

# Lab Exercise 8- Creating and Managing a ReplicaSet in Kubernetes

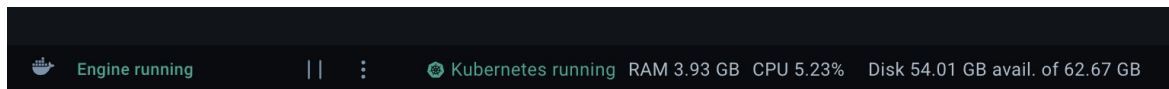
## Objective:

A ReplicaSet in Kubernetes ensures a specified number of Pod replicas are running at any given time. This exercise will guide you through creating a ReplicaSet to maintain the desired state of your application.

- Understand the syntax and structure of a Kubernetes ReplicaSet definition file (YAML).
- Learn how to create and manage a ReplicaSet to ensure application availability.
- Understand how a ReplicaSet helps in scaling applications and maintaining desired states.

## Prerequisites

- Kubernetes Cluster: Have a running Kubernetes cluster (locally using Minikube or kind, or a cloud-based service).
- kubectl: Install and configure kubectl to interact with your Kubernetes cluster.
- Basic Knowledge of YAML: Familiarity with YAML format will be helpful for understanding Kubernetes resource definitions.



<input type="checkbox"/>	Name	Container ID	Image	Port(s)	CPU (%)	Last started	Actions
<input type="checkbox"/>	minikube	dfe16e2d66cf	k8s-minikube/kicb...	50951:22 Show all ports (5)	14.38%	26 minutes ago	

## Step-by-Step Guide

### Step 1: Understanding ReplicaSet

A ReplicaSet ensures a specified number of Pod replicas are running at any given time. If a Pod crashes or is deleted, the ReplicaSet creates a new one to meet the defined number of replicas. This helps maintain application availability and ensures that your application can handle increased load by distributing traffic among multiple Pods.

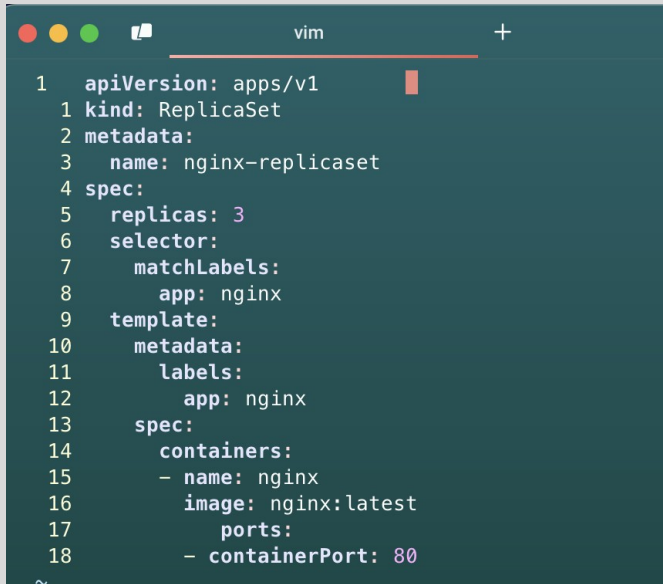
### Step 2: Create a ReplicaSet

We'll define a ReplicaSet to maintain three replicas of a simple Nginx web server Pod. Create a YAML file named `nginx-replicaset.yaml` with the following content:

```
apiVersion: apps/v1    # Specifies the API version used.
kind: ReplicaSet        # The type of resource being defined; here, it's a ReplicaSet.
metadata:
  name: nginx-replicaset # The name of the ReplicaSet.
spec:
  replicas: 3           # The desired number of Pod replicas.
  selector:
    matchLabels:        # Criteria to identify Pods managed by this ReplicaSet.
      app: nginx        # The label that should match Pods.
  template:             # The Pod template for creating new Pods.
    metadata:
      labels:
        app: nginx      # Labels applied to Pods created by this ReplicaSet.
    spec:
      containers:
        - name: nginx    # Name of the container within the Pod.
          image: nginx:latest # Docker image to use for the container.
```

ports:

- containerPort: 80 # The port the container exposes.



```
1  apiVersion: apps/v1
1  kind: ReplicaSet
2  metadata:
3    name: nginx-replicaset
4  spec:
5    replicas: 3
6    selector:
7      matchLabels:
8        app: nginx
9    template:
10     metadata:
11       labels:
12         app: nginx
13     spec:
14       containers:
15       - name: nginx
16         image: nginx:latest
17         ports:
18         - containerPort: 80
```

## Explanation:


- apiVersion: Defines the API version (apps/v1) used for the ReplicaSet resource.
- kind: Specifies that this resource is a ReplicaSet.
- metadata: Contains metadata about the ReplicaSet, including name.
  - name: The unique name for the ReplicaSet.
- spec: Provides the specification for the ReplicaSet.
  - replicas: Defines the desired number of Pod replicas.
  - selector: Criteria for selecting Pods managed by this ReplicaSet.
    - matchLabels: Labels that Pods must have to be managed by this ReplicaSet.
  - template: Defines the Pod template used for creating new Pods.
    - metadata: Contains metadata for the Pods, including labels.
      - labels: Labels applied to Pods created by this ReplicaSet.
    - spec: Specification for the Pods.
      - containers: Lists the containers that will run in the Pod.

- name: The unique name of the container within the Pod.
- image: The Docker image used for the container.
- ports: Ports exposed by the container.

### Step 3: Apply the YAML to Create the ReplicaSet

Use the kubectl apply command to create the ReplicaSet based on the YAML file.

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



A terminal window with a dark green background and light gray text. The prompt is a shell icon followed by a home icon and a tilde. The command `> vim nginx-replicaset.yaml` is entered. The prompt changes to `>` and the command `> kubectl apply -f nginx-replicaset.yaml` is entered. The output is `replicaset.apps/nginx-replicaset created`.

**Verify the ReplicaSet is running and maintaining the desired number of replicas:**

```
kubectl get replicaset
```



A terminal window with a dark green background and light gray text. The prompt is a shell icon followed by a home icon and a tilde. The command `> kubectl get replicaset` is entered. The output is a table with 5 columns: NAME, DESIRED, CURRENT, READY, and AGE. The first row of data is for the replicaset named 'nginx-replicaset'.

NAME	DESIRED	CURRENT	READY	AGE
nginx-replicaset	3	3	3	28s

This command lists all ReplicaSets in the current namespace.

## To check the Pods created by the ReplicaSet:

```
kubectl get pods -l app=nginx
```

```
Apple Home ~ .....
> kubectl get pods -l app=nginx
```

NAME	READY	STATUS	RESTARTS	AGE
nginx-replicaset-8bpfm	1/1	Running	0	59s
nginx-replicaset-dfvxk	1/1	Running	0	59s
nginx-replicaset-fnjcf	1/1	Running	0	59s

This command lists all Pods with the label app=nginx.

## Step 4: Managing the ReplicaSet

### 1. Scaling the ReplicaSet

You can scale the number of replicas managed by the ReplicaSet using the kubectl scale command.

```
kubectl scale --replicas=5 replicaset/nginx-replicaset
```

```
Apple Home ~ .....
> kubectl scale --replicas=5 replicaset/nginx-replicaset
replicaset.apps/nginx-replicaset scaled
```

This command scales the ReplicaSet to maintain 5 replicas. Verify the scaling operation:

```
kubectl get pods -l app=nginx
```

```

  Apple Home ~ .....
> kubectl get pods -l app=nginx

NAME                                READY   STATUS    RESTARTS   AGE
nginx-replicaset-45mln             1/1     Running   0           22s
nginx-replicaset-8bpfm             1/1     Running   0           107s
nginx-replicaset-dfvvk             1/1     Running   0           107s
nginx-replicaset-fnjcf             1/1     Running   0           107s
nginx-replicaset-tcm72             1/1     Running   0           22s

```

You should see that the number of Pods has increased to 5.

## 2. Updating the ReplicaSet

If you need to update the Pod template (e.g., to use a different Docker image version), modify the YAML file and apply it again. For instance, change the image to a specific version of Nginx:

```
spec:
  template:
    spec:
      containers:
      - name: nginx
        image: nginx:1.19.3 # Change to a specific version

```

```
vim
18 apiVersion: apps/v1
17 kind: ReplicaSet
16 metadata:
15   name: nginx-replicaset
14 spec:
13   replicas: 3
12   selector:
11     matchLabels:
10       app: nginx
9   template:
8     metadata:
7       labels:
6         app: nginx
5     spec:
4       containers:
3         - name: nginx
2           image: nginx:1.19.3 # Specific version as mentioned
1           ports:
19         - containerPort: 80
```

**Apply the changes:**

kubectl apply -f nginx-replicaset.yaml

```
vim nginx-replicaset.yaml
> kubectl apply -f nginx-replicaset.yaml
replicaset.apps/nginx-replicaset configured
```

**Check the status to ensure the Pods are updated:**

```
kubectl get pods -l app=nginx
```

```

🍏 🏠 ~ .....
> kubectl get pods -l app=nginx

NAME                                READY   STATUS    RESTARTS   AGE
nginx-replicaset-45mln             1/1     Running   0           8m14s
nginx-replicaset-dfvvk             1/1     Running   0           9m39s
nginx-replicaset-fnjcf             1/1     Running   0           9m39s
```

Note: Updating a ReplicaSet doesn't automatically replace existing Pods with new ones. In practice, you often create a new ReplicaSet or Deployment for updates.

### 3. Deleting the ReplicaSet

To clean up the ReplicaSet and its Pods, use the kubectl delete command:

```
kubectl delete -f nginx-replicaset.yaml
```

```

🍏 🏠 ~ .....
> kubectl delete -f nginx-replicaset.yaml
replicaset.apps "nginx-replicaset" deleted
```

This command deletes the ReplicaSet and all the Pods managed by it.