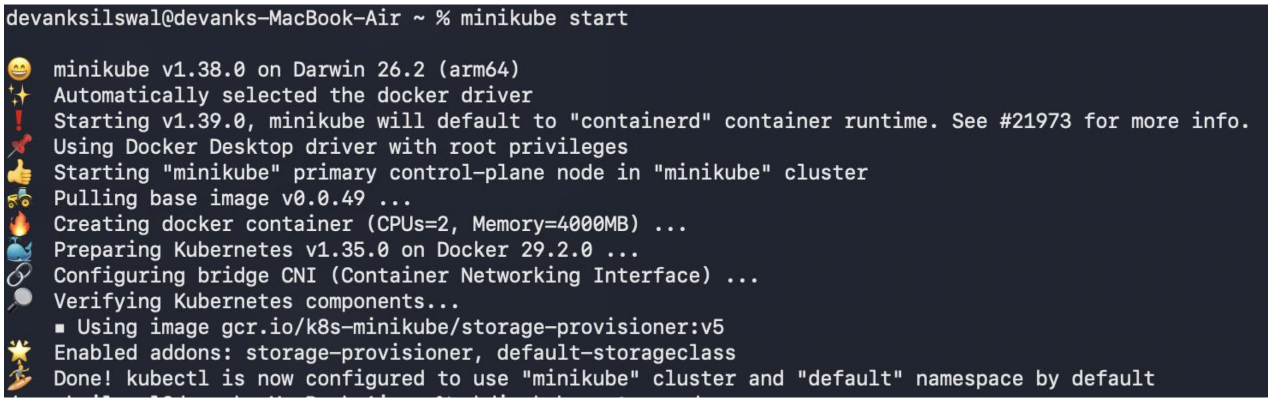
Lab Exercise 8- Create POD in Kubernetes

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.

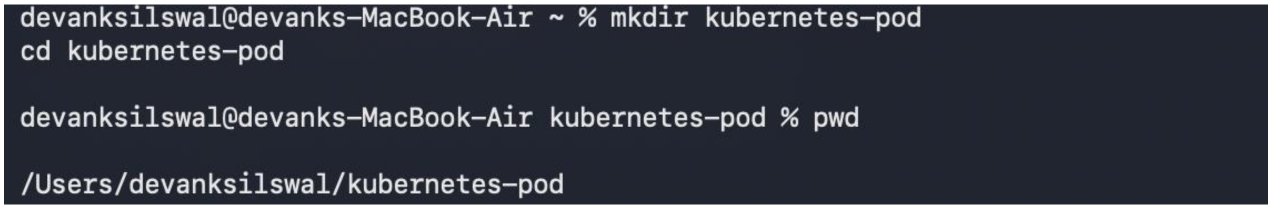


```
devanksilswal@devanks-MacBook-Air ~ % minikube start

😄  minikube v1.38.0 on Darwin 26.2 (arm64)
✨  Automatically selected the docker driver
❗  Starting v1.39.0, minikube will default to "containerd" container runtime. See #21973 for more info.
📌  Using Docker Desktop driver with root privileges
👍  Starting "minikube" primary control-plane node in "minikube" cluster
🚜  Pulling base image v0.0.49 ...
🔥  Creating docker container (CPUs=2, Memory=4000MB) ...
🐳  Preparing Kubernetes v1.35.0 on Docker 29.2.0 ...
🔗  Configuring bridge CNI (Container Networking Interface) ...
🔎  Verifying Kubernetes components...
    ▪ Using image gcr.io/k8s-minikube/storage-provisioner:v5
🌟  Enabled addons: storage-provisioner, default-storageclass
🏄  Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
```

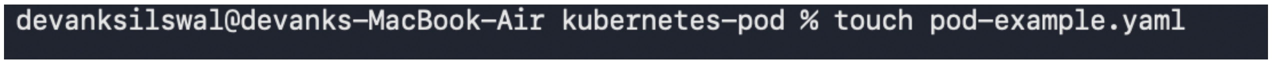
Step-by-Step Guide

Step 1: Create a YAML File for the Pod

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

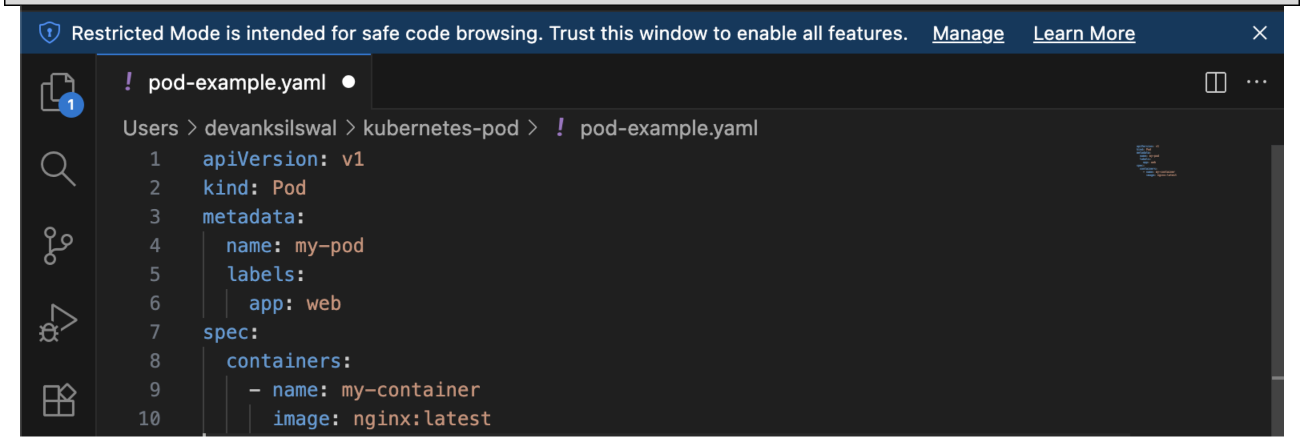


```
devanksilswal@devanks-MacBook-Air ~ % mkdir kubernetes-pod
cd kubernetes-pod

devanksilswal@devanks-MacBook-Air kubernetes-pod % pwd

/Users/devanksilswal/kubernetes-pod
```



```
devanksilswal@devanks-MacBook-Air kubernetes-pod % touch pod-example.yaml
```

```
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
  labels:
    app: web
spec:
  containers:
    - name: my-container
      image: nginx:latest
```



Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More

```
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
  labels:
    app: web
spec:
  containers:
    - name: my-container
      image: nginx:latest
```

Explanation of the YAML File

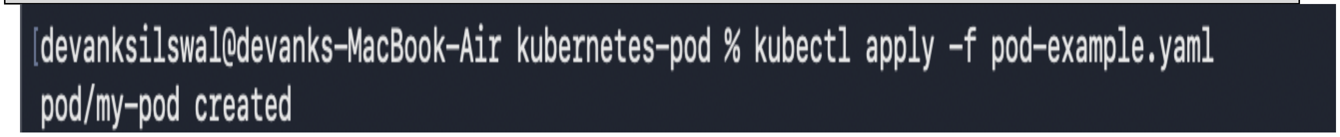
- apiVersion: Specifies the version of the Kubernetes API to use. For Pods, it's typically v1.
- kind: The type of object being created. Here it's a Pod.
- metadata: Provides metadata about the object, including name and labels. The name must be unique within the namespace, and labels help in identifying 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
```



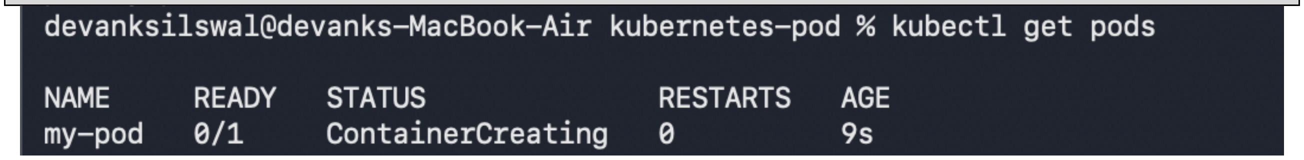
```
devanksilswal@devanks-MacBook-Air kubernetes-pod % kubectl apply -f pod-example.yaml
pod/my-pod created
```

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
```

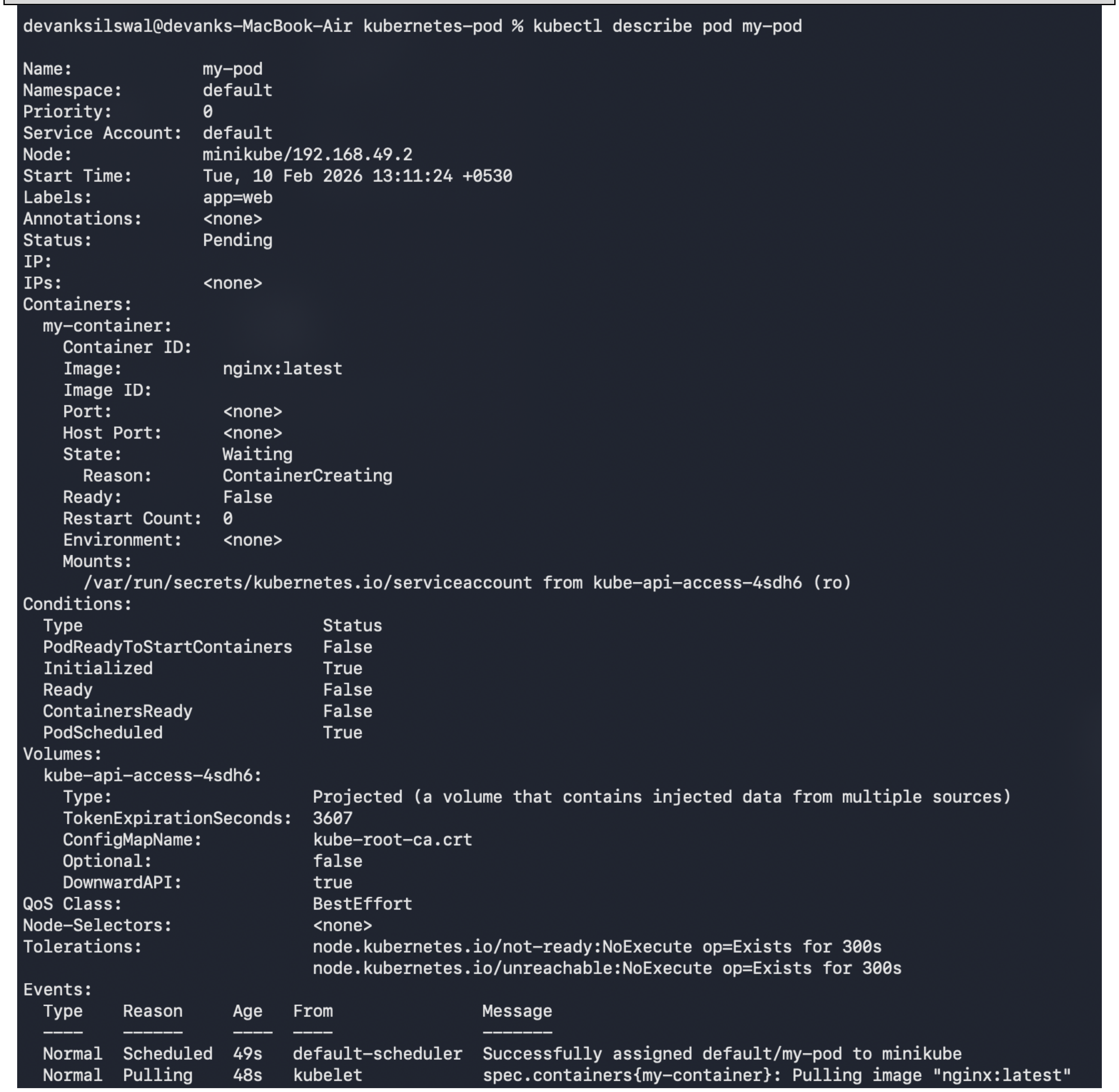


```
devanksilswal@devanks-MacBook-Air kubernetes-pod % kubectl get pods

NAME     READY   STATUS              RESTARTS   AGE
my-pod   0/1     ContainerCreating   0          9s
```

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
```



```
devanksilswal@devanks-MacBook-Air kubernetes-pod % kubectl describe pod my-pod

Name:             my-pod
Namespace:        default
Priority:         0
Service Account:  default
Node:             minikube/192.168.49.2
Start Time:       Tue, 10 Feb 2026 13:11:24 +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-4sdh6 (ro)
Conditions:
  Type                        Status
  PodReadyToStartContainers   False
  Initialized                 True
  Ready                       False
  ContainersReady             False
  PodScheduled                True
Volumes:
  kube-api-access-4sdh6:
    Type:                    Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds:  3607
    ConfigMapName:           kube-root-ca.crt
    Optional:                false
    DownwardAPI:             true
QoS Class:                   BestEffort
Node-Selectors:              <none>
Tolerations:                 node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                             node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:
  Type    Reason     Age   From               Message
  ----    ------     ----  ----               -------
  Normal  Scheduled  49s   default-scheduler  Successfully assigned default/my-pod to minikube
  Normal  Pulling    48s   kubelet            spec.containers{my-container}: Pulling image "nginx:latest"
```

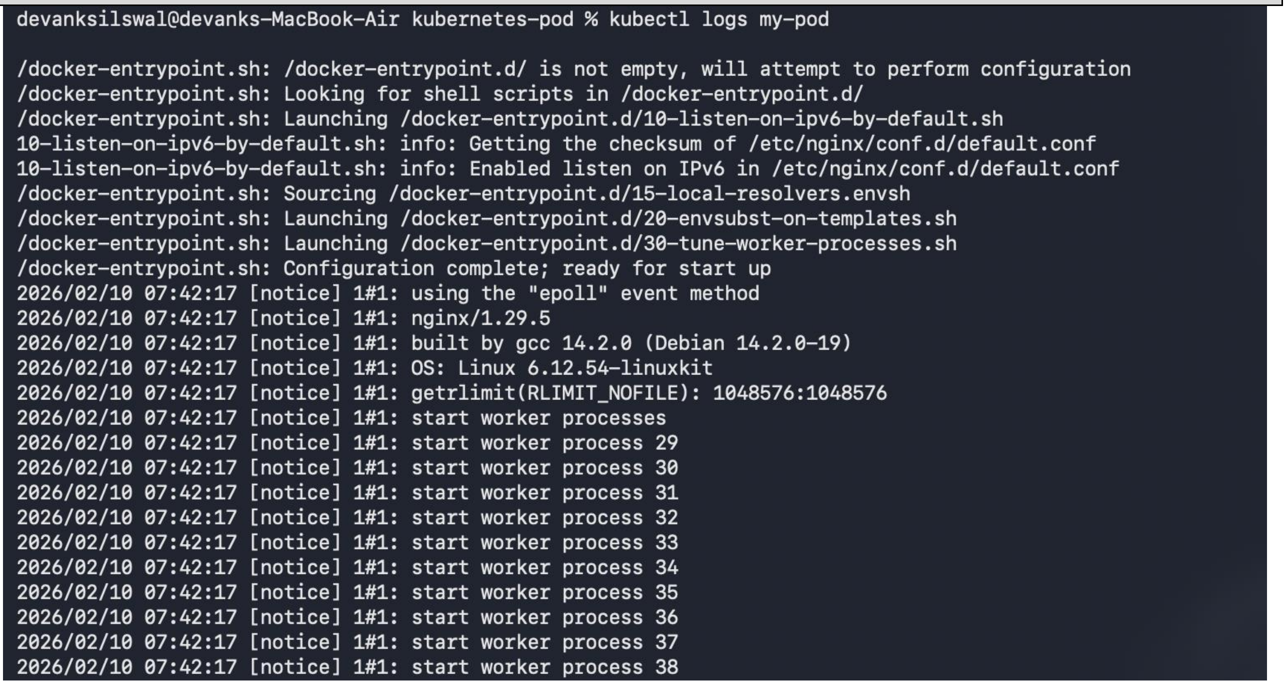
This command provides detailed information about the Pod, including its events, container specifications, and resource usage.

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:

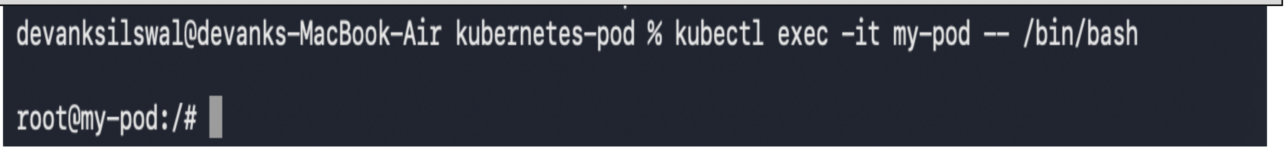
```
kubectl logs my-pod
```



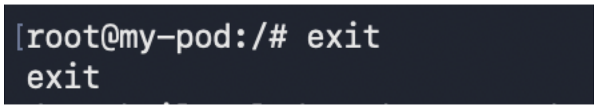
```
devanksilswal@devanks-MacBook-Air kubernetes-pod % kubectl logs my-pod

/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 07:42:17 [notice] 1#1: using the "epoll" event method
2026/02/10 07:42:17 [notice] 1#1: nginx/1.29.5
2026/02/10 07:42:17 [notice] 1#1: built by gcc 14.2.0 (Debian 14.2.0-19)
2026/02/10 07:42:17 [notice] 1#1: OS: Linux 6.12.54-linuxkit
2026/02/10 07:42:17 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2026/02/10 07:42:17 [notice] 1#1: start worker processes
2026/02/10 07:42:17 [notice] 1#1: start worker process 29
2026/02/10 07:42:17 [notice] 1#1: start worker process 30
2026/02/10 07:42:17 [notice] 1#1: start worker process 31
2026/02/10 07:42:17 [notice] 1#1: start worker process 32
2026/02/10 07:42:17 [notice] 1#1: start worker process 33
2026/02/10 07:42:17 [notice] 1#1: start worker process 34
2026/02/10 07:42:17 [notice] 1#1: start worker process 35
2026/02/10 07:42:17 [notice] 1#1: start worker process 36
2026/02/10 07:42:17 [notice] 1#1: start worker process 37
2026/02/10 07:42:17 [notice] 1#1: start worker process 38
```

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

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



```
devanksilswal@devanks-MacBook-Air kubernetes-pod % kubectl exec -it my-pod -- /bin/bash

root@my-pod:/#
```



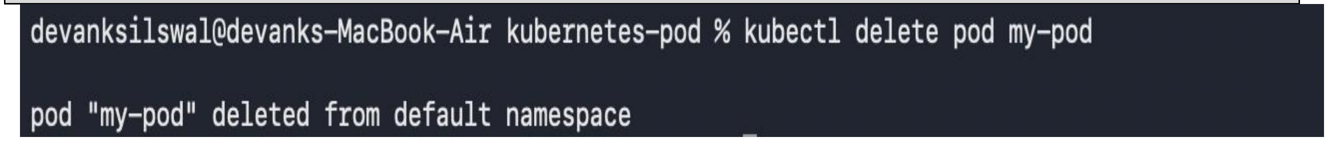
```
[root@my-pod:/# exit
 exit
```

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
```



```
devanksilswal@devanks-MacBook-Air kubernetes-pod % kubectl delete pod my-pod

pod "my-pod" deleted from default namespace
```

This command deletes the specified Pod from the cluster.