

Name – Aayush Mishra
SASP ID: 500107141

Lab Exercise 9- 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
```

Terminal

```
PS C:\Users\Slayer> kubectl get namespaces
NAME                STATUS    AGE
default             Active   15d
kube-node-lease     Active   15d
kube-public         Active   15d
kube-system         Active   15d
PS C:\Users\Slayer> 
```

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

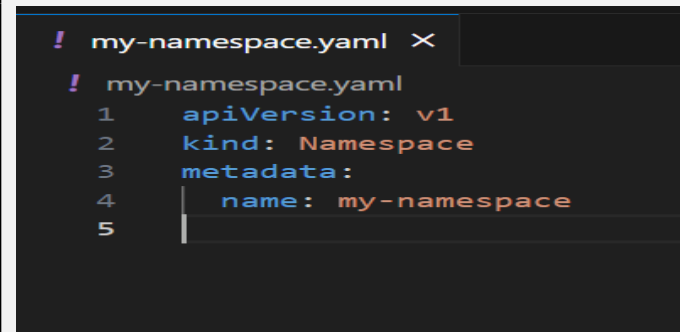
Create a file named ***my-namespace.yaml*** with the following content:

```
apiVersion: v1
```

```
kind: Namespace
```

```
metadata:
```

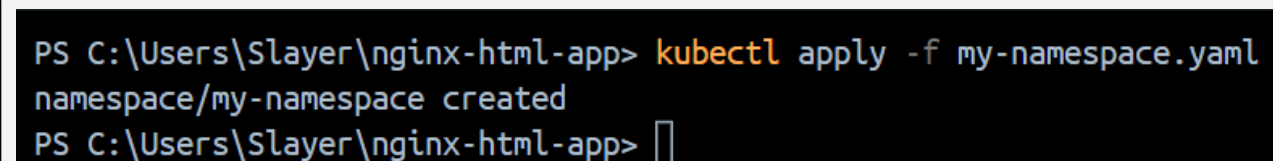
```
  name: my-namespace
```

A screenshot of a code editor window titled 'my-namespace.yaml'. The editor shows the following YAML content:

```
! my-namespace.yaml x
! my-namespace.yaml
1  apiVersion: v1
2  kind: Namespace
3  metadata:
4    name: my-namespace
5
```

Apply this YAML to create the namespace:

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

A screenshot of a terminal window showing the execution of the kubectl apply command. The output is as follows:

```
PS C:\Users\Slayer\nginx-html-app> kubectl apply -f my-namespace.yaml
namespace/my-namespace created
PS C:\Users\Slayer\nginx-html-app> █
```

Verify that the namespace is created:

```
kubectl get namespaces
```

```
PS C:\Users\Slayer\nginx-html-app> kubectl get namespaces
NAME                STATUS    AGE
default             Active    15d
kube-node-lease     Active    15d
kube-public         Active    15d
kube-system         Active    15d
my-namespace        Active    80s
```

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
```

```
PS C:\Users\Slayer\nginx-html-app> code .
```

```
! nginx-pod.yaml
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: nginx-pod
5    namespace: my-namespace # Specify the namespace for the Pod.
6  spec:
7    containers:
8      - name: nginx
9        image: nginx:latest
10       ports:
11         - containerPort: 80
12
```

Apply this YAML to create the Pod:

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

```
PS C:\Users\Slayer\nginx-html-app> kubectl apply -f nginx-pod.yaml
pod/nginx-pod created
```

Check the status of the Pod within the namespace:

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

```
PS C:\Users\Slayer\nginx-html-app> kubectl get pods -n my-namespace
NAME          READY   STATUS    RESTARTS   AGE
nginx-pod     1/1     Running   0           21s
```

To describe the Pod and see detailed information:

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

```
C:\Users\Slayer\nginx-html-app>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:    Sun, 24 Nov 2024 20:03:57 +0530
Labels:        <none>
Annotations:   <none>
Status:        Running
IP:            10.244.0.14
IPs:
  IP: 10.244.0.14
Containers:
  nginx:
    Container ID:  docker://a74e0be5159d1b5bc5388c4fee3defd8b7d16d135cc69f7f41739a2a9bdad941
    Image:         nginx:latest
    Image ID:      docker-pullable://nginx@sha256:bc5eac5eafc581aeda3008b4b1f07ebba230de2f27d47767129a6a905c84f470
    Port:          80/TCP
    Host Port:     0/TCP
    State:         Running
      Started:     Sun, 24 Nov 2024 20:04:12 +0530
      Ready:       True
      Restart Count: 0
    Environment:   <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-w4pfm (ro)
Conditions:
  Type                    Status
  PodReadyToStartContainers  True
  Initialized              True
  Ready                    True
  ContainersReady          True
  PodScheduled             True
Volumes:
  kube-api-access-w4pfm:
    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   3m35s  default-scheduler  Successfully assigned my-namespace/nginx-pod to minikube
  Normal   Pulling    3m34s  kubelet        Pulling image "nginx:latest"
  Normal   Pulled     3m20s  kubelet        Successfully pulled image "nginx:latest" in 13.627s (13.627s including waiting). Image size: 191670156 bytes.
  Normal   Created    3m20s  kubelet        Created container nginx
  Normal   Started    3m20s  kubelet        Started container nginx
```

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
```

```
PS C:\Users\Slayer\nginx-html-app> code .
```

```
! nginx-service.yaml
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: nginx-service
5    namespace: my-namespace # Specify the namespace for the Service.
6  spec:
7    selector:
8      app: nginx-pod
9    ports:
10   - protocol: TCP
11     port: 80
12     targetPort: 80
13   type: ClusterIP
14
```

Apply this YAML to create the Service:

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

```
PS C:\Users\Slayer\nginx-html-app> 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
```

```
PS C:\Users\Slayer\nginx-html-app> kubectl get services -n my-namespace
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
nginx-service	ClusterIP	10.102.110.205	<none>	80/TCP	21s

To describe the Service and see detailed information:

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

```
C:\Users\Slayer\nginx-html-app>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:            10.102.110.205
IPs:           10.102.110.205
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
```

```
PS C:\Users\Slayer\nginx-html-app> kubectl get pods -n my-namespace
```

NAME	READY	STATUS	RESTARTS	AGE
nginx-pod	1/1	Running	0	8m29s

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\Slayer\nginx-html-app> kubectl config set-context --current --namespace=my-namespace
Context "minikube" modified.
PS C:\Users\Slayer\nginx-html-app> █
```

Verify the current context's namespace:

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

```
PS C:\Users\Slayer\nginx-html-app> kubectl config view --minify | findstr namespace
>>
    namespace: my-namespace
PS C:\Users\Slayer\nginx-html-app>
```

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

Ensure that the namespace and all its resources are deleted:

kubectl get namespaces

```
PS C:\Users\Slayer\nginx-html-app> kubectl get namespaces
NAME                STATUS    AGE
default             Active   15d
kube-node-lease     Active   15d
kube-public         Active   15d
kube-system         Active   15d
PS C:\Users\Slayer\nginx-html-app> █
```