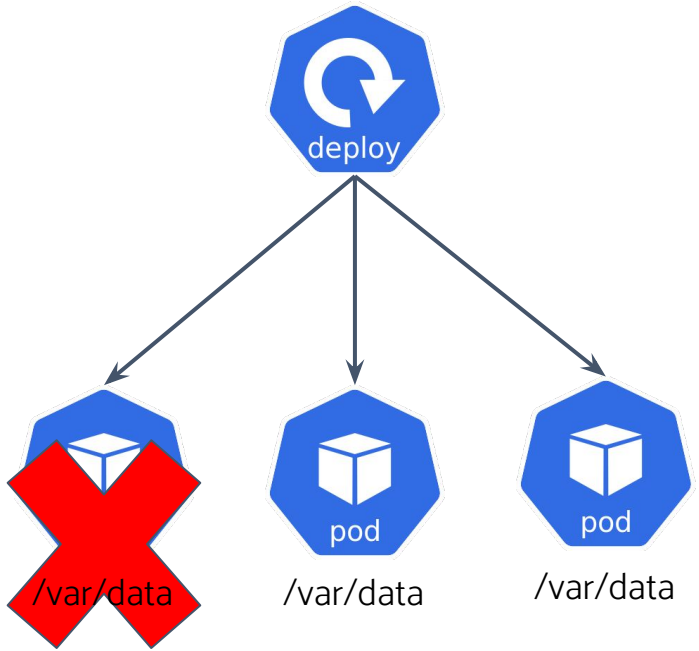
KUBERNETES ACADEMY

VOLUMES

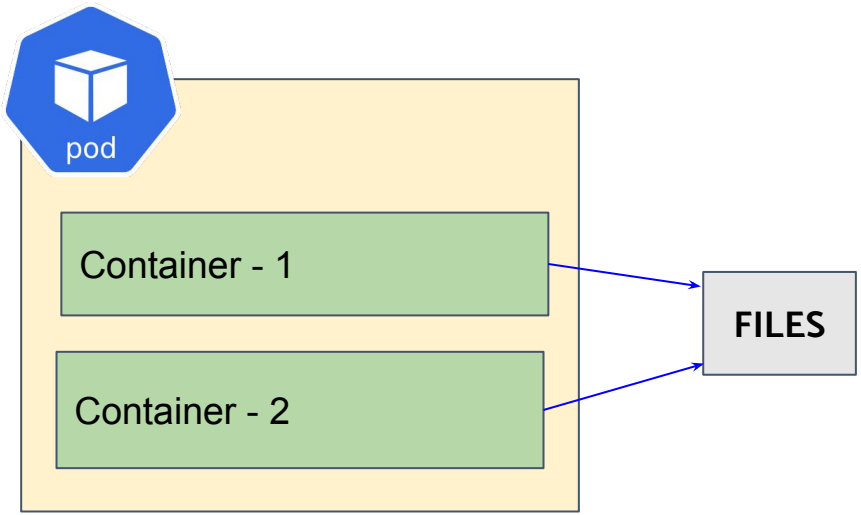
WHY VOLUME?

- **Challenge**: How can we keep files even when a pod dies and spin up again?

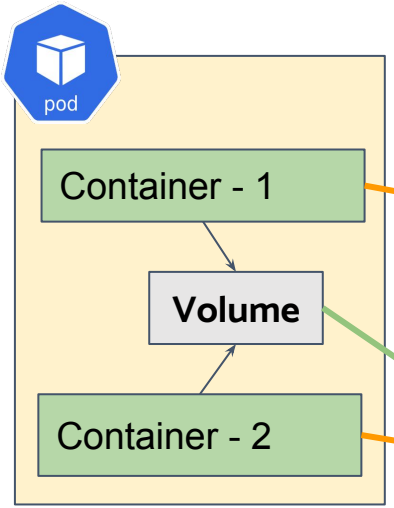


ANOTHER CASE

- How to share files between container?
- Container doesn't share files in standard configuration



Share files between containers
using “emptydir”



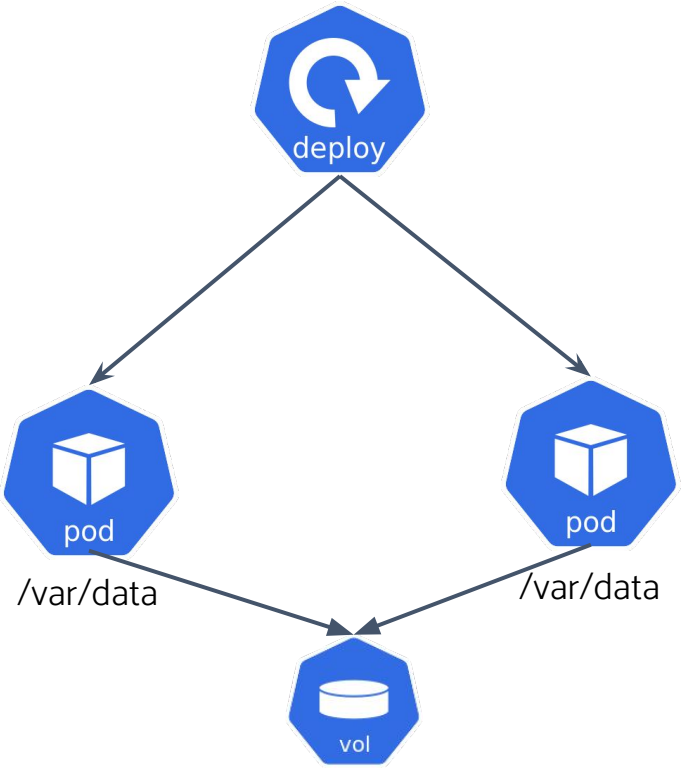

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: test-pod
spec:
  containers:
  - image: nginx
    name: container1
    volumeMounts:
    - mountPath: /data/shared-data
      name: shared-volume
  - image: nginx
    name: container2
    volumeMounts:
    - mountPath: /data/shared-data
      name: shared-volume
  volumes:
  - name: shared-volume
    emptyDir: {}
```

- Whatever the “**container1**” saves in the path “**/data/shared-data**”, “**container2**” will be able to see it
- “**MountPath**” doesn’t need to be the same path

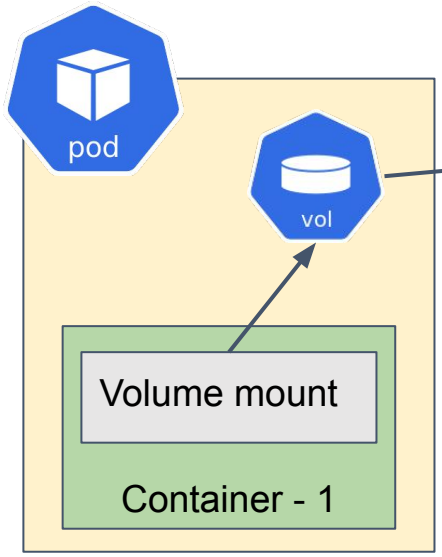
EXAMPLE OF EMPTYDIR

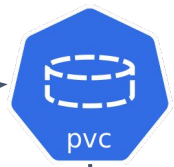
KEEP DATA

- How can we keep files even when a pod dies and spin up again?



Using PV and PVC





BIND


```yaml
volumes:
  - name: task-pv-storage
    persistentVolumeClaim:
      claimName: task-pv-claim
```

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

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: task-pv-volume
  labels:
    type: local
spec:
  storageClassName: manual
  capacity:
    storage: 3Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: "/mnt/data"
```

Access modes

- Access modes:
  - **ReadWriteOnce -** Multiples pods in the same node
  - **ReadOnlyMany** - Read only multiple nodes
  - **ReadWriteMany -** Multiple pods even in distinguish nodes
  - **ReadWriteOncePod** - the volume can be mounted as read-write by a single Pod


```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: task-pv-volume
  labels:
    type: local
spec:
  storageClassName: manual
  capacity:
    storage: 3Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: "/mnt/data"
```

PV information

- Persist Volume is rely on a cloud storage component
- **Azure** (Storage account, Azure Disk ...)
- **AWS** (AWS Disk)
- The cloud component needs to be created and configured

PV Documentation
https://kubernetes.io/docs/reference/kubernetes-api/config-and-storage-resources/persistent-volume-v1/#PersistentVolumeSpec


```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: task-pv-volume
  labels:
    type: local
spec:
  storageClassName: manual
  capacity:
    storage: 3Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: "/mnt/data"
```

EXAMPLE OF PVC and PV

PVC and PV

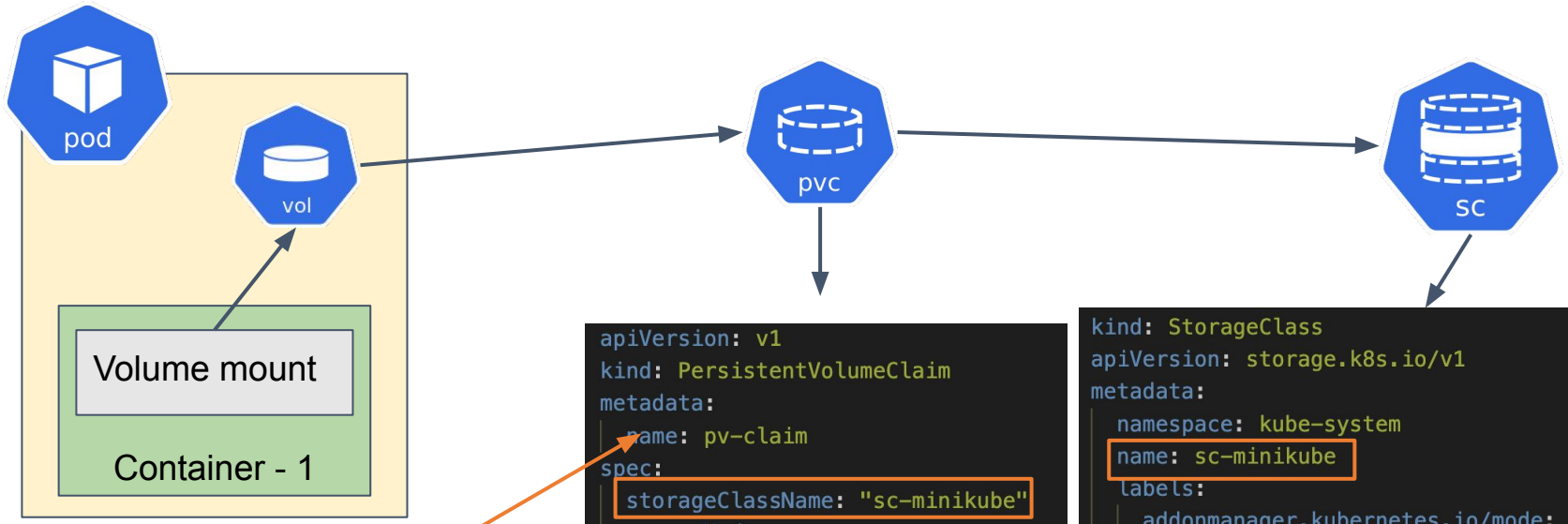
- Everytime a developer needs a new volume, it needs to be requested by the administrator
- The component configuration is manual

Storage Class

- The storage class creates the storage resources in cloud, in a dynamic way
- The provisioner is responsible to create the PV's

Volume Plugin	Internal Provisioner	Config Example
AWSElasticBlockStore	✓	AWS EBS
AzureFile	✓	Azure File
AzureDisk	✓	Azure Disk
CephFS	-	-
Cinder	✓	OpenStack Cinder
FC	-	-
FlexVolume	-	-
Flocker	✓	-
GCEPersistentDisk	✓	GCE PD
Glusterfs	✓	Glusterfs
iSCSI	-	-
Quobyte	✓	Quobyte
NFS	-	NFS
RBD	✓	Ceph RBD
VsphereVolume	✓	vSphere
PortworxVolume	✓	Portworx Volume
ScaleIO	✓	ScaleIO
StorageOS	✓	StorageOS
Local	-	Local

Storage class




```yaml
volumes:
  - name: task-pv-storage
    persistentVolumeClaim:
      claimName: task-pv-claim
```

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pv-claim
spec:
  storageClassName: "sc-minikube"
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 3Gi
```

```yaml
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  namespace: kube-system
  name: sc-minikube
  labels:
    addonmanager.kubernetes.io/mode: EnsureExists
provisioner: k8s.io/minikube-hostpath
```

LET’S DO THE TASK