Lab Exercise 8- Create POD in Kubernetes

Name- Misha
SAP ID- 500119679
Batch-2

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
kind: Pod
metadata:
  name: my-pod
  labels:
    app: web
```

```
spec:
 containers:
  - name: my-container
   image: nginx:latest
```

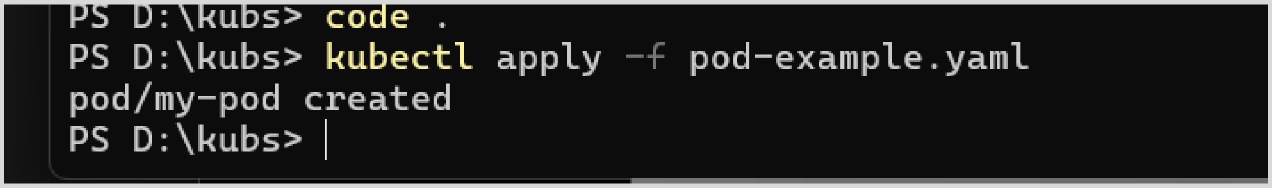
Explanation of the YAML File

- 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
```



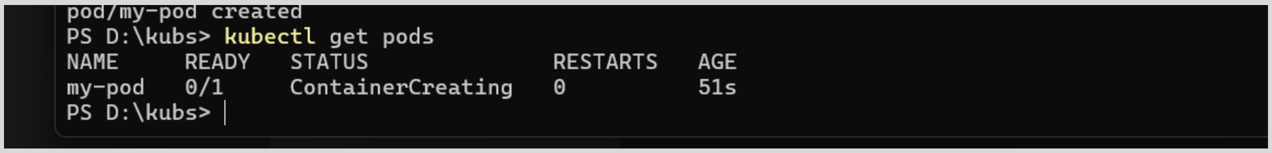
```
PS D:\kubs> code .
PS D:\kubs> kubectl apply -f pod-example.yaml
pod/my-pod created
PS D:\kubs>
```

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
```

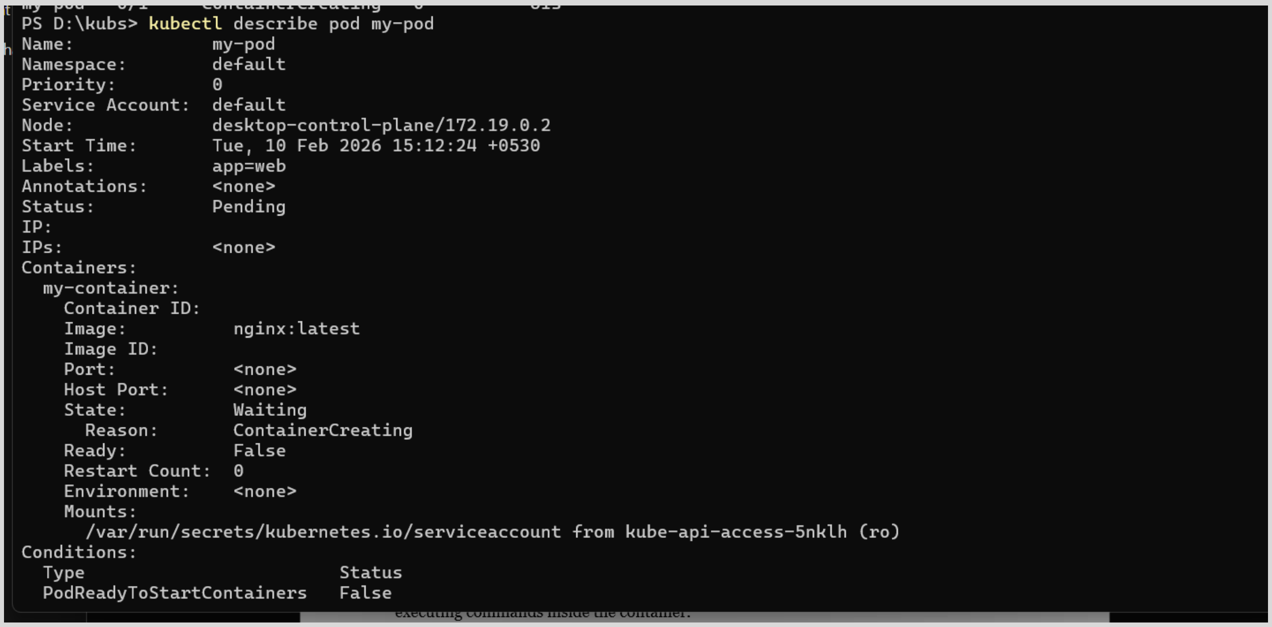


```
pod/my-pod created
PS D:\kubs> kubectl get pods
NAME     READY   STATUS              RESTARTS   AGE
my-pod   0/1     ContainerCreating   0          51s
PS D:\kubs>
```

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
```



```
PS D:\kubs> kubectl describe pod my-pod
Name:             my-pod
Namespace:        default
Priority:         0
Service Account:  default
Node:             desktop-control-plane/172.19.0.2
Start Time:       Tue, 10 Feb 2026 15:12:24 +0530
Labels:           app=web
Annotations:      <none>
Status:           Pending
IP:
IPs:              <none>
Containers:
  my-container:
    Container ID:
    Image:          nginx:latest
    Image ID:
    Port:           <none>
    Host Port:      <none>
    State:          Waiting
      Reason:       ContainerCreating
    Ready:          False
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-5nklh (ro)
Conditions:
  Type                        Status
  PodReadyToStartContainers   False
```

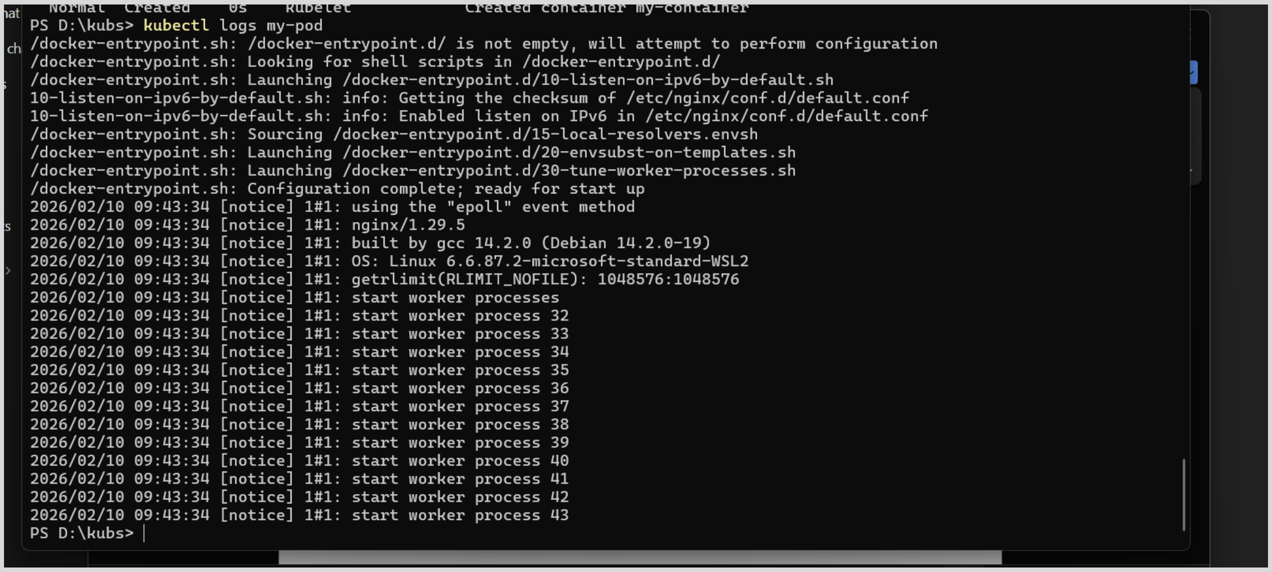
This command provides detailed information about the Pod, including its events, container specifications, and resource usage.

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:

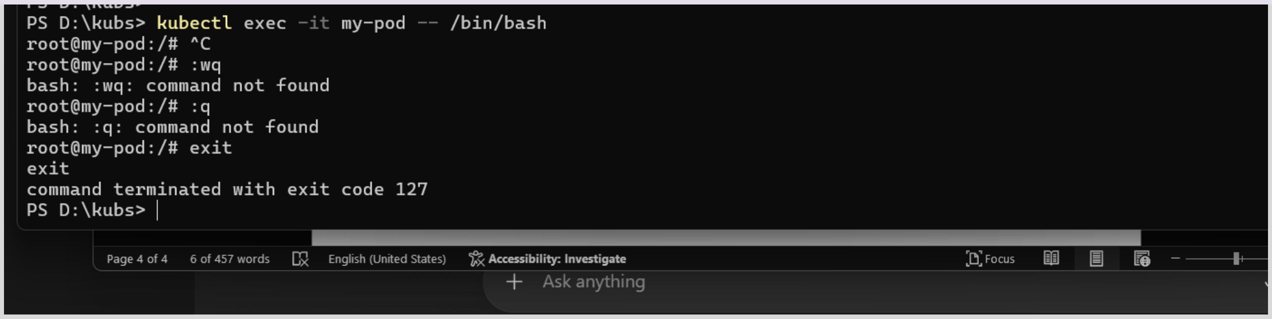
```
kubectl                                logs                                my-pod
```



```
  Normal  Created    0s    kubelet            Created container my-container
PS D:\kubs> 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
2026/02/10 09:43:34 [notice] 1#1: using the "epoll" event method
2026/02/10 09:43:34 [notice] 1#1: nginx/1.29.5
2026/02/10 09:43:34 [notice] 1#1: built by gcc 14.2.0 (Debian 14.2.0-19)
2026/02/10 09:43:34 [notice] 1#1: OS: Linux 6.6.87.2-microsoft-standard-WSL2
2026/02/10 09:43:34 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2026/02/10 09:43:34 [notice] 1#1: start worker processes
2026/02/10 09:43:34 [notice] 1#1: start worker process 32
2026/02/10 09:43:34 [notice] 1#1: start worker process 33
2026/02/10 09:43:34 [notice] 1#1: start worker process 34
2026/02/10 09:43:34 [notice] 1#1: start worker process 35
2026/02/10 09:43:34 [notice] 1#1: start worker process 36
2026/02/10 09:43:34 [notice] 1#1: start worker process 37
2026/02/10 09:43:34 [notice] 1#1: start worker process 38
2026/02/10 09:43:34 [notice] 1#1: start worker process 39
2026/02/10 09:43:34 [notice] 1#1: start worker process 40
2026/02/10 09:43:34 [notice] 1#1: start worker process 41
2026/02/10 09:43:34 [notice] 1#1: start worker process 42
2026/02/10 09:43:34 [notice] 1#1: start worker process 43
PS D:\kubs> |
```

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

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



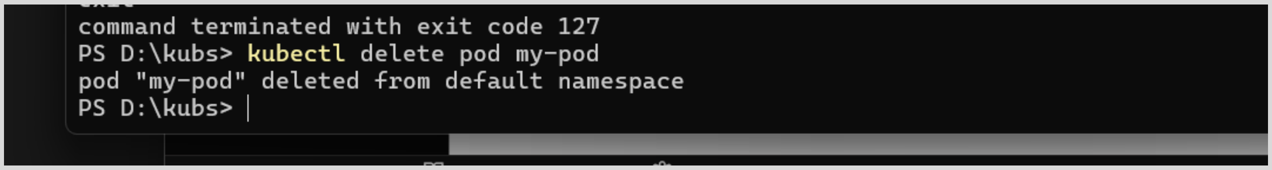
```
PS D:\kubs> kubectl exec -it my-pod -- /bin/bash
root@my-pod:/# ^C
root@my-pod:/# :wq
bash: :wq: command not found
root@my-pod:/# :q
bash: :q: command not found
root@my-pod:/# exit
exit
command terminated with exit code 127
PS D:\kubs> |
```

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



```
command terminated with exit code 127
PS D:\kubs> kubectl delete pod my-pod
pod "my-pod" deleted from default namespace
PS D:\kubs> |
```

This command deletes the specified Pod from the cluster.