Lab Exercise 14- Implementing Resource Quota in Kubernetes

Name- Misha
SAP ID- 500119679
Batch-2

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

```
PS C:\Users\Misha> d:
PS D:\> cd kubs
PS D:\kubs> code .
PS D:\kubs> kubectl apply -f quota-namespace.yaml
namespace/myns created
PS D:\kubs>
```

Verify that the namespace is created:

kubectl get namespaces

```
PS D:\kubs> kubectl get namespaces
NAME                   STATUS   AGE
default                Active   6d17h
kube-node-lease        Active   6d17h
kube-public            Active   6d17h
kube-system            Active   6d17h
kubernetes-dashboard   Active   6d17h
local-path-storage     Active   6d17h
myns                   Active   17s
PS D:\kubs>
```

You should see quota-example listed in the output.

Step 3: Define a Resource Quota

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

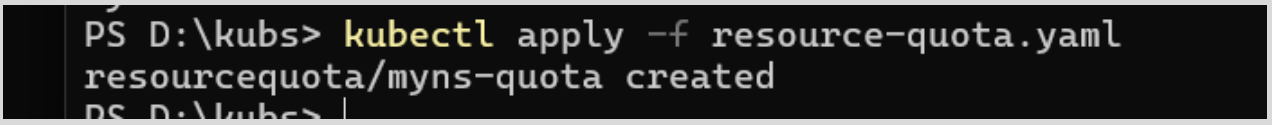
```
apiVersion: v1
kind: ResourceQuota
metadata:
  name: myns-quota    # The name of the Resource Quota.
  namespace: myns # The namespace to which the Resource Quota will apply.
spec:
```

```
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

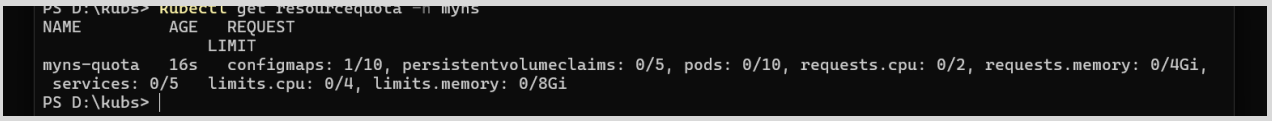
Apply the Resource Quota YAML to the namespace:

```
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
```

```
PS D:\kubs> 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
```

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

```
PS D:\kubs> kubectl apply -f nginx-replicaset-quota.yaml
replicaset.apps/nginx-replicaset created
PS D:\kubs>
```

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

kubectl get pods -n myns

```
replicaset.apps/nginx-replicaset created
PS D:\kubs> kubectl get pods -n myns
NAME                    READY   STATUS    RESTARTS   AGE
nginx-replicaset-654bd   1/1     Running   0          17s
nginx-replicaset-6lm9d   1/1     Running   0          17s
nginx-replicaset-pfc9h   1/1     Running   0          17s
nginx-replicaset-szj6v   1/1     Running   0          17s
nginx-replicaset-ww8ms   1/1     Running   0          17s
PS D:\kubs>
```

\

To describe the Pods and see their resource allocations:

kubectl describe pods -l app=nginx -n ~~quota-example~~ **myns**

```
PS D:\kubs> kubectl describe pods -l app=nginx -n myns
Name:             nginx-replicaset-654bd
Namespace:        myns
Priority:         0
Service Account:  default
Node:             desktop-control-plane/172.19.0.2
Start Time:       Mon, 23 Feb 2026 16:01:29 +0530
Labels:           app=nginx
Annotations:      <none>
Status:           Running
IP:               10.244.0.11
IPs:
  IP:           10.244.0.11
Controlled By:  ReplicaSet/nginx-replicaset
Containers:
  nginx:
    Container ID:   containerd://2a8100d103b658bc7c4e454d6064ab3748edfff31e67ab5da4e7034c7474a535
    Image:          nginx:latest
    Image ID:       docker.io/library/nginx@sha256:c881927c4077710ac4b1da63b83aa163937fb47457950c267d92f7e4dedf4aec
    Port:           80/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Mon, 23 Feb 2026 16:01:30 +0530
    Ready:          True
    Restart Count:  0
    Limits:
      cpu:     200m
      memory:  200Mi
```

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

```
PS D:\kubs> 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-quota, requested: requests.cpu=2, used: requests.cpu=500m, limited: requests.cpu=2
PS D:\kubs>
```

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

kubectl get events -n ~~quota-example~~ myns

```
PS D:\kubs> kubectl get events -n myns
LAST SEEN   TYPE     REASON             OBJECT                          MESSAGE
13m         Normal   Scheduled          pod/nginx-replicaset-654bd      Successfully assigned myns/nginx-replicaset-654bd
to desktop-control-plane
13m         Normal   Pulling            pod/nginx-replicaset-654bd      Pulling image "nginx:latest"
13m         Normal   Pulled             pod/nginx-replicaset-654bd      Successfully pulled image "nginx:latest" in 67ms (
119ms including waiting). Image size: 62870438 bytes.
13m         Normal   Created            pod/nginx-replicaset-654bd      Created container nginx
13m         Normal   Started            pod/nginx-replicaset-654bd      Started container nginx
13m         Normal   Scheduled          pod/nginx-replicaset-6lm9d      Successfully assigned myns/nginx-replicaset-6lm9d
to desktop-control-plane
13m         Normal   Pulling            pod/nginx-replicaset-6lm9d      Pulling image "nginx:latest"
13m         Normal   Pulled             pod/nginx-replicaset-6lm9d      Successfully pulled image "nginx:latest" in 46ms (
47ms including waiting). Image size: 62870438 bytes.
13m         Normal   Created            pod/nginx-replicaset-6lm9d      Created container nginx
13m         Normal   Started            pod/nginx-replicaset-6lm9d      Started container nginx
13m         Normal   Scheduled          pod/nginx-replicaset-pfc9h      Successfully assigned myns/nginx-replicaset-pfc9h
to desktop-control-plane
13m         Normal   Pulling            pod/nginx-replicaset-pfc9h      Pulling image "nginx:latest"
13m         Normal   Pulled             pod/nginx-replicaset-pfc9h      Successfully pulled image "nginx:latest" in 78ms (
176ms including waiting). Image size: 62870438 bytes.
```

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

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 D:\kubs> kubectl delete -f nginx-replicaset-quota.yaml
replicaset.apps "nginx-replicaset" deleted from myns namespace
PS D:\kubs> 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 D:\kubs> kubectl delete -f resource-quota.yaml
resourcequota "myns-quota" deleted from myns namespace
PS D:\kubs> kubectl delete namespace myns
namespace "myns" deleted
PS D:\kubs>
```