Shivang Mangal **500124202** **Batch – 2 DevOps**

Lab Exercise 14- Implementing Resource Quota in Kubernetes

Objective:

In Kubernetes, Resource Quotas are used to control the resource consumption of namespaces. They help in managing and enforcing limits on the usage of resources like CPU, memory, and the number of objects (e.g., Pods, Services) within a namespace. This exercise will guide you through creating and managing Resource Quotas to limit the resources used by applications in a specific namespace.

Step 1: Understand Resource Quotas

Resource Quotas allow you to:

- Limit the amount of CPU and memory a namespace can use.
- Control the number of certain types of resources (e.g., Pods, Services, PersistentVolumeClaims) in a namespace.
- Prevent a namespace from consuming more resources than allocated, ensuring fair usage across multiple teams or applications.

Step 2: Create a Namespace

First, create a namespace where you will apply the Resource Quota. This helps in isolating and controlling resource usage within that specific namespace.

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

```
apiVersion: v1
kind: Namespace
metadata:
  name: myns
```

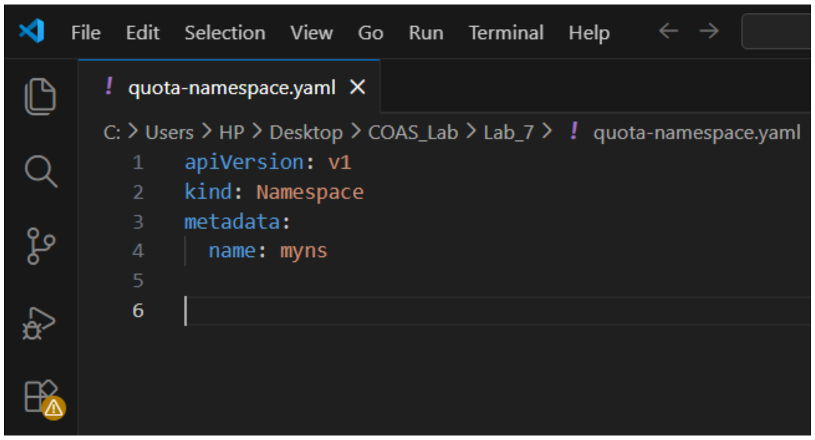
Apply the YAML to create the namespace:

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

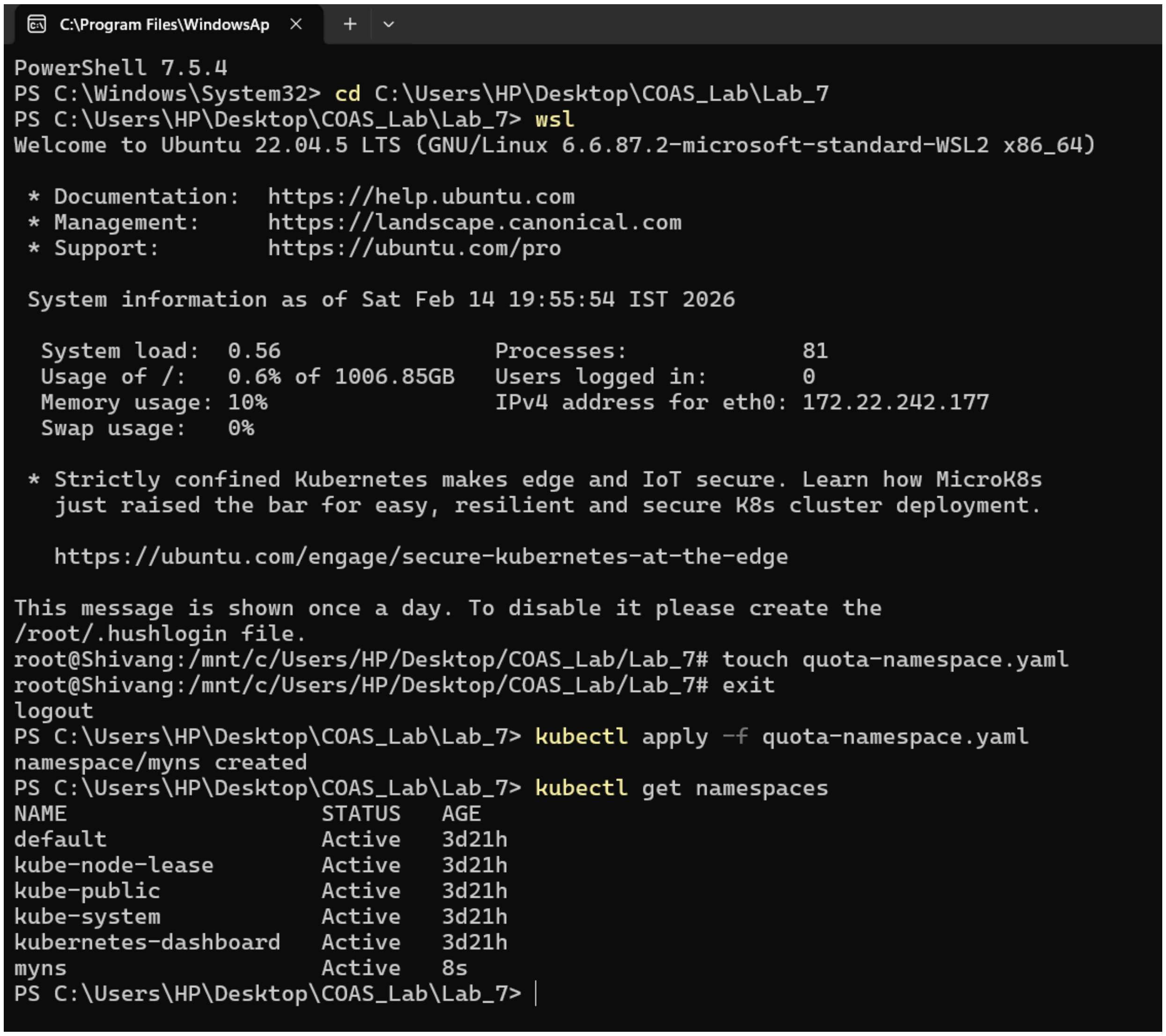
Verify that the namespace is created:

```
kubectl get namespaces
```

You should see quota-example listed in the output.


File
Edit
Selection
View
Go
Run
Terminal
Help
quota-namespace.yaml
C: > Users > HP > Desktop > COAS_Lab > Lab_7 > quota-namespace.yaml

```yaml
apiVersion: v1
kind: Namespace
metadata:
  name: myns

```



```
PowerShell 7.5.4
PS C:\Windows\System32> cd C:\Users\HP\Desktop\COAS_Lab\Lab_7
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7> wsl
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 6.6.87.2-microsoft-standard-WSL2 x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Sat Feb 14 19:55:54 IST 2026

  System load:  0.56               Processes:             81
  Usage of /:   0.6% of 1006.85GB  Users logged in:       0
  Memory usage: 10%                IPv4 address for eth0: 172.22.242.177
  Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

This message is shown once a day. To disable it please create the
/root/.hushlogin file.
root@Shivang:/mnt/c/Users/HP/Desktop/COAS_Lab/Lab_7# touch quota-namespace.yaml
root@Shivang:/mnt/c/Users/HP/Desktop/COAS_Lab/Lab_7# exit
logout
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7> kubectl apply -f quota-namespace.yaml
namespace/myns created
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7> kubectl get namespaces
NAME                   STATUS   AGE
default                Active   3d21h
kube-node-lease        Active   3d21h
kube-public            Active   3d21h
kube-system            Active   3d21h
kubernetes-dashboard   Active   3d21h
myns                   Active   8s
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7> |
```

Step 3: Define a Resource Quota

Next, create a Resource Quota YAML file named **resource-quota.yaml** with the following content:

```
apiVersion: v1
kind: ResourceQuota
metadata:
  name: myns-quota    # The name of the Resource Quota.
  namespace: myns # The namespace to which the Resource Quota will apply.
spec:
  hard:                  # The hard limits imposed by this Resource Quota.
    requests.cpu: "2"    # The total CPU resource requests allowed in the namespace (2 cores).
    requests.memory: "4Gi" # The total memory resource requests allowed in the namespace (4
GiB).
    limits.cpu: "4"      # The total CPU resource limits allowed in the namespace (4 cores).
    limits.memory: "8Gi" # The total memory resource limits allowed in the namespace (8 GiB).
    pods: "10"           # The total number of Pods allowed in the namespace.
    persistentvolumeclaims: "5" # The total number of PersistentVolumeClaims allowed in the
namespace.
    configmaps: "10"     # The total number of ConfigMaps allowed in the namespace.
    services: "5"        # The total number of Services allowed in the namespace.
```

Step 4: Apply the Resource Quota

Apply the Resource Quota YAML to the namespace:

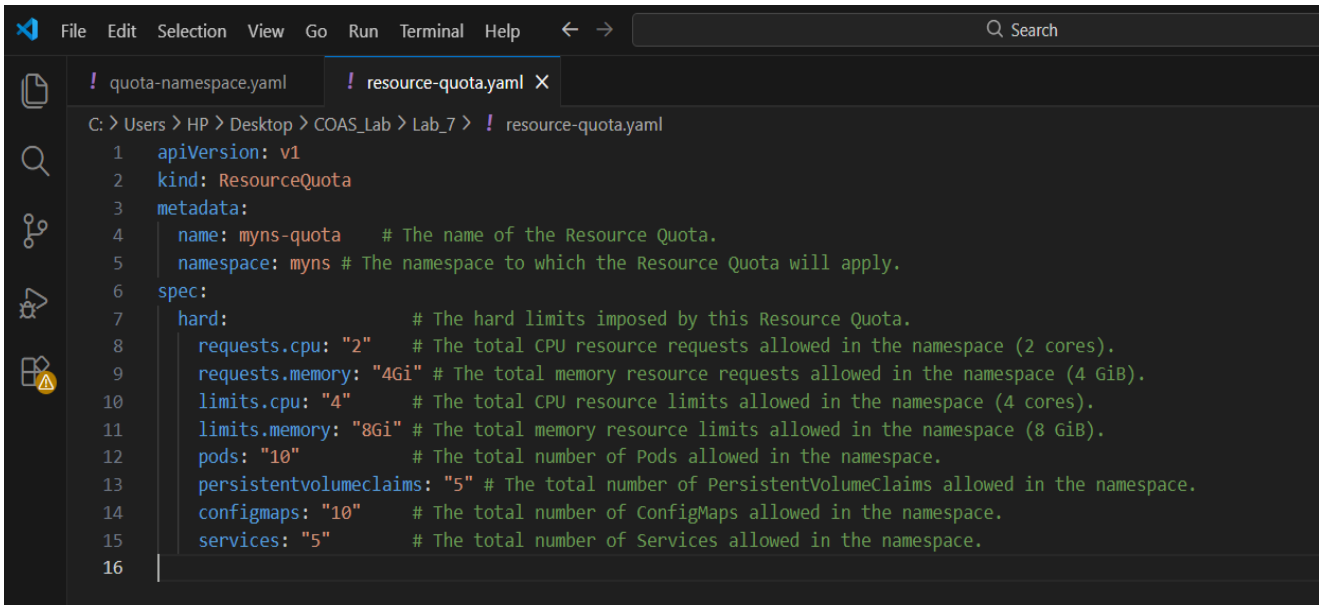
```
kubectl apply -f resource-quota.yaml
```

Verify that the Resource Quota is applied:

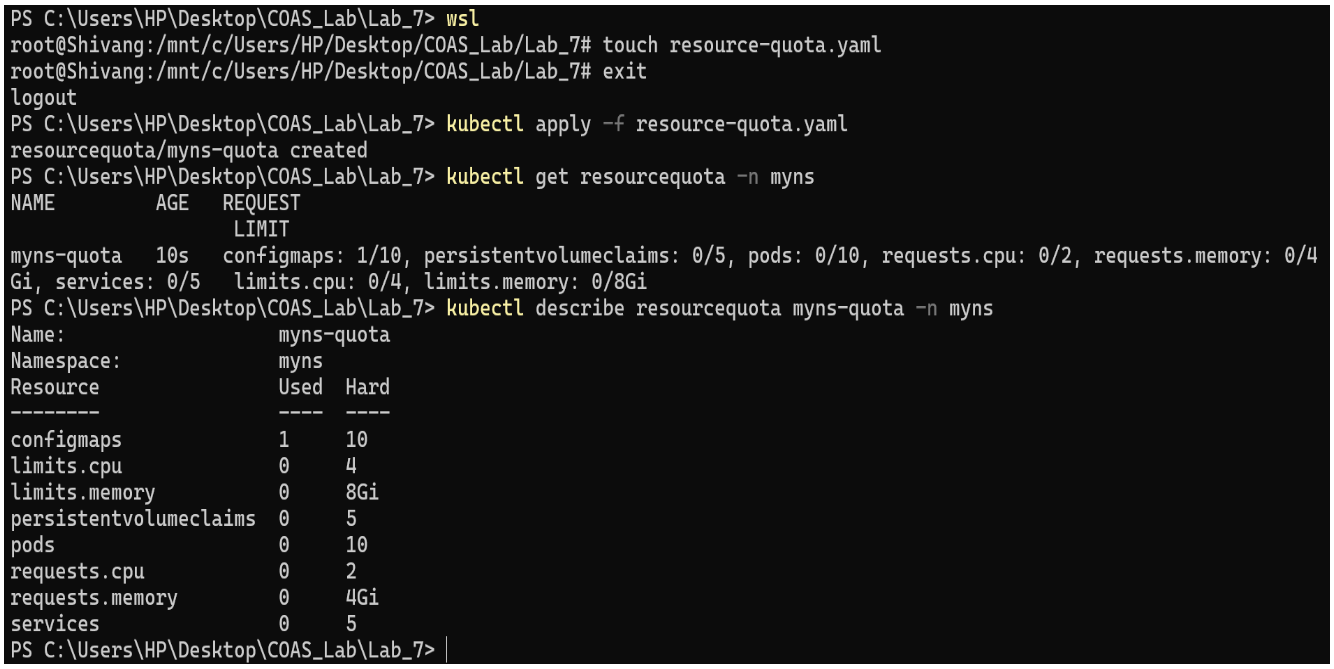
```
kubectl get resourcequota -n myns
```

To see the details of the applied Resource Quota:

```
kubectl describe resourcequota myns-quota -n myns
```


File Edit Selection View Go Run Terminal Help
Search
quota-namespace.yaml
resource-quota.yaml
C: > Users > HP > Desktop > COAS_Lab > Lab_7 > resource-quota.yaml

```yaml
apiVersion: v1
kind: ResourceQuota
metadata:
  name: myns-quota    # The name of the Resource Quota.
  namespace: myns # The namespace to which the Resource Quota will apply.
spec:
  hard:                 # The hard limits imposed by this Resource Quota.
    requests.cpu: "2"    # The total CPU resource requests allowed in the namespace (2 cores).
    requests.memory: "4Gi" # The total memory resource requests allowed in the namespace (4 GiB).
    limits.cpu: "4"      # The total CPU resource limits allowed in the namespace (4 cores).
    limits.memory: "8Gi" # The total memory resource limits allowed in the namespace (8 GiB).
    pods: "10"          # The total number of Pods allowed in the namespace.
    persistentvolumeclaims: "5" # The total number of PersistentVolumeClaims allowed in the namespace.
    configmaps: "10"     # The total number of ConfigMaps allowed in the namespace.
    services: "5"        # The total number of Services allowed in the namespace.

```



```
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7> wsl
root@Shivang:/mnt/c/Users/HP/Desktop/COAS_Lab/Lab_7# touch resource-quota.yaml
root@Shivang:/mnt/c/Users/HP/Desktop/COAS_Lab/Lab_7# exit
logout
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7> kubectl apply -f resource-quota.yaml
resourcequota/myns-quota created
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7> kubectl get resourcequota -n myns
NAME         AGE   REQUEST
                    LIMIT
myns-quota   10s   configmaps: 1/10, persistentvolumeclaims: 0/5, pods: 0/10, requests.cpu: 0/2, requests.memory: 0/4
Gi, services: 0/5   limits.cpu: 0/4, limits.memory: 0/8Gi
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7> kubectl describe resourcequota myns-quota -n myns
Name:                   myns-quota
Namespace:              myns
Resource                Used  Hard
--------                ----  ----
configmaps              1     10
limits.cpu              0     4
limits.memory           0     8Gi
persistentvolumeclaims  0     5
pods                    0     10
requests.cpu            0     2
requests.memory         0     4Gi
services                0     5
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7>
```

Step 5: Test the Resource Quota

Let's create some resources in the quota-example namespace to see how the Resource Quota affects them. Deploy a ReplicaSet with Resource Requests and Limits

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

```
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: nginx-replicaset
  namespace: myns
spec:
  replicas: 5            # Desired number of Pod replicas.
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:latest
        ports:
        - containerPort: 80
        resources:         # Define resource requests and limits.
          requests:
            memory: "100Mi"
            cpu: "100m"
          limits:
            memory: "200Mi"
            cpu: "200m"
```

Explanation:

This ReplicaSet requests a total of 500m CPU and 500Mi memory across 5 replicas.

It also limits each replica to use a maximum of 200m CPU and 200Mi memory.

Apply this YAML to create the ReplicaSet:

```
kubectl apply -f nginx-replicaset-quota.yaml
```

Check the status of the Pods and ensure they are created within the constraints of the Resource Quota:

```
kubectl get pods -n myns
```

To describe the Pods and see their resource allocations:

```
kubectl describe pods -l app=nginx -n myns
```

Attempt to Exceed the Resource Quota

Try creating additional resources to see if they are rejected when exceeding the quota. For example, create more Pods or increase the CPU/memory requests to exceed the quota limits.

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

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx-extra-pod
  namespace: myns
spec:
  containers:
  - name: nginx
    image: nginx:latest
    resources:
      requests:
        memory: "3Gi" # Requests a large amount of memory.
        cpu: "2"      # Requests a large amount of CPU.
      limits:
        memory: "4Gi"
        cpu: "2"
```

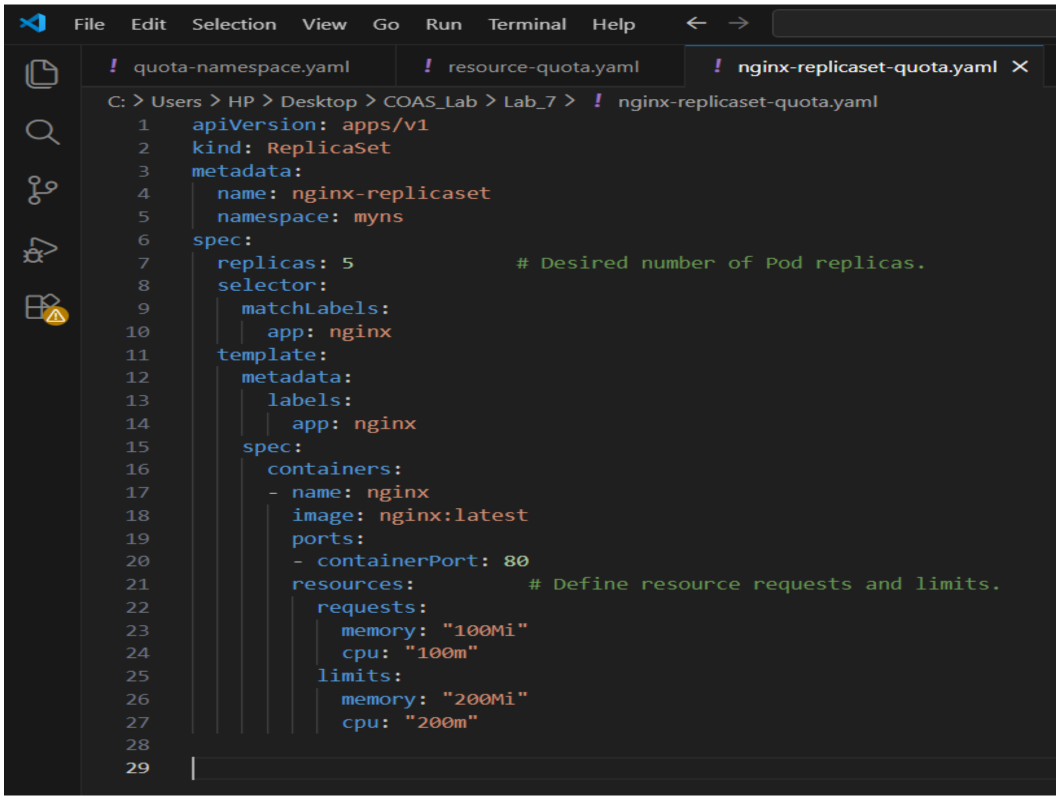
Apply this YAML to create the Pod:

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

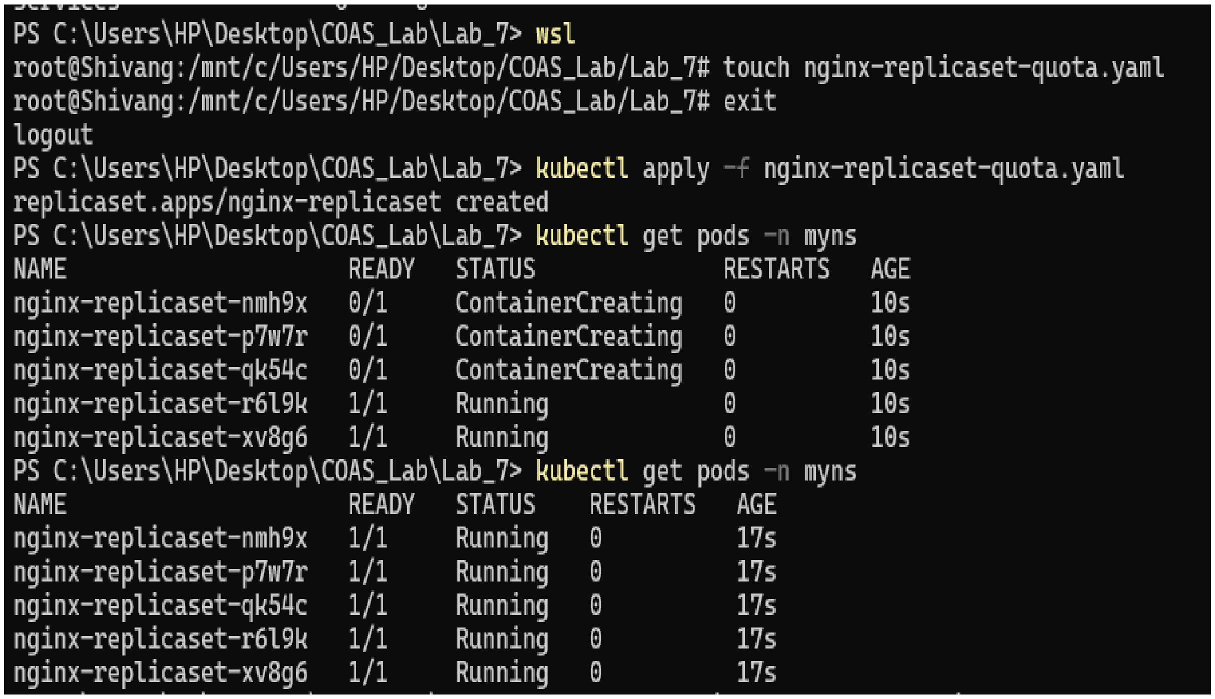
This should fail due to exceeding the Resource Quota. Check the events to see the failure reason:

```
kubectl get events -n myns
```

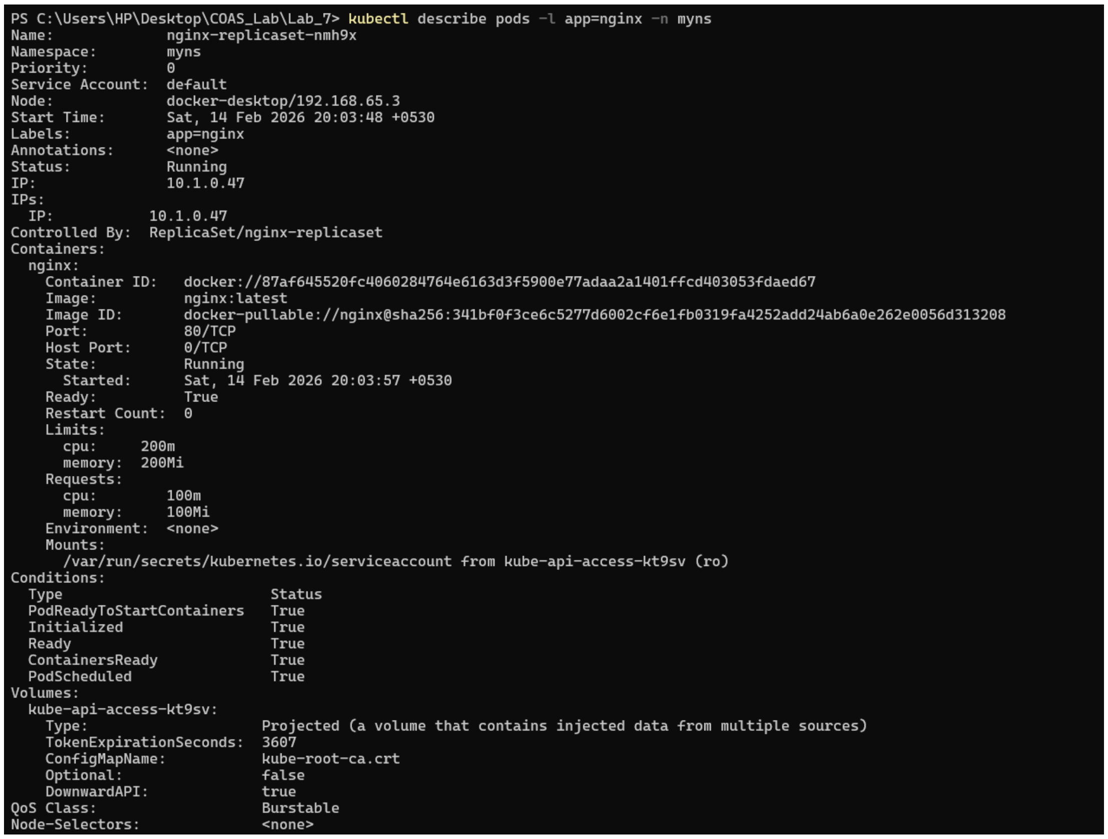
Look for error messages indicating that the Pod creation was denied due to resource constraints.


File
Edit
Selection
View
Go
Run
Terminal
Help
quota-namespace.yaml
resource-quota.yaml
nginx-replicaset-quota.yaml
C: > Users > HP > Desktop > COAS_Lab > Lab_7 > nginx-replicaset-quota.yaml

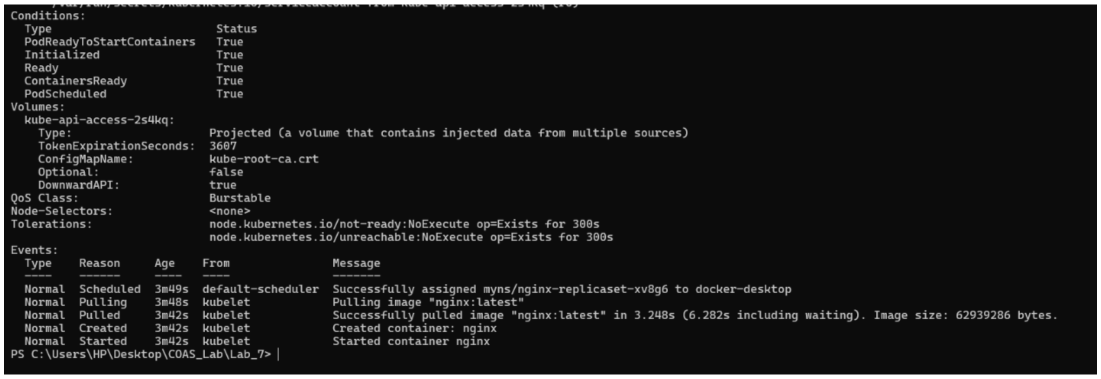
```yaml
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: nginx-replicaset
  namespace: myns
spec:
  replicas: 5            # Desired number of Pod replicas.
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:latest
        ports:
        - containerPort: 80
        resources:         # Define resource requests and limits.
          requests:
            memory: "100Mi"
            cpu: "100m"
          limits:
            memory: "200Mi"
            cpu: "200m"

```



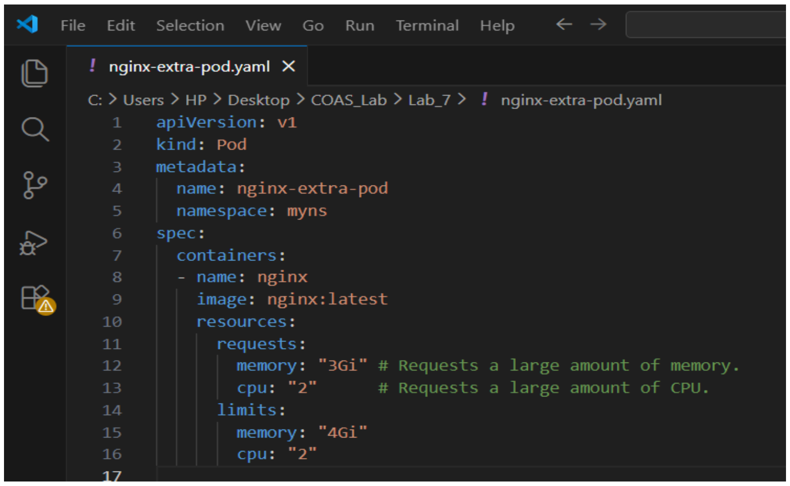
```
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7> wsl
root@Shivang:/mnt/c/Users/HP/Desktop/COAS_Lab/Lab_7# touch nginx-replicaset-quota.yaml
root@Shivang:/mnt/c/Users/HP/Desktop/COAS_Lab/Lab_7# exit
logout
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7> kubectl apply -f nginx-replicaset-quota.yaml
replicaset.apps/nginx-replicaset created
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7> kubectl get pods -n myns
NAME                     READY   STATUS              RESTARTS   AGE
nginx-replicaset-nmh9x   0/1     ContainerCreating   0          10s
nginx-replicaset-p7w7r   0/1     ContainerCreating   0          10s
nginx-replicaset-qk54c   0/1     ContainerCreating   0          10s
nginx-replicaset-r6l9k   1/1     Running             0          10s
nginx-replicaset-xv8g6   1/1     Running             0          10s
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7> kubectl get pods -n myns
NAME                     READY   STATUS    RESTARTS   AGE
nginx-replicaset-nmh9x   1/1     Running   0          17s
nginx-replicaset-p7w7r   1/1     Running   0          17s
nginx-replicaset-qk54c   1/1     Running   0          17s
nginx-replicaset-r6l9k   1/1     Running   0          17s
nginx-replicaset-xv8g6   1/1     Running   0          17s
```



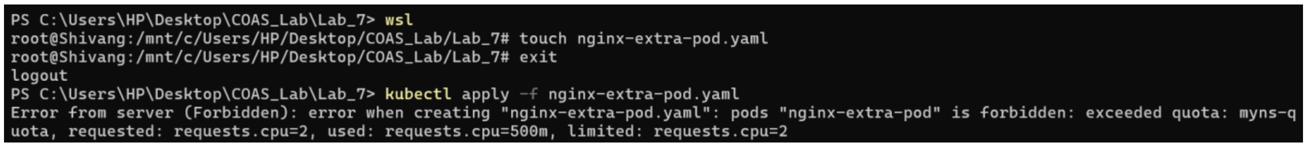
```
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7> kubectl describe pods -l app=nginx -n myns
Name:             nginx-replicaset-nmh9x
Namespace:        myns
Priority:         0
Service Account:  default
Node:             docker-desktop/192.168.65.3
Start Time:       Sat, 14 Feb 2026 20:03:48 +0530
Labels:           app=nginx
Annotations:      <none>
Status:           Running
IP:               10.1.0.47
IPs:
  IP:           10.1.0.47
Controlled By:  ReplicaSet/nginx-replicaset
Containers:
  nginx:
    Container ID:   docker://87af645520fc4060284764e6163d3f5900e77adaa2a1401ffcd403053fdaed67
    Image:          nginx:latest
    Image ID:       docker-pullable://nginx@sha256:341bf0f3ce6c5277d6002cf6e1fb0319fa4252add24ab6a0e262e0056d313208
    Port:           80/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Sat, 14 Feb 2026 20:03:57 +0530
    Ready:          True
    Restart Count:  0
    Limits:
      cpu:     200m
      memory:  200Mi
    Requests:
      cpu:        100m
      memory:     100Mi
    Environment:  <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-kt9sv (ro)
Conditions:
  Type                        Status
  PodReadyToStartContainers   True
  Initialized                 True
  Ready                       True
  ContainersReady             True
  PodScheduled                True
Volumes:
  kube-api-access-kt9sv:
    Type:                    Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds:  3607
    ConfigMapName:           kube-root-ca.crt
    Optional:                false
    DownwardAPI:             true
QoS Class:                   Burstable
Node-Selectors:              <none>
```



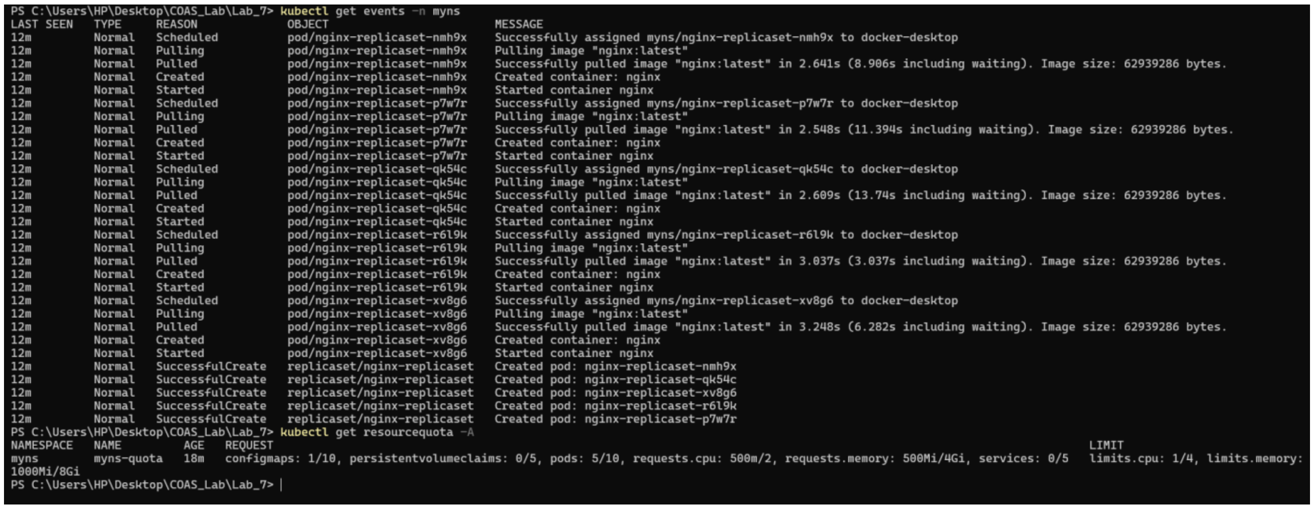
```
Conditions:
  Type                        Status
  PodReadyToStartContainers   True
  Initialized                 True
  Ready                       True
  ContainersReady             True
  PodScheduled                True
Volumes:
  kube-api-access-2s4kq:
    Type:                    Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds:  3607
    ConfigMapName:           kube-root-ca.crt
    Optional:                false
    DownwardAPI:             true
QoS Class:                   Burstable
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  3m49s  default-scheduler  Successfully assigned myns/nginx-replicaset-xv8g6 to docker-desktop
  Normal  Pulling    3m48s  kubelet            Pulling image "nginx:latest"
  Normal  Pulled     3m42s  kubelet            Successfully pulled image "nginx:latest" in 3.248s (6.282s including waiting). Image size: 62939286 bytes.
  Normal  Created    3m42s  kubelet            Created container: nginx
  Normal  Started    3m42s  kubelet            Started container nginx
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7>
```


File Edit Selection View Go Run Terminal Help
nginx-extra-pod.yaml
C: > Users > HP > Desktop > COAS_Lab > Lab_7 > nginx-extra-pod.yaml

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx-extra-pod
  namespace: myns
spec:
  containers:
  - name: nginx
    image: nginx:latest
    resources:
      requests:
        memory: "3Gi" # Requests a large amount of memory.
        cpu: "2"      # Requests a large amount of CPU.
      limits:
        memory: "4Gi"
        cpu: "2"
```



```
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7> wsl
root@Shivang:/mnt/c/Users/HP/Desktop/COAS_Lab/Lab_7# touch nginx-extra-pod.yaml
root@Shivang:/mnt/c/Users/HP/Desktop/COAS_Lab/Lab_7# exit
logout
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7> kubectl apply -f nginx-extra-pod.yaml
Error from server (Forbidden): error when creating "nginx-extra-pod.yaml": pods "nginx-extra-pod" is forbidden: exceeded quota: myns-q
uota, requested: requests.cpu=2, used: requests.cpu=500m, limited: requests.cpu=2
```

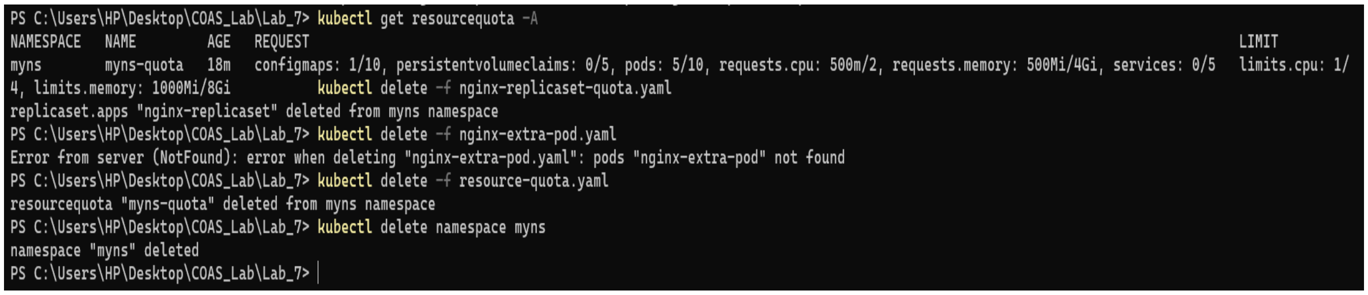


```
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7> kubectl get events -n myns
LAST SEEN   TYPE     REASON              OBJECT                           MESSAGE
12m         Normal   Scheduled           pod/nginx-replicaset-nmh9x       Successfully assigned myns/nginx-replicaset-nmh9x to docker-desktop
12m         Normal   Pulling             pod/nginx-replicaset-nmh9x       Pulling image "nginx:latest"
12m         Normal   Pulled              pod/nginx-replicaset-nmh9x       Successfully pulled image "nginx:latest" in 2.641s (8.906s including waiting). Image size: 62939286 bytes.
12m         Normal   Created             pod/nginx-replicaset-nmh9x       Created container: nginx
12m         Normal   Started             pod/nginx-replicaset-nmh9x       Started container nginx
12m         Normal   Scheduled           pod/nginx-replicaset-p7w7r       Successfully assigned myns/nginx-replicaset-p7w7r to docker-desktop
12m         Normal   Pulling             pod/nginx-replicaset-p7w7r       Pulling image "nginx:latest"
12m         Normal   Pulled              pod/nginx-replicaset-p7w7r       Successfully pulled image "nginx:latest" in 2.548s (11.394s including waiting). Image size: 62939286 bytes.
12m         Normal   Created             pod/nginx-replicaset-p7w7r       Created container: nginx
12m         Normal   Started             pod/nginx-replicaset-p7w7r       Started container nginx
12m         Normal   Scheduled           pod/nginx-replicaset-qk54c       Successfully assigned myns/nginx-replicaset-qk54c to docker-desktop
12m         Normal   Pulling             pod/nginx-replicaset-qk54c       Pulling image "nginx:latest"
12m         Normal   Pulled              pod/nginx-replicaset-qk54c       Successfully pulled image "nginx:latest" in 2.609s (13.74s including waiting). Image size: 62939286 bytes.
12m         Normal   Created             pod/nginx-replicaset-qk54c       Created container: nginx
12m         Normal   Started             pod/nginx-replicaset-qk54c       Started container nginx
12m         Normal   Scheduled           pod/nginx-replicaset-r6l9k       Successfully assigned myns/nginx-replicaset-r6l9k to docker-desktop
12m         Normal   Pulling             pod/nginx-replicaset-r6l9k       Pulling image "nginx:latest"
12m         Normal   Pulled              pod/nginx-replicaset-r6l9k       Successfully pulled image "nginx:latest" in 3.037s (3.037s including waiting). Image size: 62939286 bytes.
12m         Normal   Created             pod/nginx-replicaset-r6l9k       Created container: nginx
12m         Normal   Started             pod/nginx-replicaset-r6l9k       Started container nginx
12m         Normal   Scheduled           pod/nginx-replicaset-xv8g6       Successfully assigned myns/nginx-replicaset-xv8g6 to docker-desktop
12m         Normal   Pulling             pod/nginx-replicaset-xv8g6       Pulling image "nginx:latest"
12m         Normal   Pulled              pod/nginx-replicaset-xv8g6       Successfully pulled image "nginx:latest" in 3.248s (6.282s including waiting). Image size: 62939286 bytes.
12m         Normal   Created             pod/nginx-replicaset-xv8g6       Created container: nginx
12m         Normal   Started             pod/nginx-replicaset-xv8g6       Started container nginx
12m         Normal   SuccessfulCreate    replicaset/nginx-replicaset      Created pod: nginx-replicaset-nmh9x
12m         Normal   SuccessfulCreate    replicaset/nginx-replicaset      Created pod: nginx-replicaset-qk54c
12m         Normal   SuccessfulCreate    replicaset/nginx-replicaset      Created pod: nginx-replicaset-xv8g6
12m         Normal   SuccessfulCreate    replicaset/nginx-replicaset      Created pod: nginx-replicaset-r6l9k
12m         Normal   SuccessfulCreate    replicaset/nginx-replicaset      Created pod: nginx-replicaset-p7w7r
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7> kubectl get resourcequota -A
NAMESPACE   NAME         AGE   REQUEST                                                                                                          LIMIT
myns        myns-quota   18m   configmaps: 1/10, persistentvolumeclaims: 0/5, pods: 5/10, requests.cpu: 500m/2, requests.memory: 500Mi/4Gi, services: 0/5   limits.cpu: 1/4, limits.memory:
1000Mi/8Gi
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7> |
```

Step 6: Clean Up Resources

To delete the resources you created:

```
kubectl delete -f nginx-replicaset-quota.yaml
kubectl delete -f nginx-extra-pod.yaml
kubectl delete -f resource-quota.yaml
kubectl delete namespace myns
```



```
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7> kubectl get resourcequota -A
NAMESPACE   NAME         AGE   REQUEST                                                                                                          LIMIT
myns        myns-quota   18m   configmaps: 1/10, persistentvolumeclaims: 0/5, pods: 5/10, requests.cpu: 500m/2, requests.memory: 500Mi/4Gi, services: 0/5   limits.cpu: 1/
4, limits.memory: 1000Mi/8Gi          kubectl delete -f nginx-replicaset-quota.yaml
replicaset.apps "nginx-replicaset" deleted from myns namespace
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7> kubectl delete -f nginx-extra-pod.yaml
Error from server (NotFound): error when deleting "nginx-extra-pod.yaml": pods "nginx-extra-pod" not found
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7> kubectl delete -f resource-quota.yaml
resourcequota "myns-quota" deleted from myns namespace
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7> kubectl delete namespace myns
namespace "myns" deleted
PS C:\Users\HP\Desktop\COAS_Lab\Lab_7> |
```