Lab Exercise 10- 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: quota-example    # The name of the namespace.
```

quota-namespace.yaml

```yaml
apiVersion: v1

kind: Namespace

metadata:

  name: quota-example    # The name of the namespace.
```

Apply the YAML to create the namespace:

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

```
C:\Users\Vibhav Khaneja\OneDrive\Desktop\K8S>kubectl apply -f quota-namespace.yaml
namespace/quota-example created

C:\Users\Vibhav Khaneja\OneDrive\Desktop\K8S>
```

Verify that the namespace is created:

```
kubectl get namespaces
```

```
C:\Users\Vibhav Khaneja\OneDrive\Desktop\K8S>kubectl get namespaces
NAME              STATUS   AGE
default           Active   4d21h
kube-node-lease   Active   4d21h
kube-public       Active   4d21h
kube-system       Active   4d21h
quota-example     Active   47s

C:\Users\Vibhav Khaneja\OneDrive\Desktop\K8S>
```

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: example-quota    # The name of the Resource Quota.
  namespace: quota-example # 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.
```

resource-quota.yaml

```yaml
apiVersion: v1

kind: ResourceQuota

metadata:

  name: example-quota    # The name of the Resource Quota.

  namespace: quota-example # 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
```

```
C:\Users\Vibhav Khaneja\OneDrive\Desktop\K8S>kubectl apply -f resource-quota.yaml
resourcequota/example-quota created

C:\Users\Vibhav Khaneja\OneDrive\Desktop\K8S>
```

Verify that the Resource Quota is applied:

```
kubectl get resourcequota -n quota-example
```

```
C:\Users\Vibhav Khaneja\OneDrive\Desktop\K8S>kubectl get resourcequota -n quota-example
NAME            AGE   REQUEST
                      LIMIT
example-quota   28s   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

C:\Users\Vibhav Khaneja\OneDrive\Desktop\K8S>
```

To see the details of the applied Resource Quota:

```
kubectl describe resourcequota example-quota -n quota-example
```

```
C:\Users\Vibhav Khaneja\OneDrive\Desktop\K8S>kubectl describe resourcequota example-quota -n quota-example
Name:                   example-quota
Namespace:              quota-example
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

C:\Users\Vibhav Khaneja\OneDrive\Desktop\K8S>
```

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: quota-example
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: quota-example

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:
```

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
```

```
C:\Users\Vibhav Khaneja\OneDrive\Desktop\K8S>kubectl apply -f nginx-replicaset-quota.yaml
replicaset.apps/nginx-replicaset created

C:\Users\Vibhav Khaneja\OneDrive\Desktop\K8S>
```

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

```
kubectl get pods -n quota-example
```

```
C:\Users\Vibhav Khaneja\OneDrive\Desktop\K8S>kubectl get pods -n quota-example
NAME                     READY   STATUS    RESTARTS   AGE
nginx-replicaset-6ks8c   1/1     Running   0          28s
nginx-replicaset-lrrnf   1/1     Running   0          28s
nginx-replicaset-ptpjk   1/1     Running   0          28s
nginx-replicaset-pzpt4   1/1     Running   0          28s
nginx-replicaset-r79l8   1/1     Running   0          28s

C:\Users\Vibhav Khaneja\OneDrive\Desktop\K8S>
```

To describe the Pods and see their resource allocations:

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

```
C:\Users\Vibhav Khaneja\OneDrive\Desktop\K8S>kubectl describe pods -l app=nginx -n quota-example
Name:             nginx-replicaset-6ks8c
Namespace:        quota-example
Priority:         0
Service Account:  default
Node:             docker-desktop/192.168.65.3
Start Time:       Tue, 12 Nov 2024 13:16:43 +0530
Labels:           app=nginx
Annotations:      <none>
Status:           Running
IP:               10.1.0.22
IPs:
  IP:           10.1.0.22
Controlled By:  ReplicaSet/nginx-replicaset
Containers:
  nginx:
    Container ID:   docker://358c5675ae1ca55fc1b890e51434a2150be47c3af3f80e183118206931f949cb
    Image:          nginx:latest
    Image ID:       docker-pullable://nginx@sha256:bc5eac5eafc581aeda3008b4b1f07ebba230de2f27d47767129a6a905c84f470
    Port:           80/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Tue, 12 Nov 2024 13:17:02 +0530
    Ready:          True
    Restart Count:  0
    Limits:
      cpu:     200m
      memory:  200Mi
    Requests:
```

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: quota-example
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"
```

nginx-extra-pod.yaml

```yaml
apiVersion: v1

kind: Pod

metadata:

  name: nginx-extra-pod

  namespace: quota-example

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
```

```
C:\Users\Vibhav Khaneja\OneDrive\Desktop\K8S>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: example-quota, requested: requests.cpu=2, used: requests.cpu=500m, limited: requests.cpu=2

C:\Users\Vibhav Khaneja\OneDrive\Desktop\K8S>
```

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

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

```
C:\Users\Vibhav Khaneja\OneDrive\Desktop\K8S>kubectl get events -n quota-example
LAST SEEN   TYPE     REASON            OBJECT                          MESSAGE
4m52s       Normal   Scheduled         pod/nginx-replicaset-6ks8c    Successfully assigned quota-example/nginx-replicaset-6ks8c to docker-deskt
op
4m49s       Normal   Pulling           pod/nginx-replicaset-6ks8c    Pulling image "nginx:latest"
4m35s       Normal   Pulled            pod/nginx-replicaset-6ks8c    Successfully pulled image "nginx:latest" in 4.983s (13.878s including wait
ing). Image size: 191670156 bytes.
4m34s       Normal   Created           pod/nginx-replicaset-6ks8c    Created container nginx
4m34s       Normal   Started           pod/nginx-replicaset-6ks8c    Started container nginx
4m52s       Normal   Scheduled         pod/nginx-replicaset-lrrnf    Successfully assigned quota-example/nginx-replicaset-lrrnf to docker-deskt
op
4m49s       Normal   Pulling           pod/nginx-replicaset-lrrnf    Pulling image "nginx:latest"
4m40s       Normal   Pulled            pod/nginx-replicaset-lrrnf    Successfully pulled image "nginx:latest" in 3.048s (8.894s including waiti
ng). Image size: 191670156 bytes.
4m39s       Normal   Created           pod/nginx-replicaset-lrrnf    Created container nginx
4m39s       Normal   Started           pod/nginx-replicaset-lrrnf    Started container nginx
4m52s       Normal   Scheduled         pod/nginx-replicaset-ptpjk    Successfully assigned quota-example/nginx-replicaset-ptpjk to docker-deskt
op
4m49s       Normal   Pulling           pod/nginx-replicaset-ptpjk    Pulling image "nginx:latest"
4m43s       Normal   Pulled            pod/nginx-replicaset-ptpjk    Successfully pulled image "nginx:latest" in 2.761s (5.846s including waiti
ng). Image size: 191670156 bytes.
4m43s       Normal   Created           pod/nginx-replicaset-ptpjk    Created container nginx
4m42s       Normal   Started           pod/nginx-replicaset-ptpjk    Started container nginx
4m52s       Normal   Scheduled         pod/nginx-replicaset-pzpt4    Successfully assigned quota-example/nginx-replicaset-pzpt4 to docker-deskt
op
4m49s       Normal   Pulling           pod/nginx-replicaset-pzpt4    Pulling image "nginx:latest"
4m31s       Normal   Pulled            pod/nginx-replicaset-pzpt4    Successfully pulled image "nginx:latest" in 4.232s (18.074s including wait
ing). Image size: 191670156 bytes.
4m29s       Normal   Created           pod/nginx-replicaset-pzpt4    Created container nginx
4m28s       Normal   Started           pod/nginx-replicaset-pzpt4    Started container nginx
4m52s       Normal   Scheduled         pod/nginx-replicaset-r79l8    Successfully assigned quota-example/nginx-replicaset-r79l8 to docker-deskt
```

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 quota-example
```

```
C:\Users\Vibhav Khaneja\OneDrive\Desktop\K8S>kubectl delete -f nginx-replicaset-quota.yaml
replicaset.apps "nginx-replicaset" deleted

C:\Users\Vibhav Khaneja\OneDrive\Desktop\K8S>
```

```
C:\Users\Vibhav Khaneja\OneDrive\Desktop\K8S>kubectl delete -f resource-quota.yaml
resourcequota "example-quota" deleted

C:\Users\Vibhav Khaneja\OneDrive\Desktop\K8S>kubectl delete namespace quota-example
namespace "quota-example" deleted
```

```
C:\Users\Vibhav Khaneja\OneDrive\Desktop\K8S>kubectl get ns
NAME              STATUS   AGE
default           Active   4d22h
kube-node-lease   Active   4d22h
kube-public       Active   4d22h
kube-system       Active   4d22h

C:\Users\Vibhav Khaneja\OneDrive\Desktop\K8S>
```