Lab Exercise13-Managing Namespaces in Kubernetes

Name : Yuvraj Pratap Singh
Batch : Devops (2)
Sap id. : 500125214

Step 1: Understand Namespaces

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

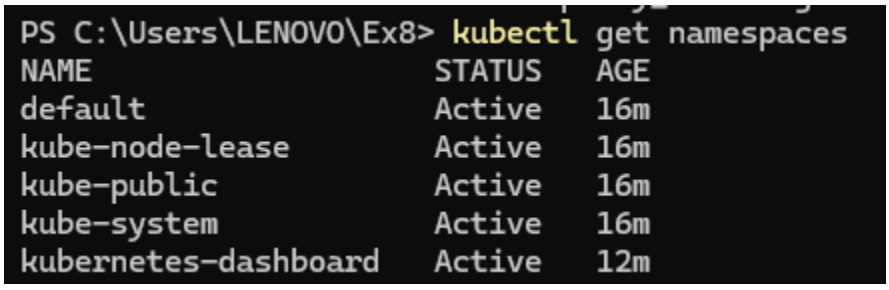
- Create environments for different applications or teams.
- Apply policies like resource quotas or network policies on a per-namespace basis.
- Separate operational environments (like development and production).

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.



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
```

```
PS C:\Users\LENOVO\Ex8> kubectl apply -f my-namespace.yaml
namespace/my-namespace created
```

Using kubectl Command

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\LENOVO\Ex8> kubectl create namespace my-namespace
Error from server (AlreadyExists): namespaces "my-namespace" already exists
PS C:\Users\LENOVO\Ex8> kubectl get namespaces
NAME                   STATUS   AGE
default                Active   18m
kube-node-lease        Active   18m
kube-public            Active   18m
kube-system            Active   18m
kubernetes-dashboard   Active   15m
my-namespace           Active   48s
```

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:

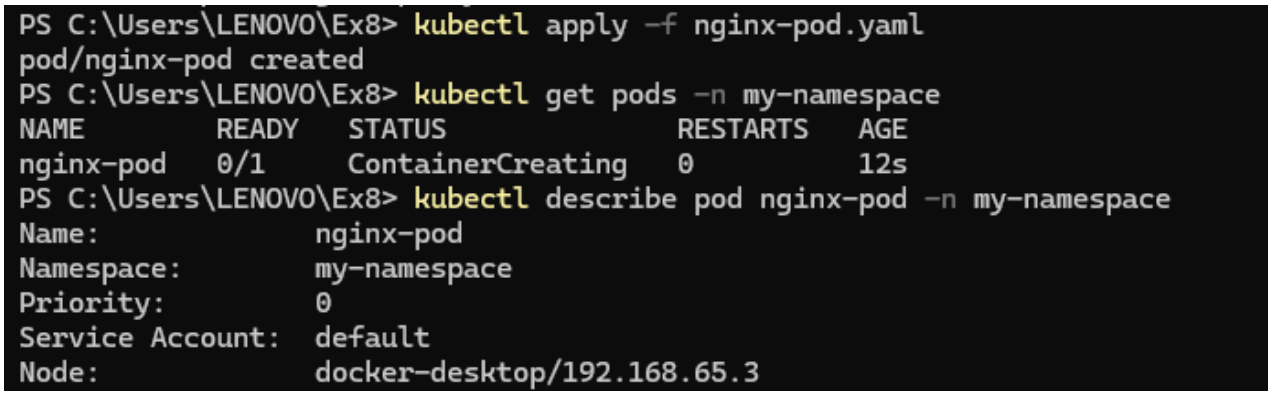
```
kubectl apply -f nginx-pod.yaml
```

Check the status of the Pod within the namespace:

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

To describe the Pod and see detailed information:

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



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
```

Check the status of the Service within the namespace:

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

To describe the Service and see detailed information:

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

```
PS C:\Users\LENOVO\Ex8> kubectl apply -f nginx-service.yaml
service/nginx-service created
PS C:\Users\LENOVO\Ex8> kubectl get services -n my-namespace
NAME            TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)   AGE
nginx-service   ClusterIP   10.101.11.222   <none>        80/TCP    9s
PS C:\Users\LENOVO\Ex8> 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:                       10.101.11.222
IPs:                      10.101.11.222
Port:                     <unset>  80/TCP
TargetPort:               80/TCP
Endpoints:
Session Affinity:         None
Internal Traffic Policy:  Cluster
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\LENOVO\Ex8> kubectl get pods -n my-namespace
NAME        READY   STATUS    RESTARTS   AGE
nginx-pod   1/1     Running   0          2m4s
```

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
```

Verify the current context's namespace:

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

```
PS C:\Users\LENOVO\Ex8> kubectl config set-context --current --namespace=my-namespace
Context "docker-desktop" modified.
PS C:\Users\LENOVO\Ex8> 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
```

```
PS C:\Users\LENOVO\Ex8> kubectl delete -f nginx-pod.yaml
pod "nginx-pod" deleted from my-namespace namespace
PS C:\Users\LENOVO\Ex8> kubectl delete -f nginx-service.yaml
service "nginx-service" deleted from my-namespace namespace
PS C:\Users\LENOVO\Ex8> 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\LENOVO\Ex8> kubectl get namespaces
NAME                   STATUS   AGE
default                Active   22m
kube-node-lease        Active   22m
kube-public            Active   22m
kube-system            Active   22m
kubernetes-dashboard   Active   19m
```