



CKAD^{Q&As}

Certified Kubernetes Application Developer (CKAD) Program

Pass Linux Foundation CKAD Exam with 100% Guarantee

Free Download Real Questions & Answers **PDF** and **VCE** file from:

<https://www.passapply.com/ckad.html>

100% Passing Guarantee
100% Money Back Assurance

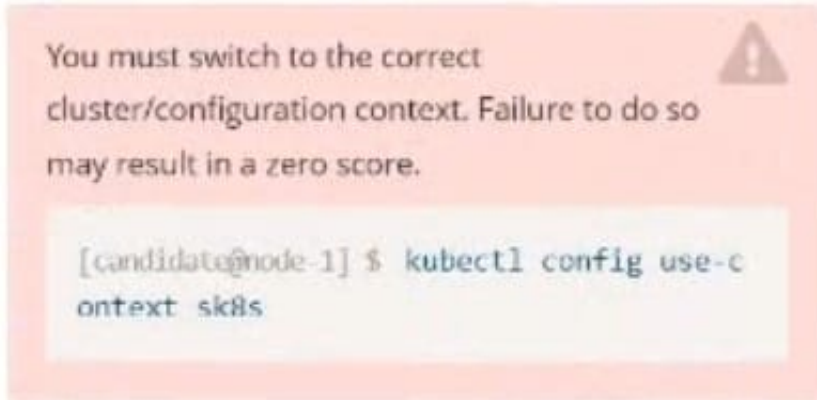
Following Questions and Answers are all new published by Linux Foundation Official Exam Center

-  **Instant Download** After Purchase
-  **100% Money Back** Guarantee
-  **365 Days** Free Update
-  **800,000+** Satisfied Customers



**QUESTION 1**

CORRECT TEXT



Task:

A pod within the Deployment named buffalo-deployment and in namespace gorilla is logging errors.

Look at the logs identify errors messages.

Find errors, including User "system:serviceaccount:gorilla:default" cannot list resource "deployment" [...] in the namespace "gorilla"

The buffalo-deployment `S manifest can be found at `-/prompt/escargot/buffalo- deployment.yaml`

A. Please check explanations

B. Place Holder

Correct Answer: A



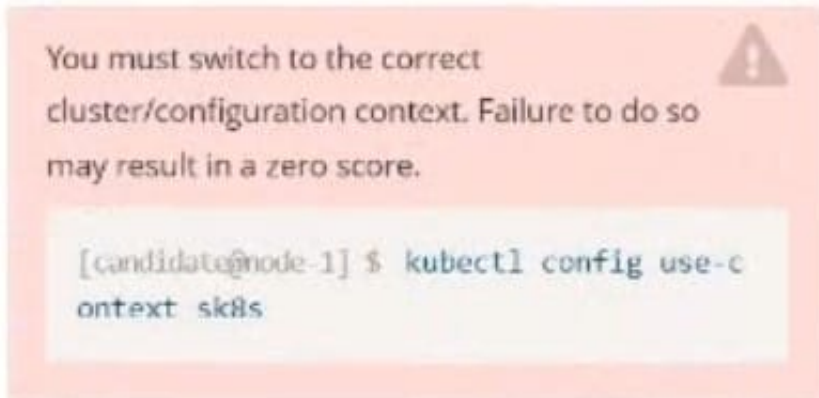
```

File Edit View Terminal Tabs Help
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:~$ 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           5h38m
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:~$

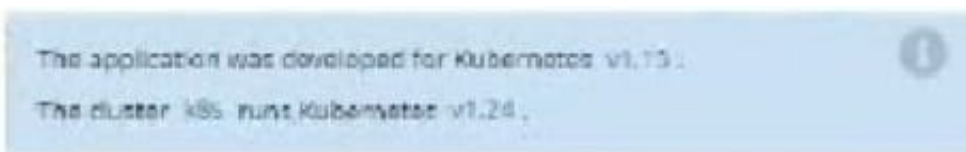
```

QUESTION 2

CORRECT TEXT



Task:



A. Please check explanations

B. Place Holder



Correct Answer: A

```
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ vim ~/credible-mite/www.yaml
```

```
File Edit View Terminal Tabs Help
apiVersion: apps/v1
kind: Deployment
metadata:
  name: www-deployment
  namespace: cobra
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: "nginx:stable"
          ports:
            - containerPort: 80
          volumeMounts:
            - mountPath: /var/log/nginx
              name: logs
          env:
            - name: NGINX_ENTRYPOINT_QUIET_LOGS
              value: "1"
      volumes:
        - name: logs
          emptyDir: {}
~
:wq
```

```
File Edit View Terminal Tabs Help
deployment.apps/expose created
candidate@node-1:~$ kubectl get pods -n ckad00014
NAME                READY   STATUS              RESTARTS   AGE
expose-85dd99d4d9-25675 0/1     ContainerCreating   0           6s
expose-85dd99d4d9-4fhcc 0/1     ContainerCreating   0           6s
expose-85dd99d4d9-fl7j  0/1     ContainerCreating   0           6s
expose-85dd99d4d9-tt6rm 0/1     ContainerCreating   0           6s
expose-85dd99d4d9-vjd8b 0/1     ContainerCreating   0           6s
expose-85dd99d4d9-vtzpq 0/1     ContainerCreating   0           6s
candidate@node-1:~$ kubectl get deploy -n ckad00014
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
expose 6/6         6             6           15s
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ vim ~/credible-mite/www.yaml
candidate@node-1:~$ vim ~/credible-mite/www.yaml
candidate@node-1:~$ kubectl apply -f ~/credible-mite/www.yaml
deployment.apps/www-deployment created
candidate@node-1:~$ kubectl get pods -n cobra
NAME                READY   STATUS              RESTARTS   AGE
www-deployment-d899c6b49-d6ccg 1/1     Running          0           6s
www-deployment-d899c6b49-f796l 0/1     ContainerCreating 0           6s
www-deployment-d899c6b49-ztfcw 0/1     ContainerCreating 0           6s
candidate@node-1:~$ kubectl get deploy -n cobra
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
www-deployment 3/3         3             3           11s
candidate@node-1:~$ kubectl get pods -n cobra
NAME                READY   STATUS              RESTARTS   AGE
www-deployment-d899c6b49-d6ccg 1/1     Running          0           14s
www-deployment-d899c6b49-f796l 1/1     Running          0           14s
www-deployment-d899c6b49-ztfcw 1/1     Running          0           14s
candidate@node-1:~$
```



QUESTION 3

CORRECT TEXT



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:

1.

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

2.

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.

1.

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

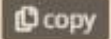
2.

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



You can access `sk8s-node-0` by  issuing the following command:

```
[student@node-1] $ | ssh sk8s-node-0
```

Ensure that you return to the base node (with hostname `node-1`) once you have completed your work on `sk8s-node-0` 

A. Please check explanations

B. Place Holder

Correct Answer: A



```
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
```



```

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

```

```

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

```

```

THE LINUX FOUNDATION
Web Terminal
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: task-pv-claim
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 100Mi
  storageClassName: storage

```

```

student@sk8a-node-01~$ kubectl create -f pv.yml
persistentvolume/task-pv-volume created
student@sk8a-node-01~$ kubectl create -f pvc.yml
persistentvolumeclaim/task-pv-claim created
student@sk8a-node-01~$ 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      9s
student@sk8a-node-01~$ kubectl get pvc
NAME          STATUS  VOLUME          CAPACITY  ACCESS MODES  STORAGECLASS  AGE
task-pv-claim  Bound   task-pv-volume  1Gi       RWO           storage        9s
student@sk8a-node-01~$ vim pod.yml

```

```

THE LINUX FOUNDATION
Web Terminal
apiVersion: v1
kind: Pod
metadata:
  name: mypod
spec:
  labels:
    app: my-storage-app
  containers:
    - name: myfrontend
      image: nginx
      volumeMounts:
        - mountPath: "/var/www/html"
          name: mypod
      volumeMounts:
        - name: mypod
          persistentVolumeClaim:
            claimName: task-pv-claim

```

```

student@sk8a-node-01~$ kubectl create -f pod.yml
pod/mypod created
student@sk8a-node-01~$ kubectl get

```

```

THE LINUX FOUNDATION
Web Terminal
student@sk8a-node-01~$ kubectl get pods
NAME    READY  STATUS             RESTARTS  AGE
mypod   0/1    ContainerCreating  0         4s
student@sk8a-node-01~$ kubectl get pods
NAME    READY  STATUS             RESTARTS  AGE
mypod   0/1    ContainerCreating  0         8s
student@sk8a-node-01~$ kubectl get pods
NAME    READY  STATUS             RESTARTS  AGE
mypod   1/1    Running            0         10s
student@sk8a-node-01~$ logout
Connection to 10.250.3.115 closed.
student@node-1~$

```




QUESTION 4

CORRECT TEXT



Context

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

Task

The desired behavior is to have Kubernetes 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:

1.

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.

2.

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.

3.

Configure the probe-pod pod provided to use these endpoints

4.

The probes should use port 8080

A. Please check explanations

B. Place Holder

Correct Answer: A

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:

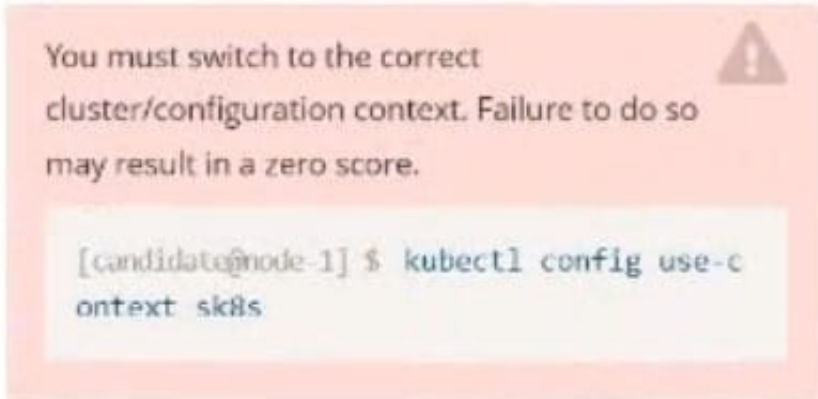
```

NAME READY STATUS RESTARTS AGE
liveness-exec 1/1 Running 1 1m

```

QUESTION 5

CORRECT TEXT



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 1Gi of memory for the Pod's container.

A. Please check explanations

B. Place Holder

Correct Answer: A



```
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
```

```
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: "1Gi"
```



```
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
```

```
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   20s   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>
```

[Latest CKAD Dumps](#)

[CKAD PDF Dumps](#)

[CKAD VCE Dumps](#)