K8s lab part 3– YAMLling your first pod and storage

1. First we create a folder where we store our yaml files:

mkdir ~/yaml

mkdir ~/yaml/iscsi

cd ~/yaml/iscsi

1. Now we build out our very first yaml file that will provision storage. Please note the space!!! Indentation in yaml is very important (like Python is too for example):

nano pvc.yml

apiVersion: v1

kind: PersistentVolumeClaim

metadata:

name: pvc-unity

spec:

storageClassName: unity-iscsi

accessModes:

- ReadWriteOnce

resources:

requests:

storage: 5Gi

1. Now it is time to execute your yaml file and see if things work. Save and close, then perform these actions:

kubectl get pvc (this should return no PVCs)

kubectl apply -f pvc.yml

kubectl get pvc (this should return your first PVC!)

* Make sure the status of the pvc goes to “bound”. If this fails, check what is going wrong by using:

Kubectl describe pvc pvc-unity

1. Now that we have a pvc deployed, it is time to create a pod using the pvc. For this we build a second yaml file:

nano nginx-pvc.yml

kind: Pod

apiVersion: v1

metadata:

name: nginx-pvc-pod

spec:

volumes:

- name: unity-volume

persistentVolumeClaim:

claimName: pvc-unity

containers:

- name: nginx-pvc-pod

image: nginx

ports:

- containerPort: 80

name: "http-server"

volumeMounts:

- mountPath: "/export/nginx"

name: unity-volume

1. Apply this yaml file. Save and close, then perform these actions:

kubectl get pods (this should only return the snapshot helper pod)

kubectl apply -f nginx-pvc.yml

kubectl get pods (this should return a pod called “nginx-pvc-pod”).

* Make sure the status of the pod goes to “ready” (for the first pvc pod this can take quite long). If this fails (stuck on “ContainerCreating”), check what is going wrong by using:

kubectl describe pod nginx-pvc-pod