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:

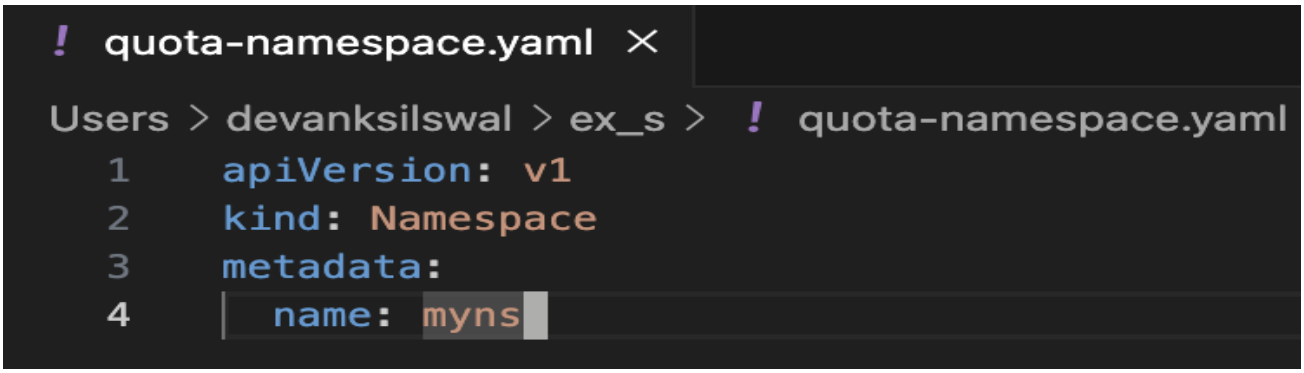
- Limit the amount of CPU and memory a namespace can use.
- Control the number of certain types of resources (e.g., Pods, Services, PersistentVolumeClaims) in a namespace.
- Prevent a namespace from consuming more resources than allocated, ensuring fair usage across multiple teams or applications.

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
```

```
devanksilswal@devanks-MacBook-Air ex_s % touch quota-namespace.yaml
```



Apply the YAML to create the namespace:

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

```
devanksilswal@devanks-MacBook-Air ex_s % kubectl apply -f quota-namespace.yaml
namespace/myns created
```

Verify that the namespace is created:

```
kubectl get namespaces
```

```
devanksilswal@devanks-MacBook-Air ex_s % kubectl get namespaces
NAME                   STATUS   AGE
default                Active   176m
kube-node-lease        Active   176m
kube-public            Active   176m
kube-system            Active   176m
kubernetes-dashboard   Active   170m
myns                   Active   5s
```

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.
```

```
devanksilswal@devanks-MacBook-Air ex_s % touch resource-quota.yaml
```

```
Users > devanksilswal > ex_s > ! resource-quota.yaml
1   apiVersion: v1
2   kind: ResourceQuota
3   metadata:
4     name: myns-quota
5     namespace: myns
6   spec:
7     hard:
8       requests.cpu: "2"
9       requests.memory: "4Gi"
10      limits.cpu: "4"
11      limits.memory: "8Gi"
12      pods: "10"
13      persistentvolumeclaims: "5"
14      configmaps: "10"
15      services: "5"
```

Step 4: Apply the Resource Quota

Apply the Resource Quota YAML to the namespace:

kubectl apply -f resource-quota.yaml

```
devanksilswal@devanks-MacBook-Air ex_s % kubectl apply -f resource-quota.yaml
resourcequota/myns-quota created
```

Verify that the Resource Quota is applied:

kubectl get resourcequota -n myns

```
devanksilswal@devanks-MacBook-Air ex_s % kubectl get resourcequota -n myns
NAME         REQUEST
           LIMIT                                  AGE
myns-quota   configmaps: 1/10, persistentvolumeclaims: 0/5, pods: 0/10, requests.cpu: 0/2, requests.memory: 0/4Gi, serv
ices: 0/5   limits.cpu: 0/4, limits.memory: 0/8Gi   5s
```

To see the details of the applied Resource Quota:

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

```
[devanksilswal@devanks-MacBook-Air ex_s % 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.

```
devanksilswal@devanks-MacBook-Air ex_s % touch nginx-replicaset-quota.yaml
```

Users > devanksilswal > ex_s > ! 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"
```

Apply this YAML to create the ReplicaSet:

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

```
devanksilswal@devanks-MacBook-Air ex_s % kubectl apply -f nginx-replicaset-quota.yaml
replicaset.apps/nginx-replicaset created
```

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

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

```
devanksilswal@devanks-MacBook-Air ex_s % kubectl get pods -n myns
NAME                     READY   STATUS    RESTARTS   AGE
nginx-replicaset-29rh8   1/1     Running   0          27s
nginx-replicaset-5njrh   1/1     Running   0          27s
nginx-replicaset-hh47m   1/1     Running   0          27s
nginx-replicaset-p4xlw   1/1     Running   0          27s
nginx-replicaset-zg7s8   1/1     Running   0          27s
```

To describe the Pods and see their resource allocations:

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

```
devanksilswal@devanks-MacBook-Air ex_s % kubectl describe pods -l app=nginx -n myns
Name:             nginx-replicaset-29rh8
Namespace:        myns
Priority:         0
Service Account:  default
Node:             docker-desktop/192.168.65.3
Start Time:       Tue, 24 Feb 2026 22:29:49 +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://f97e6d1633b4f84cbbb59b3302c0354da00699d5d0121e4bf726db64ff0ae829
    Image:          nginx:latest
    Image ID:       docker-pullable://nginx@sha256:341bf0f3ce6c5277d6002cf6e1fb0319fa4252add24ab6a0e262e0056d313208
    Port:           80/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Tue, 24 Feb 2026 22:29:54 +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-xw5sd (ro)
Conditions:
  Type                        Status
  PodReadyToStartContainers   True
  Initialized                 True
  Ready                       True
  ContainersReady             True
  PodScheduled                True
Volumes:
  kube-api-access-xw5sd:
    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  38s   default-scheduler  Successfully assigned myns/nginx-replicaset-29rh8 to docker-desktop
  Normal  Pulling    38s   kubelet            spec.containers{nginx}: Pulling image "nginx:latest"
  Normal  Pulled     33s   kubelet            spec.containers{nginx}: Successfully pulled image "nginx:latest" in 4.544s (4.544s including waiting). Image size: 61268012 bytes.
  Normal  Created    33s   kubelet            spec.containers{nginx}: Created container: nginx
  Normal  Started    33s   kubelet            spec.containers{nginx}: Started container nginx


Name:             nginx-replicaset-5njrh
Namespace:        myns
Priority:         0
Service Account:  default
Node:             docker-desktop/192.168.65.3
Start Time:       Tue, 24 Feb 2026 22:29:49 +0530
Labels:           app=nginx
Annotations:      <none>
```

```
Status:           Running
IP:               10.1.0.25
IPs:
  IP:           10.1.0.25
Controlled By:  ReplicaSet/nginx-replicaset
Containers:
  nginx:
    Container ID:   docker://e21c08a060b67358eeb17f47b7736902af0f28022ef13865f5214e69d262b795
    Image:          nginx:latest
    Image ID:       docker-pullable://nginx@sha256:341bf0f3ce6c5277d6002cf6e1fb0319fa4252add24ab6a0e262e0056d313208
    Port:           80/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Tue, 24 Feb 2026 22:30:13 +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-x6rsk (ro)
Conditions:
  Type                        Status
  PodReadyToStartContainers   True
  Initialized                 True
  Ready                       True
  ContainersReady             True
  PodScheduled                True
Volumes:
  kube-api-access-x6rsk:
    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  38s   default-scheduler  Successfully assigned myns/nginx-replicaset-5njrh to docker-desktop
  Normal  Pulling    38s   kubelet            spec.containers{nginx}: Pulling image "nginx:latest"
  Normal  Pulled     14s   kubelet            spec.containers{nginx}: Successfully pulled image "nginx:latest" in 2.869s (24.309s including waiting). Image size: 61268012 bytes.
  Normal  Created    14s   kubelet            spec.containers{nginx}: Created container: nginx
  Normal  Started    14s   kubelet            spec.containers{nginx}: Started container nginx


Name:             nginx-replicaset-hh47m
Namespace:        myns
Priority:         0
Service Account:  default
Node:             docker-desktop/192.168.65.3
Start Time:       Tue, 24 Feb 2026 22:29:49 +0530
Labels:           app=nginx
Annotations:      <none>
Status:           Running
IP:               10.1.0.21
IPs:
  IP:           10.1.0.21
Controlled By:  ReplicaSet/nginx-replicaset
Containers:
  nginx:
    Container ID:   docker://fd74ed2b400f6fbbe454d99cdf0c88b35b199ca8b7302c525a1f8bce6ff44013
    Image:          nginx:latest
```

```
    Image ID:       docker-pullable://nginx@sha256:341bf0f3ce6c5277d6002cf6e1fb0319fa4252add24ab6a0e262e0056d313208
    Port:           80/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Tue, 24 Feb 2026 22:30:00 +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-zwnp5 (ro)
Conditions:
  Type                        Status
  PodReadyToStartContainers   True
  Initialized                 True
  Ready                       True
  ContainersReady             True
  PodScheduled                True
Volumes:
  kube-api-access-zwnp5:
    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  38s   default-scheduler  Successfully assigned myns/nginx-replicaset-hh47m to docker-desktop
  Normal  Pulling    38s   kubelet            spec.containers{nginx}: Pulling image "nginx:latest"
  Normal  Pulled     27s   kubelet            spec.containers{nginx}: Successfully pulled image "nginx:latest" in 6.166s (10.707s including waiting). Image size: 61268012 bytes.
  Normal  Created    27s   kubelet            spec.containers{nginx}: Created container: nginx
  Normal  Started    27s   kubelet            spec.containers{nginx}: Started container nginx


Name:             nginx-replicaset-p4xlw
Namespace:        myns
Priority:         0
Service Account:  default
Node:             docker-desktop/192.168.65.3
Start Time:       Tue, 24 Feb 2026 22:29:49 +0530
Labels:           app=nginx
Annotations:      <none>
Status:           Running
IP:               10.1.0.23
IPs:
  IP:           10.1.0.23
Controlled By:  ReplicaSet/nginx-replicaset
Containers:
  nginx:
    Container ID:   docker://8380cbaa13da448666d6d6c00227f138b6ca558bc2b9ba00fbddae5fa3d17b42
    Image:          nginx:latest
    Image ID:       docker-pullable://nginx@sha256:341bf0f3ce6c5277d6002cf6e1fb0319fa4252add24ab6a0e262e0056d313208
    Port:           80/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Tue, 24 Feb 2026 22:30:06 +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-zbs8f (ro)
Conditions:
  Type                        Status
  PodReadyToStartContainers   True
  Initialized                 True
  Ready                       True
  ContainersReady             True
  PodScheduled                True
Volumes:
  kube-api-access-zbs8f:
    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  38s   default-scheduler  Successfully assigned myns/nginx-replicaset-p4x1w to docker-desktop
  Normal  Pulling    38s   kubelet            spec.containers{nginx}: Pulling image "nginx:latest"
  Normal  Pulled     21s   kubelet            spec.containers{nginx}: Successfully pulled image "nginx:latest" in 6.667s (17.364s including waiting). Image size: 61268012 bytes.
  Normal  Created    21s   kubelet            spec.containers{nginx}: Created container: nginx
  Normal  Started    21s   kubelet            spec.containers{nginx}: Started container nginx


Name:             nginx-replicaset-zg7s8
Namespace:        myns
Priority:         0
Service Account:  default
Node:             docker-desktop/192.168.65.3
Start Time:       Tue, 24 Feb 2026 22:29:49 +0530
Labels:           app=nginx
Annotations:      <none>
Status:           Running
IP:               10.1.0.24
IPs:
  IP:           10.1.0.24
Controlled By:  ReplicaSet/nginx-replicaset
Containers:
  nginx:
    Container ID:   docker://d5f6cb9d544ce7236297efce71e505521a79bd6c918c77746acc2d082bd7002f
    Image:          nginx:latest
    Image ID:       docker-pullable://nginx@sha256:341bf0f3ce6c5277d6002cf6e1fb0319fa4252add24ab6a0e262e0056d313208
    Port:           80/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Tue, 24 Feb 2026 22:30:11 +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-4mkf2 (ro)
Conditions:
  Type                        Status
```

```
  Type                        Status
  PodReadyToStartContainers   True
  Initialized                 True
  Ready                       True
  ContainersReady             True
  PodScheduled                True
Volumes:
  kube-api-access-4mkf2:
    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  38s   default-scheduler  Successfully assigned myns/nginx-replicaset-zg7s8 to docker-desktop
  Normal  Pulling    38s   kubelet            spec.containers{nginx}: Pulling image "nginx:latest"
  Normal  Pulled     17s   kubelet            spec.containers{nginx}: Successfully pulled image "nginx:latest" in 4.089s (21.452s including waiting). Image size: 61268012 bytes.
  Normal  Created    17s   kubelet            spec.containers{nginx}: Created container: nginx
  Normal  Started    16s   kubelet            spec.containers{nginx}: Started container nginx
```

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"
```

```
devanksilswal@devanks-MacBook-Air ex_s % touch nginx-extra-pod.yaml
```

Users > devanksilswal > ex_s > ! 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"
        cpu: "2"
      limits:
        memory: "4Gi"
        cpu: "2"
```

Apply this YAML to create the Pod:

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

```
devanksilswal@devanks-MacBook-Air ex_s % kubectl apply -f nginx-extra-pod.yaml
Error from server (Forbidden): error when creating "nginx-extra-pod.yaml": pods "nginx-extra-pod" is forbidden: exceede
d quota: myns-quota, requested: 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 myns
```

```
devanksilswal@devanks-MacBook-Air ex_s % kubectl get events -n myns
LAST SEEN   TYPE     REASON            OBJECT                          MESSAGE
111s        Normal   Scheduled         pod/nginx-replicaset-29rh8      Successfully assigned myns/nginx-replicaset-29rh8 to docker-desktop
111s        Normal   Pulling           pod/nginx-replicaset-29rh8      Pulling image "nginx:latest"
106s        Normal   Pulled            pod/nginx-replicaset-29rh8      Successfully pulled image "nginx:latest" in 4.544s (4.544s including waiting). Image size: 61268012 bytes.
106s        Normal   Created           pod/nginx-replicaset-29rh8      Created container: nginx
106s        Normal   Started           pod/nginx-replicaset-29rh8      Started container nginx
111s        Normal   Scheduled         pod/nginx-replicaset-5njrh      Successfully assigned myns/nginx-replicaset-5njrh to docker-desktop
111s        Normal   Pulling           pod/nginx-replicaset-5njrh      Pulling image "nginx:latest"
87s         Normal   Pulled            pod/nginx-replicaset-5njrh      Successfully pulled image "nginx:latest" in 2.869s (24.309s including waiting). Image size: 61268012 bytes.
87s         Normal   Created           pod/nginx-replicaset-5njrh      Created container: nginx
87s         Normal   Started           pod/nginx-replicaset-5njrh      Started container nginx
111s        Normal   Scheduled         pod/nginx-replicaset-hh47m      Successfully assigned myns/nginx-replicaset-hh47m to docker-desktop
111s        Normal   Pulling           pod/nginx-replicaset-hh47m      Pulling image "nginx:latest"
100s        Normal   Pulled            pod/nginx-replicaset-hh47m      Successfully pulled image "nginx:latest" in 6.166s (10.707s including waiting). Image size: 61268012 bytes.
100s        Normal   Created           pod/nginx-replicaset-hh47m      Created container: nginx
100s        Normal   Started           pod/nginx-replicaset-hh47m      Started container nginx
111s        Normal   Scheduled         pod/nginx-replicaset-p4xlw      Successfully assigned myns/nginx-replicaset-p4xlw to docker-desktop
111s        Normal   Pulling           pod/nginx-replicaset-p4xlw      Pulling image "nginx:latest"
94s         Normal   Pulled            pod/nginx-replicaset-p4xlw      Successfully pulled image "nginx:latest" in 6.667s (17.364s including waiting). Image size: 61268012 bytes.
94s         Normal   Created           pod/nginx-replicaset-p4xlw      Created container: nginx
94s         Normal   Started           pod/nginx-replicaset-p4xlw      Started container nginx
111s        Normal   Scheduled         pod/nginx-replicaset-zg7s8      Successfully assigned myns/nginx-replicaset-zg7s8 to docker-desktop
111s        Normal   Pulling           pod/nginx-replicaset-zg7s8      Pulling image "nginx:latest"
90s         Normal   Pulled            pod/nginx-replicaset-zg7s8      Successfully pulled image "nginx:latest" in 4.089s (21.452s including waiting). Image size: 61268012 bytes.
90s         Normal   Created           pod/nginx-replicaset-zg7s8      Created container: nginx
89s         Normal   Started           pod/nginx-replicaset-zg7s8      Started container nginx
111s        Normal   SuccessfulCreate  replicaset/nginx-replicaset     Created pod: nginx-replicaset-zg7s8
111s        Normal   SuccessfulCreate  replicaset/nginx-replicaset     Created pod: nginx-replicaset-hh47m
111s        Normal   SuccessfulCreate  replicaset/nginx-replicaset     Created pod: nginx-replicaset-p4xlw
111s        Normal   SuccessfulCreate  replicaset/nginx-replicaset     Created pod: nginx-replicaset-29rh8
111s        Normal   SuccessfulCreate  replicaset/nginx-replicaset     Created pod: nginx-replicaset-5njrh
```

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
```

```
devanksilswal@devanks-MacBook-Air ex_s % kubectl delete -f nginx-replicaset-quota.yaml
kubectl delete -f nginx-extra-pod.yaml
kubectl delete -f resource-quota.yaml
kubectl delete namespace myns
replicaset.apps "nginx-replicaset" deleted from myns namespace
Error from server (NotFound): error when deleting "nginx-extra-pod.yaml": pods "nginx-extra-pod" not found
resourcequota "myns-quota" deleted from myns namespace
namespace "myns" deleted
```

```
devanksilswal@devanks-MacBook-Air ex_s % kubectl get namespaces
NAME                   STATUS   AGE
default                Active   3h2m
kube-node-lease        Active   3h2m
kube-public            Active   3h2m
kube-system            Active   3h2m
kubernetes-dashboard   Active   176m
```