Name: Simranjeet kour

Sap Id: 500110713

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:







Apply the YAML to create the namespace:





Verify that the namespace is created:





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:





# Step 4: Apply the Resource Quota

Apply the Resource Quota YAML to the namespace:





Verify that the Resource Quota is applied:





To see the details of the applied Resource Quota:





# 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:





# 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:





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





To describe the Pods and see their resource allocations:





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:





Apply this YAML to create the Pod:





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



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:

