## **MOCK EXAM 1.0**

###### **Name: Haffiz Hissham**

###### **Date: 22 September 2024**

#### **Q1**

Create a nginx pod called dns-resolver using image nginx expose it internally with a

service called dns-resolver-service.

check if pod and service name are resolvable from within the cluster.

use the image: busybox:1.28 for dns lookup

save the result in /root/nginx.svc.A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

#### **Q2**

Create a persistent volume with name app-data, of capacity 2Gi and access mode ReadOnlyMany.

The type of volume is hostPath and its location is /srv/app- data

A screen shot of a computer

Description automatically generated

A black background with white text

Description automatically generated

#### **Q3**

Check to see how many nodes are ready (not including nodes tainted NoSchedule) and write the

number to /opt/KUSC00402/kusc00402.txt.

A screen shot of a computer

Description automatically generated

#### **Q4**

Create a new pod called mock-pod with image busy box

Allow the pod to be able to set system\_time

The container should sleep for 4000 seconds

A screenshot of a computer

Description automatically generatedA screen shot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

#### **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

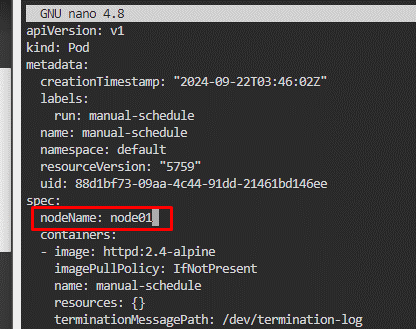
named manual-schedule2 of image httpd:2.4-alpine on controlplane

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated



A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

A computer screen with white text

Description automatically generated

A black background with white text

Description automatically generated

#### **Q6**

Create a pod called pod-cka with two containers, as given below:

Container 1 - name: cka2, image: nginx

Container2 - name: cka2, image:

busybox,

command: sleep 3000

A screenshot of a computer

Description automatically generated

A screen shot of a computer program

Description automatically generated

#### **Q7**

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

A screenshot of a computer program

Description automatically generated

#### **Q8**

create a pod as follows:

name:mongo

using image:mongo

in anew kubernetes namespacenamed:my-website

A screenshot of a computer

Description automatically generated

#### **Q9**

You’re ask to find out following information about the cluster :

How many controlplane nodes are available?

How many worker nodes are available?

how many static pods are running

Write your answers into file /opt/course/14/cluster-info, structured like this:

# /opt/course/14/cluster-info

1: [ANSWER]

2: [ANSWER]

3: [ANSWER]  
A screenshot of a computer

Description automatically generated

A screen shot of a computer

Description automatically generated

#### **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

A screenshot of a computer

Description automatically generated

#### **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.

A computer screen shot of a program

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screen shot of a computer code

Description automatically generated

#### **Q12**

create a deployment called pod-cka with two containers, as given below:

container 1 - name: cka1, image: nginx

container2 - name: cka2, image:busybox,

command: sleep 5000

A screenshot of a computer

Description automatically generatedA computer screen shot of white text

Description automatically generated

A screenshot of a computer

Description automatically generated

#### **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 nodelnfo section under status of each node.

A screen shot of a computer

Description automatically generated

#### **Q14**

create a new persistentvolumeclaim:

✑ name: pv-volume

✑ class: csi-hostpath-sc

✑ capacity: 10mi

create a new pod which mounts the persistentvolumeclaim as a volume:

✑ name: web-server

✑ image: nginx

✑ mount path: /usr/share/nginx/html

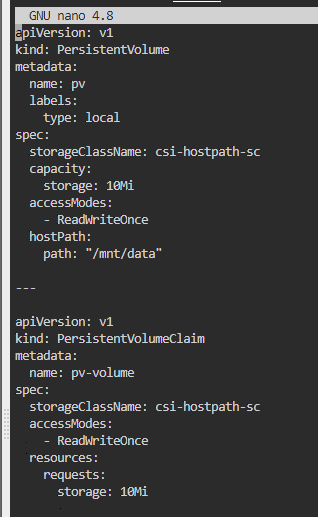
configure the new pod to have readwriteonce access on the volume.

finally, using kubectl edit or kubectl patch expand the persistentvolumeclaim to a

capacity of 70mi and record that change.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer screen

Description automatically generated A screenshot of a computer program

Description automatically generated

A screenshot of a computer error

Description automatically generated

#### **Q15**

create a static pod named static-control on the control plane node that uses the nginx:1.17

A screenshot of a computer program

Description automatically generated

**Q16**

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.

A screenshot of a computer program

Description automatically generated

A screen shot of a computer

Description automatically generated A screen shot of a computer

Description automatically generated

A screen shot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screen shot of a computer

Description automatically generated

#### **Q17**

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

**30080**.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated