

Lab Exercise 6- Create POD in Kubernetes

Objective:

- Understand the basic structure and syntax of a Kubernetes Pod definition file (YAML).
- Learn to create, inspect, and delete a Pod in a Kubernetes cluster.

Prerequisites

- Kubernetes Cluster: You need a running Kubernetes cluster. You can set up a local cluster using tools like Minikube or kind, or use a cloud-based Kubernetes service.
- kubectl: Install and configure kubectl to interact with your Kubernetes cluster.
- Basic Knowledge of YAML: Familiarity with YAML format will be helpful as Kubernetes resource definitions are written in YAML.

Step-by-Step Guide

Step 1: Create a YAML File for the Pod

We'll create a Pod configuration file named **pod-example.yaml**

```
apiVersion: v1      # The version of the Kubernetes API to use for this object.
kind: Pod           # The type of Kubernetes object. Here it's a Pod.
metadata:           # Metadata about the Pod, such as its name and labels.
  name: my-pod      # The name of the Pod. Must be unique within a namespace.
  labels:            # Labels are key-value pairs to categorize and organize Pods.
    app: my-app      # Label to categorize this Pod as part of 'my-app'.
spec:               # The specification for the Pod, detailing its containers and other settings.
  containers:        # List of containers that will run in this Pod.
    - name: my-container # The name of the container. Must be unique within the Pod.
      image: nginx:latest # The Docker image to use for this container. Here, it's the latest
                           version of Nginx.
```

Explanation of the YAML File

- **apiVersion:** Specifies the version of the Kubernetes API to use. For Pods, it's typically v1.
- **kind:** The type of object being created. Here it's a Pod.
- **metadata:** Provides metadata about the object, including name and labels. The name must be unique within the namespace, and labels help in identifying and organizing Pods.
- **spec:** Contains the specifications of the Pod, including:
 - **containers:** Lists all containers that will run inside the Pod. Each container needs:
 - **name:** A unique name within the Pod.
 - **image:** The Docker image to use for the container.
 - **ports:** The ports that this container exposes.
 - **env:** Environment variables passed to the container.

Step 2: Apply the YAML File to Create the Pod

Use the `kubectl apply` command to create the Pod based on the YAML configuration file.

```
kubectl apply -f pod-example.yaml
```

```
PS C:\Users\KHUSHI JAIN\OneDrive\Desktop\Docker_lab\lab6> kubectl apply -f pod.yaml
pod/my-pod created
PS C:\Users\KHUSHI JAIN\OneDrive\Desktop\Docker_lab\lab6> |
```

This command tells Kubernetes to create a Pod as specified in the `pod-example.yaml` file.

Step 3: Verify the Pod Creation

To check the status of the Pod and ensure it's running, use:

```
kubectl get pods
```

```
PS C:\Users\KHUSHI JAIN\OneDrive\Desktop\Docker_lab\lab6> kubectl get pods
NAME      READY   STATUS    RESTARTS   AGE
my-pod    1/1     Running   0           43s
PS C:\Users\KHUSHI JAIN\OneDrive\Desktop\Docker_lab\lab6> |
```

This command lists all the Pods in the current namespace, showing their status, restart count, and other details.

You can get detailed information about the Pod using:

```
kubectl describe pod my-pod
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\KHUSHI JAIN> kubectl get pods
NAME      READY   STATUS    RESTARTS   AGE
my-pod    1/1     Running   0           68s
PS C:\Users\KHUSHI JAIN> kubectl describe pod my-pod
Name:      my-pod
Namespace: default
Priority:   0
Service Account: default
Node:      docker-desktop/192.168.65.3
Start Time: Thu, 21 Nov 2024 17:58:11 +0530
Labels:    app=my-app
Annotations: <none>
Status:    Running
IP:        10.1.0.18
IPs:
  IP: 10.1.0.18
Containers:
  my-container:
    Container ID:  docker://b81633679fe543f475e46d8ead50e3bcfd28f109a60a2d6a75ac983703d94343
    Image:         nginx:latest
    Image ID:      docker-pullable://nginx@sha256:bc5eac5eafc581aeda3008b4b1f07ebba230de2f27d47767129a6a905c84f470
    Port:         <none>
    Host Port:     <none>
    State:         Running
    Started:       Thu, 21 Nov 2024 17:58:38 +0530
```

This command provides detailed information about the Pod, including its events, container specifications, and resource usage.

Step 4: Interact with the Pod

You can interact with the running Pod in various ways, such as accessing the logs or executing commands inside the container.

View Logs: To view the logs of the container in the Pod:

```
kubectl logs my-pod
```

```
PS C:\Users\KHUSHI JAIN> kubectl logs my-pod
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2024/11/21 12:28:38 [notice] 1#1: using the "epoll" event method
2024/11/21 12:28:38 [notice] 1#1: nginx/1.27.2
2024/11/21 12:28:38 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2024/11/21 12:28:38 [notice] 1#1: OS: Linux 5.15.133.1-microsoft-standard-WSL2
2024/11/21 12:28:38 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2024/11/21 12:28:38 [notice] 1#1: start worker processes
2024/11/21 12:28:38 [notice] 1#1: start worker process 29
2024/11/21 12:28:38 [notice] 1#1: start worker process 30
2024/11/21 12:28:38 [notice] 1#1: start worker process 31
2024/11/21 12:28:38 [notice] 1#1: start worker process 32
2024/11/21 12:28:38 [notice] 1#1: start worker process 33
2024/11/21 12:28:38 [notice] 1#1: start worker process 34
2024/11/21 12:28:38 [notice] 1#1: start worker process 35
2024/11/21 12:28:38 [notice] 1#1: start worker process 36
2024/11/21 12:28:38 [notice] 1#1: start worker process 37
2024/11/21 12:28:38 [notice] 1#1: start worker process 38
2024/11/21 12:28:38 [notice] 1#1: start worker process 39
2024/11/21 12:28:38 [notice] 1#1: start worker process 40
PS C:\Users\KHUSHI JAIN>
```

Execute a Command: To run a command inside the container:

```
kubectl exec -it my-pod -- /bin/bash
```

```
PS C:\Users\KHUSHI JAIN> kubectl exec -it my-pod -- /bin/bash
root@my-pod:/# ls
bin      dev      docker-entrypoint.sh  home  lib64  mnt  proc  run  srv  tmp  var
boot     docker-entrypoint.d  etc    lib    media  opt  root  sbin sys  usr
root@my-pod:/# pwd
/
root@my-pod:/# env
KUBERNETES_SERVICE_PORT_HTTPS=443
KUBERNETES_SERVICE_PORT=443
HOSTNAME=my-pod
PWD=/
PKG_RELEASE=1~bookworm
HOME=/root
KUBERNETES_PORT_443_TCP=tcp://10.96.0.1:443
DYNPKG_RELEASE=1~bookworm
NJS_VERSION=0.8.6
TERM=xterm
SHLVL=1
KUBERNETES_PORT_443_TCP_PROTO=tcp
KUBERNETES_PORT_443_TCP_ADDR=10.96.0.1
KUBERNETES_SERVICE_HOST=10.96.0.1
KUBERNETES_PORT=tcp://10.96.0.1:443
KUBERNETES_PORT_443_TCP_PORT=443
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
NGINX_VERSION=1.27.2
```

The -it flag opens an interactive terminal session inside the container, allowing you to run commands.

Step 5: Delete the Pod

To clean up and remove the Pod when you're done, use the following command:

```
kubectl delete pod my-pod
```

```
error: the path "my-pod" does not exist
PS C:\Users\KHUSHI JAIN\OneDrive\Desktop\Docker_lab\lab6> kubectl delete pod my-pod
pod "my-pod" deleted
PS C:\Users\KHUSHI JAIN\OneDrive\Desktop\Docker_lab\lab6> kubectl get pods
No resources found in default namespace.
PS C:\Users\KHUSHI JAIN\OneDrive\Desktop\Docker_lab\lab6> |
```

This command deletes the specified Pod from the cluster.