8.7. LABS



## Exercise 8.3: Creating a Persistent Volume Claim (PVC)

Before Pods can take advantage of the new PV we need to create a Persistent Volume Claim (PVC).

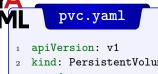
1. Begin by determining if any currently exist.

```
student@cp:~$ kubectl get pvc

1 No resources found in default namespace.
```

2. Create a YAML file for the new pvc.

```
student@cp:~$ vim pvc.yaml
```



```
kind: PersistentVolumeClaim
metadata:
name: pvc-one
spec:
accessModes:
- ReadWriteMany
resources:
requests:
storage: 200Mi
```

3. Create and verify the new pvc is bound. Note that the size is 1Gi, even though 200Mi was suggested. Only a volume of at least that size could be used.

```
student@cp:~$ kubectl create -f pvc.yaml

persistentvolumeclaim/pvc-one created
```

```
student@cp:~$ kubectl get pvc
```

```
NAME STATUS VOLUME CAPACITY ACCESSMODES STORAGECLASS AGE pvc-one Bound pvvol-1 1Gi RWX 4s
```

Look at the status of the pv again, to determine if it is in use. It should show a status of Bound.

```
student@cp:~$ kubectl get pv
```

```
NAME CAPACITY ACCESSMODES RECLAIMPOLICY STATUS CLAIM

STORAGECLASS REASON AGE
pvvol-1 1Gi RWX Retain Bound default/pvc-one

5m
```

5. Create a new deployment to use the pvc. We will copy and edit an existing deployment yaml file. We will change the deployment name then add a volumeMounts section under containers and volumes section to the general spec. The name used must match in both places, whatever name you use. The claimName must match an existing pvc. As shown in the following example. The volumes line is the same indent as containers and dnsPolicy.

```
student@cp:~$ cp first.yaml nfs-pod.yaml
student@cp:~$ vim nfs-pod.yaml
```





## nfs-pod.yaml

```
1 apiVersion: apps/v1
2 kind: Deployment
3 metadata:
     annotations:
      deployment.kubernetes.io/revision: "1"
     generation: 1
6
7
     labels:
      run: nginx
8
     name: nginx-nfs
                                        #<-- Edit name
9
     namespace: default
10
   spec:
11
     replicas: 1
12
13
     selector:
      matchLabels:
14
        run: nginx
15
     strategy:
16
      rollingUpdate:
17
        maxSurge: 1
18
19
         maxUnavailable: 1
20
       type: RollingUpdate
21
     template:
       metadata:
22
         creationTimestamp: null
23
         labels:
24
           run: nginx
25
       spec:
26
         containers:
27
         - image: nginx
28
           imagePullPolicy: Always
29
           name: nginx
30
           volumeMounts:
31
32
           - name: nfs-vol
33
             mountPath: /opt
34
           ports:
           - containerPort: 80
35
             protocol: TCP
36
           resources: {}
37
           terminationMessagePath: /dev/termination-log
38
           terminationMessagePolicy: File
         volumes:
                                              #<<-- These four lines
40
         - name: nfs-vol
41
           persistentVolumeClaim:
42
             claimName: pvc-one
43
         dnsPolicy: ClusterFirst
44
45
         restartPolicy: Always
46
         schedulerName: default-scheduler
47
         securityContext: {}
         terminationGracePeriodSeconds: 30
48
```

6. Create the pod using the newly edited file.

```
student@cp:~$ kubectl create -f nfs-pod.yaml

deployment.apps/nginx-nfs created
```

7. Look at the details of the pod. You may see the daemonset pods running as well.

```
student@cp:~$ kubectl get pods
```



8.7. LABS 3

```
NAME READY STATUS RESTARTS AGE nginx-nfs-1054709768-s8g28 1/1 Running 0 3m
```

## student@cp:~\$ kubectl describe pod nginx-nfs-1054709768-s8g28

```
Name:
                    nginx-nfs-1054709768-s8g28
                    default
   Namespace:
   Priority:
                    worker/10.128.0.5
   Node:
   <output_omitted>
6
       Mounts:
         /opt from nfs-vol (rw)
9
10
11
   <output_omitted>
12
   Volumes:
13
    nfs-vol:
14
                     PersistentVolumeClaim (a reference to a PersistentV...
       Type:
15
       ClaimName:
                          pvc-one
16
       ReadOnly:
                         false
17
   <output_omitted>
18
```

8. View the status of the PVC. It should show as bound.

## student@cp:~\$ kubectl get pvc

```
NAME STATUS VOLUME CAPACITY ACCESS MODES STORAGECLASS AGE pvc-one Bound pvvol-1 1Gi RWX 2m
```

