

Lab Exercise 10 :- Creating and Managing a ReplicaSet in Kubernetes

Name:- Vansh Bhatt

Sap ID:- 500125395

Batch:- DevOps B1

To:- Hitesh Sharma Sir

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.

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
kind: ReplicaSet
metadata:
  name: nginx-replicaset
spec:
  replicas: 3
  selector:
    matchLabels:
      app: vb-app
  template:
    metadata:
      labels:
        app: vb-app
```

spec:

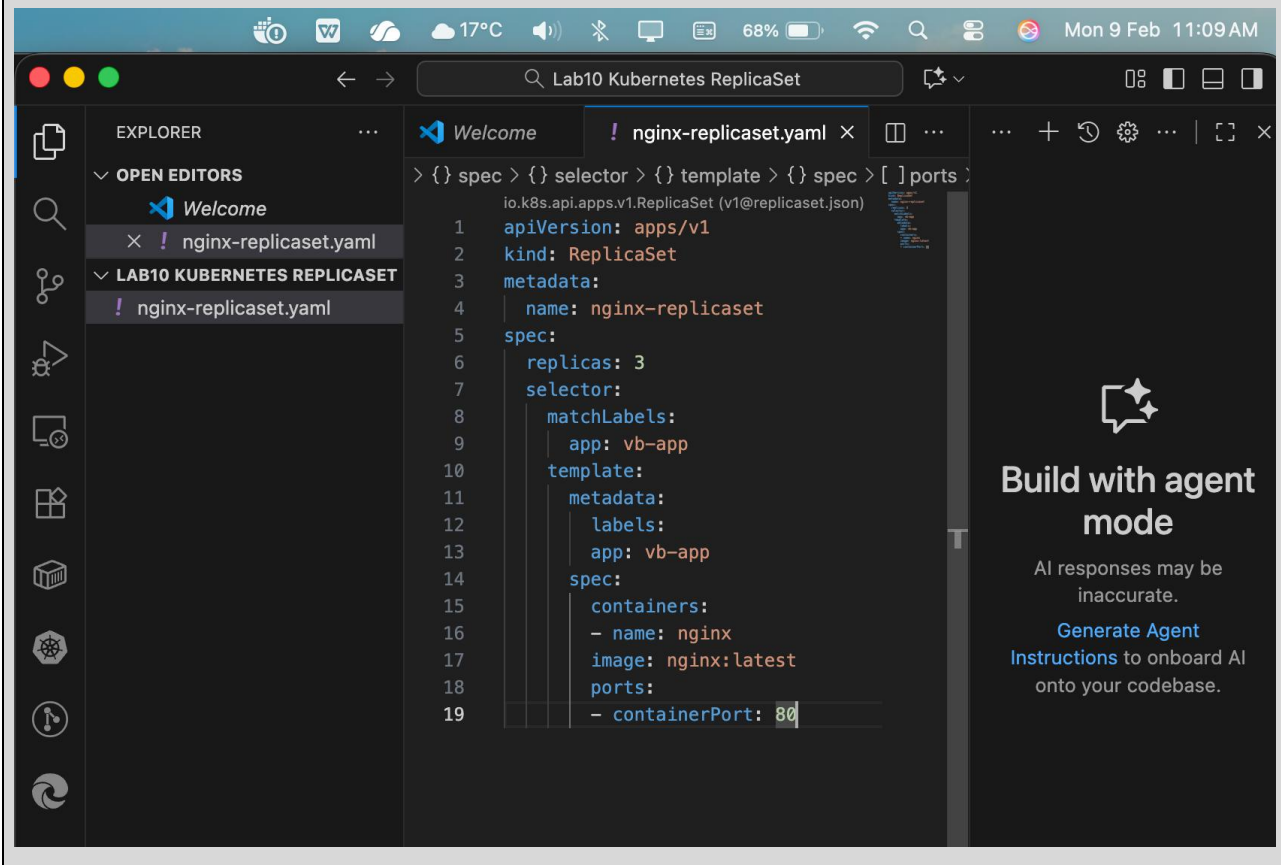
containers:

- name: nginx

image: nginx:latest

ports:

- containerPort: 80



Explanation:

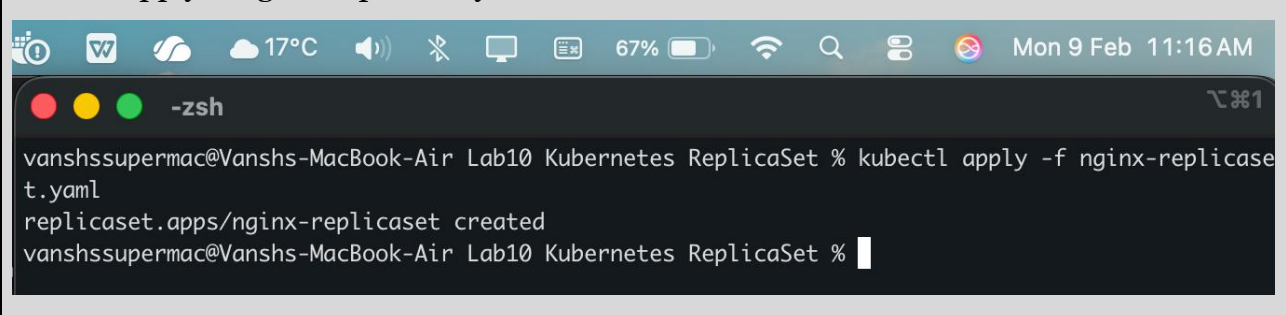
- apiVersion: Defines the API version (apps/v1) used for the ReplicaSet resource.
- kind: Specifies that this resource is a ReplicaSet.
- metadata: Contains metadata about the ReplicaSet, including name.
 - name: The unique name for the ReplicaSet.
- spec: Provides the specification for the ReplicaSet.
 - 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 screenshot of a macOS terminal window. The title bar shows system status icons (Wi-Fi, battery at 67%, temperature at 17°C, etc.) and the date/time 'Mon 9 Feb 11:16 AM'. The terminal window has a dark background and shows the command `kubectl apply -f nginx-replicaset.yaml` being executed. The output is `replicaset.apps/nginx-replicaset created`. The prompt shows the user is in a shell on a MacBook-Air.

```
vanshssupermac@Vanshs-MacBook-Air Lab10 Kubernetes ReplicaSet % kubectl apply -f nginx-replicaset.yaml
replicaset.apps/nginx-replicaset created
vanshssupermac@Vanshs-MacBook-Air Lab10 Kubernetes ReplicaSet %
```

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

```
kubectl get replicaset
```

```
Mon 9 Feb 11:17 AM
-van
vanshssupermac@Vanshs-MacBook-Air Lab10 Kubernetes ReplicaSet % kubectl get replicaset
NAME          DESIRED   CURRENT   READY   AGE
nginx-replicaset 3         3         3       46s
vanshssupermac@Vanshs-MacBook-Air Lab10 Kubernetes ReplicaSet %
```

This command lists all ReplicaSets in the current namespace.

To check the Pods created by the ReplicaSet:

```
Mon 9 Feb 11:18 AM
-kub
kubectl get pods -l app=nginx
vanshssupermac@Vanshs-MacBook-Air Lab10 Kubernetes ReplicaSet % kubectl get pods -l app=nginx
NAME                                READY   STATUS    RESTARTS   AGE
nginx-replicaset-nj7jt             1/1     Running   0           94s
nginx-replicaset-sqm6p             1/1     Running   0           94s
nginx-replicaset-v5qgm             1/1     Running   0           94s
vanshssupermac@Vanshs-MacBook-Air Lab10 Kubernetes ReplicaSet %
```

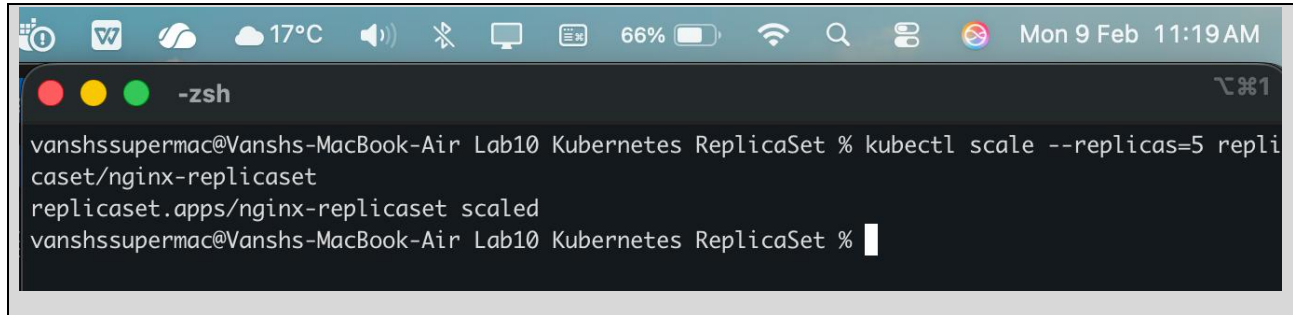
This command lists all Pods with the label app=nginx.

Step 4: Managing the ReplicaSet

1. Scaling the ReplicaSet

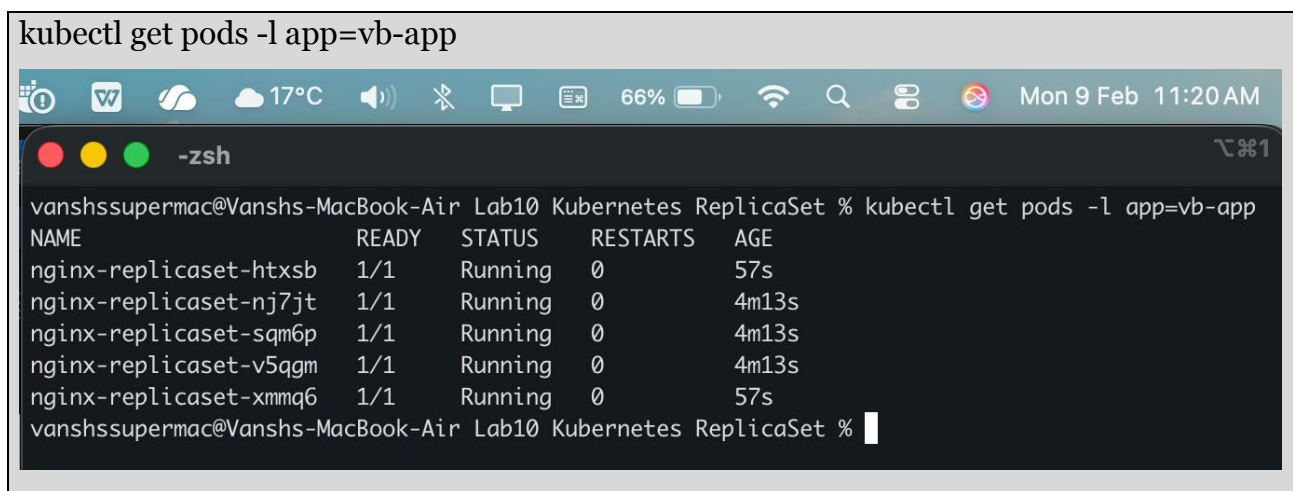
You can scale the number of replicas managed by the ReplicaSet using the `kubectl scale` command.

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

A terminal window on a Mac with a dark background. The title bar shows system icons and the time 'Mon 9 Feb 11:19 AM'. The prompt is 'vanshssupermac@Vanshs-MacBook-Air Lab10 Kubernetes ReplicaSet %'. The user enters 'kubectl scale --replicas=5 replicaset/nginx-replicaset'. The output is 'replicaset.apps/nginx-replicaset scaled'. The prompt returns to 'vanshssupermac@Vanshs-MacBook-Air Lab10 Kubernetes ReplicaSet %'.

```
vanshssupermac@Vanshs-MacBook-Air Lab10 Kubernetes ReplicaSet % kubectl scale --replicas=5 replicaset/nginx-replicaset
replicaset.apps/nginx-replicaset scaled
vanshssupermac@Vanshs-MacBook-Air Lab10 Kubernetes ReplicaSet %
```

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

A terminal window on a Mac with a dark background. The title bar shows system icons and the time 'Mon 9 Feb 11:20 AM'. The prompt is 'vanshssupermac@Vanshs-MacBook-Air Lab10 Kubernetes ReplicaSet %'. The user enters 'kubectl get pods -l app=nginx-app'. The output is a table with 5 columns: NAME, READY, STATUS, RESTARTS, and AGE. There are 5 rows of pod information, all with status 'Running'. The prompt returns to 'vanshssupermac@Vanshs-MacBook-Air Lab10 Kubernetes ReplicaSet %'.

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

vanshssupermac@Vanshs-MacBook-Air Lab10 Kubernetes ReplicaSet % kubectl get pods -l app=nginx-app
NAME                READY   STATUS    RESTARTS   AGE
nginx-replicaset-htxsb 1/1     Running   0          57s
nginx-replicaset-nj7jt 1/1     Running   0          4m13s
nginx-replicaset-sqm6p 1/1     Running   0          4m13s
nginx-replicaset-v5qgm 1/1     Running   0          4m13s
nginx-replicaset-xmmq6 1/1     Running   0          57s
vanshssupermac@Vanshs-MacBook-Air Lab10 Kubernetes ReplicaSet %
```

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

2. Updating the ReplicaSet

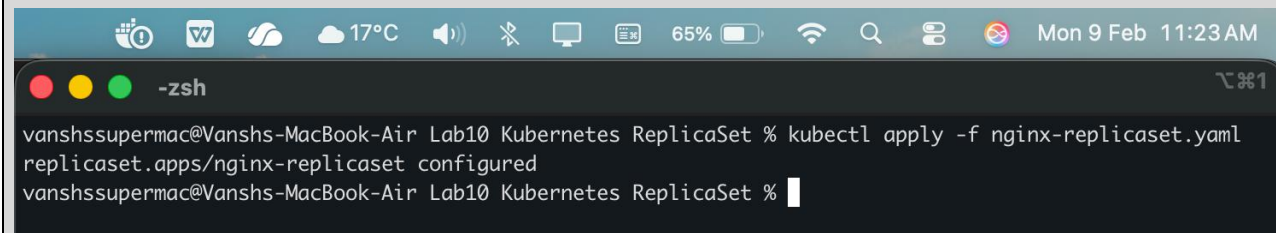
If you need to update the Pod template (e.g., to use a different Docker image version), modify the YAML file and apply it again. For instance, change the image to a specific version of Nginx:

```
spec:
  template:
    spec:
      containers:
      - name: nginx
```

```
image: nginx:1.19.3 # Change to a specific version
```

Apply the changes:

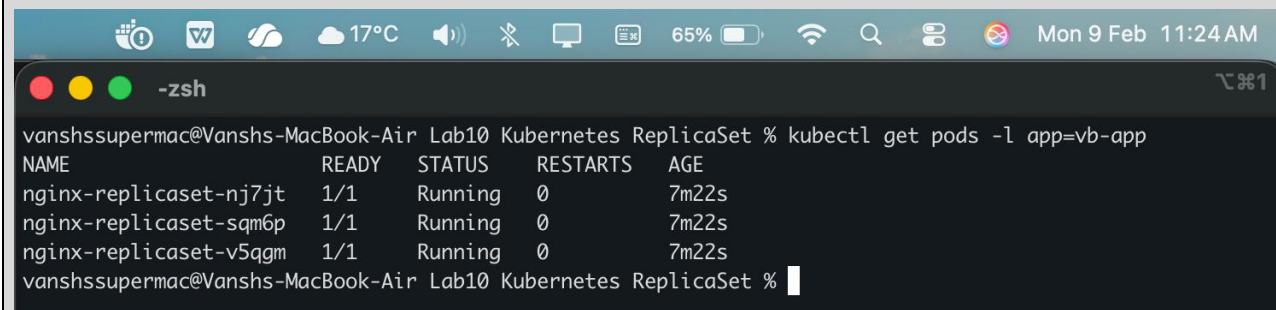
```
kubectl apply -f nginx-replicaset.yaml
```

A terminal window on a MacBook Air showing the command 'kubectl apply -f nginx-replicaset.yaml' being executed. The output indicates that the replicaset.apps/nginx-replicaset has been configured successfully.

```
vanshssupermac@Vanshs-MacBook-Air Lab10 Kubernetes ReplicaSet % kubectl apply -f nginx-replicaset.yaml
replicaset.apps/nginx-replicaset configured
vanshssupermac@Vanshs-MacBook-Air Lab10 Kubernetes ReplicaSet %
```

Check the status to ensure the Pods are updated:

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

A terminal window on a MacBook Air showing the command 'kubectl get pods -l app=vb-app'. The output displays a table with columns for NAME, READY, STATUS, RESTARTS, and AGE, showing three running pods for the nginx-replicaset.

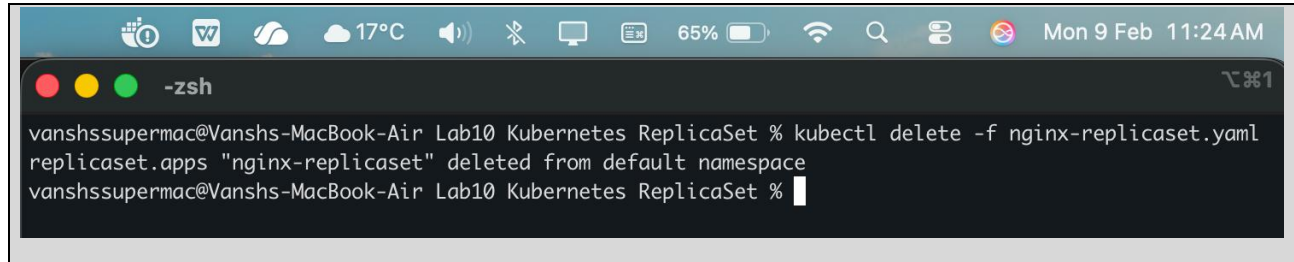
```
vanshssupermac@Vanshs-MacBook-Air Lab10 Kubernetes ReplicaSet % kubectl get pods -l app=vb-app
NAME                READY   STATUS    RESTARTS   AGE
nginx-replicaset-nj7jt 1/1     Running   0           7m22s
nginx-replicaset-sqm6p 1/1     Running   0           7m22s
nginx-replicaset-v5qgm 1/1     Running   0           7m22s
vanshssupermac@Vanshs-MacBook-Air Lab10 Kubernetes ReplicaSet %
```

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
```

A screenshot of a macOS terminal window. The title bar at the top shows standard macOS window controls (red, yellow, green buttons) and the text '-zsh'. The status bar at the very top displays system information: 17°C, 65% battery, and the date/time 'Mon 9 Feb 11:24 AM'. The terminal content shows a user prompt 'vanshssupermac@Vanshs-MacBook-Air Lab10 Kubernetes ReplicaSet %' followed by the command 'kubectl delete -f nginx-replicaset.yaml'. The output of the command is 'replicaset.apps "nginx-replicaset" deleted from default namespace'. The prompt is then repeated: 'vanshssupermac@Vanshs-MacBook-Air Lab10 Kubernetes ReplicaSet %' with a cursor at the end.

```
vanshssupermac@Vanshs-MacBook-Air Lab10 Kubernetes ReplicaSet % kubectl delete -f nginx-replicaset.yaml
replicaset.apps "nginx-replicaset" deleted from default namespace
vanshssupermac@Vanshs-MacBook-Air Lab10 Kubernetes ReplicaSet %
```

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

Thank You