DUMPSQUEEN

Certified Kubernetes Application Developer (CKAD) Program

Linux Foundation CKAD

Version Demo

Total Demo Questions: 5

Total Premium Questions: 33

Buy Premium PDF

https://dumpsqueen.com

support@dumpsqueen.com

QUESTION NO: 1 - (SIMULATION)

Context

You are tasked to create a secret and consume the secret in a pod using environment variables as follow:

Task

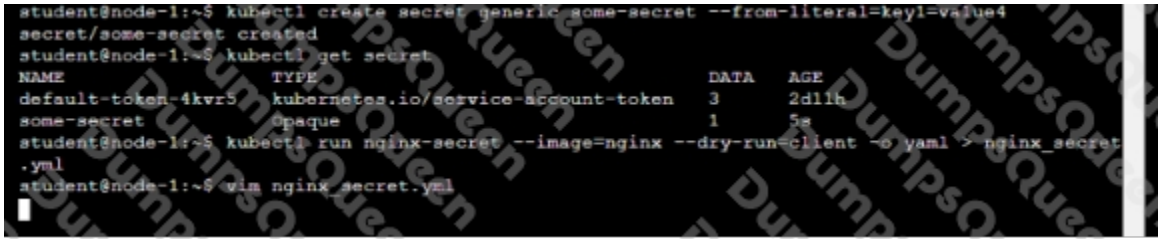
• Create a secret named another-secret with a key/value pair; key1/value4

• Start an nginx pod named nginx-secret using container image nginx, and add an environment variable exposing the value of the secret key key 1, using COOL_VARIABLE as the name for the environment variable inside the pod

ANSWER: Seethesolutionbelow.

Explanation:

Solution:




Readme
Web Terminal
THE LINUX FOUNDATION
apiVersion: v1
kind: Pod
metadata:
creationTimestamp: null
labels:
run: nginx-secret
name: nginx-secret
spec:
containers:
- image: nginx
name: nginx-secret
resources: {}
dnsPolicy: ClusterFirst
restartPolicy: Always
status: {}
"nginx_secret.yml" 15L, 253C
1,1
All


Readme
Web Terminal
THE LINUX FOUNDATION
apiVersion: v1
kind: Pod
metadata:
labels:
run: nginx-secret
name: nginx-secret
spec:
containers:
- image: nginx
name: nginx-secret
env:
- name: COOL VARIABLE
valueFrom:
secretKeyRef:
name: some-secret
key: key1
-- INSERT --
16,20
All

QUESTION NO: 2 - (SIMULATION)

Set Configuration Context:

[student@node-1] $ | kubectl

Config use-context k8s

Context

A pod is running on the cluster but it is not responding.

Task

The desired behavior is to have Kubemetes restart the pod when an endpoint returns an HTTP 500 on the /healthz endpoint. The service, probe-pod, should never send traffic to the pod while it is failing. Please complete the following:

• The application has an endpoint, /started, that will indicate if it can accept traffic by returning an HTTP 200. If the endpoint returns an HTTP 500, the application has not yet finished initialization.

• The application has another endpoint /healthz that will indicate if the application is still working as expected by returning an HTTP 200. If the endpoint returns an HTTP 500 the application is no longer responsive.

• Configure the probe-pod pod provided to use these endpoints

• The probes should use port 8080

ANSWER: Seethesolutionbelow.

Explanation:

Solution:

To have Kubernetes automatically restart a pod when an endpoint returns an HTTP 500 on the /healthz endpoint, you will need to configure liveness and readiness probes on the pod.

First, you will need to create a livenessProbe and a readinessProbe in the pod's definition yaml file. The livenessProbe will check the /healthz endpoint, and if it returns an HTTP 500, the pod will be restarted. The readinessProbe will check the /started endpoint, and if it returns an HTTP 500, the pod will not receive traffic.

Here's an example of how you can configure the liveness and readiness probes in the pod definition yaml file:

apiVersion: v1

kind: Pod

metadata:

name: probe-pod

spec:

containers:

- name: probe-pod

ports:

- containerPort: 8080

livenessProbe:

httpGet:

path: /healthz

port: 8080

initialDelaySeconds: 15

periodSeconds: 10

failureThreshold: 3

readinessProbe:

httpGet:

path: /started

port: 8080

initialDelaySeconds: 15

periodSeconds: 10

failureThreshold: 3

The httpGet specifies the endpoint to check and the port to use. The initialDelaySeconds is the amount of time the pod will wait before starting the probe. periodSeconds is the amount of time between each probe check, and the failureThreshold is the number of failed probes before the pod is considered unresponsive.

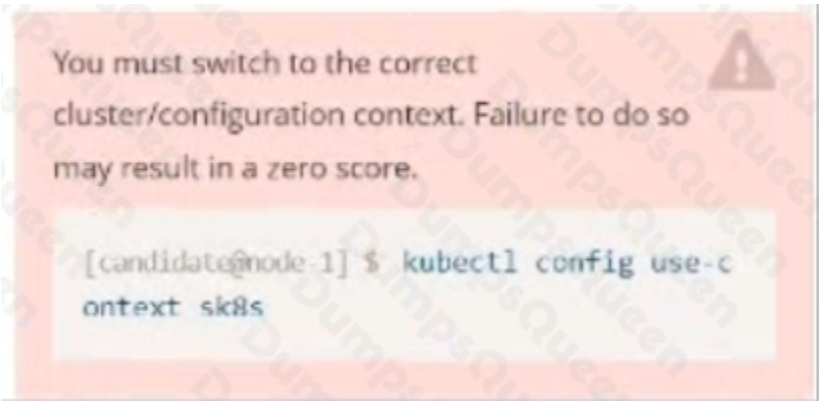
You can use kubectl to create the pod by running the following command:

Once the pod is created, Kubernetes will start monitoring it using the configured liveness and readiness probes. If the /healthz endpoint returns an HTTP 500, the pod will be restarted. If the /started endpoint returns an HTTP 500, the pod will not receive traffic.

Please note that if the failure threshold is set to 3, it means that if the probe fails 3 times consecutively it will be considered as a failure.

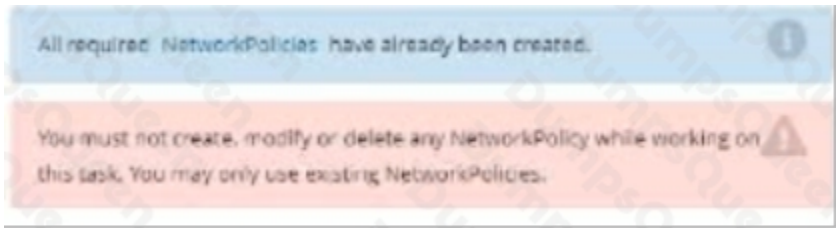
The above configuration assumes that the application is running on port 8080 and the endpoints are available on the same port.

QUESTION NO: 3 - (SIMULATION)



Task:

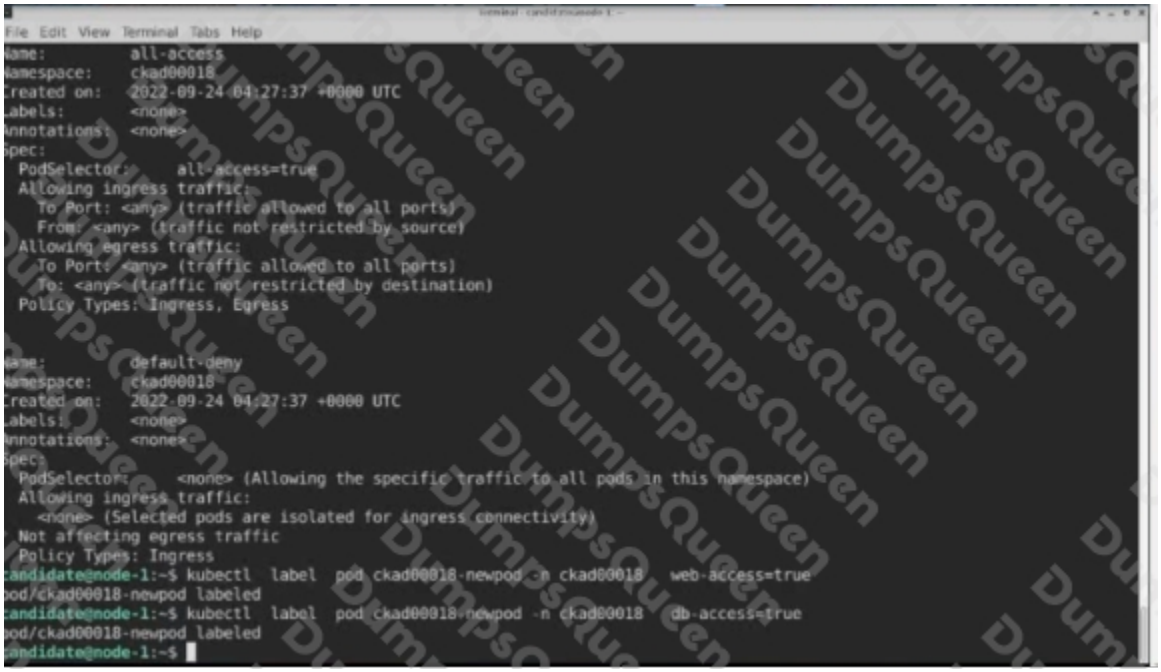
Update the Pod ckad00018-newpod in the ckad00018 namespace to use a NetworkPolicy allowing the Pod to send and receive traffic only to and from the pods web and db



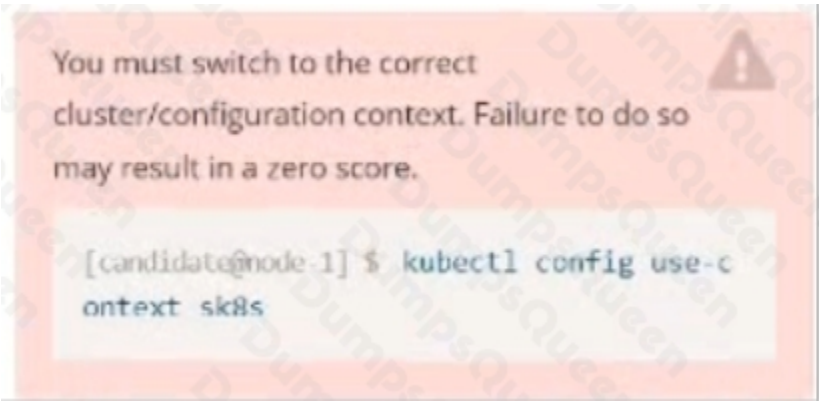
ANSWER: Seethesolutionbelow.

Explanation:

Solution:

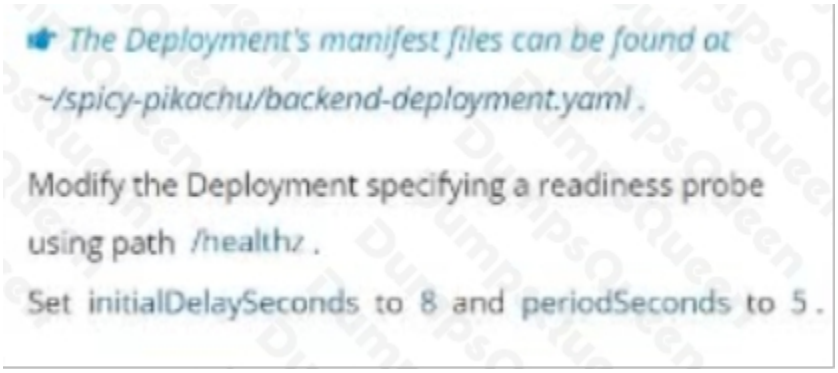


QUESTION NO: 4 - (SIMULATION)



Task

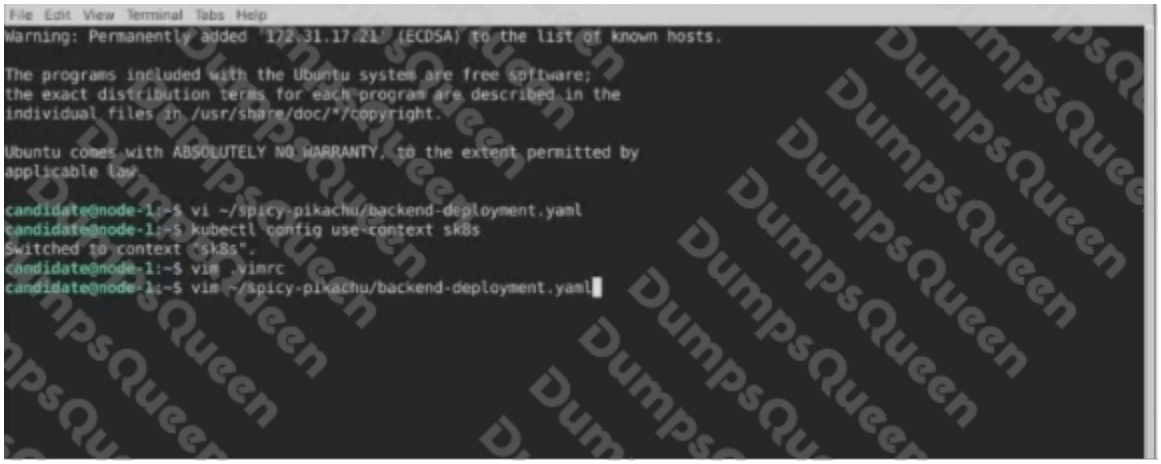
A Deployment named backend-deployment in namespace staging runs a web application on port 8081.

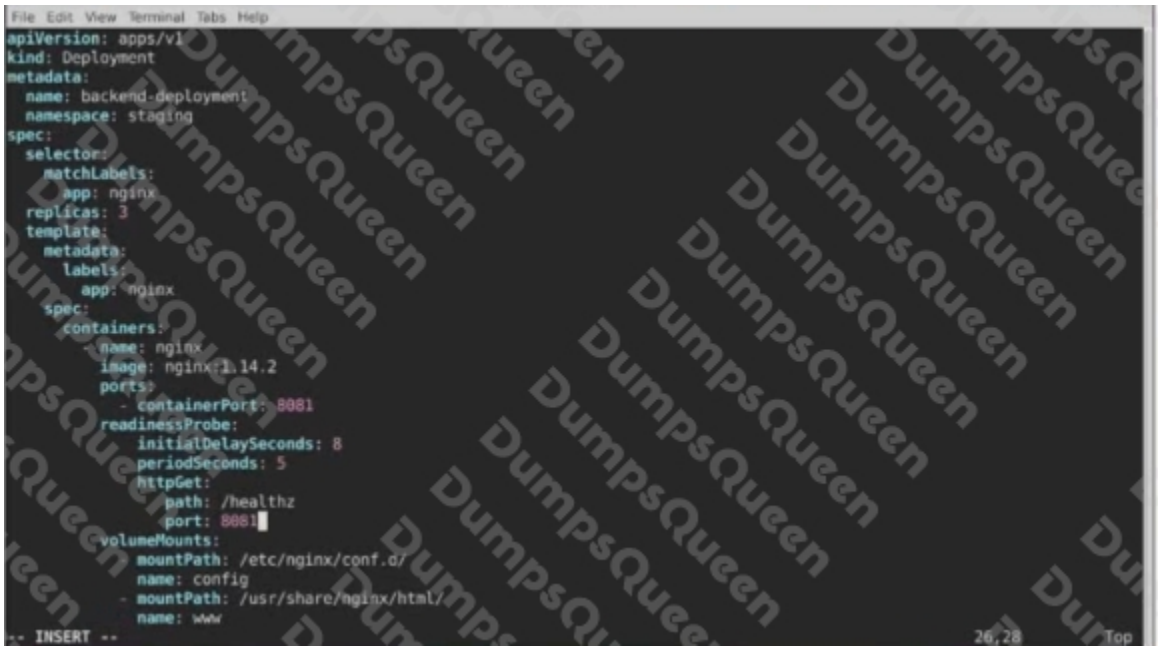


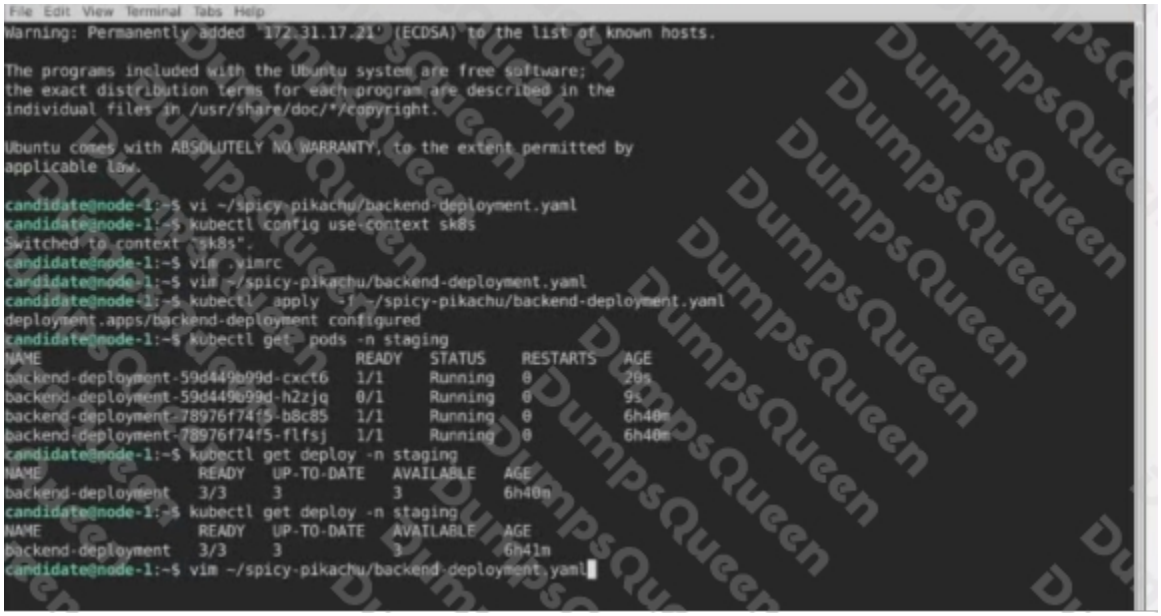
ANSWER: Seethesolutionbelow.

Explanation:

Solution:







QUESTION NO: 5 - (SIMULATION)

Context

A project that you are working on has a requirement for persistent data to be available.

Task

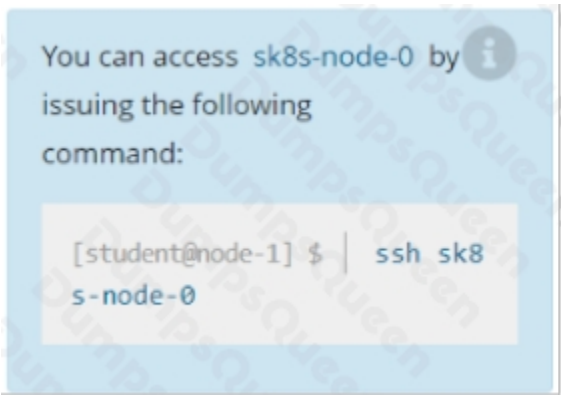
To facilitate this, perform the following tasks:

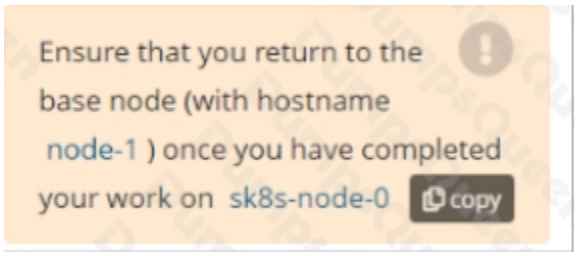
• Create a file on node sk8s-node-0 at /opt/KDSP00101/data/index.html with the content Acct=Finance

• Create a PersistentVolume named task-pv-volume using hostPath and allocate 1Gi to it, specifying that the volume is at /opt/KDSP00101/data on the cluster's node. The configuration should specify the access mode of ReadWriteOnce . It should define the StorageClass name exam for the PersistentVolume , which will be used to bind PersistentVolumeClaim requests to this PersistenetVolume.

• Create a PefsissentVolumeClaim named task-pv-claim that requests a volume of at least 100Mi and specifies an access mode of ReadWriteOnce

• Create a pod that uses the PersistentVolmeClaim as a volume with a label app: my-storage-app mounting the resulting volume to a mountPath /usr/share/nginx/html inside the pod

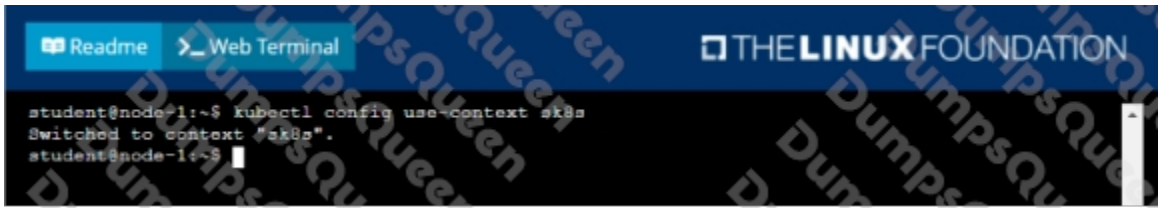


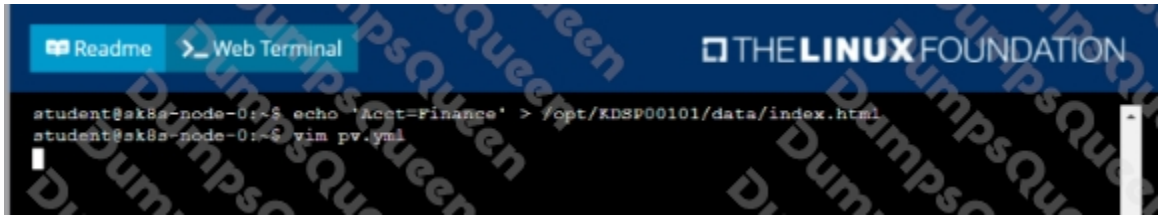


ANSWER: Seethesolutionbelow.

Explanation:

Solution:






Readme
Web Terminal
THE LINUX FOUNDATION
-- INSERT --
0,1
All


Readme
Web Terminal
THE LINUX FOUNDATION
apiVersion: v1
kind: PersistentVolume
name: task-pv-volume
storage: 1Gi
- ReadWriteOnce
storageClassName: storage
path: /opt/KDSP00101/data
type: Director


Readme
Web Terminal
THE LINUX FOUNDATION
apiVersion: v1
kind: PersistentVolumeClaim
name: task-pv-claim
- ReadWriteOnce
storage: 100Mi
storageClassName: storage

```
student@sk8s-node-0:~$ kubectl create -f pv.yml
persistentvolume/task-pv-volume created
student@sk8s-node-0:~$ kubectl create -f pvc.yml
persistentvolumeclaim/task-pv-claim created
student@sk8s-node-0:~$ kubectl get pv
NAME             CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM                   STO
RAGECLASS   REASON   AGE
task-pv-volume   1Gi        RWO            Retain           Bound    default/task-pv-claim   sto
rage                11s
student@sk8s-node-0:~$ kubectl get pvc
NAME          STATUS   VOLUME           CAPACITY   ACCESS MODES   STORAGECLASS   AGE
task-pv-claim   Bound    task-pv-volume   1Gi        RWO            storage        9s
student@sk8s-node-0:~$ vim pod.yml
```

Readme | Web Terminal — THE LINUX FOUNDATION

```
apiVersion: v1
kind: Pod
metadata:
  name: mypod
  labels:
    app: my-storage-app
spec:
  containers:
  - name: myfrontend
    image: nginx
    volumeMounts:
    - mountPath: "/usr/share/nginx/html"
      name: mypod
  volumes:
  - name: mypod
    persistentVolumeClaim:
      claimName: task-pv-clai
~
~
~
~
~
~
                                                                17,32         All
```

```
student@sk8s-node-0:~$ kubectl create -f pod.yml
pod/mypod created
student@sk8s-node-0:~$ kubectl get
```

Readme | Web Terminal — THE LINUX FOUNDATION

```
student@sk8s-node-0:~$ kubectl get pods
NAME    READY   STATUS              RESTARTS   AGE
mypod   0/1     ContainerCreating   0          4s
student@sk8s-node-0:~$ kubectl get pods
NAME    READY   STATUS              RESTARTS   AGE
mypod   0/1     ContainerCreating   0          8s
student@sk8s-node-0:~$ kubectl get pods
NAME    READY   STATUS    RESTARTS   AGE
mypod   1/1     Running   0          10s
student@sk8s-node-0:~$ logout
Connection to 10.250.3.115 closed.
student@node-1:~$
```