Storage of K8S:

By default, the storage is ephermal in nature. that means if you delete the container the storage got delete.

To store the data permanently we need to use two API persistent volume and persistent volume claim.

Using PV API we can tell kubernete that we have an storage service .This is admin work to create an storage service.After creating storage service to use the PV we need to use another API called PVC.

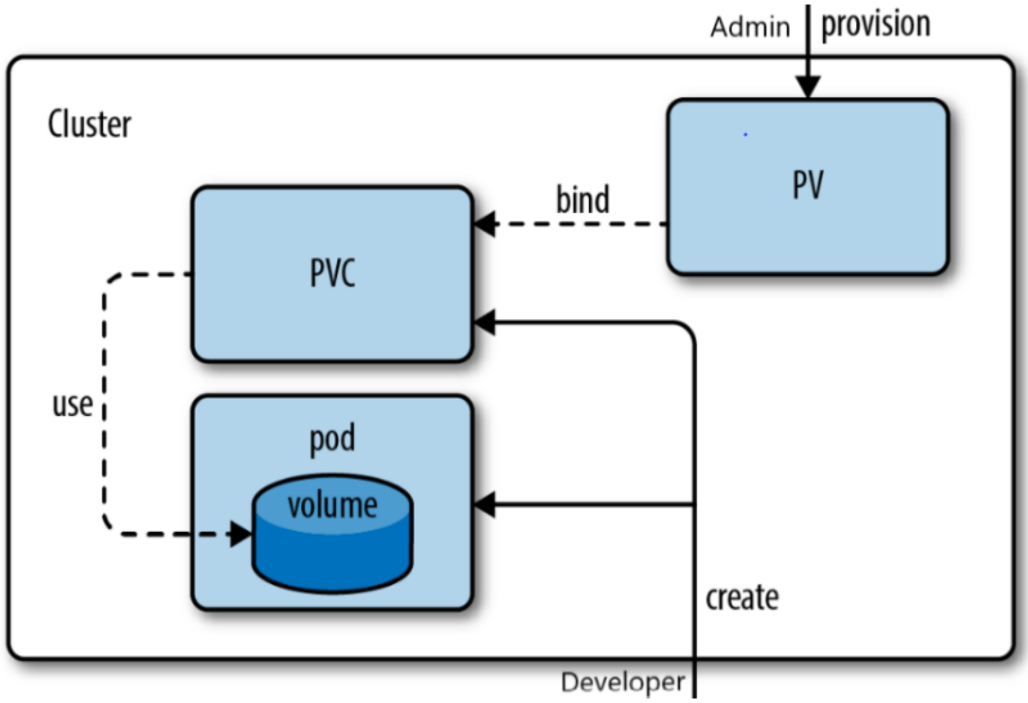
Persistent volume:

PVs are resources in the cluster that have a lifecycle independent of any individual pod that uses a PV. This is the "physical" volume on the host machine that stores persistent data. PersistentVolumes provide storage resources in a cluster, allowing the storage resource to persist even when the pods that use them are cycled.

PVs also provides facility of quota management.

Persistent volume claim:

PVCs is a request for storage by a user. It is similar to a Pod. Pods consume node resources and PVCs consume PV resources. Pods can request specific levels of resources (CPU and Memory). Claims can request specific size and access modes they can be mounted ReadWriteOnce, ReadOnlyMany or ReadWriteMany.



Pods are made up of multiple containers, so when volume attach to the pod all the containers running inside the pod can access to volume as a result for read and write purpose.

Volumes can be store in Virtual machines and Physical machines. We use PV and PVC to provide storage in Kubernetes.

Creating PV and PVC:

How to Create a PersistentVolume:

Steps to create a PersistentVolume:

Step 1: Create the YAML file.

Step 2: configuration file into the YAML manifest file.

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

Step 3: To save the persistent volume file

```
kubectl apply -f pv.yaml
```

Step 4: the created PV to see if it is available.

```
kubectl get pv
```

Step 5: To describe pv storage.

```
kubectl describe pv dev
```

How to Create a PersistentVolumeClaim:

Step 1: create a YAML file

Step 2: configuration file into the YAML manifest file.

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

storage: 10Gi
```

Step 3: To save the persistent volume Claim file

```
kubectl apply -f pvc.yaml
```

Step 4: the created PVC to see if it is available.

```
kubectl get pvC
```

Step 5: To describe pvc storage.

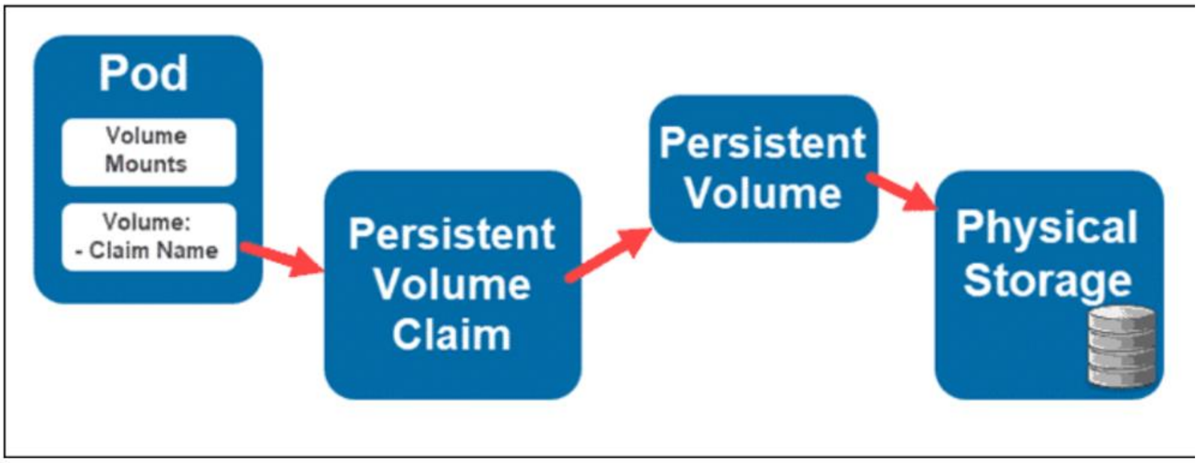
```
kubectl describe pvc dev
```

Pod Creation from PVC:

After creating the persistent volume claim, we have to use it inside of a pod. For this, we will create a new pod. To create a pod, we will create a pod.yaml file.

```
apiVersion: v1
kind: Pod
metadata:
  name: task-pv-pod
  labels:
    name: myapp
spec:
  volumes:
  - name: task-pv-storage
    persistentVolumeClaim:
      claimName: task-pv-claim
  containers:
  - name: task-pv-container
    image: nginx
    resources:
      limits:
        memory: "128Mi"
        cpu: "500m"
    ports:
      - containerPort: 80
```

```
volumeMounts:
  - mountPath: "/usr/share/nginx/html"
    name: task-pv-storage
```



Step 1: To save the pod.yaml

```
kubectl apply -f pod.yaml
```

Step 2: To check whether the pod is running.

```
kubectl get pod task-pv-pod
```

Verification:

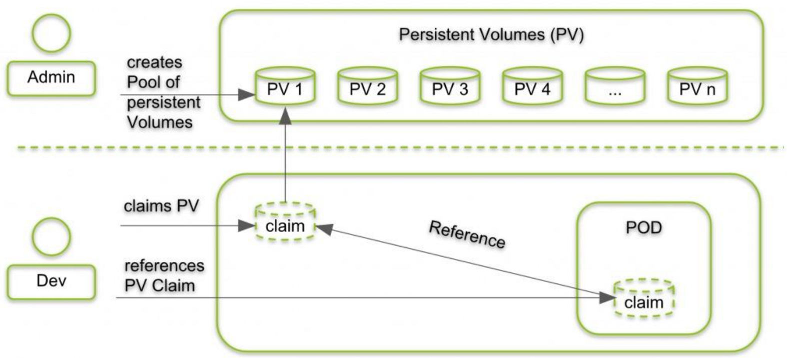
To expose the pod

```
Kubectl expose pod task-pv-pod –type=NodePort  --port=80 service/task-pv-pod exposed
```

Copy the IP address of nodemachine and copy the port number from master machine.

To get port number we use the command.

```
kubectl get svc
```


Admin
creates
Pool of
persistent
Volumes
Persistent Volumes (PV)
PV 1
PV 2
PV 3
PV 4
...
PV n
claims PV
Dev
references
PV Claim
claim
Reference
POD
claim