Creating a Pod that runs two Containers

Kubernetes is a container management tool and it automates container deployment, load balancing, and container scaling. It is open-source and developed by Google in 2014 and written in Golang. All cloud providers adopt Kubernetes. It is scheduled runs and manages isolated containers that are running on virtual, physical, and cloud machine.

Create Multiple Containers in a Pod

Step 1. Open your machine with successfully installed Kubernetes

Step 2. First of all, create a Manifest file. If a person wants to do anything in Kubernetes first required to create a Manifest. Manifest is a file using the YAML(Yet Another Markup Language)

```
root@ip-172-31-27-240:~# vi multi-container-pod.yaml
apiVersion: v1
kind: Pod
metadata:
  name: multi-container-pod
  labels:
    app: multi-container-app
spec:
  containers:
  - name: container-one
    image: nginx
    ports:
    - containerPort: 80
    volumeMounts:
    - name: shared-volume
      mountPath: /usr/share/nginx/html
```

```
  - name: container-two
    image: busybox
    command: ["sh", "-c", "echo 'Hello from Container Two!' > /data/index.html && sleep 3600"]
    volumeMounts:
    - name: shared-volume
      mountPath: /data
  volumes:
  - name: shared-volume
    emptyDir: {}
```

In the configuration file, you can see that the Pod has a Volume named shared data.

The first container listed in the configuration file runs an nginx server. The mount path for the shared Volume is /usr/share/nginx/html. The second container is based on the busybox image, and has a mount path of /pod-data. The second container runs the following command and then terminates.

echo 'Hello from Container Two `> /pod-data/index.html`

Notice that the second container writes the index.html file in the root directory of the nginx server.

Create the Pod and the two Containers:

kubectl apply -f multi-container-pod.yaml

You can see that the Container-two has terminated, and the nginx Container is still running.

Get a shell to nginx Container:

kubectl exec -it multi-container-pod -c container-one – bash

Recall that the Container-two created the index.html file in the nginx root directory. Use curl to send a GET request to the nginx server:

root@multi-container-pod:/# curl localhost

The output shows that nginx serves a web page written by the container-two:

Hello from Container Two!