Lesson 07 Demo 03

Creating a Deployment with ConfigMap as Volume

Objective: To create a deployment with ConfigMap as volume to enhance the flexibility, manageability, and scalability of your application

Tools required: kubeadm, kubectl, kubelet, and containerd

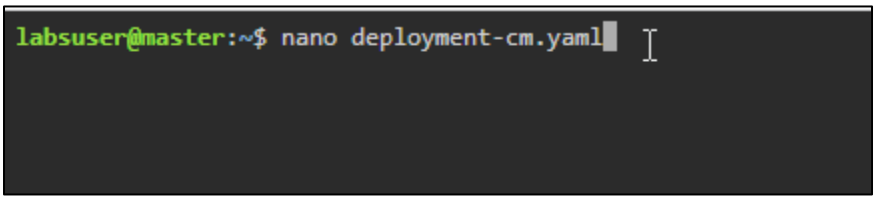
Prerequisites: A Kubernetes cluster should already be set up (refer to the steps provided in Lesson 02, Demo 01 for guidance).

Steps to be followed:

1. Create a ConfigMap
2. Create a deployment to attach a ConfigMap as volume to it

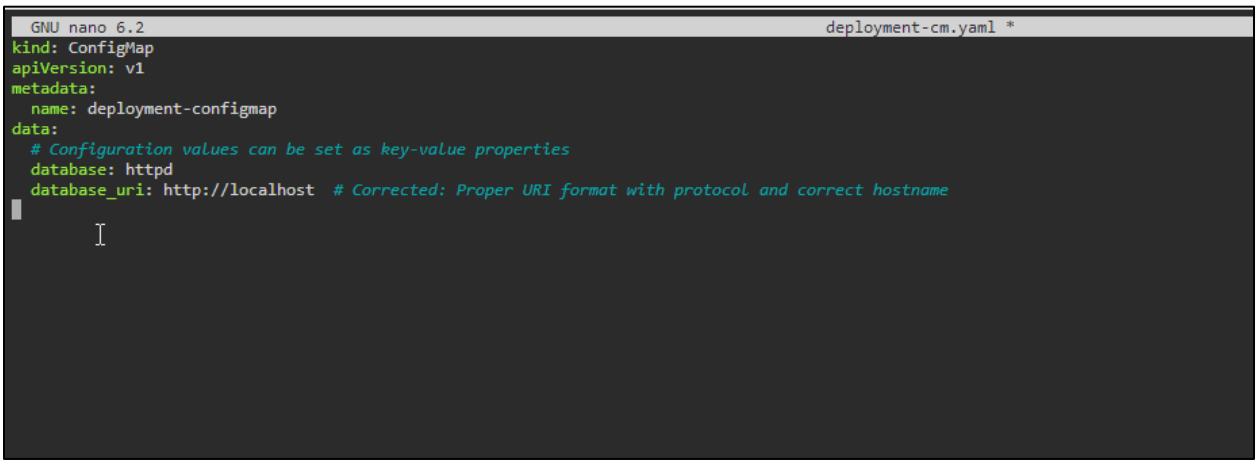
Step 1: Create a ConfigMap

1.1 On the master node, run the following command to create a YAML file:
nano deployment-cm.yaml



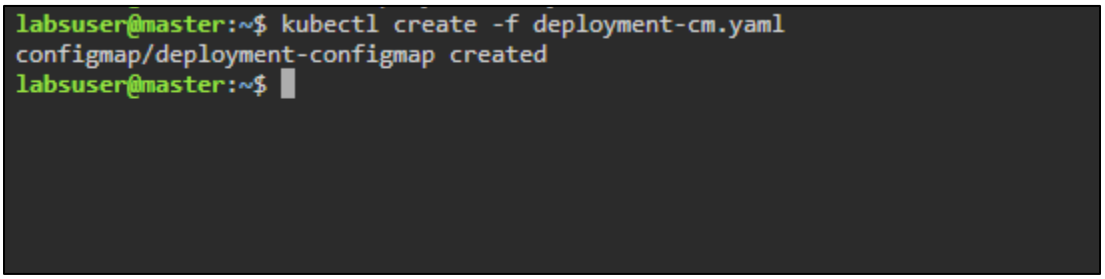
1.2 Copy the following code in the YAML file:

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: deployment-configmap
data:
  # Configuration values can be set as key-value properties
  database: httpd
  database_uri: http://localhost
```



1.3 Execute the following command to create a ConfigMap:

```
kubectl create -f deployment-cm.yaml
```



1.4 Verify the state of ConfigMap by running the following command:
kubectl get configmap

```
labsuser@master:~$ nano deployment-cm.yaml
labsuser@master:~$ kubectl create -f deployment-cm.yaml
configmap/deployment-configmap created
labsuser@master:~$ kubectl get configmap
NAME                   DATA   AGE
deployment-configmap   2      91s
kube-root-ca.crt       1      20h
labsuser@master:~$
```

Step 2: Create a deployment to attach a ConfigMap as volume to it

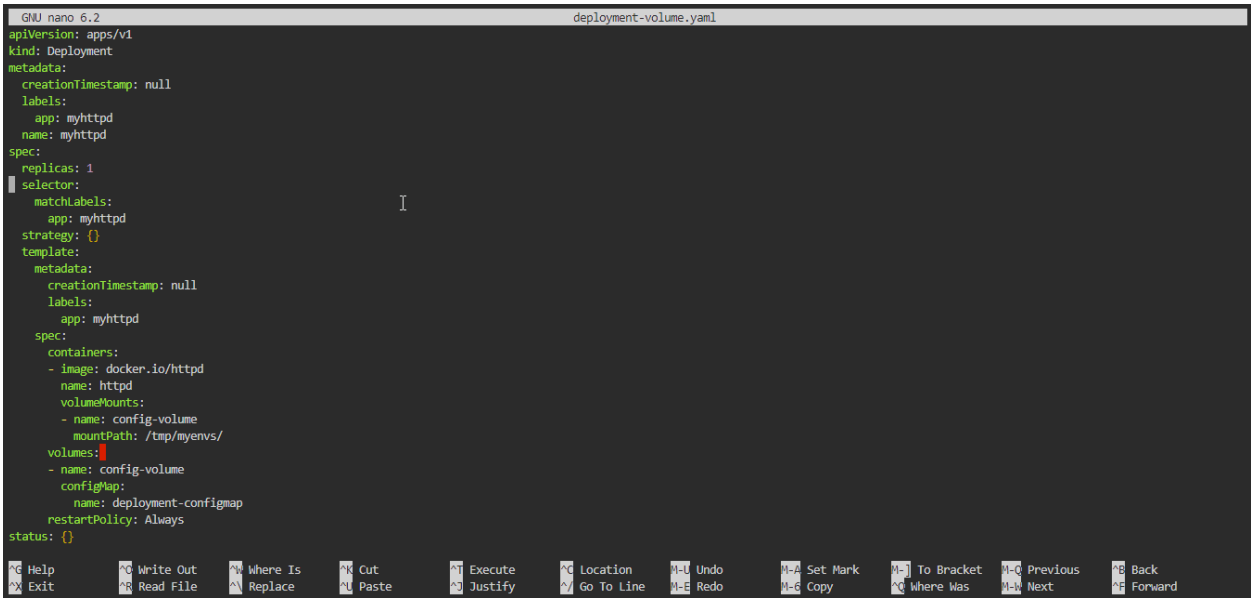
2.1 On the master node, run the following command to create a YAML file:
nano deployment-volume.yaml

```
labsuser@master:~$ nano deployment-volume.yaml
```

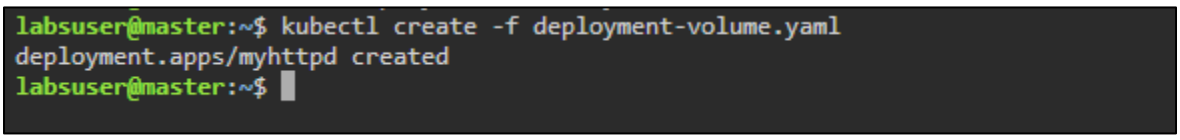
2.2 Copy the following code in the YAML file:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  creationTimestamp: null
  labels:
    app: myhttpd
  name: myhttpd
spec:
  replicas: 1
  selector:
    matchLabels:
      app: myhttpd
  strategy: {}
  template:
    metadata:
```

```
    creationTimestamp: null
    labels:
      app: myhttpd
  spec:
    containers:
    - image: docker.io/httpd
      name: httpd
      volumeMounts:
      - name: config-volume
        mountPath: /tmp/myenvs/
    volumes:    # This should be inside the spec section under template
    - name: config-volume
      configMap:
        name: deployment-configmap
    restartPolicy: Always
status: {}
```



2.3 Run the following command to create a deployment:
kubectl create -f deployment-volume.yaml



2.4 Verify the pod and deployment state by running the following commands:
kubectl get deployment
kubectl get pods

```
labsuser@master:~$ kubectl get deployment
NAME      READY   UP-TO-DATE   AVAILABLE   AGE
myhttpd   1/1     1            1           31m
labsuser@master:~$ kubectl get pods
NAME                       READY   STATUS    RESTARTS      AGE
myhttpd-9dcf74db4-bfndt    1/1     Running   0             31m
secret-pod                 1/1     Running   1 (51m ago)   2d23h
```

Note: Copy the name of the pod for the next step

2.5 Navigate to the pod using the following command and start a shell session:
kubectl exec -it <my-pod> -- /bin/sh

```
labsuser@master:~$ kubectl get deployment
NAME      READY   UP-TO-DATE   AVAILABLE   AGE
myhttpd   1/1     1            1           31m
labsuser@master:~$ kubectl get pods
NAME                       READY   STATUS    RESTARTS      AGE
myhttpd-9dcf74db4-bfndt    1/1     Running   0             31m
secret-pod                 1/1     Running   1 (51m ago)   2d23h
labsuser@master:~$ kubectl exec -it myhttpd-9dcf74db4-bfndt -- /bin/sh
#
```

Note: Replace the **<my-pod>** with your pod **NAME** as shown in the screenshot above

2.6 Inside the pod, navigate to **/tmp/myenvs** to see the ConfigMap data using the **cd** command:
cd /tmp/myenvs

```
labsuser@master:~$ kubectl get pods
NAME                       READY   STATUS    RESTARTS      AGE
myhttpd-9dcf74db4-bfndt    1/1     Running   0             31m
secret-pod                 1/1     Running   1 (51m ago)   2d23h
labsuser@master:~$ kubectl exec -it myhttpd-9dcf74db4-bfndt -- /bin/sh
# cd /tmp/myenvs
#
```

2.7 View the content of the files **database** and **database_uri** using the following commands:

cat database

cat database_uri

```
labsuser@master:~$ kubectl get pods
NAME                      READY   STATUS    RESTARTS      AGE
myhttpd-9dcf74db4-bfndt   1/1     Running   0             31m
secret-pod                1/1     Running   1 (51m ago)   2d23h
labsuser@master:~$ kubectl exec -it myhttpd-9dcf74db4-bfndt -- /bin/sh
# cd /tmp/myenvs
# cat database
httpd#
```

```
labsuser@master:~$ kubectl exec -it myhttpd-9dcf74db4-bfndt -- /bin/sh
# cd /tmp/myenvs
# cat database
httpd#
# cat database_uri
http://localhost#
```

This command should display the config value that you have provided when you created the ConfigMap.

By following these steps, you have successfully created a deployment using a ConfigMap as volume.