Exercise 9.3: 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@lfs458-node-1a0a:~$ kubectl get pvc
No resources found.
```

2. Create a YAML file for the new pvc.

```
student@lfs458-node-1a0a:~$ vim pvc.yaml


apiVersion: v1
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@lfs458-node-1a0a:~$ kubectl create -f pvc.yaml
persistentvolumeclaim "pvc-one" created

student@lfs458-node-1a0a:~$ kubectl get pvc
NAME      STATUS    VOLUME    CAPACITY   ACCESSMODES   STORAGECLASS   AGE
pvc-one   Bound     pvvol-1   1Gi        RWX                          4s
```

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

```
student@lfs458-node-1a0a:~$ 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.

```
student@lfs458-node-1a0a:~$ cp first.yaml nfs-pod.yaml

student@lfs458-node-1a0a:~$ vim nfs-pod.yaml


apiVersion: apps/v1beta1
kind: Deployment
metadata:
  annotations:
    deployment.kubernetes.io/revision: "1"
  generation: 1
```



```
    labels:
      run: nginx
    name: nginx-nfs
    namespace: default
    resourceVersion: "1411"
  spec:
    replicas: 1
    selector:
      matchLabels:
        run: nginx
    strategy:
      rollingUpdate:
        maxSurge: 1
        maxUnavailable: 1
      type: RollingUpdate
    template:
      metadata:
        creationTimestamp: null
        labels:
          run: nginx
      spec:
        containers:
        - image: nginx
          imagePullPolicy: Always
          name: nginx
          volumeMounts:
          - name: nfs-vol
            mountPath: /opt
          ports:
          - containerPort: 80
            protocol: TCP
          resources: {}
          terminationMessagePath: /dev/termination-log
          terminationMessagePolicy: File
        volumes:                          #<<-- These four lines
        - name: nfs-vol
          persistentVolumeClaim:
            claimName: pvc-one
        dnsPolicy: ClusterFirst
        restartPolicy: Always
        schedulerName: default-scheduler
        securityContext: {}
        terminationGracePeriodSeconds: 30
```

6. Create the pod using the newly edited file.

```
student@lfs458-node-1a0a:~$ kubectl create -f nfs-pod.yaml
```

7. Look at the details of the pod.

```
student@lfs458-node-1a0a:~$ kubectl get pods
NAME                          READY     STATUS    RESTARTS   AGE
nginx-nfs-1054709768-s8g28    1/1       Running   0          3m


student@lfs458-node-1a0a:~$ kubectl describe pod nginx-nfs-1054709768-s8g28
Name:               nginx-nfs-1054709768-s8g28
Namespace:        default
Node:               lfs458-node-2b2b/10.128.0.5

<output_omitted>
```



```
    Mounts:
      /opt from nfs-vol (rw)

<output_omitted>

Volumes:
  nfs-vol:
    Type:        PersistentVolumeClaim (a reference to a PersistentV...
    ClaimName:     pvc-one
    ReadOnly:      false
<output_omitted>
```

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

```
student@lfs458-node-1a0a:~$ kubectl get pvc
NAME      STATUS VOLUME  CAPACITY ACCESS MODES  STORAGECLASS  AGE
pvc-one   Bound  pvvol-1 1Gi      RWX                          2m
```

