# Lesson 07 Demo 05

# Configuring Pod Storage Using hostPath-Based PV and PVC

**Objective:** To configure pod storage using hostPath-based PersistentVolume (PV) and PersistentVolumeClaim (PVC) in Kubernetes for efficient data storage and retrieval

**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 PersistentVolume
2. Create PersistentVolumeClaim
3. Deploy a pod in a new namespace
4. Validate the pod and storage
5. Verify data persistence

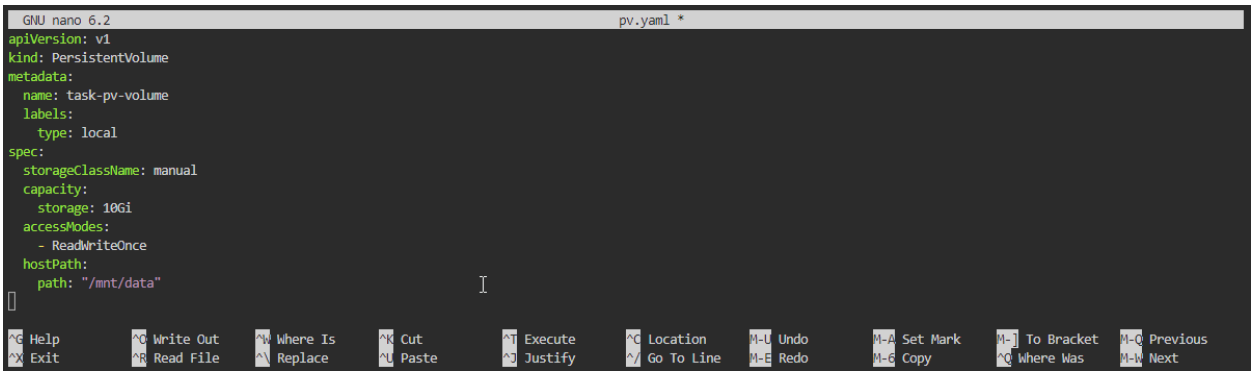
## Step 1: Create PersistentVolume

1.1 Create the YAML file using the command below:
**nano pv.yaml**

```
labsuser@master:~$ nano pv.yaml
```

1.2 Add the following code to the **pv.yaml** file to create the pod:

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: task-pv-volume
  labels:
    type: local
spec:
  storageClassName: manual
  capacity:
    storage: 10Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: "/mnt/data"
```



1.3 Use the **cat** command to validate the content of the **pv.yaml** file

```
labsuser@master:~$ nano pv.yaml
labsuser@master:~$ cat pv.yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: task-pv-volume
  labels:
    type: local
spec:
  storageClassName: manual
  capacity:
    storage: 10Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: "/mnt/data"
labsuser@master:~$
```

1.4 Apply the Kubernetes resource configuration from the YAML file
**kubectl apply -f pv.yaml**

```
labsuser@master:~$ cat pv.yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: task-pv-volume
  labels:
    type: local
spec:
  storageClassName: manual
  capacity:
    storage: 10Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: "/mnt/data"
labsuser@master:~$ kubectl apply -f pv.yaml
persistentvolume/task-pv-volume created
labsuser@master:~$
```

1.5 List all the PVs in the Kubernetes cluster using the following command:
**kubectl get pv**

```
labsuser@master:~$ kubectl apply -f pv.yaml
persistentvolume/task-pv-volume created
labsuser@master:~$ kubectl get pv
NAME             CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS      CLAIM   STORAGECLASS   REASON   AGE
task-pv-volume   10Gi       RWO            Retain           Available           manual                  3m14s
labsuser@master:~$
```

## Step 2: Create PersistentVolumeClaim

2.1 Create the YAML file using the following command:
**nano pvc.yaml**

```
labsuser@master:~$ kubectl apply -f pv.yaml
persistentvolume/task-pv-volume created
labsuser@master:~$ kubectl get pv
NAME             CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS      CLAIM   STORAGECLASS   REASON   AGE
task-pv-volume   10Gi       RWO            Retain           Available           manual                  3m14s
labsuser@master:~$ nano pvc.yaml
```

2.2 Add the following code to the **pvc.yaml** file:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: task-pv-claim
spec:
  storageClassName: manual
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 3Gi
```

```
  GNU nano 6.2                                          pvc.yaml *
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: task-pv-claim
spec:
  storageClassName: manual
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 3Gi

^G Help      ^O Write Out  ^W Where Is   ^K Cut       ^T Execute   ^C Location  M-U Undo     M-A Set Mark   M-] To Bracket  M-Q Previous
^X Exit      ^R Read File  ^\ Replace    ^U Paste     ^J Justify   ^/ Go To Line M-E Redo     M-6 Copy       ^Q Where Was    M-W Next
```

2.3 Use the **cat** command to validate the content of the **pvc.yaml** file

```
labsuser@master:~$ nano pvc.yaml
labsuser@master:~$ cat pvc.yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: task-pv-claim
spec:
  storageClassName: manual
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 3Gi
labsuser@master:~$
```

2.4 Apply the Kubernetes resource configuration defined in the **pvc.yaml** file to create the PVC using the following command:
**kubectl apply -f pvc.yaml**

```
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 3Gi
labsuser@master:~$ kubectl apply -f pvc.yaml
persistentvolumeclaim/task-pv-claim created
labsuser@master:~$
```

2.5 List all the PVs in the Kubernetes cluster using the following command:
**kubectl get pv**

```
labsuser@master:~$ kubectl apply -f pvc.yaml
persistentvolumeclaim/task-pv-claim created
labsuser@master:~$ kubectl get pv
NAME             CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM                   STORAGECLASS   REASON   AGE
task-pv-volume   10Gi       RWO            Retain           Bound    default/task-pv-claim   manual                  17m
labsuser@master:~$
```

2.6 List all the PVCs in the Kubernetes cluster using the following command:
**kubectl get pvc**

```
labsuser@master:~$ kubectl apply -f pvc.yaml
persistentvolumeclaim/task-pv-claim created
labsuser@master:~$ kubectl get pv
NAME             CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM                   STORAGECLASS   REASON   AGE
task-pv-volume   10Gi       RWO            Retain           Bound    default/task-pv-claim   manual                  17m
labsuser@master:~$ kubectl get pvc
NAME            STATUS   VOLUME           CAPACITY   ACCESS MODES   STORAGECLASS   AGE
task-pv-claim   Bound    task-pv-volume   10Gi       RWO            manual         88s
labsuser@master:~$
```

## Step 3: Deploy a pod in a new namespace

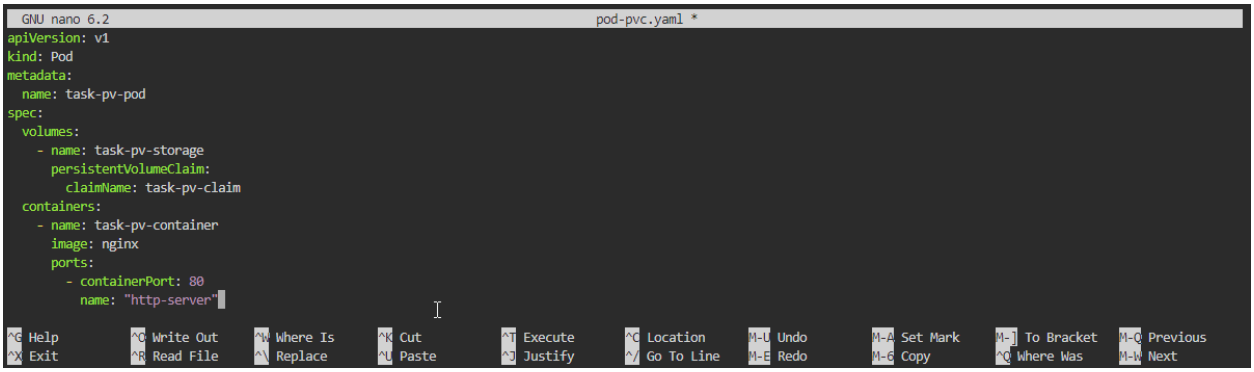
3.1 Create the YAML file using the following command:
**nano pod-pvc.yaml**

```
labsuser@master:~$ kubectl apply -f pvc.yaml
persistentvolumeclaim/task-pv-claim created
labsuser@master:~$ kubectl get pv
NAME             CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM                   STORAGECLASS   REASON   AGE
task-pv-volume   10Gi       RWO            Retain           Bound    default/task-pv-claim   manual                  17m
labsuser@master:~$ kubectl get pvc
NAME            STATUS   VOLUME           CAPACITY   ACCESS MODES   STORAGECLASS   AGE
task-pv-claim   Bound    task-pv-volume   10Gi       RWO            manual         88s
labsuser@master:~$ nano pod-pvc.yaml
```

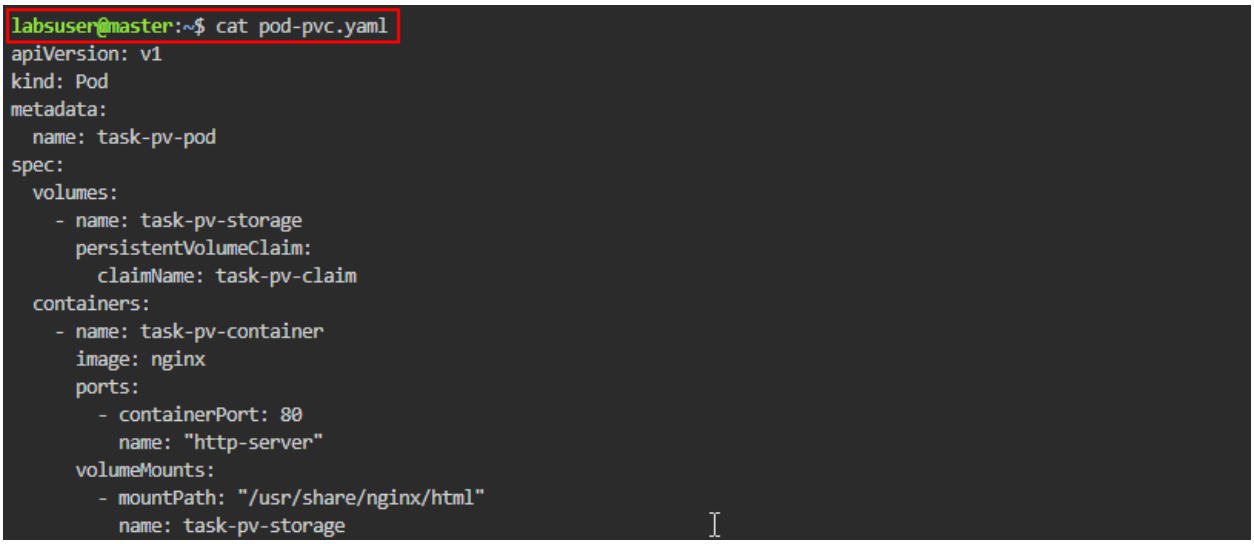
3.2 Add the following code to the **pod-pvc.yaml** file:

```
apiVersion: v1
kind: Pod
metadata:
  name: task-pv-pod
spec:
  volumes:
    - name: task-pv-storage
      persistentVolumeClaim:
        claimName: task-pv-claim
```

```
containers:
  - name: task-pv-container
    image: nginx
    ports:
      - containerPort: 80
        name: "http-server"
    volumeMounts:
      - mountPath: "/usr/share/nginx/html"
        name: task-pv-storage
```




3.3 Use the **cat** command to validate the content of the **pod-pvc.yaml** file



3.4 Apply the Kubernetes pod with the configuration defined in the **pod-pvc.yaml** file using the following command:
**kubectl apply -f pod-pvc.yaml**

```
      ports:
        - containerPort: 80
          name: "http-server"
      volumeMounts:
        - mountPath: "/usr/share/nginx/html"
          name: task-pv-storage
labsuser@master:~$ kubectl apply -f pod-pvc.yaml
pod/task-pv-pod created
labsuser@master:~$
```

3.5 List all the PVCs in the Kubernetes cluster using the following command:
**kubectl get pvc**

```
labsuser@master:~$ kubectl apply -f pod-pvc.yaml
pod/task-pv-pod created
labsuser@master:~$ kubectl get pvc
NAME            STATUS   VOLUME           CAPACITY   ACCESS MODES   STORAGECLASS   AGE
task-pv-claim   Bound    task-pv-volume   10Gi       RWO            manual         16m
labsuser@master:~$
```

## Step 4: Validate the pod and storage

4.1 Create an empty **index.html** file in the **/usr/share/nginx/html** directory within the **task-pv-pod** using the following command:
**kubectl exec -it task-pv-pod -- touch /usr/share/nginx/html/index.html**

```
labsuser@master:~$ kubectl get pvc
NAME            STATUS   VOLUME           CAPACITY   ACCESS MODES   STORAGECLASS   AGE
task-pv-claim   Bound    task-pv-volume   10Gi       RWO            manual         16m
labsuser@master:~$ kubectl exec -it task-pv-pod -- touch /usr/share/nginx/html/index.html
labsuser@master:~$
```

4.2 List the content of the **/usr/share/nginx/html/** directory within the **task-pv-pod** using the following command:
**kubectl exec -it task-pv-pod -- ls /usr/share/nginx/html/**

```
labsuser@master:~$ kubectl get pvc
NAME            STATUS   VOLUME           CAPACITY   ACCESS MODES   STORAGECLASS   AGE
task-pv-claim   Bound    task-pv-volume   10Gi       RWO            manual         16m
labsuser@master:~$ kubectl exec -it task-pv-pod -- touch /usr/share/nginx/html/index.html
labsuser@master:~$ kubectl exec -it task-pv-pod -- ls /usr/share/nginx/html/
index.html
labsuser@master:~$
```

4.3 Retrieve the status and details about the **task-pv-pod** using the following command:
**kubectl get pod task-pv-pod -o wide**

```
labsuser@master:~$ kubectl exec -it task-pv-pod -- touch /usr/share/nginx/html/index.html
labsuser@master:~$ kubectl exec -it task-pv-pod -- ls /usr/share/nginx/html/
index.html
labsuser@master:~$ kubectl get pod task-pv-pod -o wide
NAME          READY   STATUS    RESTARTS   AGE     IP               NODE                     NOMINATED NODE   READINESS GATES
task-pv-pod   1/1     Running   0          9m21s   192.168.47.129   worker-node-1.example.com   <none>           <none>
labsuser@master:~$
```

4.4 List the contents of the **/mnt/data** directory within the **task-pv-pod** by running the following command:
**ls /mnt/data**

```
labsuser@worker-node-1:~$ ls /mnt/data
index.html
labsuser@worker-node-1:~$
```

**Note:** Execute the above command in the **worker-node-1**

## Step 5: Verify data persistence

5.1 On the master node, delete the Kubernetes pod created from the configuration defined in the **pod-pvc.yaml** file using the following command:
**kubectl delete -f pod-pvc.yaml**

```
labsuser@master:~$ kubectl apply -f pod-pvc.yaml
pod/task-pv-pod created
labsuser@master:~$ kubectl get pvc
NAME            STATUS   VOLUME           CAPACITY   ACCESS MODES   STORAGECLASS   AGE
task-pv-claim   Bound    task-pv-volume   10Gi       RWO            manual         16m
labsuser@master:~$ kubectl exec -it task-pv-pod -- touch /usr/share/nginx/html/index.html
labsuser@master:~$ kubectl exec -it task-pv-pod -- ls /usr/share/nginx/html/
index.html
labsuser@master:~$ kubectl get pod task-pv-pod -o wide
NAME          READY   STATUS    RESTARTS   AGE     IP               NODE                        NOMINATED NODE   READINESS GATES
task-pv-pod   1/1     Running   0          9m21s   192.168.47.129   worker-node-1.example.com   <none>           <none>
labsuser@master:~$ kubectl delete -f pod-pvc.yaml
pod "task-pv-pod" deleted
labsuser@master:~$
```

5.2 On the worker-node-1, list the contents of the **/mnt/data** directory within the **task-pv-pod** by running the following command:
**ls /mnt/data**

```
labsuser@worker-node-1:~$ ls /mnt/data
index.html
labsuser@worker-node-1:~$ ls /mnt/data
index.html
labsuser@worker-node-1:~$
```

5.3 Deploy the Kubernetes pod with the configuration defined in the **pod-pvc.yaml** file using the following command:
**k apply -f pod-pvc.yaml**

```
labsuser@master:~$ kubectl get pod task-pv-pod -o wide
NAME          READY   STATUS    RESTARTS   AGE     IP               NODE                        NOMINATED NODE   READINESS GATES
task-pv-pod   1/1     Running   0          9m21s   192.168.47.129   worker-node-1.example.com   <none>           <none>
labsuser@master:~$ kubectl delete -f pod-pvc.yaml
pod "task-pv-pod" deleted
labsuser@master:~$ kubectl apply -f pod-pvc.yaml
pod/task-pv-pod created
```

5.4 List the content of the **/usr/share/nginx/html/** directory within the **task-pv-pod** using the following command:
**k exec -it task-pv-pod -- ls /usr/share/nginx/html/**

```
labsuser@master:~$ kubectl delete -f pod-pvc.yaml
pod "task-pv-pod" deleted
labsuser@master:~$ kubectl apply -f pod-pvc.yaml
pod/task-pv-pod created
labsuser@master:~$ kubectl exec -it task-pv-pod -- ls /usr/share/nginx/html/
index.html
labsuser@master:~$
```

By following these steps, you have successfully configured a Kubernetes pod for data storage and retrieval using hostPath-based PV and PVC.