

# 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.

```
PS C:\Users\namit> notepad quota-namespace.yaml
PS C:\Users\namit> kubectl apply -f quota-namespace.yaml
namespace/myns created
PS C:\Users\namit> kubectl get namespaces
NAME                STATUS    AGE
default             Active   14d
kube-node-lease     Active   14d
kube-public         Active   14d
kube-system         Active   14d
kubernetes-dashboard Active   108m
myns                Active   11s
PS C:\Users\namit>
```

### 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
```

```

PS C:\Users\namit> notepad resource-quota.yaml
PS C:\Users\namit> kubectl apply -f resource-quota.yaml
resourcequota/myns-quota created
PS C:\Users\namit> kubectl get resourcequota -n myns
NAME      AGE      REQUEST
myns-quota 42s      configmaps: 1/10, persistentvolumeclaims: 0/5, pods: 0/10, requests.cpu: 0/2, requests.memory: 0/4Gi, services: 0/5
limits.cpu: 0/4, limits.memory: 0/8Gi
PS C:\Users\namit>

```

To see the details of the applied Resource Quota:

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

```

PS C:\Users\namit> 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\namit>

```

## 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 C:\Users\namit> notepad nginx-replicaset-quota.yaml
PS C:\Users\namit> kubectl apply -f nginx-replicaset-quota.yaml
replicaset.apps/nginx-replicaset created
PS C:\Users\namit>
```

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

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

```
PS C:\Users\namit> kubectl get pods -n myns
```

NAME	READY	STATUS	RESTARTS	AGE
nginx-replicaset-67b4m	1/1	Running	0	73s
nginx-replicaset-b76tj	1/1	Running	0	73s
nginx-replicaset-q96g5	1/1	Running	0	73s
nginx-replicaset-wgj45	1/1	Running	0	73s
nginx-replicaset-xvdvb	1/1	Running	0	73s

```
PS C:\Users\namit>
```

To describe the Pods and see their resource allocations:

```
kubectl describe pods -l app=nginx -n quota-example
```

```
PS C:\Users\namit> kubectl describe pods -l app=nginx -n myns
Name:          nginx-replicaset-67b4m
Namespace:     myns
Priority:       0
Service Account: default
Node:          docker-desktop/192.168.65.3
Start Time:    Tue, 24 Feb 2026 18:17:31 +0530
Labels:        app=nginx
Annotations:    <none>
Status:        Running
IP:            10.1.0.57
IPs:           10.1.0.57
Controlled By: ReplicaSet/nginx-replicaset
Containers:
  nginx:
    Container ID:  docker://c931addfb399fb99afbcf511fb87877bd876376ff32014ef17103d407f138ce0
    Image:          nginx:latest
    Image ID:       docker-pullable://nginx@sha256:341bf0f3ce6c5277d6002cf6e1fb0319fa4252add24ab6a0e262e0056d313208
    Port:           80/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Tue, 24 Feb 2026 18:17:36 +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-76s78 (ro)
Conditions:
  Type                               Status
  PodReadyToStartContainers          True
  Initialized                         True
  Ready                             True
  ContainersReady                    True
  PodScheduled                       True
```

```
Host Port:      0/TCP
State:          Running
Started:        Tue, 24 Feb 2026 18:17:38 +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-8psvl (ro)
Conditions:
  Type                               Status
  PodReadyToStartContainers          True
  Initialized                         True
  Ready                             True
  ContainersReady                    True
  PodScheduled                       True
Volumes:
  kube-api-access-8psvl:
    Type:          Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds: 3607
    ConfigMapName:    kube-root-ca.crt
    ConfigMapOptional: <nil>
    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   4m21s default-scheduler Successfully assigned myns/nginx-replicaset-xvdb to docker-desktop
  Normal    Pulling    4m19s kubelet        Pulling image "nginx:latest"
  Normal    Pulled     4m14s kubelet        Successfully pulled image "nginx:latest" in 2.557s (5.26s including waiting). Image size: 62939286 bytes.
  Normal    Created    4m14s kubelet        Created container: nginx
  Normal    Started    4m13s kubelet        Started container nginx
PS C:\Users\namit>
```

## 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 C:\Users\namit> 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 C:\Users\namit>
```

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

```
kubectl get events -n quota-example
```

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



```

PS C:\Users\namit> kubectl get events --n myns
LAST SEEN   TYPE      REASON              OBJECT                               MESSAGE
18s         Normal    Scheduled            pod/nginx-replicaset-26zdk         Successfully assigned myns/nginx-replicaset-26zdk to docker-desktop
16s         Normal    Pulling             pod/nginx-replicaset-26zdk         Pulling image "nginx:latest"
20m         Normal    Scheduled            pod/nginx-replicaset-67b4m         Successfully assigned myns/nginx-replicaset-67b4m to docker-desktop
20m         Normal    Pulling             pod/nginx-replicaset-67b4m         Pulling image "nginx:latest"
20m         Normal    Pulled             pod/nginx-replicaset-67b4m         Successfully pulled image "nginx:latest" in 2.722s (2.722s including waiting). Image
size: 62939286 bytes.
20m         Normal    Created             pod/nginx-replicaset-67b4m         Created container: nginx
20m         Normal    Started            pod/nginx-replicaset-67b4m         Started container nginx
18s         Normal    Scheduled            pod/nginx-replicaset-75tjx         Successfully assigned myns/nginx-replicaset-75tjx to docker-desktop
16s         Normal    Pulling             pod/nginx-replicaset-75tjx         Pulling image "nginx:latest"
2s          Normal    Pulled             pod/nginx-replicaset-75tjx         Successfully pulled image "nginx:latest" in 3.192s (3.192s including waiting). Image
size: 62939286 bytes.
2s          Normal    Created             pod/nginx-replicaset-75tjx         Created container: nginx
2s          Normal    Started            pod/nginx-replicaset-75tjx         Started container nginx
18s         Normal    Scheduled            pod/nginx-replicaset-7cj4z         Successfully assigned myns/nginx-replicaset-7cj4z to docker-desktop
16s         Normal    Pulling             pod/nginx-replicaset-7cj4z         Pulling image "nginx:latest"
9s          Normal    Pulled             pod/nginx-replicaset-7cj4z         Successfully pulled image "nginx:latest" in 2.969s (2.969s including waiting). Image
size: 62939286 bytes.
9s          Normal    Created             pod/nginx-replicaset-7cj4z         Created container: nginx
8s          Normal    Started            pod/nginx-replicaset-7cj4z         Started container nginx
20m         Normal    Scheduled            pod/nginx-replicaset-b76tj         Successfully assigned myns/nginx-replicaset-b76tj to docker-desktop
20m         Normal    Pulling             pod/nginx-replicaset-b76tj         Pulling image "nginx:latest"
19m         Normal    Pulled             pod/nginx-replicaset-b76tj         Successfully pulled image "nginx:latest" in 2.599s (2.599s including waiting). Image
size: 62939286 bytes.
19m         Normal    Created             pod/nginx-replicaset-b76tj         Created container: nginx
19m         Normal    Started            pod/nginx-replicaset-b76tj         Started container nginx
20m         Normal    Scheduled            pod/nginx-replicaset-q96g5         Successfully assigned myns/nginx-replicaset-q96g5 to docker-desktop
20m         Normal    Pulling             pod/nginx-replicaset-q96g5         Pulling image "nginx:latest"
19m         Normal    Pulled             pod/nginx-replicaset-q96g5         Successfully pulled image "nginx:latest" in 2.76s (2.76s including waiting). Image
size: 62939286 bytes.
19m         Normal    Created             pod/nginx-replicaset-q96g5         Created container: nginx
19m         Normal    Started            pod/nginx-replicaset-q96g5         Started container nginx
18s         Normal    Scheduled            pod/nginx-replicaset-sn82c         Successfully assigned myns/nginx-replicaset-sn82c to docker-desktop
16s         Normal    Pulling             pod/nginx-replicaset-sn82c         Pulling image "nginx:latest"
12s         Normal    Pulled             pod/nginx-replicaset-sn82c         Successfully pulled image "nginx:latest" in 4.592s (4.592s including waiting). Image
size: 62939286 bytes.
12s         Normal    Created             pod/nginx-replicaset-sn82c         Created container: nginx
11s         Normal    Started            pod/nginx-replicaset-sn82c         Started container nginx
20m         Normal    Scheduled            pod/nginx-replicaset-wgj45         Successfully assigned myns/nginx-replicaset-wgj45 to docker-desktop

ize: 62939286 bytes.
20m         Normal    Created             pod/nginx-replicaset-xvdvb         Created container: nginx
20m         Normal    Started            pod/nginx-replicaset-xvdvb         Started container nginx
18s         Normal    Scheduled            pod/nginx-replicaset-zbq6w         Successfully assigned myns/nginx-replicaset-zbq6w to docker-desktop
16s         Normal    Pulling             pod/nginx-replicaset-zbq6w         Pulling image "nginx:latest"
6s          Normal    Pulled             pod/nginx-replicaset-zbq6w         Successfully pulled image "nginx:latest" in 3.27s (3.27s including waiting). Image
size: 62939286 bytes.
5s          Normal    Created             pod/nginx-replicaset-zbq6w         Created container: nginx
5s          Normal    Started            pod/nginx-replicaset-zbq6w         Started container nginx
20m         Warning   FailedCreate        replicaset/nginx-replicaset        Error creating: Unauthorized
20m         Normal    SuccessfulCreate    replicaset/nginx-replicaset        Created pod: nginx-replicaset-xvdvb
20m         Normal    SuccessfulCreate    replicaset/nginx-replicaset        Created pod: nginx-replicaset-67b4m
20m         Normal    SuccessfulCreate    replicaset/nginx-replicaset        Created pod: nginx-replicaset-q96g5
20m         Normal    SuccessfulCreate    replicaset/nginx-replicaset        Created pod: nginx-replicaset-wgj45
20m         Normal    SuccessfulCreate    replicaset/nginx-replicaset        Created pod: nginx-replicaset-b76tj
18s         Normal    SuccessfulCreate    replicaset/nginx-replicaset        Created pod: nginx-replicaset-sn82c
18s         Normal    SuccessfulCreate    replicaset/nginx-replicaset        Created pod: nginx-replicaset-7cj4z
18s         Normal    SuccessfulCreate    replicaset/nginx-replicaset        Created pod: nginx-replicaset-zbq6w
18s         Warning   FailedCreate        replicaset/nginx-replicaset        Error creating: pods "nginx-replicaset-f4c5d" is forbidden: exceeded quota: myns-quot
a, requested: pods=1, used: pods=10, limited: pods=10
18s         Warning   FailedCreate        replicaset/nginx-replicaset        Error creating: pods "nginx-replicaset-r8s7m" is forbidden: exceeded quota: myns-quot
a, requested: pods=1, used: pods=10, limited: pods=10
18s         Normal    SuccessfulCreate    replicaset/nginx-replicaset        Created pod: nginx-replicaset-26zdk
18s         Normal    SuccessfulCreate    replicaset/nginx-replicaset        Created pod: nginx-replicaset-75tjx
18s         Warning   FailedCreate        replicaset/nginx-replicaset        Error creating: pods "nginx-replicaset-kh9vg" is forbidden: exceeded quota: myns-quot
a, requested: pods=1, used: pods=10, limited: pods=10
18s         Warning   FailedCreate        replicaset/nginx-replicaset        Error creating: pods "nginx-replicaset-tgscs" is forbidden: exceeded quota: myns-quot
a, requested: pods=1, used: pods=10, limited: pods=10
18s         Warning   FailedCreate        replicaset/nginx-replicaset        Error creating: pods "nginx-replicaset-99bvs" is forbidden: exceeded quota: myns-quot
a, requested: pods=1, used: pods=10, limited: pods=10
18s         Warning   FailedCreate        replicaset/nginx-replicaset        Error creating: pods "nginx-replicaset-b9dpk" is forbidden: exceeded quota: myns-quot
a, requested: pods=1, used: pods=10, limited: pods=10
18s         Warning   FailedCreate        replicaset/nginx-replicaset        Error creating: pods "nginx-replicaset-bpns4" is forbidden: exceeded quota: myns-quot
a, requested: pods=1, used: pods=10, limited: pods=10
18s         Warning   FailedCreate        replicaset/nginx-replicaset        Error creating: pods "nginx-replicaset-qz5gk" is forbidden: exceeded quota: myns-quot
a, requested: pods=1, used: pods=10, limited: pods=10
17s         Warning   FailedCreate        replicaset/nginx-replicaset        Error creating: pods "nginx-replicaset-8w82w" is forbidden: exceeded quota: myns-quot
a, requested: pods=1, used: pods=10, limited: pods=10
1s          Warning   FailedCreate        replicaset/nginx-replicaset        (combined from similar events): Error creating: pods "nginx-replicaset-lpcrn" is forb
idden: exceeded quota: myns-quota, requested: pods=1, used: pods=10, limited: pods=10
PS C:\Users\namit>

```

## 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\namit> kubectl delete -f nginx-replicaset-quota.yaml
replicaset.apps "nginx-replicaset" deleted
PS C:\Users\namit> 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\namit> kubectl delete -f resource-quota.yaml
resourcequota "myns-quota" deleted
PS C:\Users\namit> kubectl delete namespace myns
namespace "myns" deleted
PS C:\Users\namit>
```