CKA MOCK EXAM 2.0

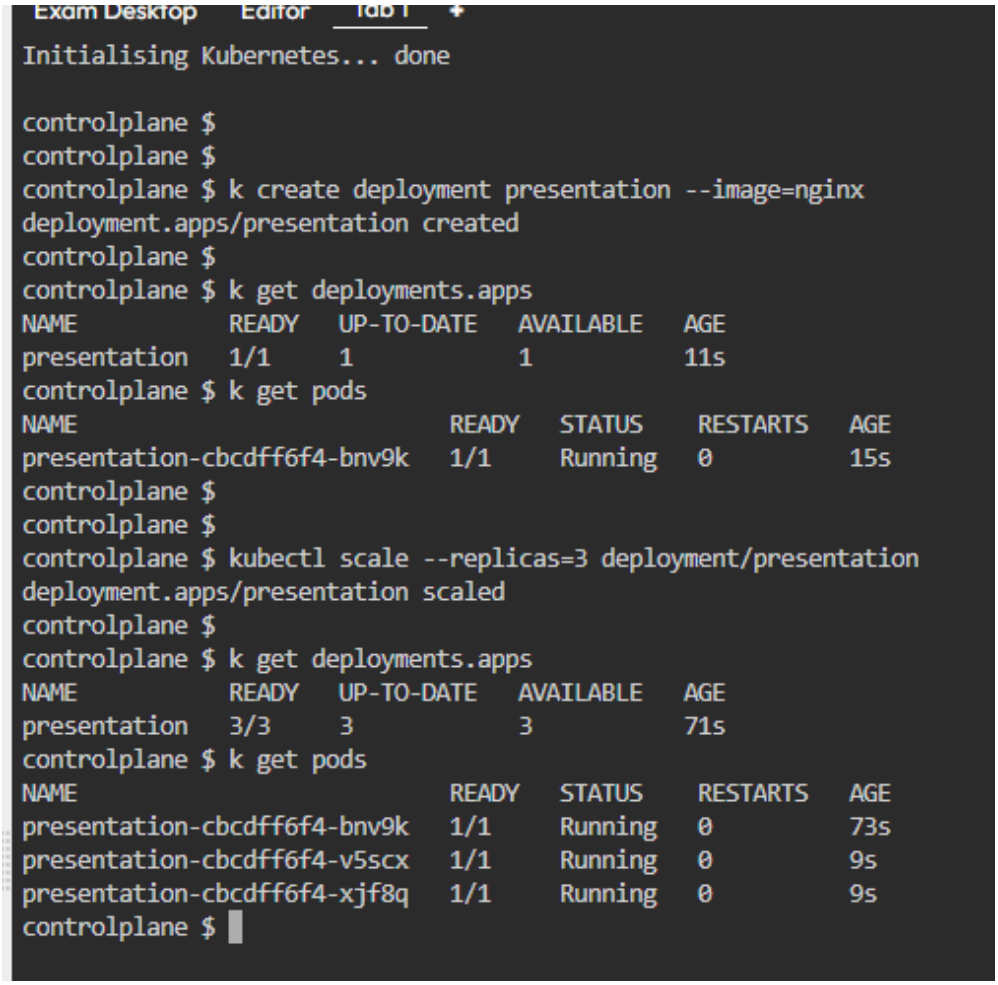
TIME:2 HOURS

Name: Haffiz Hissham

Date: 20 September 2024

Q1

Scale the deployment presentation to 3 pods.



Q2

Create a Persistent Volume with the given specification.
Volume Name: pv-demo
Storage:100Mi
Access modes: ReadWriteMany
HostPath: /pv/host-data

```
controlplane $
controlplane $
controlplane $ nano q2.yaml
controlplane $
controlplane $ k apply -f q2.yaml
persistentvolume/pv-demo created
controlplane $
controlplane $ k get pv
NAME      CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS      CLAIM   STORAGECLASS   VOLUMEATTRIBUTESCLASS   REASON   AGE
pv-demo   100Mi      RWX            Retain           Available           manual         <unset>                          3s
controlplane $
controlplane $
```

KILLERCODA PLUS

Exam Desktop Editor Tab 1 +

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv-demo
spec:
  capacity:
    storage: 100Mi
  accessModes:
  - ReadWriteMany
  storageClassName: manual
  hostPath:
    path: "/pv/host-data"
~
```

Q3

Monitor the logs of pod foo and:

⇨ Extract log lines

⇨ Write them to /opt/KUTR00101/foo

```
controlplane $
controlplane $ k run foo --image=nginx
pod/foo created
controlplane $
controlplane $ k get pods
NAME   READY   STATUS              RESTARTS   AGE
foo    0/1     ContainerCreating   0          3s
controlplane $
controlplane $ k logs foo
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2024/09/19 23:32:50 [notice] 1#1: using the "epoll" event method
2024/09/19 23:32:50 [notice] 1#1: nginx/1.27.1
2024/09/19 23:32:50 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2024/09/19 23:32:50 [notice] 1#1: OS: Linux 5.4.0-131-generic
2024/09/19 23:32:50 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2024/09/19 23:32:50 [notice] 1#1: start worker processes
2024/09/19 23:32:50 [notice] 1#1: start worker process 28
controlplane $
controlplane $ mkdir -p /opt/KUTR00101
controlplane $
controlplane $ k logs foo > /opt/KUTR00101/foo
controlplane $
controlplane $ cat /opt/KUTR00101/foo
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2024/09/19 23:32:50 [notice] 1#1: using the "epoll" event method
2024/09/19 23:32:50 [notice] 1#1: nginx/1.27.1
2024/09/19 23:32:50 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2024/09/19 23:32:50 [notice] 1#1: OS: Linux 5.4.0-131-generic
2024/09/19 23:32:50 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2024/09/19 23:32:50 [notice] 1#1: start worker processes
2024/09/19 23:32:50 [notice] 1#1: start worker process 28
controlplane $
```

Q4

There is a pod running in node my-pod
Take a backup of the pod ETCD database on /root/apna-backup.db
and then delete the pod and restore the backup and pod
/var/lib/apnabackup
And check the file apnabackup in /var/lib
pod must be running.

```
controlplane $
controlplane $ k run my-pod --image=nginx
pod/my-pod created
controlplane $
controlplane $ k get pods
NAME     READY   STATUS    RESTARTS   AGE
foo      1/1     Running   0          3m15s
my-pod   1/1     Running   0          4s
controlplane $
controlplane $
```

```
controlplane $
controlplane $ cat /etc/kubernetes/manifests/etcd.yaml | grep "trusted-ca-file" -B 20
spec:
  containers:
  - command:
    - etcd
    - --advertise-client-urls=https://172.30.1.2:2379
    - --cert-file=/etc/kubernetes/pki/etcd/server.crt
    - --client-cert-auth=true
    - --data-dir=/var/lib/etcd
    - --experimental-initial-corrupt-check=true
    - --experimental-watch-progress-notify-interval=5s
    - --initial-advertise-peer-urls=https://172.30.1.2:2380
    - --initial-cluster=controlplane=https://172.30.1.2:2380
    - --key-file=/etc/kubernetes/pki/etcd/server.key
    - --listen-client-urls=https://127.0.0.1:2379,https://172.30.1.2:2379
    - --listen-metrics-urls=http://127.0.0.1:2381
    - --listen-peer-urls=https://172.30.1.2:2380
    - --name=controlplane
    - --peer-cert-file=/etc/kubernetes/pki/etcd/peer.crt
    - --peer-client-cert-auth=true
    - --peer-key-file=/etc/kubernetes/pki/etcd/peer.key
    - --peer-trusted-ca-file=/etc/kubernetes/pki/etcd/ca.crt
    - --snapshot-count=10000
    - --trusted-ca-file=/etc/kubernetes/pki/etcd/ca.crt
```

```
controlplane $ ETCDCTL_API=3 etcdctl --endpoints=https://127.0.0.1:2379 \
>   --cacert=/etc/kubernetes/pki/etcd/ca.crt --cert=/etc/kubernetes/pki/etcd/server.crt --key=/etc/kubernetes/pki/etcd/server.key
 \
>   snapshot save /root/apna-backup.db
{"level":"info","ts":1726789123.5158374,"caller":"snapshot/v3_snapshot.go:68","msg":"created temporary db file","path":"/root/apn
a-backup.db.part"}
{"level":"info","ts":1726789123.5256088,"logger":"client","caller":"v3/maintenance.go:211","msg":"opened snapshot stream; downloa
ding"}
{"level":"info","ts":1726789123.5257561,"caller":"snapshot/v3_snapshot.go:76","msg":"fetching snapshot","endpoint":"https://127.0
.0.1:2379"}
{"level":"info","ts":1726789123.6728082,"logger":"client","caller":"v3/maintenance.go:219","msg":"completed snapshot read; closin
g"}
{"level":"info","ts":1726789123.687718,"caller":"snapshot/v3_snapshot.go:91","msg":"fetched snapshot","endpoint":"https://127.0.0
.1:2379","size":"6.6 MB","took":"now"}
{"level":"info","ts":1726789123.6877682,"caller":"snapshot/v3_snapshot.go:100","msg":"saved","path":"/root/apna-backup.db"}
Snapshot saved at /root/apna-backup.db
controlplane $
controlplane $ export ETCDCTL_API=3
controlplane $ etcdctl --write-out=table snapshot status /root/apna-backup.db
Deprecated: Use `etcdutl snapshot status` instead.

+----------+----------+------------+------------+
|   HASH   | REVISION | TOTAL KEYS | TOTAL SIZE |
+----------+----------+------------+------------+
| 8f8b3a74 |    21298 |        890 |     6.6 MB |
+----------+----------+------------+------------+
controlplane $
```

```
controlplane $
controlplane $ k delete pods my-pod
pod "my-pod" deleted
controlplane $
controlplane $ k delete pods foo
pod "foo" deleted
controlplane $
controlplane $
controlplane $ k get pods
No resources found in default namespace.
controlplane $
```

```
controlplane $ ETCDCTL_API=3 etcdctl --data-dir /var/lib/apnabackup --endpoints=https://127.0.0.1:2379 \
>   --cacert=/etc/kubernetes/pki/etcd/ca.crt --cert=/etc/kubernetes/pki/etcd/server.crt --key=/etc/kubernetes/pki/etcd/server.key
 \
>   snapshot restore /root/apna-backup.db
Deprecated: Use `etcdutl snapshot restore` instead.

2024-09-19T23:41:50Z    info    snapshot/v3_snapshot.go:251     restoring snapshot      {"path": "/root/apna-backup.db", "wal-dir
": "/var/lib/apnabackup/member/wal", "data-dir": "/var/lib/apnabackup", "snap-dir": "/var/lib/apnabackup/member/snap", "stack": "
go.etcd.io/etcd/etcdutl/v3/snapshot.(*v3Manager).Restore\n\t/tmp/etcd-release-3.5.0/etcd/release/etcd/etcdutl/snapshot/v3_snapsho
t.go:257\ngo.etcd.io/etcd/etcdutl/v3/etcdutl.SnapshotRestoreCommandFunc\n\t/tmp/etcd-release-3.5.0/etcd/release/etcd/etcdutl/etcd
utl/snapshot_command.go:147\ngo.etcd.io/etcd/etcdctl/v3/ctlv3/command.snapshotRestoreCommandFunc\n\t/tmp/etcd-release-3.5.0/etcd/
release/etcd/etcdctl/ctlv3/command/snapshot_command.go:128\ngithub.com/spf13/cobra.(*Command).execute\n\t/home/remote/sbatsche/.g
vm/pkgsets/go1.16.3/global/pkg/mod/github.com/spf13/cobra@v1.1.3/command.go:856\ngithub.com/spf13/cobra.(*Command).ExecuteC\n\t/h
ome/remote/sbatsche/.gvm/pkgsets/go1.16.3/global/pkg/mod/github.com/spf13/cobra@v1.1.3/command.go:960\ngithub.com/spf13/cobra.(*C
ommand).Execute\n\t/home/remote/sbatsche/.gvm/pkgsets/go1.16.3/global/pkg/mod/github.com/spf13/cobra@v1.1.3/command.go:897\ngo.et
cd.io/etcd/etcdctl/v3/ctlv3.Start\n\t/tmp/etcd-release-3.5.0/etcd/release/etcd/etcdctl/ctlv3/ctl.go:107\ngo.etcd.io/etcd/etcdctl/
v3/ctlv3.MustStart\n\t/tmp/etcd-release-3.5.0/etcd/release/etcd/etcdctl/ctlv3/ctl.go:111\nmain.main\n\t/tmp/etcd-release-3.5.0/et
cd/release/etcd/etcdctl/main.go:59\nruntime.main\n\t/home/remote/sbatsche/.gvm/gos/go1.16.3/src/runtime/proc.go:225"}
2024-09-19T23:41:50Z    info    membership/store.go:119 Trimming membership information from the backend...
2024-09-19T23:41:50Z    info    membership/cluster.go:393       added member    {"cluster-id": "cdf818194e3a8c32", "local-member-
id": "0", "added-peer-id": "8e9e05c52164694d", "added-peer-peer-urls": ["http://localhost:2380"]}
2024-09-19T23:41:51Z    info    snapshot/v3_snapshot.go:272     restored snapshot       {"path": "/root/apna-backup.db", "wal-dir
": "/var/lib/apnabackup/member/wal", "data-dir": "/var/lib/apnabackup", "snap-dir": "/var/lib/apnabackup/member/snap"}
controlplane $
controlplane $ ls /var/lib | grep backup
apnabackup
controlplane $ ls /var/lib | grep etcd
etcd
controlplane $
```

```
      name: etcd-data
    - mountPath: /etc/kubernetes/pki/etcd
      name: etcd-certs
  hostNetwork: true
  priority: 2000001000
  priorityClassName: system-node-critical
  securityContext:
    seccompProfile:
      type: RuntimeDefault
  volumes:
  - hostPath:
      path: /etc/kubernetes/pki/etcd
      type: DirectoryOrCreate
    name: etcd-certs
  - hostPath:
      path: /var/lib/apnabackup
      type: DirectoryOrCreate
    name: etcd-data
status: {}

File Name to Write: /etc/kubernetes/manifests/etcd.yaml
^G Get Help                    M-D DOS Format
```

```
controlplane $
controlplane $ nano /etc/kubernetes/manifests/etcd.yaml
controlplane $
controlplane $
controlplane $
controlplane $ k get pods
^C
controlplane $
controlplane $ k get pods

^C
controlplane $ k get pods
NAME     READY   STATUS    RESTARTS   AGE
foo      1/1     Running   0          11m
my-pod   1/1     Running   0          8m44s
controlplane $
```

Q5

Temporarily stop the kube-scheduler, this means in a way that you can start it again afterwards.

Create a single Pod named manual-schedule of image httpd:2.4-alpine, confirm it's created but not scheduled on any node.

Now you're the scheduler and have all its power, manually schedule that Pod on node with nodename. Make sure it's running.

Start the kube-scheduler again and confirm it's running correctly by creating a second Pod named manual-schedule2 of image httpd:2.4-alpine on controlplane

```
controlplane $
controlplane $ k get pods -A
NAMESPACE            NAME                                       READY   STATUS    RESTARTS        AGE
default              foo                                        1/1     Running   0               12m
default              my-pod                                     1/1     Running   0               9m29s
kube-system          calico-kube-controllers-75bdb5b75d-zhhrq   1/1     Running   3 (48s ago)     10d
kube-system          canal-fzfpm                                2/2     Running   2 (3h58m ago)   10d
kube-system          canal-szcfj                                2/2     Running   2 (3h58m ago)   10d
kube-system          coredns-5c69dbb7bd-298pn                   1/1     Running   1 (3h58m ago)   10d
kube-system          coredns-5c69dbb7bd-f6vzw                   1/1     Running   1 (3h58m ago)   10d
kube-system          etcd-controlplane                          0/1     Pending   0               45s
kube-system          kube-apiserver-controlplane                1/1     Running   2 (3h58m ago)   10d
kube-system          kube-controller-manager-controlplane       1/1     Running   3 (105s ago)    10d
kube-system          kube-proxy-ffdml                           1/1     Running   1 (3h58m ago)   10d
kube-system          kube-proxy-mvqrk                           1/1     Running   2 (3h58m ago)   10d
kube-system          kube-scheduler-controlplane                1/1     Running   3 (100s ago)    10d
local-path-storage   local-path-provisioner-75655fcf79-6xrsw    1/1     Running   2 (3h58m ago)   10d
controlplane $
controlplane $ k -n kube-system get kube-scheduler-controlplane -o yaml > schedule.yaml
error: the server doesn't have a resource type "kube-scheduler-controlplane"
controlplane $ k -n kube-system get pod kube-scheduler-controlplane -o yaml > schedule.yaml
controlplane $
controlplane $ nano schedule.yaml
controlplane $
```

```
controlplane $
controlplane $ k -n kube-system delete pods kube-scheduler-controlplane
pod "kube-scheduler-controlplane" deleted



^Ccontrolplane $
controlplane $ k get pods -A
NAMESPACE            NAME                                       READY   STATUS        RESTARTS        AGE
default              foo                                        1/1     Running       0               15m
default              my-pod                                     1/1     Running       0               11m
kube-system          calico-kube-controllers-75bdb5b75d-zhhrq   1/1     Running       3 (3m10s ago)   10d
kube-system          canal-fzfpm                                2/2     Running       2 (4h ago)      10d
kube-system          canal-szcfj                                2/2     Running       2 (4h ago)      10d
kube-system          coredns-5c69dbb7bd-298pn                   1/1     Running       1 (4h ago)      10d
kube-system          coredns-5c69dbb7bd-f6vzw                   1/1     Running       1 (4h ago)      10d
kube-system          etcd-controlplane                          0/1     Pending       0               3m7s
kube-system          kube-apiserver-controlplane                1/1     Running       2 (4h ago)      10d
kube-system          kube-controller-manager-controlplane       1/1     Running       3 (4m7s ago)    10d
kube-system          kube-proxy-ffdml                           1/1     Running       1 (4h ago)      10d
kube-system          kube-proxy-mvqrk                           1/1     Running       2 (4h ago)      10d
kube-system          kube-scheduler-controlplane                1/1     Terminating   3 (4m2s ago)    10d
local-path-storage   local-path-provisioner-75655fcf79-6xrsw    1/1     Running       2 (4h ago)      10d
controlplane $ k get pods -A
NAMESPACE            NAME                                       READY   STATUS        RESTARTS        AGE
```

Q6

Create a pod called pod-cka with two containers, as given below:
Container 1 - name: cool1, image: nginx
Container2 - name: cool2, image:
busybox,
command: sleep 3000

```
Exam Desktop   Editor   Tab 1   +
apiVersion: v1
kind: Pod
metadata:
  name: pod-cka
spec:
  containers:
  - name: cool1
    image: nginx
  - name: cool2
    image: busybox
    args:
    - sleep
    - "3000"
~
```

```
controlplane $ k get pods | grep cka
pod-cka          2/2     Running   0          38s
controlplane $
controlplane $ k describe pods pod-cka
Name:             pod-cka
Namespace:        default
Priority:         0
Service Account:  default
Node:             node01/172.30.2.2
Start Time:       Thu, 19 Sep 2024 23:52:43 +0000
Labels:           <none>
Annotations:      cni.projectcalico.org/containerID: 5abc6b1673f1e69f7e68a24dc10e311252e0340
                  cni.projectcalico.org/podIP: 192.168.1.7/32
                  cni.projectcalico.org/podIPs: 192.168.1.7/32
Status:           Running
IP:               192.168.1.7
IPs:
  IP:  192.168.1.7
Containers:
  cool1:
    Container ID:   containerd://03c0508dc121e9e2e94b0a99bca25bf07104a20e353e301e38f9a769bf4
    Image:          nginx
    Image ID:       docker.io/library/nginx@sha256:04ba374043ccd2fc5c593885c0eacddebabd5ca37
    Port:           <none>
    Host Port:      <none>
    State:          Running
      Started:      Thu, 19 Sep 2024 23:52:44 +0000
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-kgmbq (ro)
  cool2:
    Container ID:  containerd://88dc9f511c877eddf6ddb37a88319606559e00e0b2b368551557403a2728
    Image:         busybox
    Image ID:      docker.io/library/busybox@sha256:c230832bd3b0be59a6c47ed64294f9ce71e91b32
    Port:          <none>
    Host Port:     <none>
    Args:
      sleep
      3000
    State:          Running
```

Q7

create a deployment named source-ip-app that uses the image registry.k8s.io/echoserver:1.4 .

```
controlplane $
controlplane $ k create deployment source-ip-app --image=registry.k8s.io/echoserver:1.4
deployment.apps/source-ip-app created
controlplane $
controlplane $ k get deployments.apps
NAME            READY   UP-TO-DATE   AVAILABLE   AGE
source-ip-app   0/1     1            0           5s
controlplane $
controlplane $ k get pods | grep source
source-ip-app-f547dd7d4-llxrw   1/1     Running   0          22s
controlplane $
controlplane $ k describe deployments.apps source-ip-app
Name:                   source-ip-app
Namespace:              default
CreationTimestamp:      Thu, 19 Sep 2024 23:54:33 +0000
Labels:                 app=source-ip-app
Annotations:            deployment.kubernetes.io/revision: 1
Selector:               app=source-ip-app
Replicas:               1 desired | 1 updated | 1 total | 1 available | 0 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=source-ip-app
  Containers:
   echoserver:
    Image:        registry.k8s.io/echoserver:1.4
    Port:         <none>
    Host Port:    <none>
```

Q8

create a pod that will have two containers, one main container and another sidecar container that will collect the main containers logs

using kubectl, view the logs from the container named “sidecar”

main container args
['sh', '-c', 'while true do echo "$(date)\n" >> var/log/main-container.log; sleep5; done']

sidecar args
args: [/bin/sh, -c, 'tail -f /var/log/main-container.log']

```
Exam Desktop   Editor   Tab 1   +
  GNU nano 4.8                                    q8.yaml
apiVersion: v1
kind: Pod
metadata:
  name: sidecar-logs
spec:
  containers:
  - name: main
    image: busybox
    args: ['sh', '-c', 'while truel do echo "$(date)\n" >> var/log/main-container.log; sleep5; done']
  - name: sidecar
    image: busybox
    args: [/bin/sh, -c, 'tail -f /var/log/main-container.log']
```

Q9

Create a new PersistentVolume named safari-pv. It should have a capacity of 2Gi, accessMode ReadWriteOnce, hostPath /Volumes/Data and no storageClassName defined.

Next create a new PersistentVolumeClaim in Namespace project-tiger named safari-pvc . It should request 2Gi storage, accessMode ReadWriteOnce and should not define a storageClassName. The PVC should bound to the PV correctly.

Finally create a new Deployment safari in Namespace project-tiger which mounts that volume at /tmp/safari-data. The Pods of that Deployment should be of image httpd:2.4.41-alpine.

```
controlplane $
controlplane $ k create ns project-tiger
namespace/project-tiger created
controlplane $
controlplane $ k get ns
NAME                 STATUS   AGE
default              Active   10d
kube-node-lease      Active   10d
kube-public          Active   10d
kube-system          Active   10d
local-path-storage   Active   10d
project-tiger        Active   5s
controlplane $
```

Exam Desktop Editor Tab 1 +

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: safari-pv
  labels:
    type: local
spec:
  storageClassName: ""
  capacity:
    storage: 2Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: "/Volumes/Data"

---

apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: safari-pvc
  namespace: project-tiger
spec:
  storageClassName: ""
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 2Gi
~
```

```
controlplane $
controlplane $ vim pv-pvc.yaml
controlplane $ k apply -f pv-pvc.yaml
persistentvolume/safari-pv created
persistentvolumeclaim/safari-pvc created
controlplane $
controlplane $ k get pvc
No resources found in default namespace.
controlplane $ k get pvc -n project-tiger
NAME         STATUS   VOLUME      CAPACITY   ACCESS MODES   STORAGECLASS   VOLUMEATTRIBUTESCLASS   AGE
safari-pvc   Bound    safari-pv   2Gi        RWO                           <unset>                 12s
controlplane $
```

Exam Desktop Editor Tab 1 +

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  creationTimestamp: null
  labels:
    app: safari
  name: safari
  namespace: project-tiger
spec:
  replicas: 1
  selector:
    matchLabels:
      app: safari
  strategy: {}
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: safari
    spec:
      volumes:
        - name: task-pv-storage
          persistentVolumeClaim:
            claimName: safari-pvc
      containers:
      - image: httpd:2.4.41-alpine
        name: httpd
        volumeMounts:
          - mountPath: "/tmp/safari-data"
            name: task-pv-storage
        resources: {}
status: {}
~
```

```
controlplane $
controlplane $ vim safari.yaml
controlplane $
controlplane $ k apply -f safari.yaml
deployment.apps/safari created
controlplane $
controlplane $ k get deployments.apps -n project-tiger
NAME     READY   UP-TO-DATE   AVAILABLE   AGE
safari   1/1     1            1           15s
controlplane $
controlplane $ k describe deployments.apps safari -n project-tiger
Name:                   safari
Namespace:              project-tiger
CreationTimestamp:      Fri, 20 Sep 2024 00:12:35 +0000
Labels:                 app=safari
Annotations:            deployment.kubernetes.io/revision: 1
Selector:               app=safari
Replicas:               1 desired | 1 updated | 1 total | 1 available | 0 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=safari
  Containers:
   httpd:
    Image:        httpd:2.4.41-alpine
    Port:         <none>
    Host Port:    <none>
    Environment:  <none>
    Mounts:
      /tmp/safari-data from task-pv-storage (rw)
  Volumes:
   task-pv-storage:
    Type:          PersistentVolumeClaim (a reference to a PersistentVolumeClaim in the same namespace)
    ClaimName:     safari-pvc
    ReadOnly:      false
  Node-Selectors:  <none>
  Tolerations:     <none>
Conditions:
  Type           Status  Reason
  ----           ------  ------
```

Q10

Create a new deployment called mockpod, with image nginx:1.16 and 1 replica.
Next upgrade the deployment to version 1.17 using rolling update
Make sure that the version upgrade is recorded in the resource annotation

```
controlplane $
controlplane $ k create deployment mockpod --image=nginx:1.16 --replicas=1
deployment.apps/mockpod created
controlplane $
controlplane $ k get deployments.apps
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
mockpod        0/1     1            0           5s
source-ip-app  1/1     1            1           19m
controlplane $ k get deployments.apps
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
mockpod        1/1     1            1           8s
source-ip-app  1/1     1            1           19m
controlplane $
controlplane $ kubectl set image deployment/mockpod nginx=nginx:1.17 --record
Flag --record has been deprecated, --record will be removed in the future
deployment.apps/mockpod image updated
controlplane $
controlplane $ kubectl rollout history deployment/mockpod
deployment.apps/mockpod
REVISION  CHANGE-CAUSE
1         <none>
2         kubectl set image deployment/mockpod nginx=nginx:1.17 --record=true

controlplane $
controlplane $
```

Q11

write a command into /opt/course/100/cluster_events.sh which shows the latest events in the whole cluster, ordered by time (metadata.creationtimestamp). use kubectl for it.
now delete the kube-proxy pod running on node controlpane node and write the events this caused
into /opt/course/100/pod_kill.log.

```
controlplane $
controlplane $ mkdir -p /opt/course/100/
controlplane $ echo -e "kubectl get events --sort-by=.metadata.creationTimestamp
> ^C
controlplane $
controlplane $
controlplane $ echo -e "kubectl get events --sort-by=.metadata.creationTimestamp" > /opt/course/100/cluster_events.sh
controlplane $
controlplane $ cat /opt/course/100/cluster_events.sh
kubectl get events --sort-by=.metadata.creationTimestamp
controlplane $
```

```
controlplane $
controlplane $ k get pods -A -o wide | grep proxy
kube-system          kube-proxy-ffdml                          1/1     Running
  node01          <none>           <none>
kube-system          kube-proxy-mvqrk                          1/1     Running
  controlplane    <none>           <none>
controlplane $
controlplane $
controlplane $ k delete pods kube-proxy-mvqrk -n kube-system
pod "kube-proxy-mvqrk" deleted
controlplane $
controlplane $ k get pods -A -o wide | grep proxy
kube-system          kube-proxy-6g2m5                          1/1     Running
  controlplane    <none>           <none>
kube-system          kube-proxy-ffdml                          1/1     Running
  node01          <none>           <none>
controlplane $
controlplane $ k delete pods kube-proxy-6g2m5 -n kube-system
pod "kube-proxy-6g2m5" deleted
controlplane $
controlplane $ k get pods -A -o wide | grep proxy
kube-system          kube-proxy-ffdml                          1/1     Running
 node01          <none>           <none>
kube-system          kube-proxy-jvlkz                          1/1     Running
 controlplane    <none>           <none>
controlplane $
```

```
controlplane $
controlplane $ bash /opt/course/100/cluster_events.sh > /opt/course/100/pod_kill.log
controlplane $
controlplane $ cat /opt/course/100/pod_kill.log | tail
4m58s       Normal    ScalingReplicaSet   deployment/mockpod                Scaled up replica set mockpod-
4m57s       Normal    Scheduled           pod/mockpod-76cc984cd-b67p9       Successfully assigned default/
7p9 to node01
4m52s       Normal    Started             pod/mockpod-76cc984cd-b67p9       Started container nginx
4m52s       Normal    Created             pod/mockpod-76cc984cd-b67p9       Created container nginx
4m52s       Normal    Pulled              pod/mockpod-76cc984cd-b67p9       Successfully pulled image "ng
5.884s including waiting). Image size: 51030575 bytes.
4m51s       Normal    Killing             pod/mockpod-7686cdbc85-pw2c5      Stopping container nginx
4m51s       Normal    SuccessfulDelete    replicaset/mockpod-7686cdbc85     Deleted pod: mockpod-7686cdbc
4m51s       Normal    ScalingReplicaSet   deployment/mockpod                Scaled down replica set mockpo
om 1
2m10s       Normal    Starting            node/controlplane
104s        Normal    Starting            node/controlplane
controlplane $
```

Q12

create a new user “sam” .grant him access to the cluster.

user “sam” should have permission to create and delete secrets. the private key exists at location: EXPIRATIONSECONDS 65 DAYS

/root/sam/.key and csr at /root/sam.csr

```
controlplane $
controlplane $ openssl genrsa -out sam.key 2048 && openssl req -new -key sam.key -out sam.csr -subj "/CN=sam"
Generating RSA private key, 2048 bit long modulus (2 primes)
.....................................................................+++++
.....................................................................................................................
.....................................................+++++
e is 65537 (0x010001)
controlplane $ ls
filesystem  sam.csr  sam.key  snap
controlplane $
controlplane $ cat sam.csr | base64 | tr -d "\n\n"
LS0tLS1CRUdJTiBDRVJUSUZJQ0FURSBSRVFVRVNULS0tLS0KTUlJQ1V6Q0NBVHNDQVFBd0RqRU1NQW9HQTFVRUF3d0RjMkZ0TUlJQklqQU5CZ2txaGtpRzl3MEJBUUVGQ
UFPQwpBUThBTUlJQkNnS0NBUUVBdVFWVW9hSDI4L25LcGdnNlhUSEhvWVRHYUZnK1p5a2V0dkQ3MllWbHc1cXdvaUxiClVvRkdoMkE4NTdPdllsVTdkL2hLMVlEWU5nYn
NNODFxWnJWei9DbklwdHZaOG5aSUo3MDBHT0tabEs4Y1dDVlgKbk9yOXpRU3U3ak1ubzFRTlhhZDZGU2QrdG9HZlg0ekxVSFNqQU82b3d1Z0c2TEZDMjZmQWJGMXhPei9
vYW1pZwoxd0NLakNUcEFCYkJ4VFdLQVZtbGkwbkVQZTNhUTd0QkNXV0FKamxkSk1LWlcvZGw4QjhnaUMyWkp2RUZ3d2NvClZHWVFWL2FWc1Vad3FEcDdKOTVCMklTU0FW
TFRMZjhxZFQwR2huTElDdjdQa3MwSWtDYmROaURMMTVVUEtlb0cKZ2hpbjJmZHhmUlBiS2tKRG90YmVBM2pkNFRoVUF4TTh6MUk5eVFJREFRQUJvQUF3RFFZSktvWklod
mNOQVFFTApCUUFEZ2dFQkFFbHF4N29RanlZb0hKbURiWFR2OHEraE1jWkdVbkExRnRnYUI3QTFuS21Db1QxaUhRam1jV0ZSCmhPTmFkOUNwRmd4OFlDRlBrYkhKeFNoMD
lsdk02TEk4b1RqMG9Pb0xzT1JDMi9mVmV1b3MrQW4wNkpDcjFPTkwKZzVHaDhYclF3THU4bG1nK3hRYTQzc29Jb0YybUhlNUVNR2c3WnUrck5tbmg5bWdIcnFZSzloYnN
xR0V6RENpbgoxNGFwRld2UUl1cFJSY0ZXRGdPOUY1MW9tRVpGR2NNVnFHa09FVmUwakk0T2F5bG5aNk9iaTlnVXRlV2VrNDBDCkNWRCtDY243dzVNeEFiWGNlRTZUeWZP
T1NYWnZqQmtib3F6dDhTd1EwV0RqS1RDaWorbzVNRGhXZGpEQ0kxZjQKQnd0UlJROXlER1FqSGxzQVRlT3pwaThlc2VsYnJDST0KLS0tLS1FTkQgQ0VSVElGSUNBVEUgU
kVRVUVTVC0tLS0tCg==controlplane $ cat sam.csr | base64 | tr -d "\n"
LS0tLS1CRUdJTiBDRVJUSUZJQ0FURSBSRVFVRVNULS0tLS0KTUlJQ1V6Q0NBVHNDQVFBd0RqRU1NQW9HQTFVRUF3d0RjMkZ0TUlJQklqQU5CZ2txaGtpRzl3MEJBUUVGQ
UFPQwpBUThBTUlJQkNnS0NBUUVBdVFWVW9hSDI4L25LcGdnNlhUSEhvWVRHYUZnK1p5a2V0dkQ3MllWbHc1cXdvaUxiClVvRkdoMkE4NTdPdllsVTdkL2hLMVlEWU5nYn
NNODFxWnJWei9DbklwdHZaOG5aSUo3MDBHT0tabEs4Y1dDVlgKbk9yOXpRU3U3ak1ubzFRTlhhZDZGU2QrdG9HZlg0ekxVSFNqQU82b3d1Z0c2TEZDMjZmQWJGMXhPei9
vYW1pZwoxd0NLakNUcEFCYkJ4VFdLQVZtbGkwbkVQZTNhUTd0QkNXV0FKamxkSk1LWlcvZGw4QjhnaUMyWkp2RUZ3d2NvClZHWVFWL2FWc1Vad3FEcDdKOTVCMklTU0FW
TFRMZjhxZFQwR2huTElDdjdQa3MwSWtDYmROaURMMTVVUEtlb0cKZ2hpbjJmZHhmUlBiS2tKRG90YmVBM2pkNFRoVUF4TTh6MUk5eVFJREFRQUJvQUF3RFFZSktvWklod
mNOQVFFTApCUUFEZ2dFQkFFbHF4N29RanlZb0hKbURiWFR2OHEraE1jWkdVbkExRnRnYUI3QTFuS21Db1QxaUhRam1jV0ZSCmhPTmFkOUNwRmd4OFlDRlBrYkhKeFNoMD
lsdk02TEk4b1RqMG9Pb0xzT1JDMi9mVmV1b3MrQW4wNkpDcjFPTkwKZzVHaDhYclF3THU4bG1nK3hRYTQzc29Jb0YybUhlNUVNR2c3WnUrck5tbmg5bWdIcnFZSzloYnN
xR0V6RENpbgoxNGFwRld2UUl1cFJSY0ZXRGdPOUY1MW9tRVpGR2NNVnFHa09FVmUwakk0T2F5bG5aNk9iaTlnVXRlV2VrNDBDCkNWRCtDY243dzVNeEFiWGNlRTZUeWZP
T1NYWnZqQmtib3F6dDhTd1EwV0RqS1RDaWorbzVNRGhXZGpEQ0kxZjQKQnd0UlJROXlER1FqSGxzQVRlT3pwaThlc2VsYnJDST0KLS0tLS1FTkQgQ0VSVElGSUNBVEUgU
kVRVUVTVC0tLS0tCg==controlplane $
controlplane $
controlplane $
```

```
Exam Desktop    Editor    Tab 1    +
  GNU nano 4.8                                                  csr.yaml
apiVersion: certificates.k8s.io/v1
kind: CertificateSigningRequest
metadata:
  name: sam
spec:
  request: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURSBSRVFVRVNULS0tLS0KTUlJQ1V6Q0NBVHND
  signerName: kubernetes.io/kube-apiserver-client
  expirationSeconds: 5616000
  usages:
  - client auth
```

```
controlplane $
controlplane $
controlplane $ nano csr.yaml
controlplane $ nano csr.yaml
controlplane $ k apply -f csr.yaml
certificatesigningrequest.certificates.k8s.io/sam created
controlplane $ kubectl get csr
NAME        AGE   SIGNERNAME                                    REQUESTOR                 REQUESTEDDURATION   CONDITION
csr-rg496   10d   kubernetes.io/kube-apiserver-client-kubelet   system:bootstrap:4y3sei   <none>              Approved,Issued
csr-tmfl9   11d   kubernetes.io/kube-apiserver-client-kubelet   system:node:controlplane  <none>              Approved,Issued
sam         5s    kubernetes.io/kube-apiserver-client           kubernetes-admin          65d                 Pending
controlplane $ kubectl certificate approve sam
certificatesigningrequest.certificates.k8s.io/sam approved
controlplane $ kubectl get csr
NAME        AGE   SIGNERNAME                                    REQUESTOR                 REQUESTEDDURATION   CONDITION
csr-rg496   10d   kubernetes.io/kube-apiserver-client-kubelet   system:bootstrap:4y3sei   <none>              Approved,Issued
csr-tmfl9   11d   kubernetes.io/kube-apiserver-client-kubelet   system:node:controlplane  <none>              Approved,Issued
sam         19s   kubernetes.io/kube-apiserver-client           kubernetes-admin          65d                 Approved,Issued
controlplane $
```

```
controlplane $
controlplane $ kubectl get csr sam -o jsonpath='{.status.certificate}'| base64 -d > sam.crt
controlplane $ ls
csr.yaml  filesystem  sam.crt  sam.csr  sam.key  snap
controlplane $
controlplane $ kubectl create role developer --verb=create --verb=get --verb=delete --resource=secrets
role.rbac.authorization.k8s.io/developer created
controlplane $ kubectl create rolebinding developer-binding-sam --role=developer --user=sam
rolebinding.rbac.authorization.k8s.io/developer-binding-sam created
controlplane $
controlplane $ kubectl config set-credentials sam --client-key=sam.key --client-certificate=sam.crt --embed-certs=true
User "sam" set.
controlplane $ kubectl config set-context sam --cluster=kubernetes --user=sam
Context "sam" created.
controlplane $
controlplane $ kubectl config set-context
error: you must specify a non-empty context name or --current
controlplane $ kubectl config get-context
error: unknown command "get-context"
See 'kubectl config -h' for help and examples
controlplane $ kubectl config get-contexts
CURRENT   NAME                          CLUSTER      AUTHINFO           NAMESPACE
*         kubernetes-admin@kubernetes   kubernetes   kubernetes-admin
          sam                           kubernetes   sam
controlplane $
```

Q13

use json path query to retrieve the osimages of all the nodes and store it in a file “all-nodes-os-info.txt” at root location.

note: the osimage are under the nodeInfo section under status of each node.

```
controlplane $
controlplane $ kubectl get nodes -o=jsonpath='{.items[*].status.nodeInfo.osImage}'
Ubuntu 20.04.5 LTS Ubuntu 20.04.5 LTScontrolplane $
controlplane $
controlplane $ kubectl get nodes -o=jsonpath='{.items[*].status.nodeInfo.osImage}' > &amp;quot;all-nodes-os-
bash: syntax error near unexpected token `&'
controlplane $ kubectl get nodes -o=jsonpath='{.items[*].status.nodeInfo.osImage}' > all-nodes-os-info.txt
controlplane $
controlplane $ cat all-nodes-os-info.txt
Ubuntu 20.04.5 LTS Ubuntu 20.04.5 LTScontrolplane $
```

Q14

list the services on your linux operating system that are associated with kubernetes. save the output to a file named services.csv.

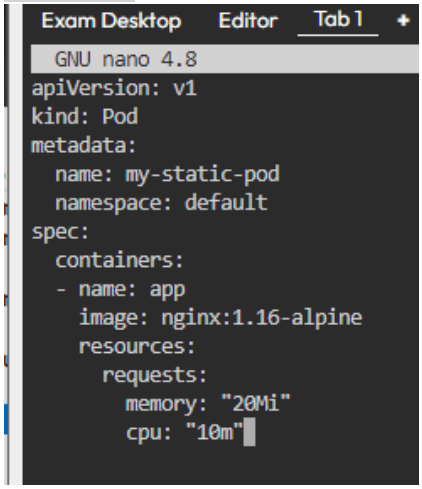
```
controlplane $
controlplane $ systemctl status | grep kubernetes > services.csv
controlplane $ cat services.csv
           │ │ └─1611 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --kubeconfig=/etc/kubernetes
/kubelet.conf --config=/var/lib/kubelet/config.yaml --container-runtime-endpoint=unix:///var/run/containerd/containerd.sock --pod
-infra-container-image=registry.k8s.io/pause:3.9 --container-runtime-endpoint unix:///run/containerd/containerd.sock --cgroup-dri
ver=systemd --eviction-hard imagefs.available<5%,memory.available<100Mi,nodefs.available<5% --fail-swap-on=false
           │ │ └─13327 grep --color=auto kubernetes
           │ │   └─2027 kube-controller-manager --allocate-node-cidrs=true --authentication-kubeconfig=/etc/kubernetes/controll
er-manager.conf --authorization-kubeconfig=/etc/kubernetes/controller-manager.conf --bind-address=127.0.0.1 --client-ca-file=/etc
/kubernetes/pki/ca.crt --cluster-cidr=192.168.0.0/16 --cluster-name=kubernetes --cluster-signing-cert-file=/etc/kubernetes/pki/ca
.crt --cluster-signing-key-file=/etc/kubernetes/pki/ca.key --controllers=*,bootstrapsigner,tokencleaner --kubeconfig=/etc/kuberne
tes/controller-manager.conf --leader-elect=true --requestheader-client-ca-file=/etc/kubernetes/pki/front-proxy-ca.crt --root-ca-f
ile=/etc/kubernetes/pki/ca.crt --service-account-private-key-file=/etc/kubernetes/pki/sa.key --service-cluster-ip-range=10.96.0.0
/12 --use-service-account-credentials=true
           │ │ └─2079 etcd --advertise-client-urls=https://172.30.1.2:2379 --cert-file=/etc/kubernetes/pki/etcd/server.crt --
client-cert-auth=true --data-dir=/var/lib/etcd --experimental-initial-corrupt-check=true --experimental-watch-progress-notify-int
erval=5s --initial-advertise-peer-urls=https://172.30.1.2:2380 --initial-cluster=controlplane=https://172.30.1.2:2380 --key-file=
/etc/kubernetes/pki/etcd/server.key --listen-client-urls=https://127.0.0.1:2379,https://172.30.1.2:2379 --listen-metrics-urls=htt
p://127.0.0.1:2381 --listen-peer-urls=https://172.30.1.2:2380 --name=controlplane --peer-cert-file=/etc/kubernetes/pki/etcd/peer.
crt --peer-client-cert-auth=true --peer-key-file=/etc/kubernetes/pki/etcd/peer.key --peer-trusted-ca-file=/etc/kubernetes/pki/etc
d/ca.crt --snapshot-count=10000 --trusted-ca-file=/etc/kubernetes/pki/etcd/ca.crt
           │ │ └─2051 kube-scheduler --authentication-kubeconfig=/etc/kubernetes/scheduler.conf --authorization-kubeconfig=/e
tc/kubernetes/scheduler.conf --bind-address=127.0.0.1 --kubeconfig=/etc/kubernetes/scheduler.conf --leader-elect=true
           │ │ └─2014 kube-apiserver --advertise-address=172.30.1.2 --allow-privileged=true --authorization-mode=Node,RBAC --
client-ca-file=/etc/kubernetes/pki/ca.crt --enable-admission-plugins=NodeRestriction --enable-bootstrap-token-auth=true --etcd-ca
file=/etc/kubernetes/pki/etcd/ca.crt --etcd-certfile=/etc/kubernetes/pki/apiserver-etcd-client.crt --etcd-keyfile=/etc/kubernetes
/pki/apiserver-etcd-client.key --etcd-servers=https://127.0.0.1:2379 --kubelet-client-certificate=/etc/kubernetes/pki/apiserver-k
ubelet-client.crt --kubelet-client-key=/etc/kubernetes/pki/apiserver-kubelet-client.key --kubelet-preferred-address-types=Interna
lIP,ExternalIP,Hostname --proxy-client-cert-file=/etc/kubernetes/pki/front-proxy-client.crt --proxy-client-key-file=/etc/kubernet
es/pki/front-proxy-client.key --requestheader-allowed-names=front-proxy-client --requestheader-client-ca-file=/etc/kubernetes/pki
/front-proxy-ca.crt --requestheader-extra-headers-prefix=X-Remote-Extra- --requestheader-group-headers=X-Remote-Group --requesthe
ader-username-headers=X-Remote-User --secure-port=6443 --service-account-issuer=https://kubernetes.default.svc.cluster.local --se
rvice-account-key-file=/etc/kubernetes/pki/sa.pub --service-account-signing-key-file=/etc/kubernetes/pki/sa.key --service-cluster
-ip-range=10.96.0.0/12 --tls-cert-file=/etc/kubernetes/pki/apiserver.crt --tls-private-key-file=/etc/kubernetes/pki/apiserver.key
controlplane $
```

Q15

Q16

create a static pod named my-static-pod in namespace default on cluster3-controlplane1. it should be of image nginx:1.16-alpine and have resource requests for 10m cpu and 20mi memory.

then create a nodeport service named static-pod-service which exposes that static pod on port 80 and check if it's reachable through the controlplane internal ip address




```
controlplane $
controlplane $ k run my-pod --image=nginx
pod/my-pod created
controlplane $
controlplane $ k get pods
NAME                          READY   STATUS              RESTARTS       AGE
my-pod                        0/1     ContainerCreating   0              4s
my-static-pod-controlplane    1/1     Running             0              2m21s
output-pod                    0/1     CrashLoopBackOff    6 (2m2s ago)   7m42s
controlplane $
```

```
controlplane $
controlplane $ cd /etc/kubernetes/manifests/
controlplane $ pwd
/etc/kubernetes/manifests
controlplane $
controlplane $ nano my-static-pod
controlplane $
controlplane $ ls
etcd.yaml  kube-apiserver.yaml  kube-controller-manager.yaml  kube-scheduler.yaml  my-static-pod
controlplane $
controlplane $ k get pods -A
NAMESPACE           NAME                                       READY   STATUS             RESTARTS        AGE
default             my-static-pod-controlplane                 1/1     Running            0               11s
default             output-pod                                 0/1     CrashLoopBackOff   5 (2m43s ago)   5m32s
kube-system         calico-kube-controllers-75bdb5b75d-zhhrq   1/1     Running            2 (30m ago)     11d
kube-system         canal-fzfpm                                2/2     Running            2 (30m ago)     11d
kube-system         canal-szcfj                                2/2     Running            2 (30m ago)     11d
kube-system         coredns-5c69dbb7bd-298pn                   1/1     Running            1 (30m ago)     11d
kube-system         coredns-5c69dbb7bd-f6vzw                   1/1     Running            1 (30m ago)     11d
kube-system         etcd-controlplane                          1/1     Running            2 (30m ago)     11d
kube-system         kube-apiserver-controlplane                1/1     Running            2 (30m ago)     11d
kube-system         kube-controller-manager-controlplane       1/1     Running            2 (30m ago)     11d
kube-system         kube-proxy-ffdml                           1/1     Running            1 (30m ago)     11d
kube-system         kube-proxy-mvqrk                           1/1     Running            2 (30m ago)     11d
kube-system         kube-scheduler-controlplane                1/1     Running            2 (30m ago)     11d
local-path-storage  local-path-provisioner-75655fcf79-6xrsw    1/1     Running            2 (30m ago)     11d
controlplane $
```

Q17

Create a NodePort service to expose a pod named my-pod on port 8080, with the NodePort set to
30060.

```
Exam Desktop   Editor   Tab 1   +
  GNU nano 4.8
apiVersion: v1
kind: Service
metadata:
  creationTimestamp: null
  labels:
    app: my-pod-svc
  name: my-pod-svc
spec:
  ports:
  - name: my-pod-svc
    nodePort: 30060
    port: 8080
    protocol: TCP
    targetPort: 8080
  selector:
    app.kubernetes.io/name: my-pod
  type: NodePort
status:
  loadBalancer: {}
```

```
controlplane $
controlplane $ nano svc.yaml
controlplane $ k apply -f svc.yaml
service/my-pod-svc configured
controlplane $
controlplane $ k describe svc my-pod-svc
Name:                     my-pod-svc
Namespace:                default
Labels:                   app=my-pod-svc
Annotations:              <none>
Selector:                 app.kubernetes.io/name=my-pod
Type:                     NodePort
IP Family Policy:         SingleStack
IP Families:              IPv4
IP:                       10.104.49.163
IPs:                      10.104.49.163
Port:                     my-pod-svc  8080/TCP
TargetPort:               8080/TCP
NodePort:                 my-pod-svc  30060/TCP
Endpoints:                <none>
Session Affinity:         None
External Traffic Policy:  Cluster
Events:                   <none>
controlplane $ k get svc
NAME         TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
kubernetes   ClusterIP   10.96.0.1       <none>        443/TCP          11d
my-pod-svc   NodePort    10.104.49.163   <none>        8080:30060/TCP   4m21s
controlplane $
```