pod -n my-namespace
```

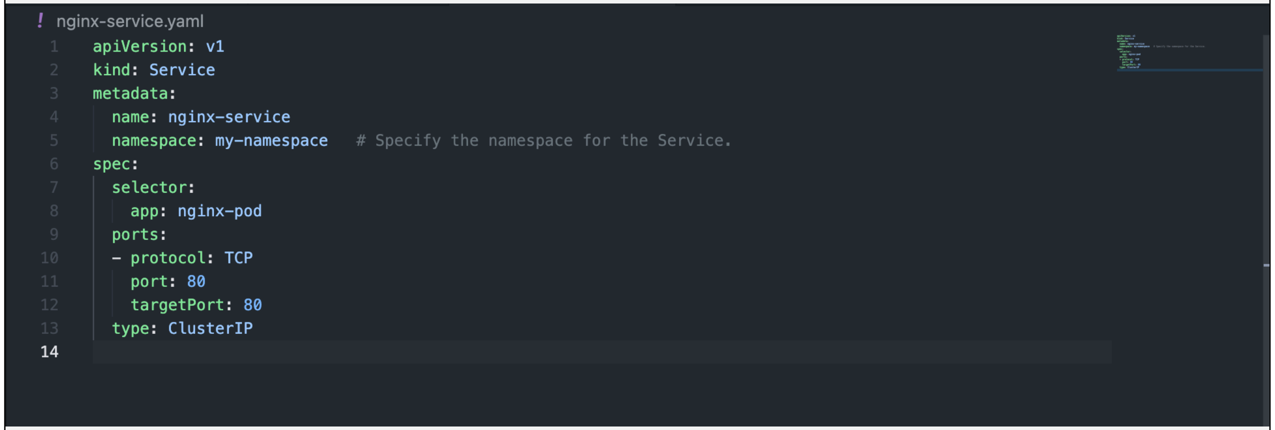


```
adityatomar@Mac Kubernetes % kubectl describe pod nginx-pod -n my-namespace
Name:             nginx-pod
Namespace:        my-namespace
Priority:         0
Service Account:  default
Node:             docker-desktop/192.168.65.3
Start Time:       Mon, 11 Nov 2024 11:39:09 +0530
Labels:           <none>
Annotations:      <none>
Status:           Running
IP:               10.1.0.15
IPs:
  IP:  10.1.0.15
Containers:
  nginx:
    Container ID:   docker://078b4f8ddc238b74ab23cb83cebfd364c6490a0214ff562525d
785a98195c1ff
    Image:          nginx:latest
    Image ID:       docker-pullable://nginx@sha256:28402db69fec7c17e179ea8788266
7f1e054391138f77ffaf0c3eb388efc3ffb
    Port:           80/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Mon, 11 Nov 2024 11:39:12 +0530
```

Create a Service in the Namespace

Create a YAML file named nginx-service.yaml with the following content:

```
apiVersion: v1
kind: Service
metadata:
  name: nginx-service
  namespace: my-namespace   # Specify the namespace for the Service.
spec:
  selector:
    app: nginx-pod
  ports:
  - protocol: TCP
    port: 80
    targetPort: 80
  type: ClusterIP
```

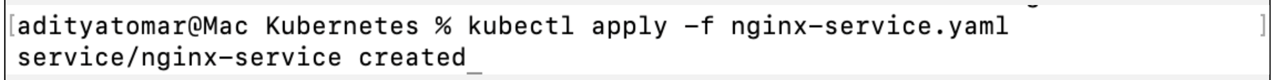


nginx-service.yaml

```
apiVersion: v1
kind: Service
metadata:
  name: nginx-service
  namespace: my-namespace   # Specify the namespace for the Service.
spec:
  selector:
    app: nginx-pod
  ports:
  - protocol: TCP
    port: 80
    targetPort: 80
  type: ClusterIP
```

Apply this YAML to create the Service:

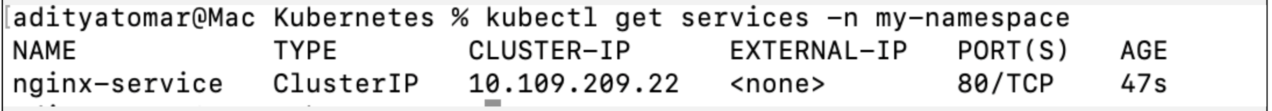
```
kubectl apply -f nginx-service.yaml
```



```
adityatomar@Mac Kubernetes % kubectl apply -f nginx-service.yaml
service/nginx-service created
```

Check the status of the Service within the namespace:

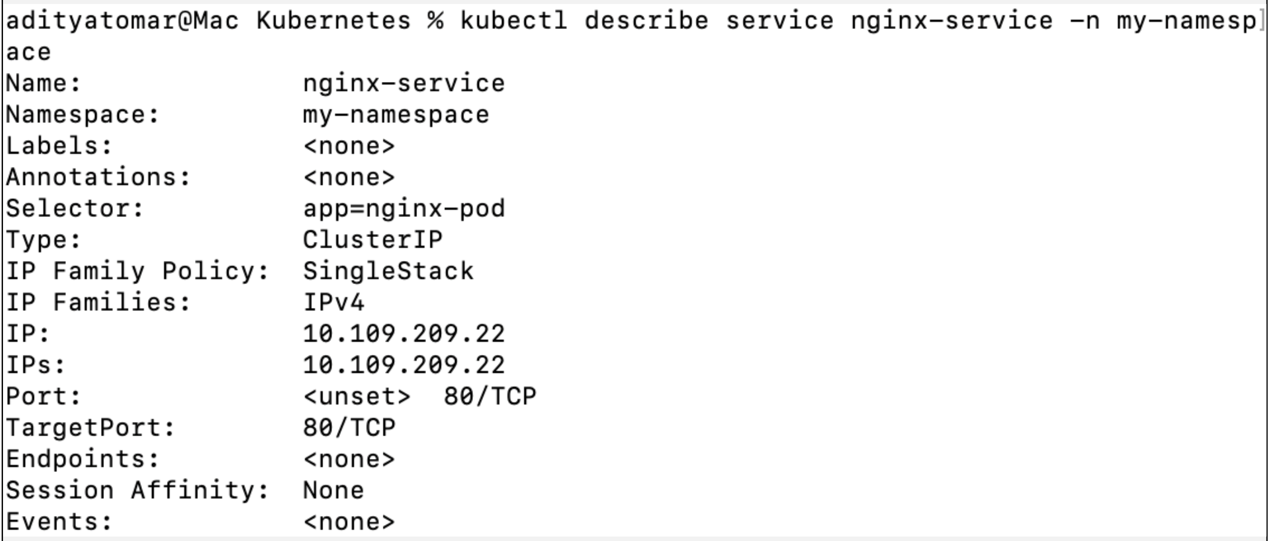
```
kubectl get services -n my-namespace
```



```
adityatomar@Mac Kubernetes % kubectl get services -n my-namespace
NAME            TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)   AGE
nginx-service   ClusterIP   10.109.209.22   <none>        80/TCP    47s
```

To describe the Service and see detailed information:

```
kubectl describe service nginx-service -n my-namespace
```



```
adityatomar@Mac Kubernetes % kubectl describe service nginx-service -n my-namesp
ace
Name:              nginx-service
Namespace:         my-namespace
Labels:            <none>
Annotations:       <none>
Selector:          app=nginx-pod
Type:              ClusterIP
IP Family Policy:  SingleStack
IP Families:       IPv4
IP:                10.109.209.22
IPs:               10.109.209.22
Port:              <unset>  80/TCP
TargetPort:        80/TCP
Endpoints:         <none>
Session Affinity:  None
Events:            <none>
```

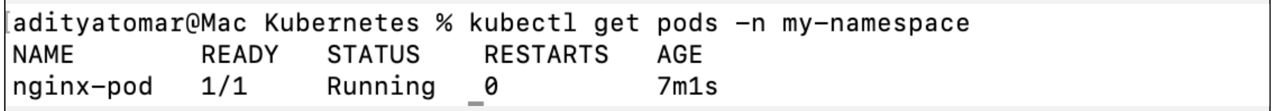
Step 5: Switching Context Between Namespaces

When working with multiple namespaces, you can specify the namespace in kubectl commands or switch the default context.

Specify Namespace in Commands

You can specify the namespace directly in kubectl commands using the -n or --namespace flag:

```
kubectl get pods -n my-namespace
```

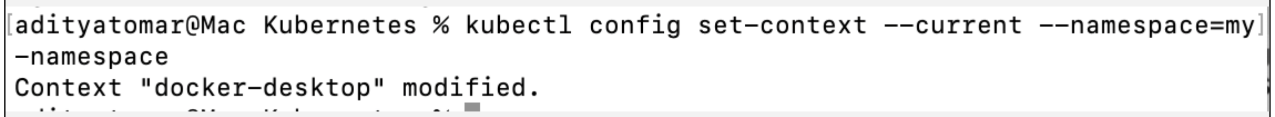


```
adityatomar@Mac Kubernetes % kubectl get pods -n my-namespace
NAME        READY   STATUS    RESTARTS   AGE
nginx-pod   1/1     Running   0          7m1s
```

Set Default Namespace for kubectl Commands

To avoid specifying the namespace every time, you can set the default namespace for the current context:

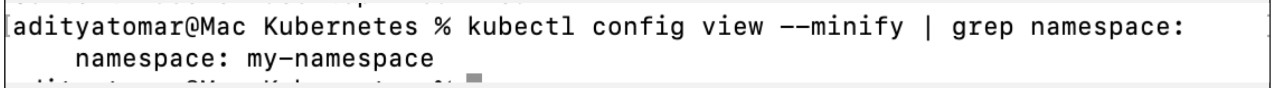
```
kubectl config set-context --current —namespace=my-namespace
```



```
adityatomar@Mac Kubernetes % kubectl config set-context --current --namespace=my
-namespace
Context "docker-desktop" modified.
```

Verify the current context's namespace:

```
kubectl config view --minify | grep namespace:
```

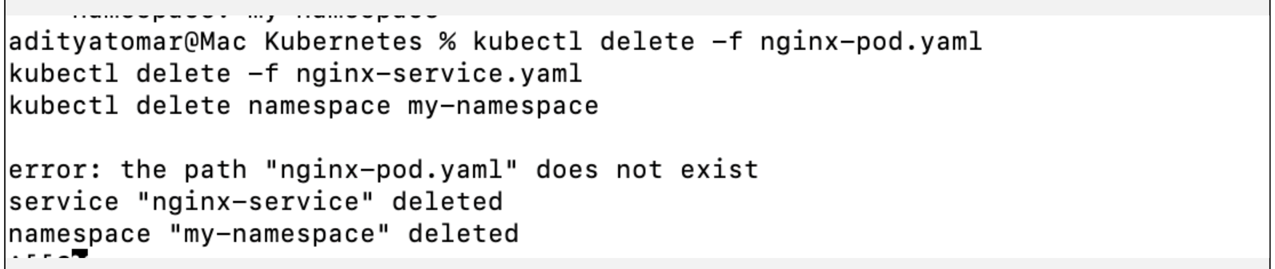


```
adityatomar@Mac Kubernetes % kubectl config view --minify | grep namespace:
    namespace: my-namespace
```

Step 6: Clean Up Resources

To delete the resources and the namespace you created:

```
kubectl delete -f nginx-pod.yaml
kubectl delete -f nginx-service.yaml
kubectl delete namespace my-namespace
```

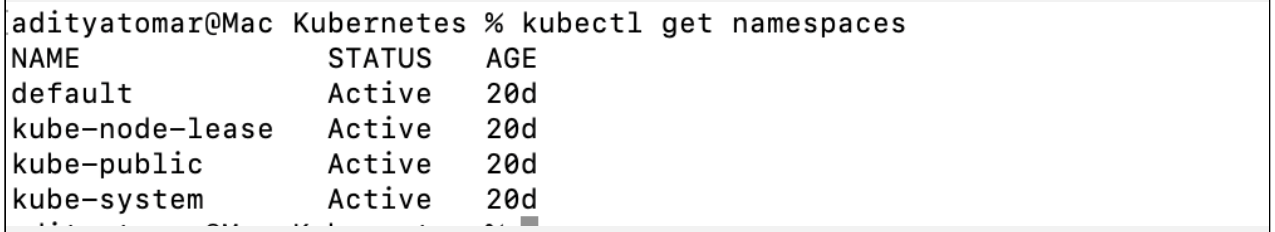


```
adityatomar@Mac Kubernetes % kubectl delete -f nginx-pod.yaml
kubectl delete -f nginx-service.yaml
kubectl delete namespace my-namespace

error: the path "nginx-pod.yaml" does not exist
service "nginx-service" deleted
namespace "my-namespace" deleted
```

Ensure that the namespace and all its resources are deleted:

```
kubectl get namespaces
```



```
adityatomar@Mac Kubernetes % kubectl get namespaces
NAME              STATUS   AGE
default           Active   20d
kube-node-lease   Active   20d
kube-public       Active   20d
kube-system       Active   20d
```