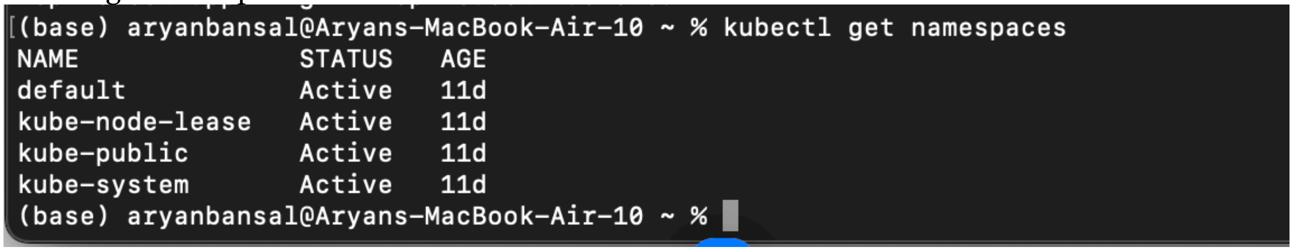
# EXPERIMENT 9

## AIM: 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

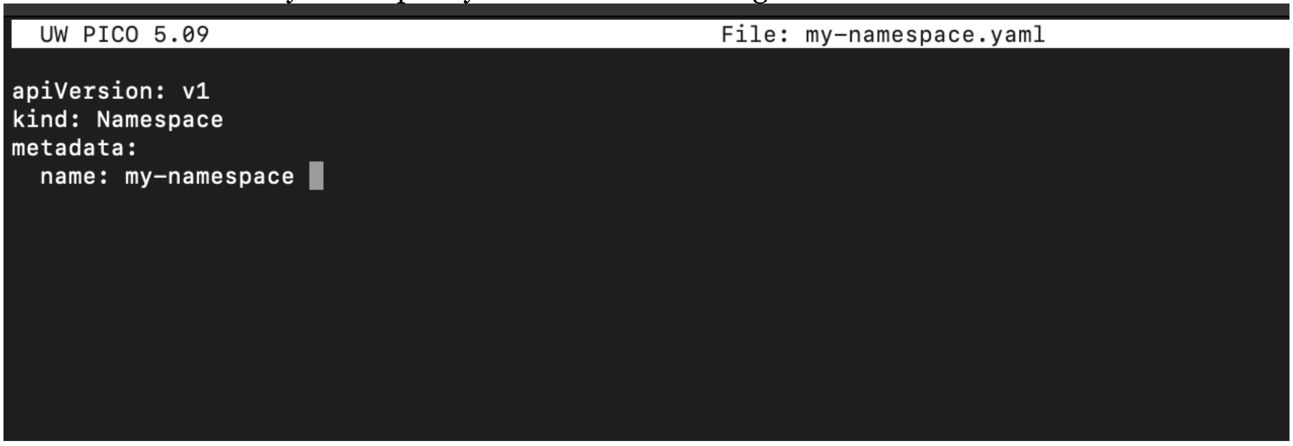


```
(base) aryanbansal@Aryans-MacBook-Air-10 ~ % kubectl get namespaces
NAME              STATUS   AGE
default           Active   11d
kube-node-lease   Active   11d
kube-public       Active   11d
kube-system       Active   11d
(base) aryanbansal@Aryans-MacBook-Air-10 ~ %
```

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

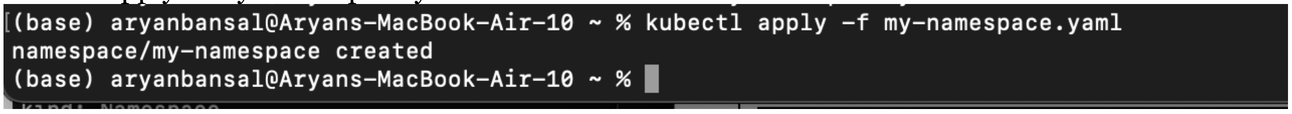
## Step 3: Create a Namespace
You can create a namespace using a YAML file or directly with the kubectl command.

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



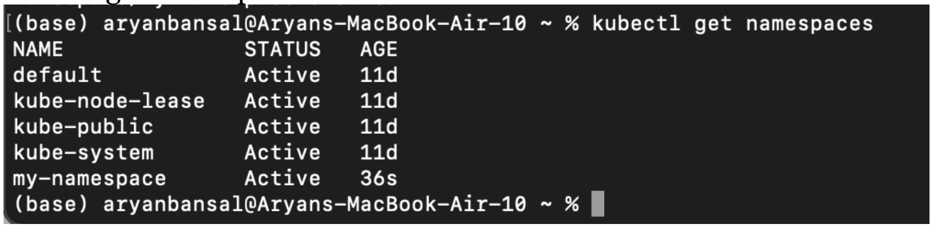
```
  UW PICO 5.09                                   File: my-namespace.yaml

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

Apply this YAML to create the namespace:
kubectl apply -f my-namespace.yaml



```
(base) aryanbansal@Aryans-MacBook-Air-10 ~ % kubectl apply -f my-namespace.yaml
namespace/my-namespace created
(base) aryanbansal@Aryans-MacBook-Air-10 ~ %
```

Verify that the namespace is created:
kubectl get namespaces

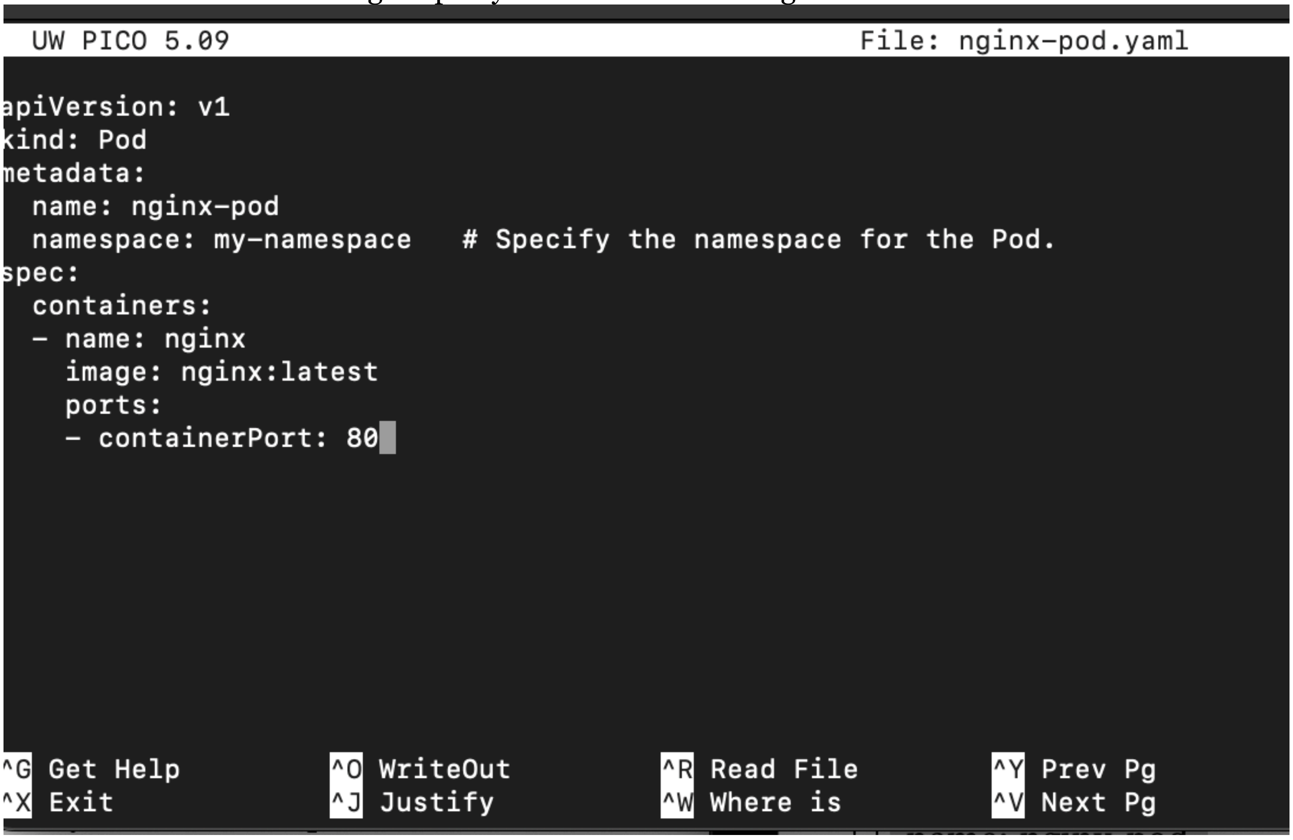


```
(base) aryanbansal@Aryans-MacBook-Air-10 ~ % kubectl get namespaces
NAME              STATUS   AGE
default           Active   11d
kube-node-lease   Active   11d
kube-public       Active   11d
kube-system       Active   11d
my-namespace      Active   36s
(base) aryanbansal@Aryans-MacBook-Air-10 ~ %
```

You should see my-namespace listed in the output.

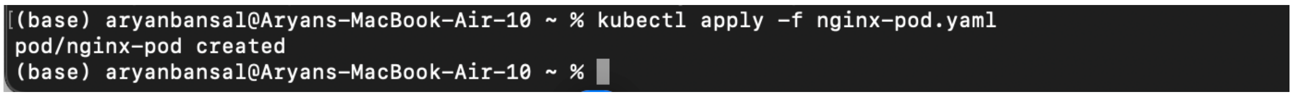
## Step 4: Deploy Resources in a Namespace

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:



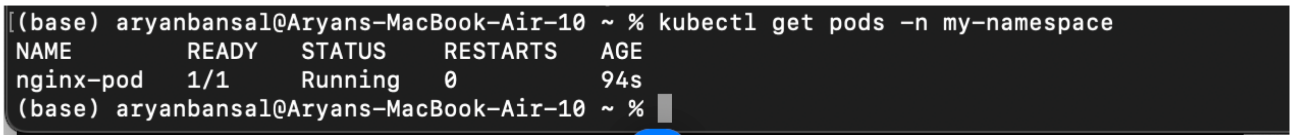
```
  UW PICO 5.09                                           File: nginx-pod.yaml

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

^G Get Help          ^O WriteOut          ^R Read File         ^Y Prev Pg
^X Exit              ^J Justify           ^W Where is          ^V Next Pg
```

Apply this YAML to create the Pod:
kubectl apply -f nginx-pod.yaml



```
(base) aryanbansal@Aryans-MacBook-Air-10 ~ % kubectl apply -f nginx-pod.yaml
pod/nginx-pod created
(base) aryanbansal@Aryans-MacBook-Air-10 ~ %
```

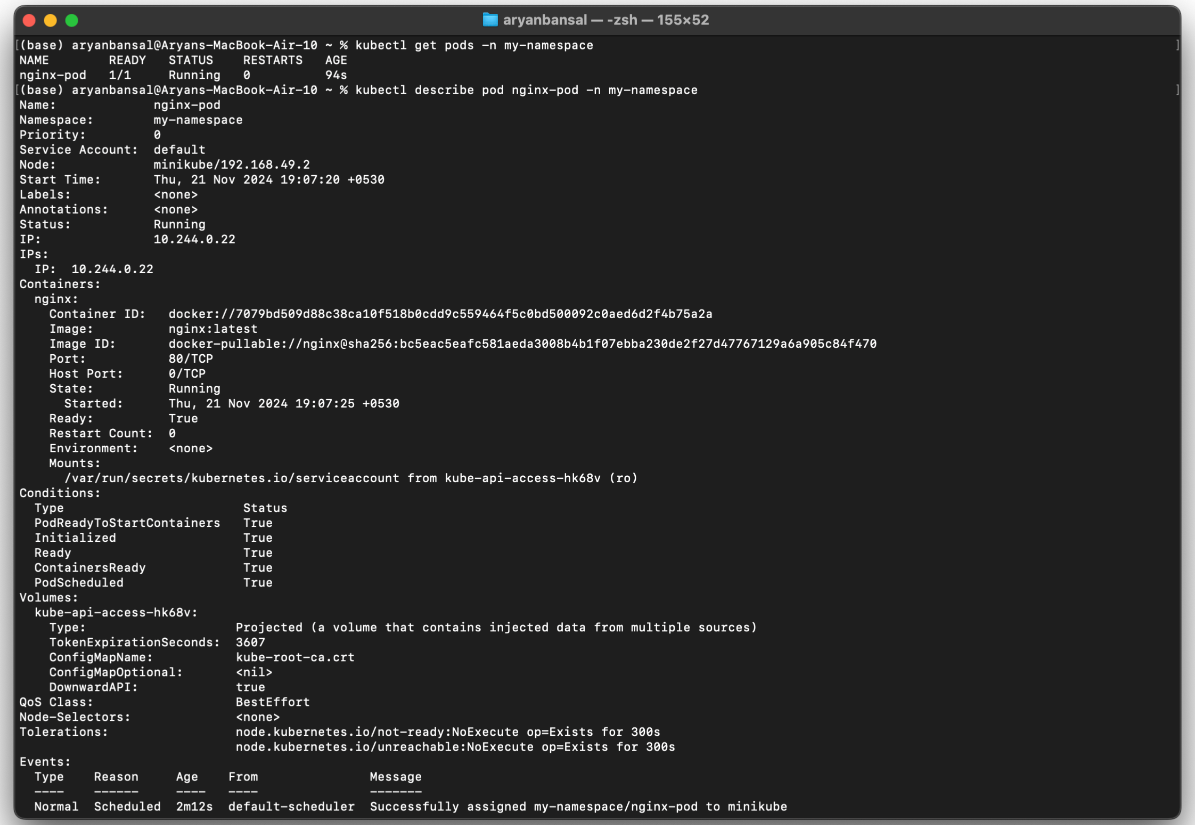
Check the status of the Pod within the namespace:
kubectl get pods -n my-namespace



```
(base) aryanbansal@Aryans-MacBook-Air-10 ~ % kubectl get pods -n my-namespace
NAME        READY   STATUS    RESTARTS   AGE
nginx-pod   1/1     Running   0          94s
(base) aryanbansal@Aryans-MacBook-Air-10 ~ %
```

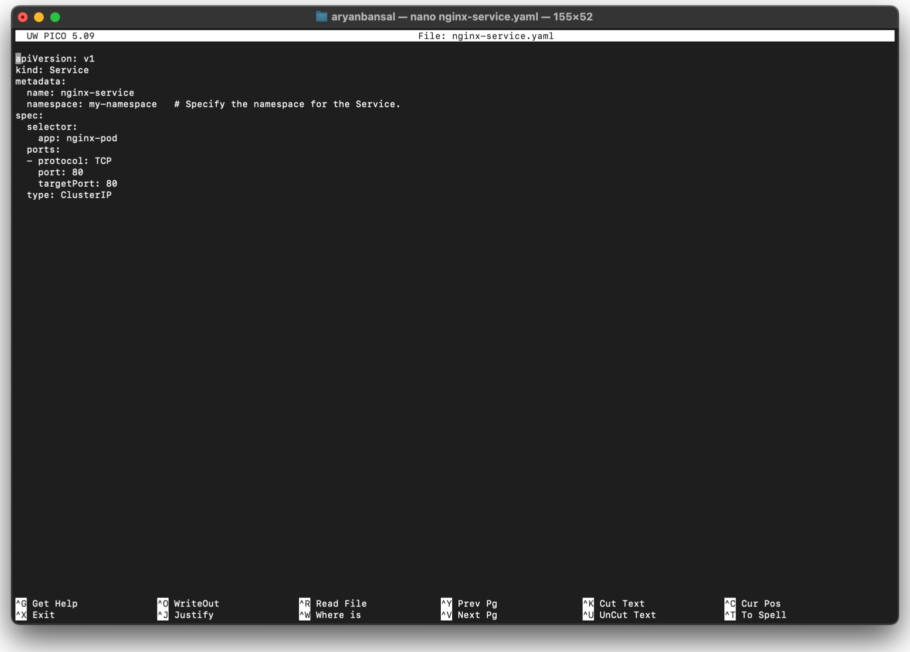
To describe the Pod and see detailed information:

kubectl describe pod nginx-pod -n my-namespace



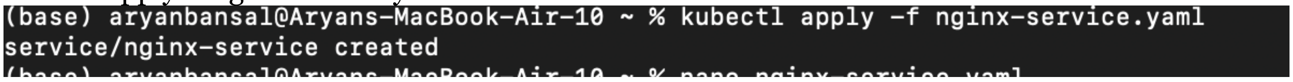
```
(base) aryanbansal@Aryans-MacBook-Air-10 ~ % kubectl get pods -n my-namespace
NAME        READY   STATUS    RESTARTS   AGE
nginx-pod   1/1     Running   0          94s
(base) aryanbansal@Aryans-MacBook-Air-10 ~ % kubectl describe pod nginx-pod -n my-namespace
Name:             nginx-pod
Namespace:        my-namespace
Priority:         0
Service Account:  default
Node:             minikube/192.168.49.2
Start Time:       Thu, 21 Nov 2024 19:07:20 +0530
Labels:           <none>
Annotations:      <none>
Status:           Running
IP:               10.244.0.22
IPs:
  IP:  10.244.0.22
Containers:
  nginx:
    Container ID:   docker://7079bd509d88c38ca10f518b0cdd9c559464f5c0bd500092c0aed6d2f4b75a2a
    Image:          nginx:latest
    Image ID:       docker-pullable://nginx@sha256:bc5eac5eafc581aeda3008b4b1f07ebba230de2f27d47767129a6a905c84f470
    Port:           80/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Thu, 21 Nov 2024 19:07:25 +0530
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-hk68v (ro)
Conditions:
  Type                        Status
  PodReadyToStartContainers   True
  Initialized                 True
  Ready                       True
  ContainersReady             True
  PodScheduled                True
Volumes:
  kube-api-access-hk68v:
    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  2m12s  default-scheduler  Successfully assigned my-namespace/nginx-pod to minikube
```

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



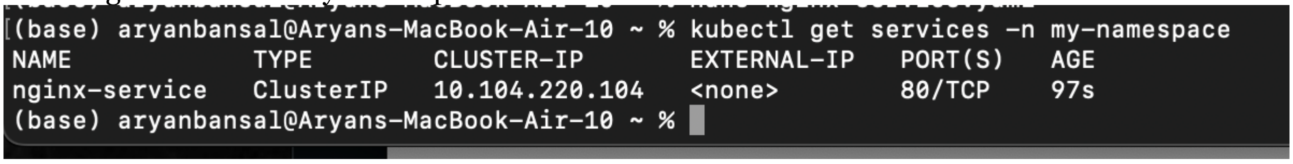
```
apiVersion: v1
kind: Service
metadata:
  name: nginx-service
  namespace: my-namespace   # Specify the namespace for the Service.
spec:
  selector:
    app: nginx-pod
  ports:
  - protocol: TCP
    port: 80
    targetPort: 80
  type: ClusterIP
```

Apply this YAML to create the Service:
kubectl apply -f nginx-service.yaml



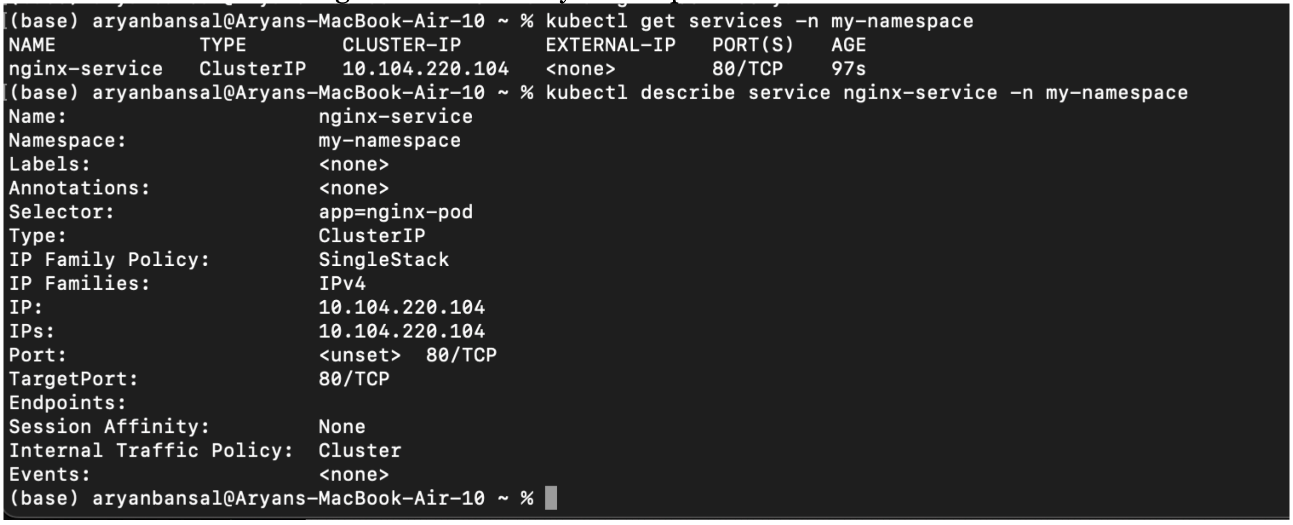
```
(base) aryanbansal@Aryans-MacBook-Air-10 ~ % kubectl apply -f nginx-service.yaml
service/nginx-service created
```

Check the status of the Service within the namespace:
kubectl get services -n my-namespace



```
(base) aryanbansal@Aryans-MacBook-Air-10 ~ % kubectl get services -n my-namespace
NAME            TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)   AGE
nginx-service   ClusterIP   10.104.220.104   <none>        80/TCP    97s
(base) aryanbansal@Aryans-MacBook-Air-10 ~ %
```

To describe the Service and see detailed information:
kubectl describe service nginx-service -n my-namespace



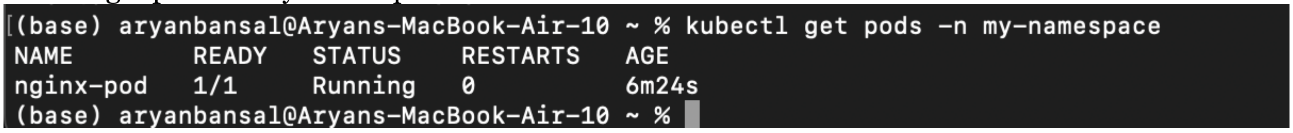
```
(base) aryanbansal@Aryans-MacBook-Air-10 ~ % kubectl get services -n my-namespace
NAME            TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)   AGE
nginx-service   ClusterIP   10.104.220.104   <none>        80/TCP    97s
(base) aryanbansal@Aryans-MacBook-Air-10 ~ % kubectl describe service 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:                       10.104.220.104
IPs:                      10.104.220.104
Port:                     <unset>  80/TCP
TargetPort:               80/TCP
Endpoints:
Session Affinity:         None
Internal Traffic Policy:  Cluster
Events:                   <none>
(base) aryanbansal@Aryans-MacBook-Air-10 ~ %
```

## Step 5: Switching Context Between Namespaces

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

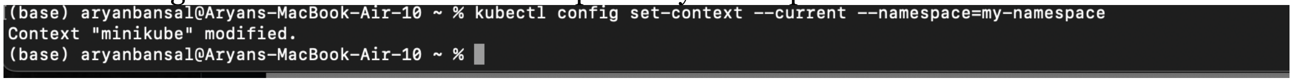
### Specify Namespace in Commands

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



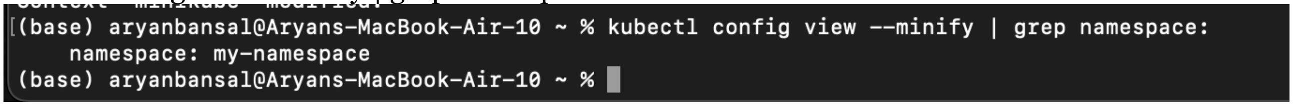
```
(base) aryanbansal@Aryans-MacBook-Air-10 ~ % kubectl get pods -n my-namespace
NAME        READY   STATUS    RESTARTS   AGE
nginx-pod   1/1     Running   0          6m24s
(base) aryanbansal@Aryans-MacBook-Air-10 ~ %
```

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



```
(base) aryanbansal@Aryans-MacBook-Air-10 ~ % kubectl config set-context --current --namespace=my-namespace
Context "minikube" modified.
(base) aryanbansal@Aryans-MacBook-Air-10 ~ %
```

Verify the current context's namespace:
kubectl config view --minify | grep namespace:

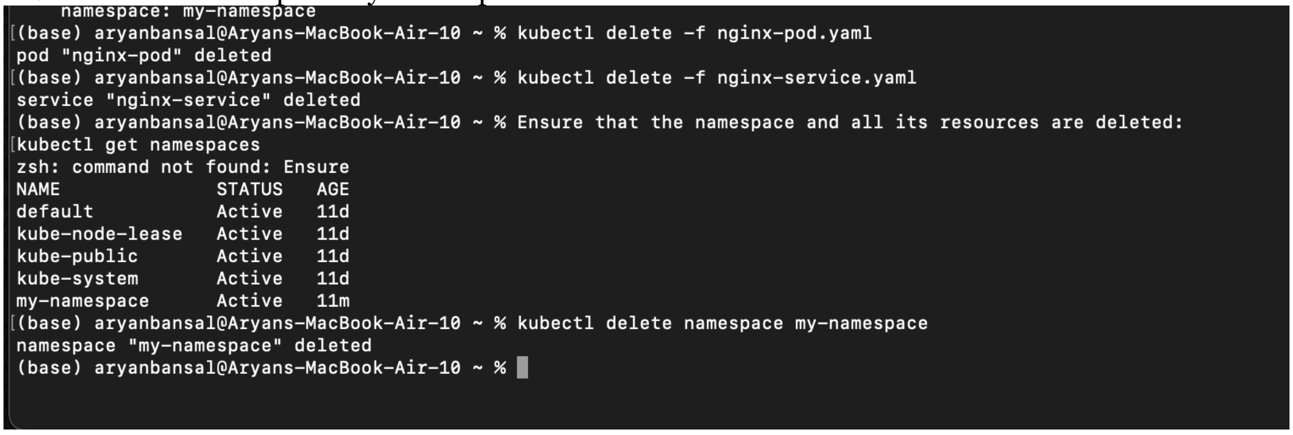


```
(base) aryanbansal@Aryans-MacBook-Air-10 ~ % kubectl config view --minify | grep namespace:
    namespace: my-namespace
(base) aryanbansal@Aryans-MacBook-Air-10 ~ %
```

## 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

Ensure that the namespace and all its resources are deleted:
kubectl delete namespace my-namespace



```
    namespace: my-namespace
[(base) aryanbansal@Aryans-MacBook-Air-10 ~ % kubectl delete -f nginx-pod.yaml
pod "nginx-pod" deleted
[(base) aryanbansal@Aryans-MacBook-Air-10 ~ % kubectl delete -f nginx-service.yaml
service "nginx-service" deleted
(base) aryanbansal@Aryans-MacBook-Air-10 ~ % Ensure that the namespace and all its resources are deleted:
[kubectl get namespaces
zsh: command not found: Ensure
NAME              STATUS   AGE
default           Active   11d
kube-node-lease   Active   11d
kube-public       Active   11d
kube-system       Active   11d
my-namespace      Active   11m
[(base) aryanbansal@Aryans-MacBook-Air-10 ~ % kubectl delete namespace my-namespace
namespace "my-namespace" deleted
(base) aryanbansal@Aryans-MacBook-Air-10 ~ %
```