

# Linux-Foundation

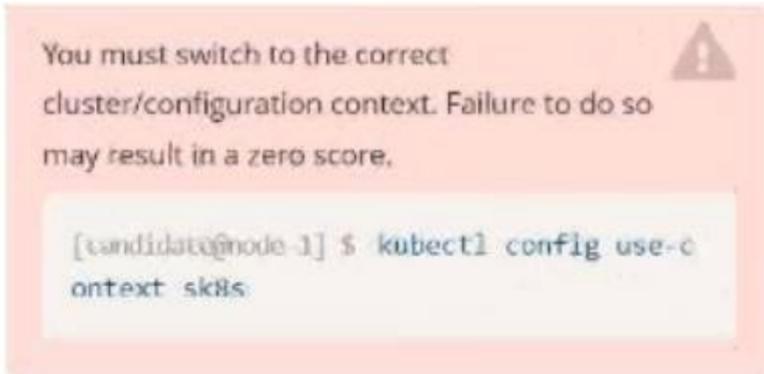
## Exam Questions CKAD

Certified Kubernetes Application Developer (CKAD) Program



**NEW QUESTION 1**

Exhibit:



Task:

Update the Deployment app-1 in the frontend namespace to use the existing ServiceAccount app.

- A. Mastered
- B. Not Mastered

**Answer: A**

**Explanation:**

Solution:

Text Description automatically generated

```

File Edit View Terminal Tabs Help
The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

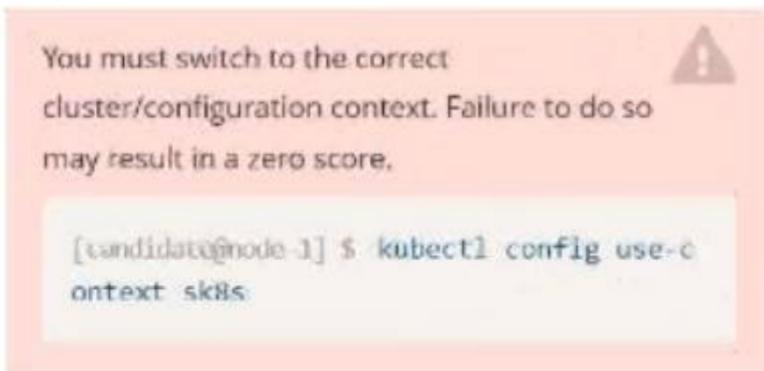
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

candidate@node-1:~$ vi ~/spicy-pikachu/backend-deployment.yaml
candidate@node-1:~$ kubectl config use-context sk8s
Switched to context "sk8s".
candidate@node-1:~$ vim .vimrc
candidate@node-1:~$ vim ~/spicy-pikachu/backend-deployment.yaml
candidate@node-1:~$ kubectl apply -f ~/spicy-pikachu/backend-deployment.yaml
deployment.apps/backend-deployment configured
candidate@node-1:~$ kubectl get pods -n staging
NAME                                READY   STATUS    RESTARTS   AGE
backend-deployment-59d449b99d-cxct6  1/1     Running   0           20s
backend-deployment-59d449b99d-h2zjq  0/1     Running   0           9s
backend-deployment-78976f74f5-b8c85  1/1     Running   0           6h40m
backend-deployment-78976f74f5-flfsj  1/1     Running   0           6h40m
candidate@node-1:~$ kubectl get deploy -n staging
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
backend-deployment  3/3     3             3           6h40m
candidate@node-1:~$ kubectl get deploy -n staging
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
backend-deployment  3/3     3             3           6h41m
candidate@node-1:~$ vim ~/spicy-pikachu/backend-deployment.yaml
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ kubectl set serviceaccount deploy app-1 app -n frontend
deployment.apps/app-1 serviceaccount updated
candidate@node-1:~$

```

**NEW QUESTION 2**

Exhibit:



Task:

Key3: value1

Add an environment variable named BEST\_VARIABLE consuming the value of the secret key3.

- A. Mastered
- B. Not Mastered

**Answer: A**

**Explanation:**

Solution:

```
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ kubectl create secret generic app-secret -n default --from-literal=key3=value1
secret/app-secret created
candidate@node-1:~$ kubectl get secrets
NAME      TYPE      DATA   AGE
app-secret Opaque    1       4s
candidate@node-1:~$ kubectl run nginx-secret -n default --image=nginx:stable --dry-run=client -o yaml > sec.yaml
candidate@node-1:~$ vim sec.yaml
```

Text Description automatically generated

```
File Edit View Terminal Tabs Help
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: nginx-secret
  name: nginx-secret
  namespace: default
spec:
  containers:
  - image: nginx:stable
    name: nginx-secret
    env:
    - name: BEST_VARIABLE
      valueFrom:
        secretKeyRef:
          name: app-secret
          key: key3
```

Text Description automatically generated

```
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ kubectl create secret generic app-secret -n default --from-literal=key3=value1
secret/app-secret created
candidate@node-1:~$ kubectl get secrets
NAME      TYPE      DATA   AGE
app-secret Opaque    1       4s
candidate@node-1:~$ kubectl run nginx-secret -n default --image=nginx:stable --dry-run=client -o yaml > sec.yaml
candidate@node-1:~$ vim sec.yaml
candidate@node-1:~$ kubectl create -f sec.yaml
pod/nginx-secret created
candidate@node-1:~$ kubectl get pods
NAME      READY   STATUS    RESTARTS   AGE
nginx-secret 1/1     Running   0          7s
candidate@node-1:~$
```

### NEW QUESTION 3

Exhibit:



Context

A web application requires a specific version of redis to be used as a cache. Task

Create a pod with the following characteristics, and leave it running when complete:

- The pod must run in the web namespace. The namespace has already been created
- The name of the pod should be cache
- Use the lfcncf/redis image with the 3.2 tag
- Expose port 6379

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:

```

Readme > Web Terminal THE LINUX FOUNDATION
student@node-1:~$ kubectl run cache --image=lfcncf/redis:3.2 --port=6379 -n web
pod/cache created
student@node-1:~$ kubectl get pods -n web
NAME    READY   STATUS             RESTARTS   AGE
cache   0/1     ContainerCreating  0           6s
student@node-1:~$ kubectl get pods -n web
NAME    READY   STATUS    RESTARTS   AGE
cache   1/1     Running   0           9s
student@node-1:~$ █

```

**NEW QUESTION 4**

Exhibit:



Context

You sometimes need to observe a pod's logs, and write those logs to a file for further analysis. Task

Please complete the following;

- Deploy the counter pod to the cluster using the provided YAMLSpec file at /opt/KDOB00201/counter.yaml
- Retrieve all currently available application logs from the running pod and store them in the file /opt/KDOB00201/log\_Output.txt, which has already been created

- A. Mastered
- B. Not Mastered

**Answer: A**

**Explanation:**

Solution:

```

student@node-1:~$ kubectl create -f /opt/KDOB00201/counter.yaml
pod/counter created
student@node-1:~$ kubectl get pods
NAME           READY   STATUS    RESTARTS   AGE
counter        1/1     Running   0           10s
liveness-http  1/1     Running   0           6h45m
nginx-101      1/1     Running   0           6h46m
nginx-configmap 1/1     Running   0           107s
nginx-secret   1/1     Running   0           7m21s
poller         1/1     Running   0           6h46m
student@node-1:~$ kubectl logs counter
1: 2b305101817ae25ca60ae46510fb6d11
2: 3648cf2eae95ab680dba8f195f891af4
3: 65c8bbd4dbf70bf81f2a0984a3a44ede
4: 40d3a9c8e46f5533bb4828fbc5c8d038
5: 390442d2530a90c3602901e3fe999ac8
6: b71d95187417e139effb33af77681040
7: 66a8e55a6491e756d2d0549ad6ab90a7
8: ff2b3d583b64125d2f9129c443bb37ff
9: b6c6a12b6e77944ed8baaaf6c242dae4
10: bfcc9a894a0604fc4b814b37d0a200a4
student@node-1:~$ kubectl logs counter > /opt/KDOB00201/log_output.txt
student@node-1:~$ █

student@node-1:~$ kubectl logs counter > /opt/KDOB00201/log_output.txt
student@node-1:~$ kubectl logs counter > /opt/KDOB00201/log_output.txt
student@node-1:~$ cat /opt/KDOB00201/log_output.txt

```

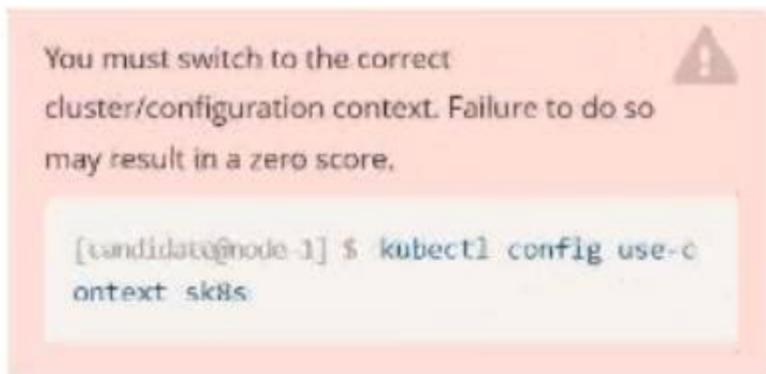
Readme
Web Terminal
THE LINUX FOUNDATION

```

student@node-1:~$ kubectl logs counter > /opt/KDOB00201/log_output.txt
student@node-1:~$ cat /opt/KDOB00201/log_output.txt
1: 2b305101817ae25ca60ae46510fb6d11
2: 3648cf2eae95ab680dba8f195f891af4
3: 65c8bbd4dbf70bf81f2a0984a3a44ede
4: 40d3a9c8e46f5533bb4828fbe5c8d038
5: 390442d2530a90c3602901e3fe999ac8
6: b71d95187417e139effb33af77681040
7: 66a8e55a6491e756d2d0549ad6ab90a7
8: ff2b3d583b64125d2f9129c443bb37ff
9: b6c6a12b6e77944ed8baaaf6c242dae4
10: bfcc9a894a0604fc4b814b37d0a200a4
11: 5493cd16a1790a5fb9512b0c9d4c5dd1
12: 03f169e93e6143438e6dfe4ecb3cc9ed
13: 764b37fe611373c42d0b47154041f6eb
14: 1a56fbe1896b0ee6394136166281839e
15: ecc492eb17715de090c47345a98d98d3
16: 7974a6bec0fb44b6b8bbfc71aa3fbe74
17: 9ae01bef01748b12cc9f97a5f9f72cd6
18: 23fb22ee34d4272e4c9e005f1774515f
19: ec7e1a5d314da9a0ad45d53be5a7acae
20: 0bccdd8ee02cd42029e8162cd1c1197c
21: d6851ea43546216b95bcb81ced997102
22: 7ed9a38ea8bf0d86206569481442af44
23: 29b8416ddc63dbfcb987ab3c8198e9fe
24: 1f2062001df51a108ab25010f506716f
student@node-1:~$
    
```

**NEW QUESTION 5**

Exhibit:



Task:

- > To run 2 replicas of the pod
- > Add the following label on the pod:  
Role userUI

- A. Mastered
- B. Not Mastered

**Answer: A**

**Explanation:**

Solution:  
Text Description automatically generated

```
File Edit View Terminal Tabs Help
# reopened with the relevant failures.
#
apiVersion: apps/v1
kind: Deployment
metadata:
  annotations:
    deployment.kubernetes.io/revision: "1"
  creationTimestamp: "2022-09-24T04:27:03Z"
  generation: 1
  labels:
    app: nginx
  name: ckad00017-deployment
  namespace: ckad00017
  resourceVersion: "3349"
  uid: 1cd67613-fade-46e9-b741-94298b9c6e7c
spec:
  progressDeadlineSeconds: 600
  replicas: 2
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app: nginx
  strategy:
    rollingUpdate:
      maxSurge: 25%
      maxUnavailable: 25%
    type: RollingUpdate
  template:
    metadata:
      creationTimestamp: null
      labels:
-- INSERT --
```

33,14 5%

Text Description automatically generated

```
File Edit View Terminal Tabs Help
name: ckad00017-deployment
namespace: ckad00017
resourceVersion: "3349"
uid: 1cd67613-fade-46e9-b741-94298b9c6e7c
spec:
  progressDeadlineSeconds: 600
  replicas: 2
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app: nginx
  strategy:
    rollingUpdate:
      maxSurge: 25%
      maxUnavailable: 25%
    type: RollingUpdate
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: nginx
        role: userUI
    spec:
      containers:
      - image: nginx:latest
        imagePullPolicy: Always
        name: nginx
        ports:
        - containerPort: 80
          protocol: TCP
        resources: {}
-- INSERT --
```

35,21 33%

Text Description automatically generated

```

File Edit View Terminal Tabs Help
backend-deployment-59d449b99d-h2zjq 0/1 Running 0 9s
backend-deployment-78976f74f5-b8c85 1/1 Running 0 6h40m
backend-deployment-78976f74f5-flfsj 1/1 Running 0 6h40m
candidate@node-1:~$ kubectl get deploy -n staging
NAME READY UP-TO-DATE AVAILABLE AGE
backend-deployment 3/3 3 3 6h40m
candidate@node-1:~$ kubectl get deploy -n staging
NAME READY UP-TO-DATE AVAILABLE AGE
backend-deployment 3/3 3 3 6h41m
candidate@node-1:~$ vim ~/spicy-pikachu/backend-deployment.yaml
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ kubectl set serviceaccount deploy app-1 app -n frontend
deployment.apps/app-1 serviceaccount updated
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ vim ~/prompt-escargot/buffalo-deployment.yaml
candidate@node-1:~$ vim ~/prompt-escargot/buffalo-deployment.yaml
candidate@node-1:~$ kubectl apply -f ~/prompt-escargot/buffalo-deployment.yaml
deployment.apps/buffalo-deployment configured
candidate@node-1:~$ kubectl get pods -n gorilla
NAME READY STATUS RESTARTS AGE
buffalo-deployment-776844df7f-r5fsb 1/1 Running 0 6h38m
buffalo-deployment-859898c6f5-zx5gj 0/1 ContainerCreating 0 8s
candidate@node-1:~$ kubectl get deploy -n gorilla
NAME READY UP-TO-DATE AVAILABLE AGE
buffalo-deployment 1/1 1 1 6h38m
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ kubectl edit deploy ckad00017-deployment -n ckad00017
deployment.apps/ckad00017-deployment edited
candidate@node-1:~$

```

```

File Edit View Terminal Tabs Help
candidate@node-1:~$ kubectl get pods -n gorilla
NAME READY STATUS RESTARTS AGE
buffalo-deployment-776844df7f-r5fsb 1/1 Running 0 6h38m
buffalo-deployment-859898c6f5-zx5gj 0/1 ContainerCreating 0 8s
candidate@node-1:~$ kubectl get deploy -n gorilla
NAME READY UP-TO-DATE AVAILABLE AGE
buffalo-deployment 1/1 1 1 6h38m
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ kubectl edit deploy ckad00017-deployment -n ckad00017
deployment.apps/ckad00017-deployment edited
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad0001
ckad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad0001
ckad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad0001
ckad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad0001
ckad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad0001
ckad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad0001
ckad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad0001
ckad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose service/cherry -n ckad00017 --name=cherry --port=8888 --type=NodePort
service/cherry exposed
candidate@node-1:~$

```

```

candidate@node-1:~$ kubectl get svc
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 77d
candidate@node-1:~$ kubectl get svc -n ckad00017
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
cherry NodePort 10.100.100.176 <none> 8888:30683/TCP 24s
candidate@node-1:~$ kubectl expose service deploy ckad00017-deployment -n ckad00017 --name=cherry --port=8888 --type=NodePort
Error from server (NotFound): services "deploy" not found
Error from server (NotFound): services "ckad00017-deployment" not found
candidate@node-1:~$ kubectl get svc -n ckad00017
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
cherry NodePort 10.100.100.176 <none> 8888:30683/TCP 46s
candidate@node-1:~$

```

```

File Edit View Terminal Tabs Help
candidate@node-1:~$ kubectl expose service deploy ckad00017-deployment -n ckad00017 --name=cherry --port=8888 --type=NodePort
Error from server (NotFound): services "deploy" not found
Error from server (NotFound): services "ckad00017-deployment" not found
candidate@node-1:~$ kubectl get svc -n ckad00017
NAME      TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
cherry    NodePort  10.100.100.176 <none>        8888:30683/TCP  46s
candidate@node-1:~$ history
 1 vi ~/spicy-pikachu/backend-deployment.yaml
 2 kubectl config use-context sk8s
 3 vim .vimrc
 4 vim ~/spicy-pikachu/backend-deployment.yaml
 5 kubectl apply -f ~/spicy-pikachu/backend-deployment.yaml
 6 kubectl get pods -n staging
 7 kubectl get deploy -n staging
 8 vim ~/spicy-pikachu/backend-deployment.yaml
 9 kubectl config use-context k8s
10 kubectl set serviceaccount deploy app-1 app -n frontend
11 kubectl config use-context k8s
12 vim ~/prompt-escargot/buffalo-deployment.yaml
13 kubectl apply -f ~/prompt-escargot/buffalo-deployment.yaml
14 kubectl get pods -n gorilla
15 kubectl get deploy -n gorilla
16 kubectl config use-context k8s
17 kubectl edit deploy ckad00017-deployment -n ckad00017
18 kubectl expose deploy ckad00017-deployment -n ckad00017 --name=cherry --port=8888 --type=NodePort
19 kubectl get svc
20 kubectl get svc -n ckad00017
21 kubectl expose service deploy ckad00017-deployment -n ckad00017 --name=cherry --port=8888 --type=NodePort
22 kubectl get svc -n ckad00017
23 history
candidate@node-1:~$

```

**NEW QUESTION 6**

Exhibit:



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

- A. Mastered
- B. Not Mastered

**Answer: A**

**Explanation:**

Solution:

apiVersion: v1 kind: Pod metadata: labels:

test: liveness

name: liveness-exec

spec: containers:

- name: liveness

image: k8s.gcr.io/busybox

args:

- /bin/sh

- -c

- touch /tmp/healthy; sleep 30; rm -rf /tmp/healthy; sleep 600

livenessProbe: exec: command:

- cat

- /tmp/healthy

initialDelaySeconds: 5

periodSeconds: 5

In the configuration file, you can see that the Pod has a single Container. The periodSeconds field specifies that the kubelet should perform a liveness probe every 5 seconds. The initialDelaySeconds field tells the kubelet that it should wait 5 seconds before performing the first probe. To perform a probe, the kubelet executes the command cat /tmp/healthy in the target container. If the command succeeds, it returns 0, and the kubelet considers the container to be alive and healthy. If the command returns a non-zero value, the kubelet kills the container and restarts it.

When the container starts, it executes this command:

/bin/sh -c "touch /tmp/healthy; sleep 30; rm -rf /tmp/healthy; sleep 600"

For the first 30 seconds of the container's life, there is a /tmp/healthy file. So during the first 30 seconds, the command `cat /tmp/healthy` returns a success code. After 30 seconds, `cat /tmp/healthy` returns a failure code.

Create the Pod:

`kubectl apply -f https://k8s.io/examples/pods/probe/exec-liveness.yaml` Within 30 seconds, view the Pod events:

`kubectl describe pod liveness-exec`

The output indicates that no liveness probes have failed yet:

FirstSeen LastSeen Count From SubobjectPath Type Reason Message

```
-----
24s 24s 1 {default-scheduler } Normal Scheduled Successfully assigned liveness-exec to worker0
23s 23s 1 {kubelet worker0} spec.containers{liveness} Normal Pulling pulling image "k8s.gcr.io/busybox"
23s 23s 1 {kubelet worker0} spec.containers{liveness} Normal Pulled Successfully pulled image "k8s.gcr.io/busybox"
23s 23s 1 {kubelet worker0} spec.containers{liveness} Normal Created Created container with docker id 86849c15382e; Security:[seccomp=unconfined]
23s 23s 1 {kubelet worker0} spec.containers{liveness} Normal Started Started container with docker id 86849c15382e
```

After 35 seconds, view the Pod events again: `kubectl describe pod liveness-exec`

At the bottom of the output, there are messages indicating that the liveness probes have failed, and the containers have been killed and recreated.

FirstSeen LastSeen Count From SubobjectPath Type Reason Message

```
-----
37s 37s 1 {default-scheduler } Normal Scheduled Successfully assigned liveness-exec to worker0
36s 36s 1 {kubelet worker0} spec.containers{liveness} Normal Pulling pulling image "k8s.gcr.io/busybox"
36s 36s 1 {kubelet worker0} spec.containers{liveness} Normal Pulled Successfully pulled image "k8s.gcr.io/busybox"
36s 36s 1 {kubelet worker0} spec.containers{liveness} Normal Created Created container with docker id 86849c15382e; Security:[seccomp=unconfined]
36s 36s 1 {kubelet worker0} spec.containers{liveness} Normal Started Started container with docker id 86849c15382e
```

2s 2s 1 {kubelet worker0} spec.containers{liveness} Warning Unhealthy Liveness probe failed: cat: can't open '/tmp/healthy': No such file or directory

Wait another 30 seconds, and verify that the container has been restarted: `kubectl get pod liveness-exec`

The output shows that RESTARTS has been incremented: NAME READY STATUS RESTARTS AGE

`liveness-exec 1/1 Running 1 1m`

### NEW QUESTION 7

Exhibit:

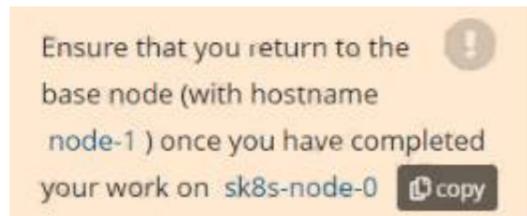
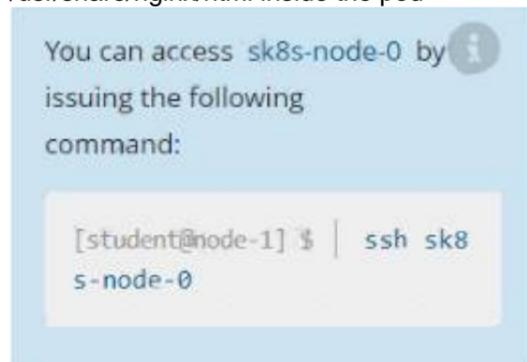


Context

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 PersistentVolume.
- Create a PersistentVolumeClaim 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 PersistentVolumeClaim as a volume with a label `app: my-storage-app` mounting the resulting volume to a `mountPath /usr/share/nginx/html` inside the pod



- A. Mastered
- B. Not Mastered

**Answer:** A

**Explanation:**

**Solution:**

```

Readme Web Terminal THE LINUX FOUNDATION
student@node-1:~$ kubectl config use-context sk8s
Switched to context "sk8s".
student@node-1:~$

```

```

Readme Web Terminal THE LINUX FOUNDATION
* Documentation: https://help.ubuntu.com
* Management:   https://landscape.canonical.com
* Support:      https://ubuntu.com/advantage

System information as of Fri Oct 9 08:52:09 UTC 2020

System load: 2.02          Users logged in:      0
Usage of /:  10.3% of 242.29GB IP address for eth0:  10.250.3.115
Memory usage: 2%          IP address for docker0: 172.17.0.1
Swap usage:  0%           IP address for cni0:   10.244.1.1
Processes:   38

* Kubernetes 1.19 is out! Get it in one command with:

  sudo snap install microk8s --channel=1.19 --classic

https://microk8s.io/ has docs and details.

7 packages can be updated.
1 update is a security update.

New release '20.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

student@sk8s-node-0:~$

```

```

Readme Web Terminal THE LINUX FOUNDATION
student@sk8s-node-0:~$ echo 'Acct=Finance' > /opt/KDSP00101/data/index.html
student@sk8s-node-0:~$ vim pv.yml

```

```

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

```

```

Readme Web Terminal THE LINUX FOUNDATION
apiVersion: v1
kind: PersistentVolume
metadata:
  name: task-pv-volume
spec:
  capacity:
    storage: 1Gi
  accessModes:
  - ReadWriteOnce
  storageClassName: storage
  hostPath:
    path: /opt/KDSP00101/data
    type: Directory

```

```

Readme Web Terminal THE LINUX FOUNDATION
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: task-pv-claim
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      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                STORAGECLASS  AGE
task-pv-volume      1Gi       RWO           Retain          Bound   default/task-pv-claim  storage        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-claim

```

```

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:~$

```

**NEW QUESTION 8**

Exhibit:



Context

A container within the poller pod is hard-coded to connect the nginxsvc service on port 90. As this port changes to 5050 an additional container needs to be added to the poller pod which adapts the container to connect to this new port. This should be realized as an ambassador container within the pod.

Task

- Update the nginxsvc service to serve on port 5050.
- Add an HAProxy container named haproxy bound to port 90 to the poller pod and deploy the enhanced pod. Use the image haproxy and inject the configuration located at /opt/KDMC00101/haproxy.cfg, with a ConfigMap named haproxy-config, mounted into the container so that haproxy.cfg is available at /usr/local/etc/haproxy/haproxy.cfg. Ensure that you update the args of the poller container to connect to localhost instead of nginxsvc so that the connection is correctly proxied to the new service endpoint. You must not modify the port of the endpoint in poller's args . The spec file used to create the initial poller pod is available in /opt/KDMC00101/poller.yaml

- A. Mastered
- B. Not Mastered

**Answer: A**

**Explanation:**

Solution:

```

apiVersion: apps/v1 kind: Deployment metadata:
name: my-nginx spec:
selector:
matchLabels: run: my-nginx replicas: 2 template: metadata: labels:
run: my-nginx spec: containers:
- name: my-nginx image: nginx ports:
- containerPort: 90

```

This makes it accessible from any node in your cluster. Check the nodes the Pod is running on: `kubectl apply -f ./run-my-nginx.yaml`

`kubectl get pods -l run=my-nginx -o wide`

```

NAME READY STATUS RESTARTS AGE IP NODE

```

```

my-nginx-3800858182-jr4a2 1/1 Running 0 13s 10.244.3.4 kubernetes-minion-905m my-nginx-3800858182-kna2y 1/1 Running 0 13s 10.244.2.5 kubernetes-minion-ljyd

```

Check your pods' IPs:

```

kubectl get pods -l run=my-nginx -o yaml | grep podIP podIP: 10.244.3.4
podIP: 10.244.2.5

```

**NEW QUESTION 9**

Exhibit:



Context

Developers occasionally need to submit pods that run periodically. Task

Follow the steps below to create a pod that will start at a predetermined time and which runs to completion only once each time it is started:

- Create a YAML formatted Kubernetes manifest /opt/KDPD00301/periodic.yaml that runs the following shell command: `date` in a single busybox container. The command should run every minute and must complete within 22 seconds or be terminated by Kubernetes. The Cronjob name and container name should both be

hello

- Create the resource in the above manifest and verify that the job executes successfully at least once

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:

```

student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "*" * * * * --dry-run=
client -o yml > /opt/KDPD00301/periodic.yaml
error: unable to match a printer suitable for the output format "yml", allowed formats are: go-t
emplate,go-template-file,json,jsonpath,jsonpath-as-json,jsonpath-file,name,template,templatefile
,yaml
student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "*" * * * * --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
student@node-1:~$ vim /opt/KDPD00301/periodic.yaml
    
```

```

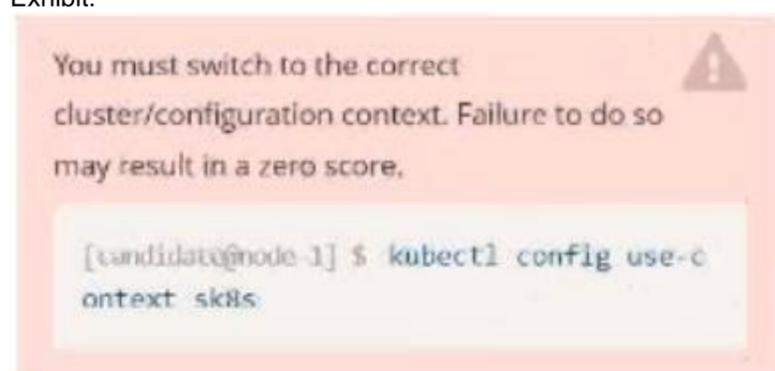
apiVersion: batch/v1beta1
kind: CronJob
metadata:
  name: hello
spec:
  jobTemplate:
    metadata:
      name: hello
    spec:
      template:
        spec:
          containers:
            - image: busybox
              name: hello
              args: ["/bin/sh", "-c", "date"]
          restartPolicy: Never
  schedule: */1 * * * *
  startingDeadlineSeconds: 22
  concurrencyPolicy: Allow
    
```

```

student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "*" * * * * --dry-run=
client -o yml > /opt/KDPD00301/periodic.yaml
error: unable to match a printer suitable for the output format "yml", allowed formats are: go-t
emplate,go-template-file,json,jsonpath,jsonpath-as-json,jsonpath-file,name,template,templatefile
,yaml
student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "*" * * * * --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
student@node-1:~$ vim /opt/KDPD00301/periodic.yaml
student@node-1:~$ kubectl create -f /opt/KDPD00301/periodic.yaml
cronjob.batch/hello created
student@node-1:~$ kubectl get cronjob
NAME          SCHEDULE          SUSPEND   ACTIVE   LAST SCHEDULE   AGE
hello        */1 * * * *      False    0        <none>          6s
student@node-1:~$
    
```

**NEW QUESTION 10**

Exhibit:



**Task:**

Create a Pod named nginx resources in the existing pod resources namespace. Specify a single container using nginx:stable image. Specify a resource request of 300m cpus and 1G1 of memory for the Pod's container.

- A. Mastered
- B. Not Mastered

**Answer:** A

**Explanation:**

**Solution:**

```
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ kubectl run nginx-resources -n pod-resources --image=nginx:stable --dry-run=client -o yaml > hw.yaml
candidate@node-1:~$ vim hw.yaml
```

Text Description automatically generated with medium confidence

```
File Edit View Terminal Tabs Help
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: nginx-resources
  name: nginx-resources
  namespace: pod-resources
spec:
  containers:
  - image: nginx:stable
    name: nginx-resources
    resources:
      requests:
        cpu: 300m
        memory: "1G1"
```

Text Description automatically generated

```
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ kubectl run nginx-resources -n pod-resources --image=nginx:stable --dry-run=client -o yaml > hw.yaml
candidate@node-1:~$ vim hw.yaml
candidate@node-1:~$ kubectl create -f hw.yaml
pod/nginx-resources created
candidate@node-1:~$ kubectl get pods -n pod-resources
NAME          READY   STATUS    RESTARTS   AGE
nginx-resources 1/1     Running   0           13s
candidate@node-1:~$ kubectl describe pods -n pod-resources
```

Text Description automatically generated

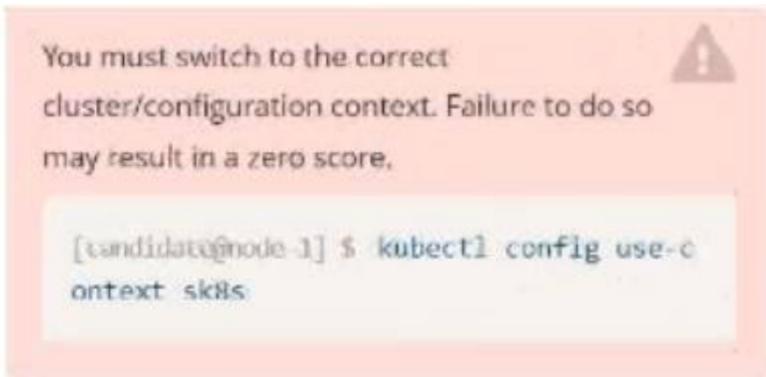
```

File Edit View Terminal Tabs Help
memory: 1Gi
Environment: <none>
Mounts:
  /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-dmx9j (ro)
Conditions:
  Type           Status
  Initialized     True
  Ready           True
  ContainersReady True
  PodScheduled    True
Volumes:
  kube-api-access-dmx9j:
    Type:          Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds: 3607
    ConfigMapName:  kube-root-ca.crt
    ConfigMapOptional: <nil>
    DownwardAPI:    true
QoS Class:       Burstable
Node-Selectors:  <none>
Tolerations:     node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                  node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:
  Type     Reason      Age   From          Message
  ----     -
  Normal   Scheduled   28s   default-scheduler   Successfully assigned pod-resources/nginx-resources to k8s-node-0
  Normal   Pulling     19s   kubelet        Pulling image "nginx:stable"
  Normal   Pulled      13s   kubelet        Successfully pulled image "nginx:stable" in 6.55664052s
  Normal   Created     13s   kubelet        Created container nginx-resources
  Normal   Started     12s   kubelet        Started container nginx-resources
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ kubectl create deploy expose -n ckad00014 --image lfccncf/nginx:1.13.7 --dry-run=client -o yaml>

```

**NEW QUESTION 10**

Exhibit:



Task:

Modify the existing Deployment named broker-deployment running in namespace quetzal so that its containers. The broker-deployment is manifest file can be found at:

```
~/daring-mocasin/broker-deployment.yaml
```

- A. Mastered
- B. Not Mastered

**Answer: A**

**Explanation:**

Solution:

```

candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ vim

```

Text Description automatically generated

```

File Edit View Terminal Tabs Help
containers:
- name: broker
  image: redis:alpine
  ports:
  - containerPort: 6379
securityContext:
  runAsUser: 30000
  privileged: false

candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ vim ~/daring-moccasin/broker-deployment.yaml
candidate@node-1:~$ kubectl apply -f ~/daring-moccasin/broker-deployment.yaml
deployment.apps/broker-deployment configured
candidate@node-1:~$ kubectl get pods -n quetzal
NAME                                READY   STATUS    RESTARTS   AGE
broker-deployment-65446d6d94-868p6  1/1     Running   0           30s
broker-deployment-65446d6d94-8dn7l  1/1     Running   0           32s
broker-deployment-65446d6d94-p4h4l  1/1     Running   0           31s
candidate@node-1:~$ kubectl get deploy -n quetzal
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
broker-deployment  3/3     3             3           7h3m
candidate@node-1:~$

```

**NEW QUESTION 11**

Exhibit:



Context

As a Kubernetes application developer you will often find yourself needing to update a running application. Task Please complete the following:

- Update the app deployment in the kdpd00202 namespace with a maxSurge of 5% and a maxUnavailable of 2%
- Perform a rolling update of the web1 deployment, changing the Ifccncf/ngmx image version to 1.13
- Roll back the app deployment to the previous version

- A. Mastered
- B. Not Mastered

**Answer: A**

**Explanation:**

Solution:

```

Readme  Web Terminal  THE LINUX FOUNDATION

uid: 1dfa2527-5c61-46a9-8dd3-e24643d3ce14
spec:
  progressDeadlineSeconds: 600
  replicas: 10
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app: nginx
  strategy:
    rollingUpdate:
      maxSurge: 5%
      maxUnavailable: 2
    type: RollingUpdate
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: nginx
    spec:
      containers:
      - image: lfcncf/nginx:1.13
        imagePullPolicy: IfNotPresent
        name: nginx
        ports:
        - containerPort: 80
          protocol: TCP
:wq!

```

```

Readme  Web Terminal  THE LINUX FOUNDATION

student@node-1:~$ kubectl edit deployment app -n kdpd00202
deployment.apps/app edited
student@node-1:~$ kubectl rollout status deployment app -n kdpd00202
Waiting for deployment "app" rollout to finish: 6 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 7 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 7 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 7 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 8 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 8 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 8 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 8 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 9 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 9 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 9 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 1 old replicas are pending termination...
Waiting for deployment "app" rollout to finish: 8 of 10 updated replicas are available...
Waiting for deployment "app" rollout to finish: 9 of 10 updated replicas are available...
deployment "app" successfully rolled out
student@node-1:~$ kubectl rollout undo deployment app -n kdpd00202
deployment.apps/app rolled back
student@node-1:~$ kubectl rollout status deployment app -n kdpd00202

```

```

student@node-1:~$ kubectl rollout status deployment app -n kdpd00202
Waiting for deployment "app" rollout to finish: 6 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 6 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 6 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 6 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 7 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 7 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 9 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 9 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 1 old replicas are pending termination...
Waiting for deployment "app" rollout to finish: 1 old replicas are pending termination...
Waiting for deployment "app" rollout to finish: 1 old replicas are pending termination...
Waiting for deployment "app" rollout to finish: 8 of 10 updated replicas are available...
Waiting for deployment "app" rollout to finish: 9 of 10 updated replicas are available...
deployment "app" successfully rolled out
student@node-1:~$

```

**NEW QUESTION 12**

Exhibit:



Task

You have rolled out a new pod to your infrastructure and now you need to allow it to communicate with the web and storage pods but nothing else. Given the running pod kdsn00201 -newpod edit it to use a network policy that will allow it to send and receive traffic only to and from the web and storage pods.

All work on this item should be conducted in the `kdsn00201` namespace.

All required `NetworkPolicy` resources are already created and ready for use as appropriate. You should not create, modify or delete any network policies whilst completing this item.

- A. Mastered
- B. Not Mastered

**Answer:** A

**Explanation:**

```
apiVersion: networking.k8s.io/v1 kind: NetworkPolicy
metadata:
name: internal-policy namespace: default spec:
podSelector: matchLabels: name: internal policyTypes:
- Egress
- Ingress ingress:
- {}
egress:
- to:
- podSelector: matchLabels: name: mysql ports:
- protocol: TCP port: 3306
- to:
- podSelector: matchLabels:
name: payroll ports:
- protocol: TCP port: 8080
- ports:
- port: 53 protocol: UDP
- port: 53 protocol: TCP
```

**NEW QUESTION 17**

.....

## **Thank You for Trying Our Product**

### **We offer two products:**

1st - We have Practice Tests Software with Actual Exam Questions

2nd - Questions and Answers in PDF Format

### **CKAD Practice Exam Features:**

- \* CKAD Questions and Answers Updated Frequently
- \* CKAD Practice Questions Verified by Expert Senior Certified Staff
- \* CKAD Most Realistic Questions that Guarantee you a Pass on Your First Try
- \* CKAD Practice Test Questions in Multiple Choice Formats and Updates for 1 Year

**100% Actual & Verified — Instant Download, Please Click**  
**[Order The CKAD Practice Test Here](#)**