**Persistent Volume**

* Create on volume in aws account (eg:-EBS )
* Copy that volume id in YAML file
* Vi mypvm.yml

( eg:- apiVersion: v1

kind: PersistentVolume

metadata:

name: myebsvol

spec:

capacity:

storage: 1Gi

accessModes:

- ReadWriteOnce

persistentVolumeReclaimPolicy: Recycle

awsElasticBlockStore:

volumeID: # YAHAN APNI EBS VOLUME ID DAALO

fsType: ext4 )

* Kubectl apply -f mypv.yml
* Kubectl get pv
* Vi mypvc.yml (create one YAML file for claiming the volume )

( eg:- apiVersion: v1

kind: PersistentVolumeClaim

metadata:

name: myebsvolclaim

spec:

accessModes:

- ReadWriteOnce

resources:

requests:

storage: 1Gi )

* Kubectl apply -f mypvc.yml
* Kubectl get pvc ( your volume bounded now you can use )
* Vi deploypvc.yml

(eg:- apiVersion: apps/v1

kind: Deployment

metadata:

name: pvdeploy

spec:

replicas: 1

selector: # tells the controller which pods to watch/belong to

matchLabels:

app: mypv

template:

metadata:

labels:

app: mypv

spec:

containers:

- name: shell

image: centos

command: ["bin/bash", "-c", "sleep 10000"]

volumeMounts:

- name: mypd

mountPath: "/tmp/persistent"

volumes:

- name: mypd

persistentVolumeClaim:

claimName: myebsvolclaim )

* Kubectl apply -f deploypvc.yml ( your volume mount with pod )
* Kubectl get deploy
* Kubectl get rs
* Kubectl get pods
* Kubectl exec pvcdeploy-fdhfkhd -it -- /bin/bash ( to go inside the container )
* Cd /tmp/persistent/
* Ls
* Vi testfile ( make one file )
* Ls (o/p:- testfile )
* Kubectl pod pvcdeploy-dfbdf
* Kubectl get pods ( after deleting pod one new pod will be created due to raplicaset )
* Kubectl exec pvcdeploy-fdjfdskf -it -- /bin/bash ( go inside the container )
* Cd /tmp/persistent/ ( you will see that persistent volume )
* Ls
* Cat testfile

**LivenessProbe**

* Vi liveness .yml

( eg:- apiVersion: v1

kind: Pod

metadata:

labels:

test: liveness

name: mylivenessprobe

spec:

containers:

- name: liveness

image: ubuntu

args:

- /bin/sh

- -c

- touch /tmp/healthy; sleep 1000

livenessProbe:

exec:

command:

- cat

- /tmp/healthy

initialDelaySeconds: 5

periodSeconds: 5

timeoutSeconds: 30 )

* Kubectl apply -f liveness.yml
* Kubectl get pods
* Kubectl describe “pod name” ( you can see the detail of health pod )
* Kubectl get pods
* Kubectl exec mylivenessprobe -it -- /bin/bash ( go inside the container )
* Cat /tmp/healthy ( proper working livnessprobe)
* Echo $? ( o/p:- 0 “pod is healthy” if your pod is un healthy it will show you another number )