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

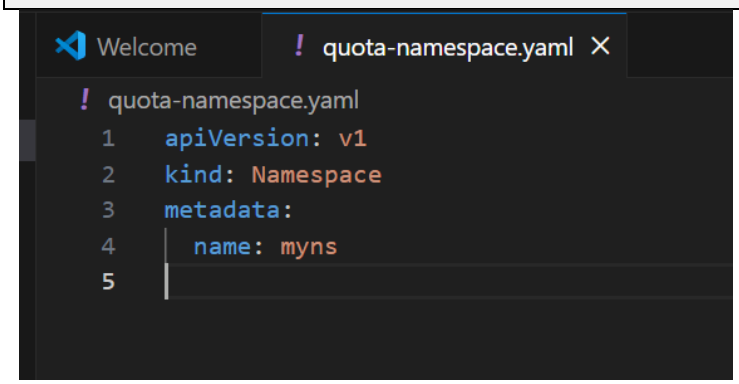
Resource Quotas allow you to:

- 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
```

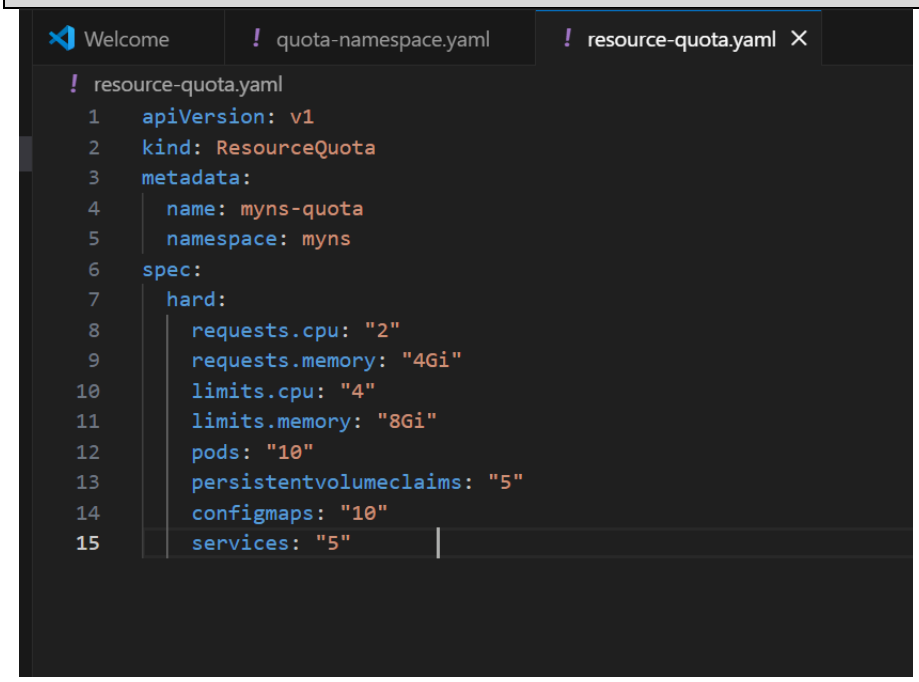
```
PS C:\Users\Devanshi\Desktop\lab_14> kubectl apply -f quota-namespace.yaml
namespace/myns created
PS C:\Users\Devanshi\Desktop\lab_14> kubectl get namespaces
NAME              STATUS   AGE
default           Active   42m
kube-node-lease   Active   42m
kube-public       Active   42m
kube-system       Active   42m
myns              Active   14s
```

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 C:\Users\Devanshi\Desktop\lab_14> kubectl apply -f resource-quota.yaml
resourcequota/myns-quota created
PS C:\Users\Devanshi\Desktop\lab_14> kubectl get resourcequota -n myns
NAME         AGE   REQUEST
    LIMIT
myns-quota   6s    configmaps: 1/10, persistentvolumeclaims: 0/5, pods: 0/10, requests.cpu: 0/2, requests.memory: 0/4Gi, serv
5   limits.cpu: 0/4, limits.memory: 0/8Gi
PS C:\Users\Devanshi\Desktop\lab_14> 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"
```

nginx-replicaset-quota.yaml

```yaml
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: nginx-replicaset
  namespace: myns
spec:
  replicas: 5
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:latest
        ports:
        - containerPort: 80
        resources:
          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 quota-example
```

```
PS C:\Users\Devanshi\Desktop\lab_14> kubectl apply -f nginx-replicaset-quota.yaml
replicaset.apps/nginx-replicaset created
PS C:\Users\Devanshi\Desktop\lab_14> kubectl get pods -n myns
NAME                     READY   STATUS              RESTARTS   AGE
nginx-replicaset-lz65r   1/1     Running             0          10s
nginx-replicaset-mqxps   1/1     Running             0          10s
nginx-replicaset-q2krr   0/1     ContainerCreating   0          10s
nginx-replicaset-vwttx   1/1     Running             0          10s
nginx-replicaset-xwl2z   1/1     Running             0          10s
PS C:\Users\Devanshi\Desktop\lab_14> kubectl describe pods -l app=nginx -n quota-example
No resources found in quota-example namespace.
```

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"
```

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"
        cpu: "2"
      limits:
        memory: "4Gi"
        cpu: "2"
```

Apply this YAML to create the Pod:

kubectl apply -f nginx-extra-pod.yaml

```
PS C:\Users\Devanshi\Desktop\lab_14> kubectl apply -f pod.yaml
Error from server (Forbidden): error when creating "pod.yaml": pods "nginx-extra-pod" is forbidden: exceeded quota: myns-quota, reque
sted: requests.cpu=2, used: requests.cpu=500m, limited: requests.cpu=2
```

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

kubectl get events -n quota-example

```
PS C:\Users\Devanshi\Desktop\lab_14> kubectl get events -n myns
LAST SEEN   TYPE     REASON              OBJECT                          MESSAGE
5m22s       Normal   Scheduled           pod/nginx-replicaset-lz65r      Successfully assigned myns/nginx-replicaset-lz65r to docker-desktop
5m21s       Normal   Pulling             pod/nginx-replicaset-lz65r      Pulling image "nginx:latest"
5m17s       Normal   Pulled              pod/nginx-replicaset-lz65r      Successfully pulled image "nginx:latest" in 1.887s (3.777s including waiti
ng). Image size: 62939286 bytes.
5m17s       Normal   Created             pod/nginx-replicaset-lz65r      Created container: nginx
5m17s       Normal   Started             pod/nginx-replicaset-lz65r      Started container nginx
5m22s       Normal   Scheduled           pod/nginx-replicaset-mqxps      Successfully assigned myns/nginx-replicaset-mqxps to docker-desktop
5m21s       Normal   Pulling             pod/nginx-replicaset-mqxps      Pulling image "nginx:latest"
5m13s       Normal   Pulled              pod/nginx-replicaset-mqxps      Successfully pulled image "nginx:latest" in 1.934s (7.528s including waiti
ng). Image size: 62939286 bytes.
5m13s       Normal   Created             pod/nginx-replicaset-mqxps      Created container: nginx
5m13s       Normal   Started             pod/nginx-replicaset-mqxps      Started container nginx
5m22s       Normal   Scheduled           pod/nginx-replicaset-q2krr      Successfully assigned myns/nginx-replicaset-q2krr to docker-desktop
5m21s       Normal   Pulling             pod/nginx-replicaset-q2krr      Pulling image "nginx:latest"
5m11s       Normal   Pulled              pod/nginx-replicaset-q2krr      Successfully pulled image "nginx:latest" in 1.903s (9.427s including waiti
ng). Image size: 62939286 bytes.
5m11s       Normal   Created             pod/nginx-replicaset-q2krr      Created container: nginx
5m11s       Normal   Started             pod/nginx-replicaset-q2krr      Started container nginx
5m22s       Normal   Scheduled           pod/nginx-replicaset-vwttx      Successfully assigned myns/nginx-replicaset-vwttx to docker-desktop
5m21s       Normal   Pulling             pod/nginx-replicaset-vwttx      Pulling image "nginx:latest"
5m19s       Normal   Pulled              pod/nginx-replicaset-vwttx      Successfully pulled image "nginx:latest" in 1.943s (1.943s including waiti
ng). Image size: 62939286 bytes.
5m19s       Normal   Created             pod/nginx-replicaset-vwttx      Created container: nginx
5m19s       Normal   Started             pod/nginx-replicaset-vwttx      Started container nginx
5m22s       Normal   Scheduled           pod/nginx-replicaset-xwl2z      Successfully assigned myns/nginx-replicaset-xwl2z to docker-desktop
5m21s       Normal   Pulling             pod/nginx-replicaset-xwl2z      Pulling image "nginx:latest"
5m15s       Normal   Pulled              pod/nginx-replicaset-xwl2z      Successfully pulled image "nginx:latest" in 1.862s (5.606s including waiti
ng). Image size: 62939286 bytes.
5m15s       Normal   Created             pod/nginx-replicaset-xwl2z      Created container: nginx
5m15s       Normal   Started             pod/nginx-replicaset-xwl2z      Started container nginx
5m22s       Normal   SuccessfulCreate    replicaset/nginx-replicaset     Created pod: nginx-replicaset-xwl2z
5m22s       Normal   SuccessfulCreate    replicaset/nginx-replicaset     Created pod: nginx-replicaset-mqxps
5m22s       Normal   SuccessfulCreate    replicaset/nginx-replicaset     Created pod: nginx-replicaset-lz65r
5m22s       Normal   SuccessfulCreate    replicaset/nginx-replicaset     Created pod: nginx-replicaset-vwttx
5m22s       Normal   SuccessfulCreate    replicaset/nginx-replicaset     Created pod: nginx-replicaset-q2krr
```

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

```
replicaset.apps "nginx-replicaset" deleted
PS C:\Users\Devanshi\Desktop\lab_14> kubectl delete -f pod.yaml
Error from server (NotFound): error when deleting "pod.yaml": pods "nginx-extra-pod" not found
PS C:\Users\Devanshi\Desktop\lab_14> kubectl delete -f resource-quota.yaml
resourcequota "myns-quota" deleted
PS C:\Users\Devanshi\Desktop\lab_14> kubectl delete namespace myns
namespace "myns" deleted
PS C:\Users\Devanshi\Desktop\lab_14>
```