Lab Exercise 13- Managing Namespaces in Kubernetes

Step 1: Understand Namespaces

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

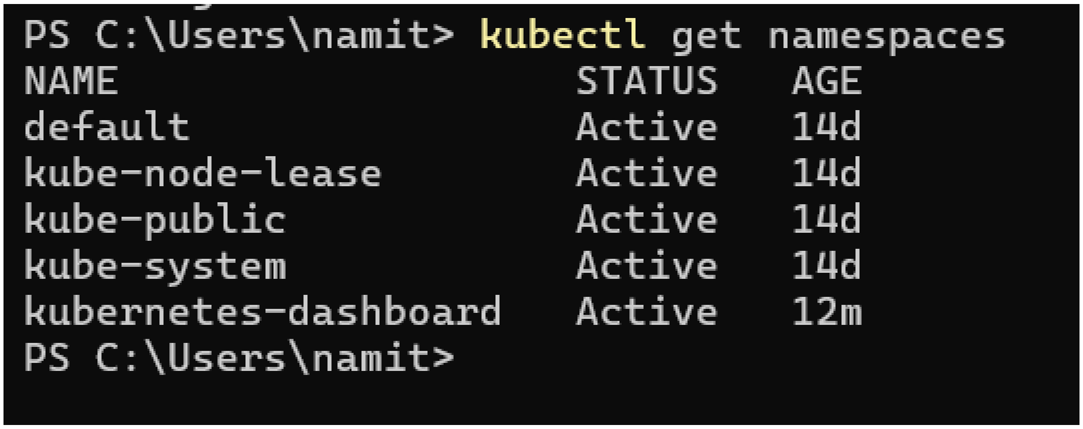
- Create environments for different applications or teams.
- Apply policies like resource quotas or network policies on a per-namespace basis.
- Separate operational environments (like development and production).

Step 2: List Existing Namespaces

To list all the namespaces in your Kubernetes cluster:

```
kubectl get namespaces
```

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



```
PS C:\Users\namit> kubectl get namespaces
NAME                   STATUS   AGE
default                Active   14d
kube-node-lease        Active   14d
kube-public            Active   14d
kube-system            Active   14d
kubernetes-dashboard   Active   12m
PS C:\Users\namit>
```

Step 3: Create a Namespace

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

Using YAML File

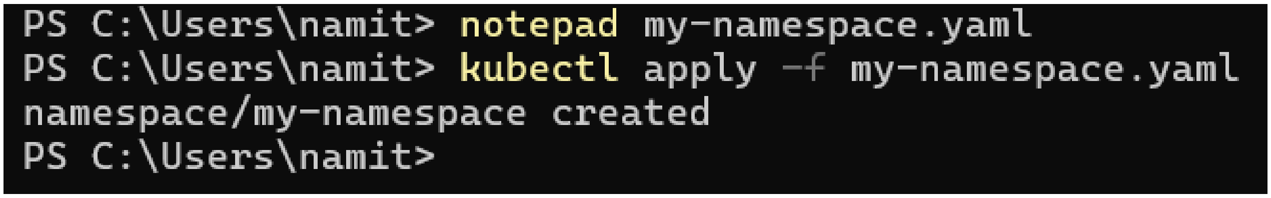
Create a file named my-namespace.yaml with the following content:

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

Apply this YAML to create the namespace:

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

Using kubectl Command



```
PS C:\Users\namit> notepad my-namespace.yaml
PS C:\Users\namit> kubectl apply -f my-namespace.yaml
namespace/my-namespace created
PS C:\Users\namit>
```

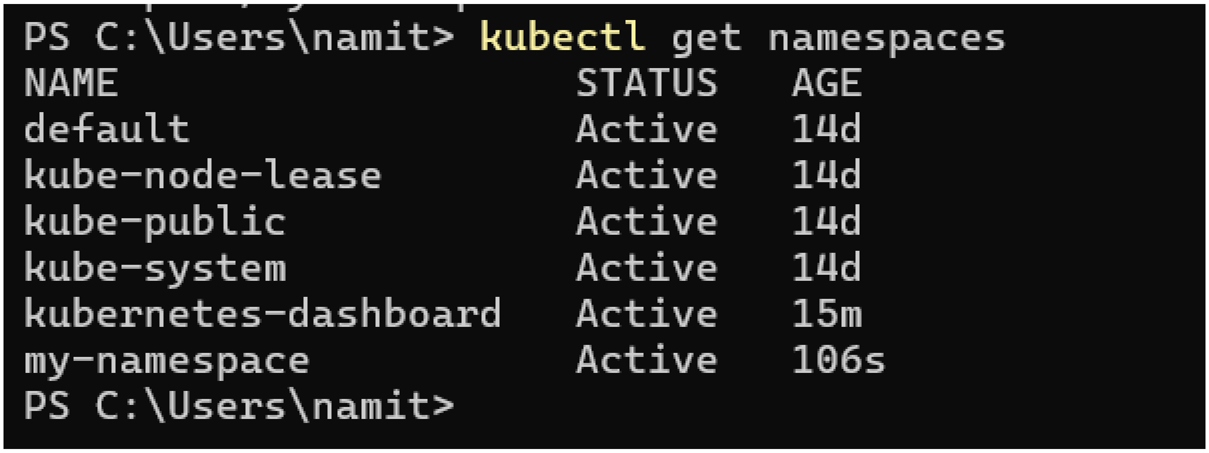
Alternatively, create a namespace using the kubectl command:

```
kubectl create namespace my-namespace
```

Verify that the namespace is created:

```
kubectl get namespaces
```

You should see my-namespace listed in the output.



```
PS C:\Users\namit> kubectl get namespaces
NAME                   STATUS   AGE
default                Active   14d
kube-node-lease        Active   14d
kube-public            Active   14d
kube-system            Active   14d
kubernetes-dashboard   Active   15m
my-namespace           Active   106s
PS C:\Users\namit>
```

Step 4: Deploy Resources in a Namespace

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

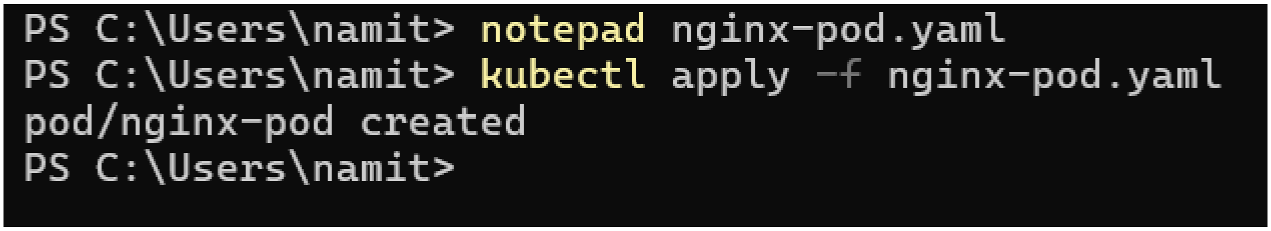
Deploy a Pod in the Namespace

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

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
  namespace: my-namespace
spec:
  containers:
  - name: nginx
    image: nginx:latest
    ports:
    - containerPort: 80
```

Apply this YAML to create the Pod:

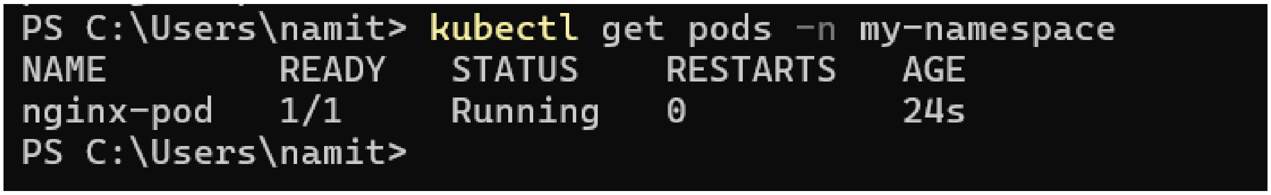
```
kubectl apply -f nginx-pod.yaml
```



```
PS C:\Users\namit> notepad nginx-pod.yaml
PS C:\Users\namit> kubectl apply -f nginx-pod.yaml
pod/nginx-pod created
PS C:\Users\namit>
```

Check the status of the Pod within the namespace:

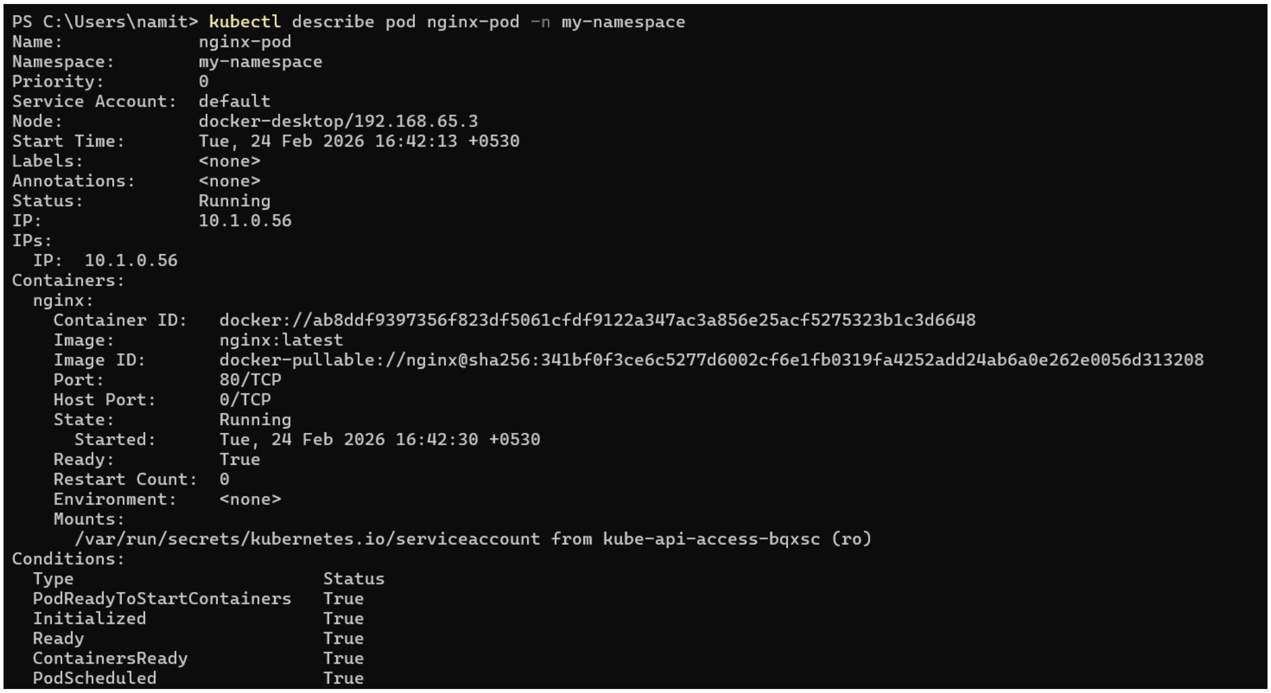
```
kubectl get pods -n my-namespace
```



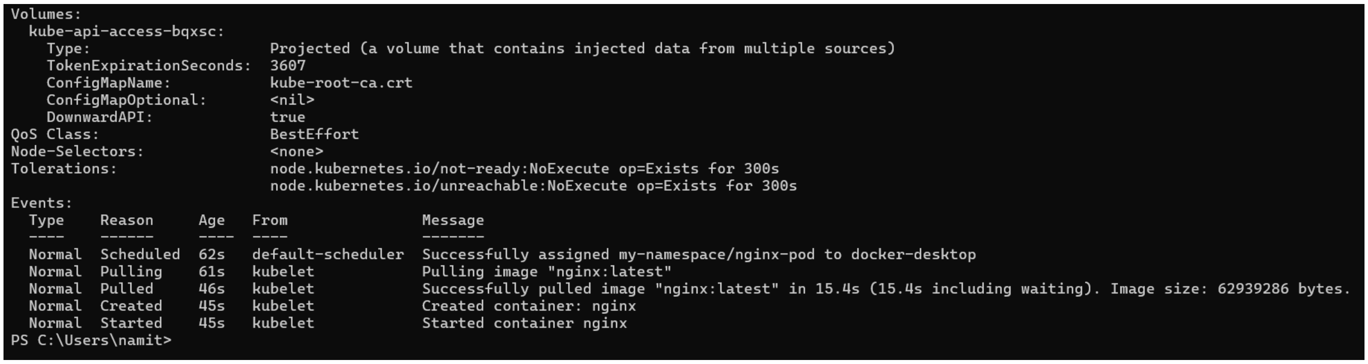
```
PS C:\Users\namit> kubectl get pods -n my-namespace
NAME        READY   STATUS    RESTARTS   AGE
nginx-pod   1/1     Running   0          24s
PS C:\Users\namit>
```

To describe the Pod and see detailed information:

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



```
PS C:\Users\namit> 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:       Tue, 24 Feb 2026 16:42:13 +0530
Labels:           <none>
Annotations:      <none>
Status:           Running
IP:               10.1.0.56
IPs:
  IP:  10.1.0.56
Containers:
  nginx:
    Container ID:   docker://ab8ddf9397356f823df5061cfdf9122a347ac3a856e25acf5275323b1c3d6648
    Image:          nginx:latest
    Image ID:       docker-pullable://nginx@sha256:341bf0f3ce6c5277d6002cf6e1fb0319fa4252add24ab6a0e262e0056d313208
    Port:           80/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Tue, 24 Feb 2026 16:42:30 +0530
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-bqxsc (ro)
Conditions:
  Type                        Status
  PodReadyToStartContainers   True
  Initialized                 True
  Ready                       True
  ContainersReady             True
  PodScheduled                True
```



```
Volumes:
  kube-api-access-bqxsc:
    Type:                    Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds:  3607
    ConfigMapName:           kube-root-ca.crt
    ConfigMapOptional:       <nil>
    DownwardAPI:             true
QoS Class:                   BestEffort
Node-Selectors:              <none>
Tolerations:                 node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                             node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:
  Type    Reason     Age   From               Message
  ----    ------     ----  ----               -------
  Normal  Scheduled  62s   default-scheduler  Successfully assigned my-namespace/nginx-pod to docker-desktop
  Normal  Pulling    61s   kubelet            Pulling image "nginx:latest"
  Normal  Pulled     46s   kubelet            Successfully pulled image "nginx:latest" in 15.4s (15.4s including waiting). Image size: 62939286 bytes.
  Normal  Created    45s   kubelet            Created container: nginx
  Normal  Started    45s   kubelet            Started container nginx
PS C:\Users\namit>
```

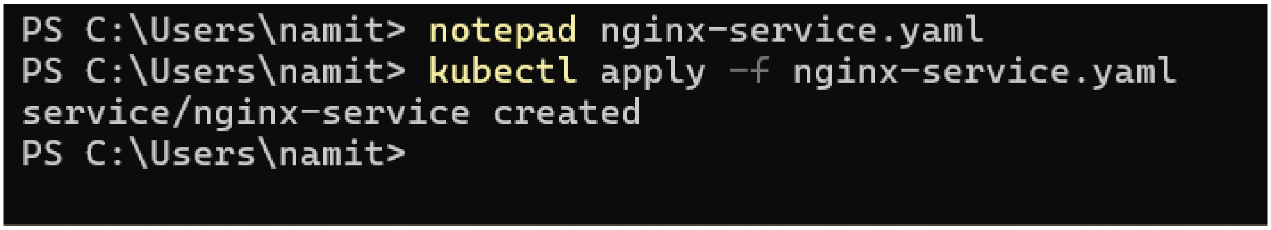
Create a Service in the Namespace

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

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

Apply this YAML to create the Service:

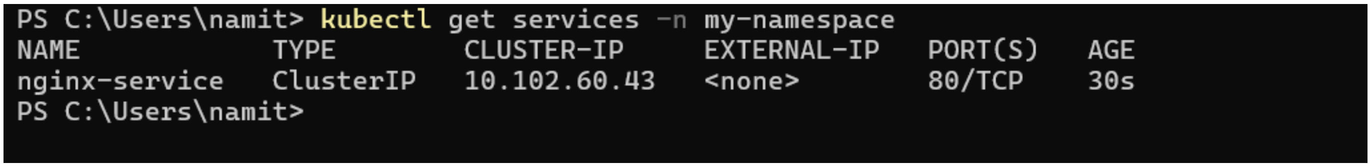
```
kubectl apply -f nginx-service.yaml
```



```
PS C:\Users\namit> notepad nginx-service.yaml
PS C:\Users\namit> kubectl apply -f nginx-service.yaml
service/nginx-service created
PS C:\Users\namit>
```

Check the status of the Service within the namespace:

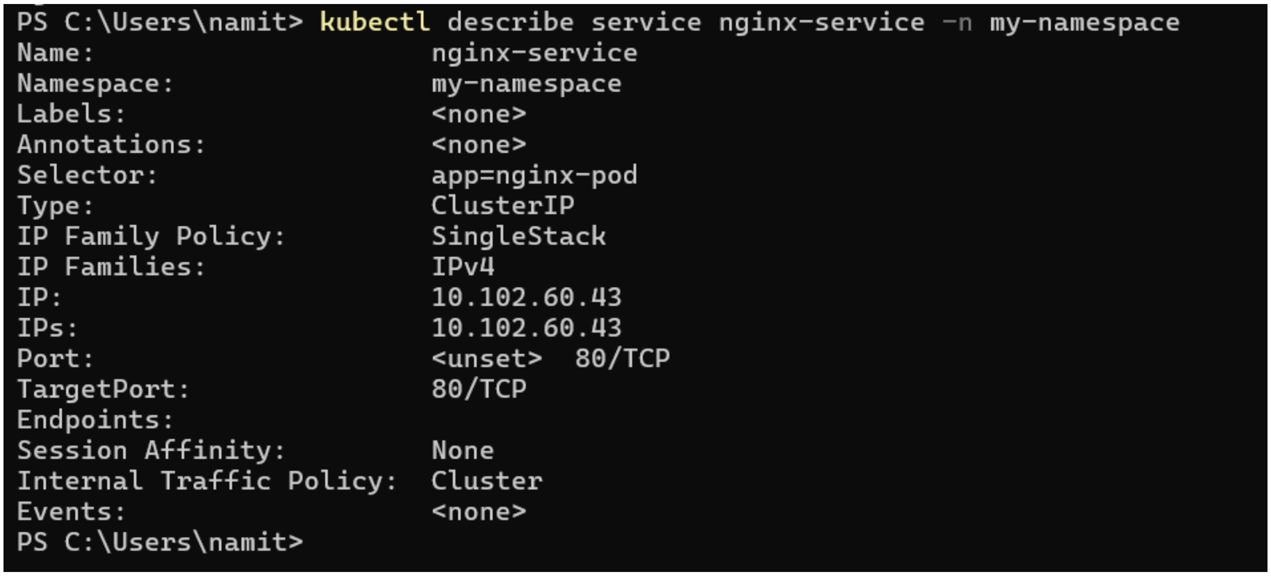
```
kubectl get services -n my-namespace
```



```
PS C:\Users\namit> kubectl get services -n my-namespace
NAME            TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)   AGE
nginx-service   ClusterIP   10.102.60.43   <none>        80/TCP    30s
PS C:\Users\namit>
```

To describe the Service and see detailed information:

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



```
PS C:\Users\namit> kubectl describe service nginx-service -n my-namespace
Name:                     nginx-service
Namespace:                my-namespace
Labels:                   <none>
Annotations:              <none>
Selector:                 app=nginx-pod
Type:                     ClusterIP
IP Family Policy:         SingleStack
IP Families:              IPv4
IP:                       10.102.60.43
IPs:                      10.102.60.43
Port:                     <unset>  80/TCP
TargetPort:               80/TCP
Endpoints:
Session Affinity:         None
Internal Traffic Policy:  Cluster
Events:                   <none>
PS C:\Users\namit>
```

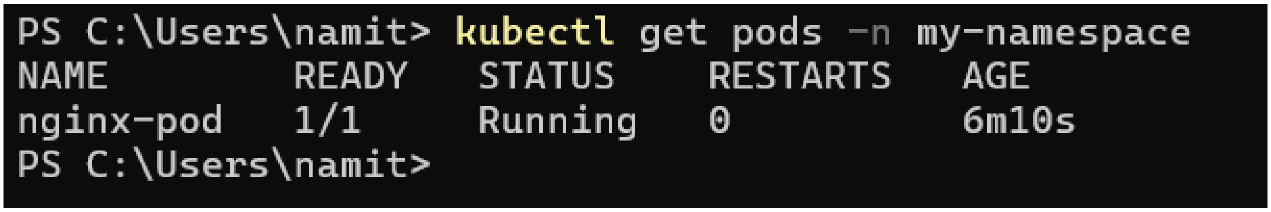
Step 5: Switching Context Between Namespaces

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
```

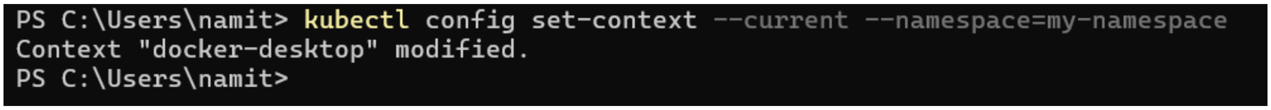


```
PS C:\Users\namit> kubectl get pods -n my-namespace
NAME        READY   STATUS    RESTARTS   AGE
nginx-pod   1/1     Running   0          6m10s
PS C:\Users\namit>
```

Set Default Namespace for kubectl Commands

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

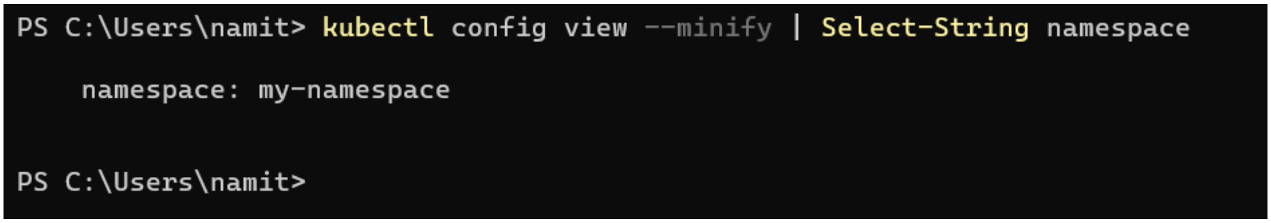
```
kubectl config set-context --current --namespace=my-namespace
```



```
PS C:\Users\namit> kubectl config set-context --current --namespace=my-namespace
Context "docker-desktop" modified.
PS C:\Users\namit>
```

Verify the current context’s namespace:

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

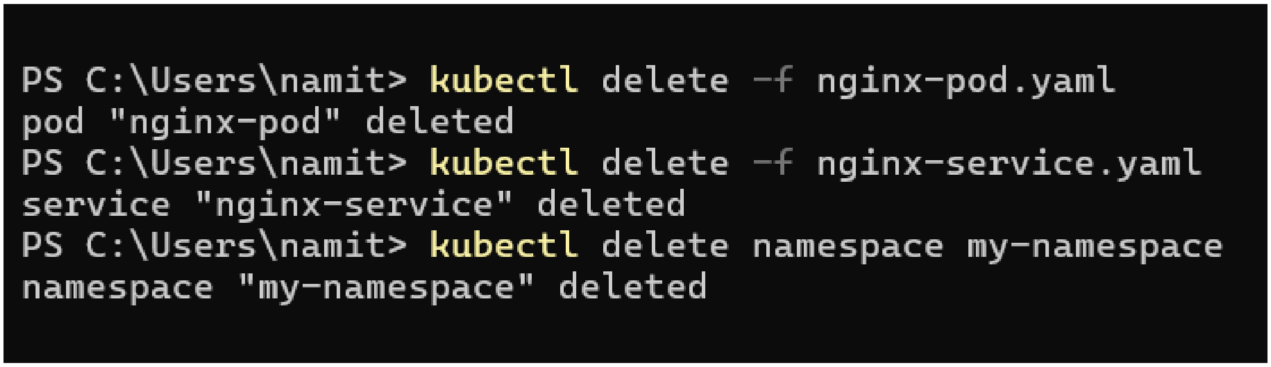


```
PS C:\Users\namit> kubectl config view --minify | Select-String namespace

    namespace: my-namespace


PS C:\Users\namit>
```

Step 6: Clean Up Resources

To delete the resources and the namespace you created:

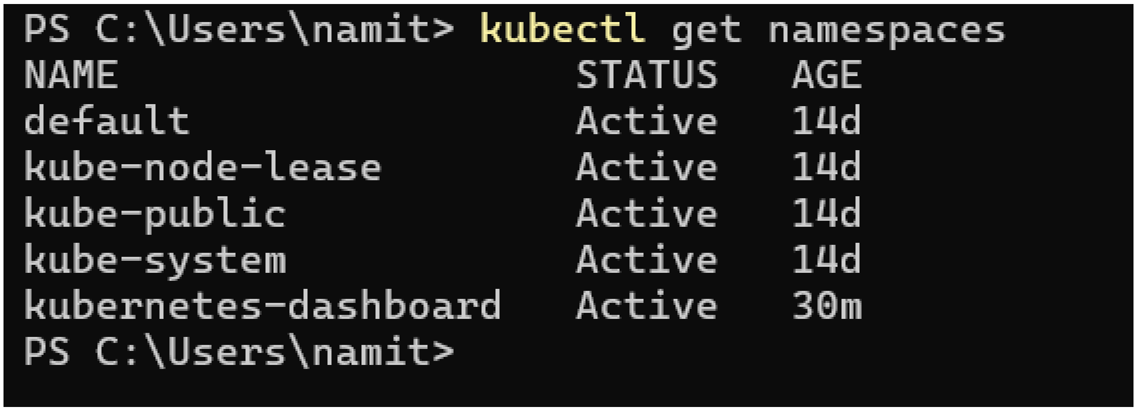
```
kubectl delete -f nginx-pod.yaml
kubectl delete -f nginx-service.yaml
kubectl delete namespace my-namespace
```



```
PS C:\Users\namit> kubectl delete -f nginx-pod.yaml
pod "nginx-pod" deleted
PS C:\Users\namit> kubectl delete -f nginx-service.yaml
service "nginx-service" deleted
PS C:\Users\namit> kubectl delete namespace my-namespace
namespace "my-namespace" deleted
```

Ensure that the namespace and all its resources are deleted:

```
kubectl get namespaces
```



```
PS C:\Users\namit> kubectl get namespaces
NAME                   STATUS   AGE
default                Active   14d
kube-node-lease        Active   14d
kube-public            Active   14d
kube-system            Active   14d
kubernetes-dashboard   Active   30m
PS C:\Users\namit>
```