Testpassport Q&A

La meilleure qualité le meilleur service

http://www.testpassport.fr

Service de mise à jour gratuit pendant un an

Exam : **CKA**

Title : Certified Kubernetes Administrator

Version : DEMO

1.CORRECT TEXT

Perform the following tasks:

➪ Add an init container to hungry-bear (which has been defined in spec file /opt/KUCC00108/pod-spec-KUCC00108.yaml)

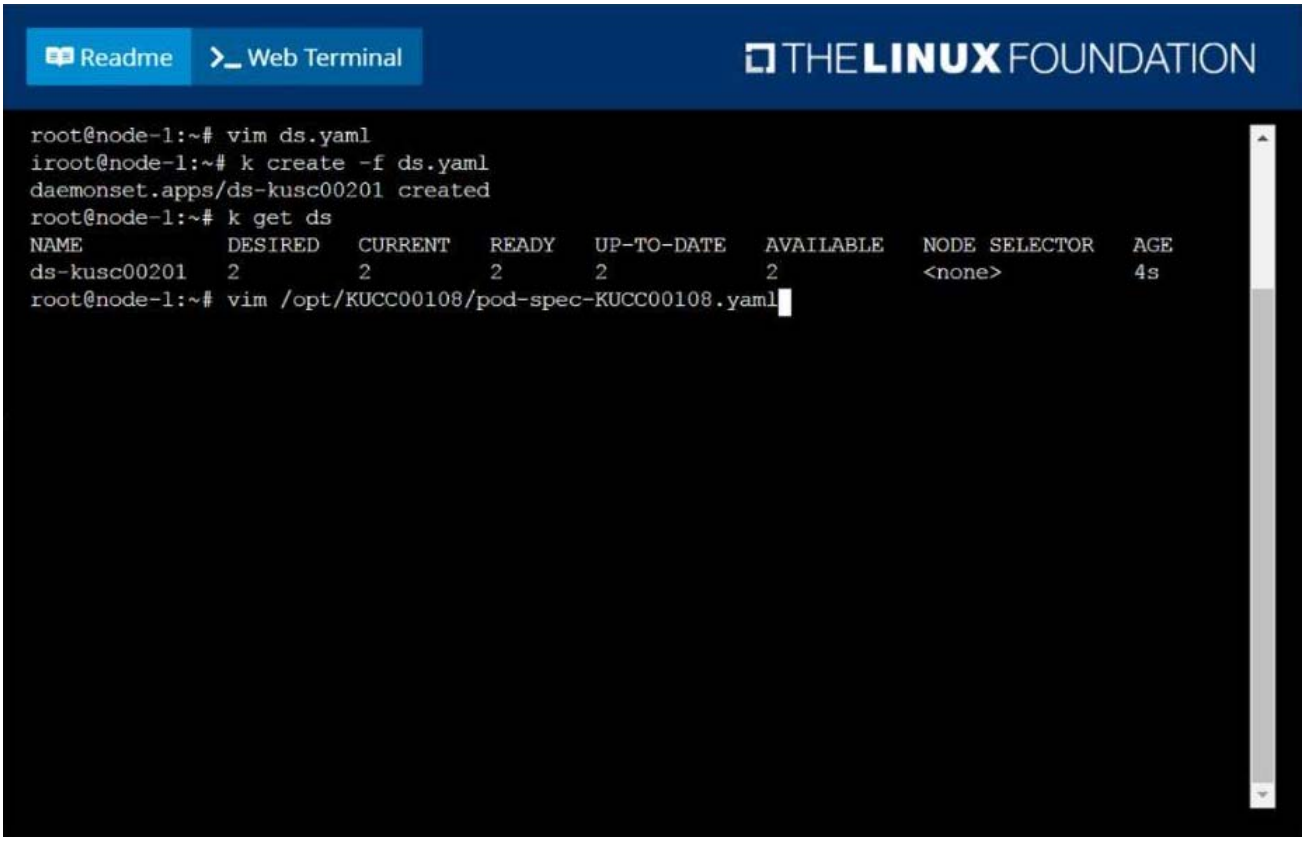
➪ The init container should create an empty file named/workdir/calm.txt

➪ If /workdir/calm.txt is not detected, the pod should exit

➪ Once the spec file has been updated with the init container definition, the pod should be created

Answer:

solution



Readme | Web Terminal — THE LINUX FOUNDATION

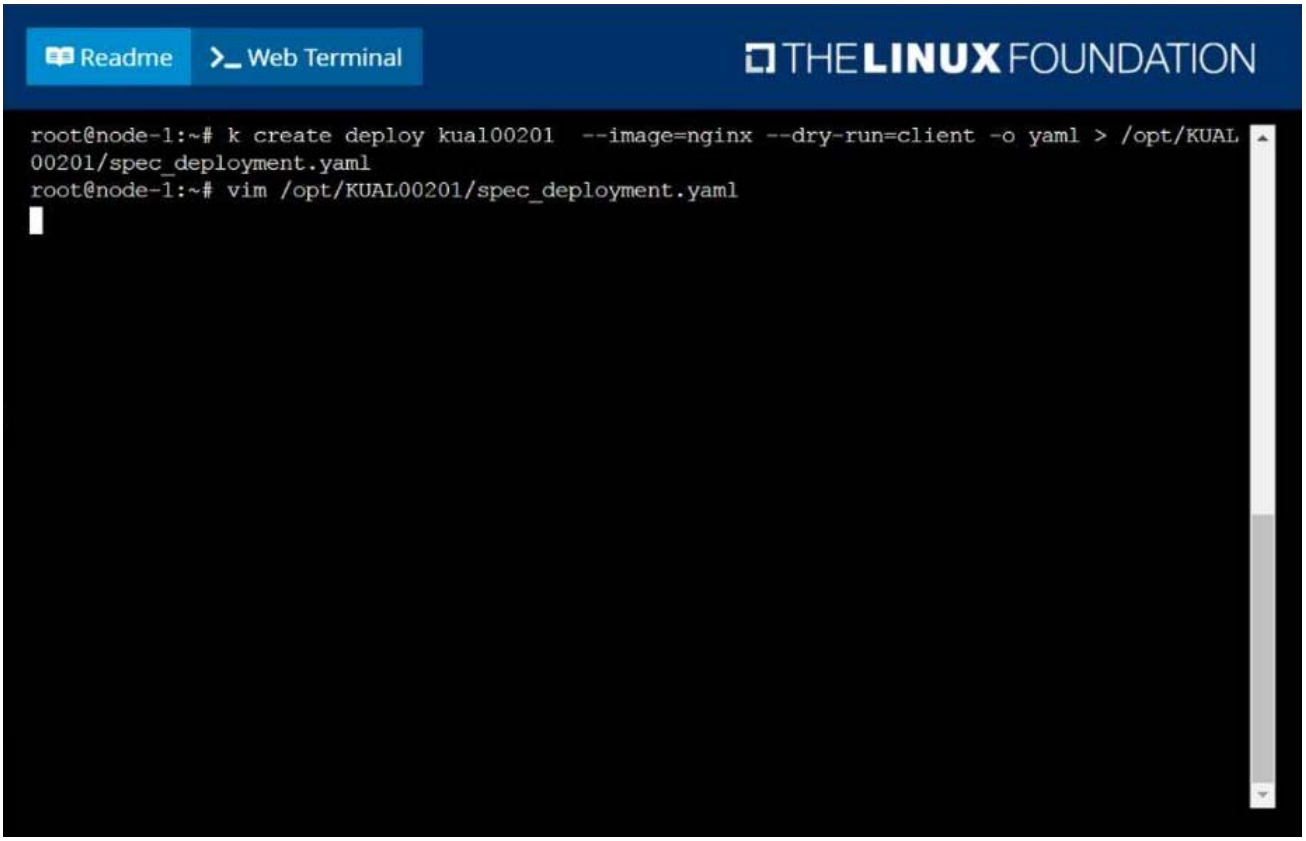
```
apiVersion: v1
kind: Pod
metadata:
  name: hungry-bear
spec:
  volumes:
    - name: workdir
      emptyDir:
  containers:
  - name: checker
    image: alpine
    command: ["/bin/sh", "-c", "if [ -f /workdir/calm.txt ];
              then sleep 100000; else exit 1; fi"]
    volumeMounts:
    - name: workdir
      mountPath: /workdir
  initContainers:
  - name: create
    image: alpine
    command: ["/bin/sh", "-c", "touch /workdir/calm.txt"]
    volumeMounts:
    - name: workdir
      mountPath: /workdir
:wq
```

Readme | Web Terminal — THE LINUX FOUNDATION

```
root@node-1:~# vim ds.yaml
iroot@node-1:~# k create -f ds.yaml
daemonset.apps/ds-kusc00201 created
root@node-1:~# k get ds
NAME           DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR   AGE
ds-kusc00201   2         2         2       2            2           <none>          4s
root@node-1:~# vim /opt/KUCC00108/pod-spec-KUCC00108.yaml
root@node-1:~# k create -f /opt/KUCC00108/pod-spec-KUCC00108.yaml
pod/hungry-bear created
root@node-1:~#
```

2.CORRECT TEXT

Create a deployment spec file that will:
➪ Launch 7 replicas of the nginx Image with the labelapp_runtime_stage=dev
➪ deployment name: kual00201
Save a copy of this spec file to /opt/KUAL00201/spec_deployment.yaml
(or /opt/KUAL00201/spec_deployment.json).
When you are done, clean up (delete) any new Kubernetes API object that you produced during this task.
Answer:
solution



Readme | Web Terminal

THE LINUX FOUNDATION

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app_runtime_stage: dev
  name: kua100201
spec:
  replicas: 7
  selector:
    matchLabels:
      app_runtime_stage: dev
  template:
    metadata:
      labels:
        app_runtime_stage: dev
    spec:
      containers:
      - image: nginx
        name: nginx
~
~
~
~
~
"/opt/KUAL00201/spec_deployment.yaml" 19L, 320C written
```

3.CORRECT TEXT

From the pod label name=cpu-utilizer, find pods running high CPU workloads and write the name of the pod consuming most CPU to the file /opt/KUTR00102/KUTR00102.txt (which already exists).

Answer:

solution

Readme Web Terminal THE LINUX FOUNDATION

```
root@node-1:~# k top po -l name=cpu-utilizer
NAME                    CPU(cores)   MEMORY(bytes)
cpu-utilizer-98b9se     60m          7Mi
cpu-utilizer-ab2d3s     14m          7Mi
cpu-utilizer-kipb9a     45m          7Mi
root@node-1:~# vim /opt/KUTR00102/KUTR00102.txt
```

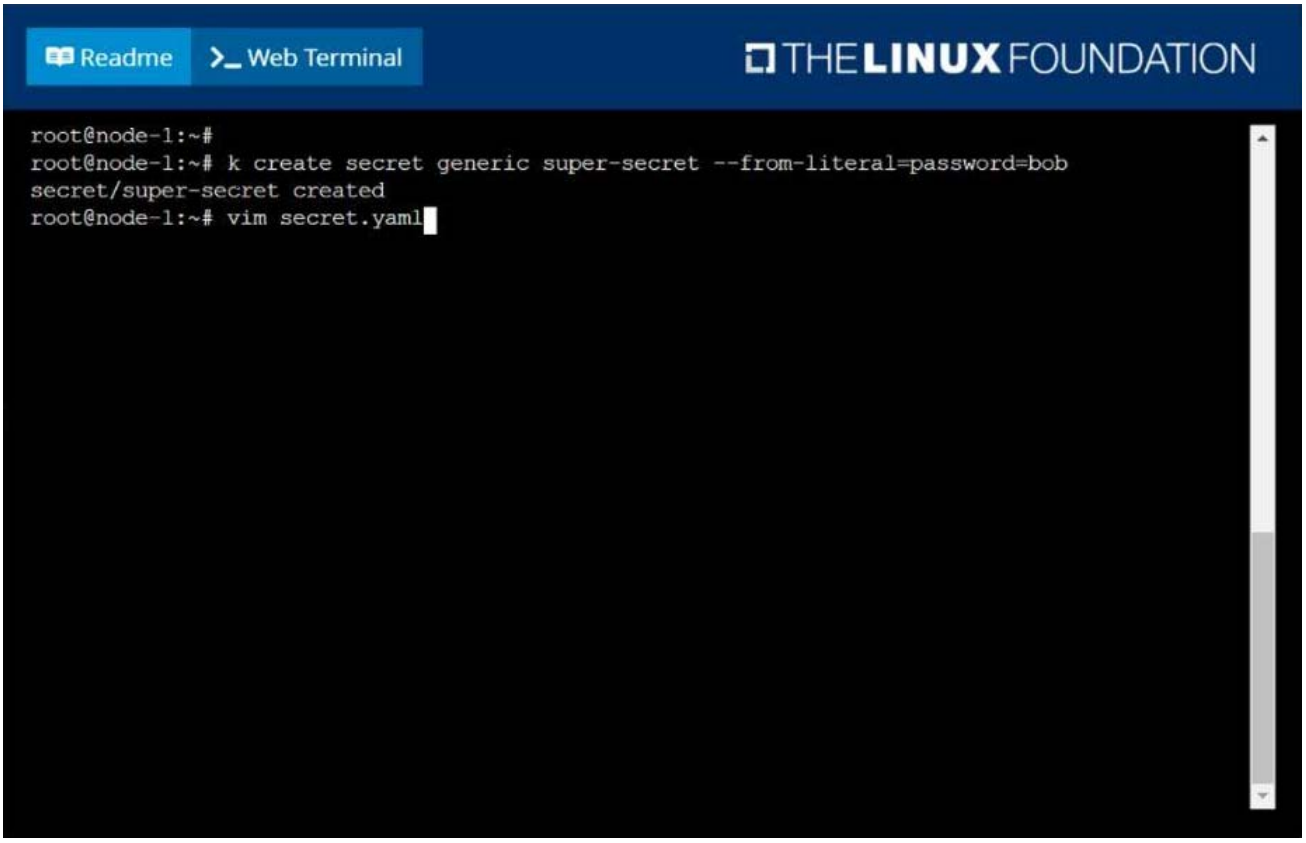
Readme Web Terminal THE LINUX FOUNDATION

```
cpu-utilizer-98b9se
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
:wq
```

4.CORRECT TEXT

Create a Kubernetes secret as follows:

➪ Name: super-secret
➪ password: bob
Create a pod named pod-secrets-via-file, using the redis Image, which mounts a secret named super-secret at /secrets.
Create a second pod named pod-secrets-via-env, using the redis Image, which exports password as CONFIDENTIAL
Answer:
solution



Readme | Web Terminal — THE LINUX FOUNDATION

```
apiVersion: v1
kind: Pod
metadata:
  name: pod-secrets-via-file
spec:
  containers:
  - name: redis
    image: redis
    volumeMounts:
    - name: foo
      mountPath: "/secrets"
  volumes:
  - name: foo
    secret:
      secretName: super-secret
~
~
~
~
~
~
~
~
~
:w
```

Readme | Web Terminal — THE LINUX FOUNDATION

```
root@node-1:~# k create -f secret.yaml
pod/pod-secrets-via-file created
root@node-1:~# vim secret1.yaml
root@node-1:~# k create -f secret1.yaml
pod/pod-secrets-via-env created
root@node-1:~# k get po
NAME                          READY   STATUS    RESTARTS   AGE
cpu-utilizer-98b9se           1/1     Running   0          6h25m
cpu-utilizer-ab2d3s           1/1     Running   0          6h25m
cpu-utilizer-kipb9a           1/1     Running   0          6h25m
ds-kusc00201-2r2k9            1/1     Running   0          40m
ds-kusc00201-hzm9q            1/1     Running   0          40m
foo                           1/1     Running   0          6h28m
front-end                     1/1     Running   0          6h27m
hungry-bear                   1/1     Running   0          36m
kucc8                         3/3     Running   0          34m
nginx-app-848cfcf495-9prjh    1/1     Running   0          19m
nginx-app-848cfcf495-gl2kh    1/1     Running   0          19m
nginx-app-848cfcf495-pg2c8    1/1     Running   0          19m
nginx-kusc00101               1/1     Running   0          26m
pod-secrets-via-env           1/1     Running   0          4s
pod-secrets-via-file          1/1     Running   0          106s
webserver-84c55967f4-qzjcv    1/1     Running   0          6h43m
webserver-84c55967f4-t4791    1/1     Running   0          6h43m
root@node-1:~#
```

5.CORRECT TEXT

Score: 7%

Set configuration context:

```
[student@node-1] $ kube
ctl config use-context k
8s
```

Task
Create a new nginx Ingress resource as follows:
• Name: ping
• Namespace: ing-internal
• Exposing service hi on path /hi using service port 5678

The availability of service hi can be checked using the following command, which should return hi :

```
[student@node-1] $ curl
-kL <INTERNAL_IP>/hi
```

Answer:
Solution:
vi ingress.yaml
#
apiVersion: networking.k8s.io/v1
kind: Ingress metadata: name: ping
namespace: ing-internal
spec:
rules:
- http:
paths:
- path: /hi pathType: Prefix backend: service: name: hi
port:
number: 5678
#
kubectl create -f ingress.yaml