Lab Exercise 9- Managing Namespaces in Kubernetes

Name : Ankur Sharma
Sap id : 500110541
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

```
C:\Users\Dell>kubectl get namespaces
E1111 11:19:54.600118    5220 memcache.go:265] couldn't get current server API group list: Get "https://kubernetes.docker.internal:64
43/api?timeout=32s": dial tcp 127.0.0.1:6443: connectex: No connection could be made because the target machine actively refused it.
E1111 11:19:54.601802    5220 memcache.go:265] couldn't get current server API group list: Get "https://kubernetes.docker.internal:64
43/api?timeout=32s": dial tcp 127.0.0.1:6443: connectex: No connection could be made because the target machine actively refused it.
E1111 11:19:54.602464    5220 memcache.go:265] couldn't get current server API group list: Get "https://kubernetes.docker.internal:64
43/api?timeout=32s": dial tcp 127.0.0.1:6443: connectex: No connection could be made because the target machine actively refused it.
E1111 11:19:54.603520    5220 memcache.go:265] couldn't get current server API group list: Get "https://kubernetes.docker.internal:64
43/api?timeout=32s": dial tcp 127.0.0.1:6443: connectex: No connection could be made because the target machine actively refused it.
E1111 11:19:54.605510    5220 memcache.go:265] couldn't get current server API group list: Get "https://kubernetes.docker.internal:64
43/api?timeout=32s": dial tcp 127.0.0.1:6443: connectex: No connection could be made because the target machine actively refused it.
Unable to connect to the server: dial tcp 127.0.0.1:6443: connectex: No connection could be made because the target machine actively
refused it.
```

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
1   apiVersion: v1
2
3   kind: Namespace
4
5   metadata:
6
7     name: my-namespace
```

Apply this YAML to create the namespace:

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

```
PS C:\Users\Dell\OneDrive\Documents\kubernetes> kubectl apply -f my-namespace.yaml
namespace/my-namespace created
```

Verify that the namespace is created:

```
kubectl get namespaces
```

```
PS C:\Users\Dell\OneDrive\Documents\kubernetes> kubectl get namespaces
NAME              STATUS   AGE
default           Active   21d
kube-node-lease   Active   21d
kube-public       Active   21d
kube-system       Active   21d
my-namespace      Active   42s
```

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

```yaml
! 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

```
PS C:\Users\Dell\OneDrive\Documents\kubernetes> 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\Dell\OneDrive\Documents\kubernetes> kubectl get pods -n my-namespace
NAME        READY   STATUS    RESTARTS   AGE
nginx-pod   1/1     Running   0          41s
```

To describe the Pod and see detailed information:

kubectl describe pod nginx-pod -n my-namespace

```
PS C:\Users\Dell\OneDrive\Documents\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 11:36:31 +0530
Labels:           <none>
Annotations:      <none>
Status:           Running
IP:               10.1.0.17
IPs:
  IP:  10.1.0.17
Containers:
  nginx:
    Container ID:   docker://5de1c14e4a57c91e4d0153e95eefee17b6adb4e8f68ffcb97e34739e669c368d
    Image:          nginx:latest
    Image ID:       docker-pullable://nginx@sha256:28402db69fec7c17e179ea87882667f1e054391138f77ffaf0c3eb388efc3ffb
    Port:           80/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Mon, 11 Nov 2024 11:36:36 +0530
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-67klx (ro)
Conditions:
  Type                        Status
  PodReadyToStartContainers   True
  Initialized                 True
  Ready                       True
  ContainersReady             True
```

Create a Service in the Namespace

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

apiVersion: v1

```yaml
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

```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

```
PS C:\Users\Dell\OneDrive\Documents\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

```
PS C:\Users\Dell\OneDrive\Documents\kubernetes> kubectl get services -n my-namespace
NAME            TYPE        CLUSTER-IP    EXTERNAL-IP   PORT(S)   AGE
nginx-service   ClusterIP   10.96.70.70   <none>        80/TCP    39s
```

To describe the Service and see detailed information:

kubectl describe service nginx-service -n my-namespace

```
PS C:\Users\Dell\OneDrive\Documents\kubernetes> 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:                10.96.70.70
IPs:               10.96.70.70
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\Dell\OneDrive\Documents\kubernetes> kubectl get pods -n my-namespace
NAME        READY   STATUS    RESTARTS   AGE
nginx-pod   1/1     Running   0          8m53s
```

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\Dell\OneDrive\Documents\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:
```

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\Dell\OneDrive\Documents\kubernetes> kubectl delete -f nginx-pod.yaml
pod "nginx-pod" deleted
PS C:\Users\Dell\OneDrive\Documents\kubernetes>
PS C:\Users\Dell\OneDrive\Documents\kubernetes> kubectl delete -f nginx-service.yaml
service "nginx-service" deleted
PS C:\Users\Dell\OneDrive\Documents\kubernetes>
PS C:\Users\Dell\OneDrive\Documents\kubernetes> 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\Dell\OneDrive\Documents\kubernetes> kubectl get namespaces
NAME              STATUS   AGE
default           Active   21d
kube-node-lease   Active   21d
kube-public       Active   21d
kube-system       Active   21d
```