CKAD Reliable Exam Cram - Dump CKAD Check

Exam Linux Foundation CKAD Reference, Dumps CKAD Vce

[illegible]

[illegible]

Dumps CKAD Vce - Exam CKAD Bootcamp

[illegible]

What's more, part of that ValidDumps CKAD dumps now are free: https://drive.google.com/open?id=1mkHmMo_l6Iojox7ij3An05FO-hQ5edMg

The information technology market has become very competitive. Linux Foundation CKAD technologies and services are constantly evolving. Therefore, the Linux Foundation CKAD certification has become very important to advance one's career. Success in the Linux Foundation Certified Kubernetes Application Developer Exam CKAD exam validates and upgrades your skills in Linux Foundation CKAD technologies. It is the main reason behind the popularity of the Linux Foundation CKAD certification exam. You must put all your efforts to clear the challenging Linux Foundation CKAD examination. However, cracking the CKAD test is not an easy task.

Maybe you are unfamiliar with our CKAD latest material, but our CKAD real questions are applicable to this exam with high passing rate up to 98 percent and over. Choosing from a wide assortment of practice materials, rather than aiming solely to make a profit from our CKAD latest material, we are determined to offer help. Quick purchase process, free demos and various versions and high quality CKAD Real Questions are al features of our advantageous practice materials. With passing rate up to 98 to 100 percent, you will get through the CKAD practice exam with ease. So they can help you save time and cut down additional time to focus on the CKAD practice exam review only.

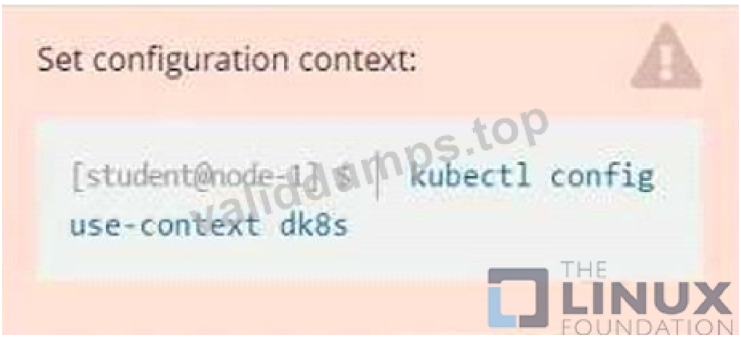
>> CKAD Reliable Exam Cram <<

Tips to Crack the CKAD Exam
]

If you are forced to pass exams and obtain certification by your manger, our CKAD original questions will be a good choice for you. Our products can help you clear exams at first shot. We promise that we provide you with best quality CKAD original questions and competitive prices. We offer 100% pass products with excellent service. We provide one year studying assist service and one year free updates downloading of Linux Foundation CKAD Exam Questions. If you fail exam we support to exchange and full refund.

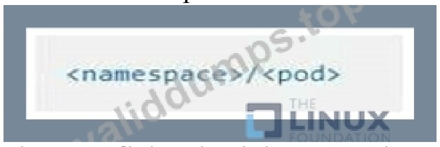
Linux Foundation Certified Kubernetes Application Developer Exam Sample Questions (Q70-Q75):

NEW QUESTION # 70
Context


Set configuration context:

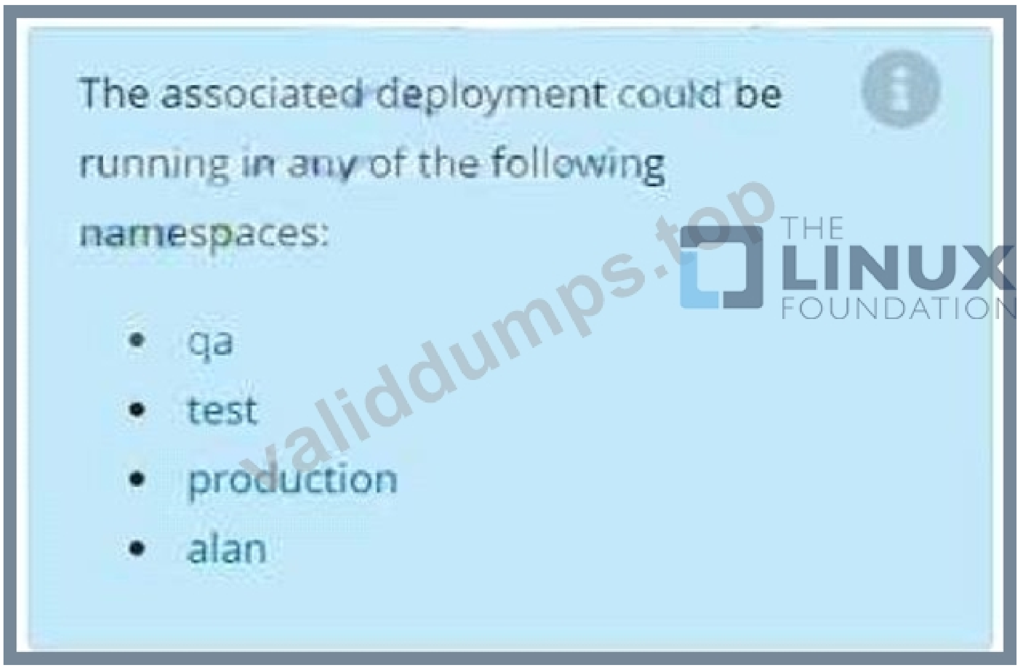
```
[student@node-1] $ | kubectl config
use-context dk8s
```

Context
A user has reported an aopticauon is unteachable due to a failing livenessProbe .
Task
Perform the following tasks:
* Find the broken pod and store its name and namespace to /opt/KDOB00401/broken.txt in the format:



```
<namespace>/<pod>
```

The output file has already been created
* Store the associated error events to a file /opt/KDOB00401/error.txt, The output file has already been created. You will need to use the -o wide output specifier with your command
* Fix the issue.


The associated deployment could be running in any of the following namespaces:

- qa
- test
- production
- alan

Answer:

Explanation:
Solution:
Create the Pod:

kubectl create -f http://k8s.io/docs/tasks/configure-pod-container/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 "gcr.io/google_containers/busybox"
23s 23s 1 {kubelet worker0} spec.containers{liveness} Normal Pulled Successfully pulled image
"gcr.io/google_containers/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 "gcr.io/google_containers/busybox"
36s 36s 1 {kubelet worker0} spec.containers{liveness} Normal Pulled Successfully pulled image
"gcr.io/google_containers/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 m
```

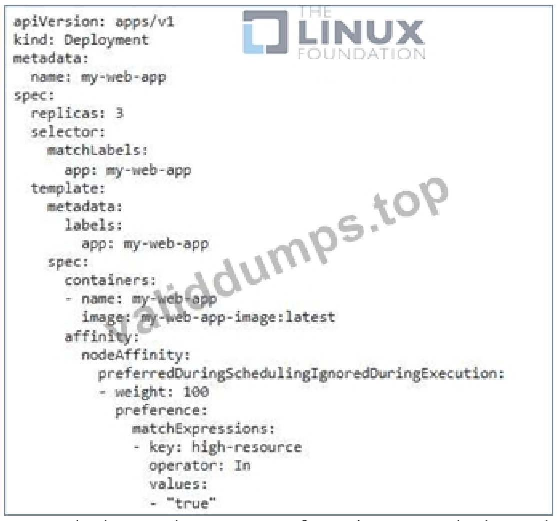
NEW QUESTION # 71
You have a Deployment running a web application that is scaling dynamically based on traffic. However, the application occasionally experiences Slow response times during peak traffic periods. You suspect that the pods are being scheduled on nodes that are already under pressure. To improve the performance, you want to implement node affinity, ensuring that pods are scheduled on nodes with specific labels that indicate high resources and low utilization.

Answer:

Explanation:
See the solution below with Step by Step Explanation.
Explanation:
Solution (Step by Step) :
1. Define Node Labels:
- Identify nodes with high resources and low utilization.
- Label these nodes with a specific label like 'high-resource':
bash

```
kubectl label nodes node-name high-resource=true
```

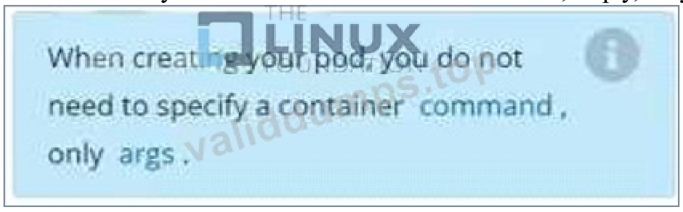
2. Configure Node Affinity in Deployment
- Update tne Deployment YAML to include node affinity rules.
- preferredDuringSchedulingIgnoredDuringExecution: This affinity rule indicates a preference for scheduling pods on nodes with specific labels. It doesn't prevent scheduling on other nodes if preferred nodes are unavailable.



```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-web-app
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-web-app
  template:
    metadata:
      labels:
        app: my-web-app
    spec:
      containers:
      - name: my-web-app
        image: my-web-app-image:latest
      affinity:
        nodeAffinity:
          preferredDuringSchedulingIgnoredDuringExecution:
          - weight: 100
            preference:
              matchExpressions:
              - key: high-resource
                operator: In
                values:
                - "true"
```

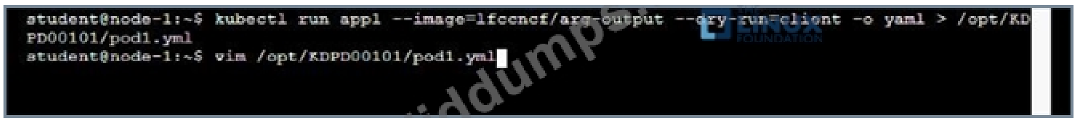
3. Apply the Deployment Configuration: - Apply the updated Deployment configuration to your Kubernetes cluster: bash kubectl apply -f my-web-app-deployment.yaml 4. Monitor Pod Scheduling: - Use 'kubectl get pods -l app=my-web-app' to monitor the pod scheduling. - Verity that the pods are being scheduled on nodes with the 'high-resource' label.

NEW QUESTION # 72
Context
Anytime a team needs to run a container on Kubernetes they will need to define a pod within which to run the container.
Task
Please complete the following:
* Create a YAML formatted pod manifest
/opt/KDPD00101/podl.yml to create a pod named app1 that runs a container named app1cont using image Ifccncf/arg-output with these command line arguments: -lines 56 -F
* Create the pod with the kubect1 command using the YAML file created in the previous step
* When the pod is running display summary data about the pod in JSON format using the kubect1 command and redirect the output to a file named /opt/KDPD00101/out1.json
* All of the files you need to work with have been created, empty, for your convenience

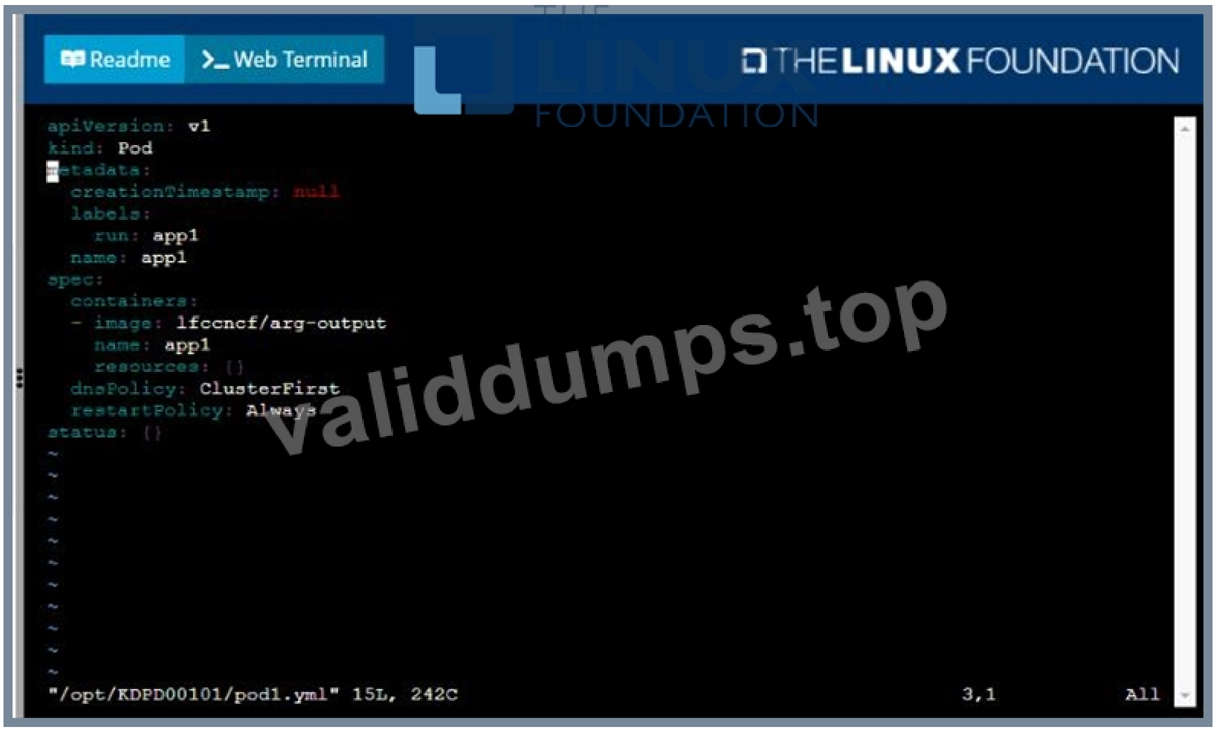


When creating your pod, you do not need to specify a container command, only args.

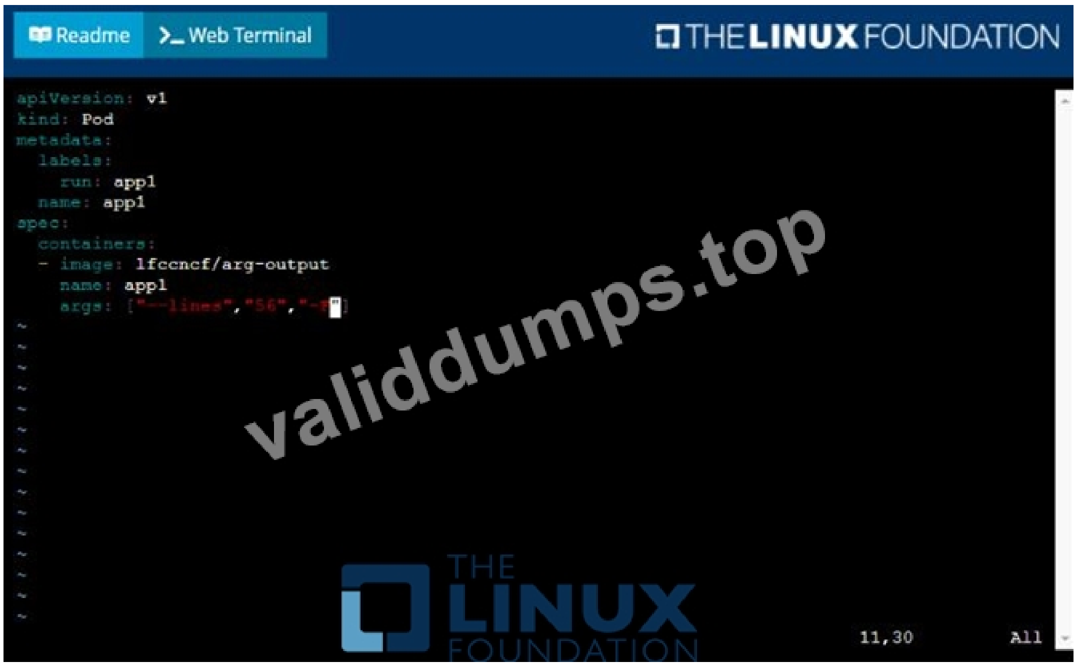
- A. Solution:



```
student@node-1:~$ kubectl run app1 --image=lfccncf/arg-output --dry-run=client -o yaml > /opt/KD
PD00101/pod1.yml
student@node-1:~$ vim /opt/KDPD00101/pod1.yml
```

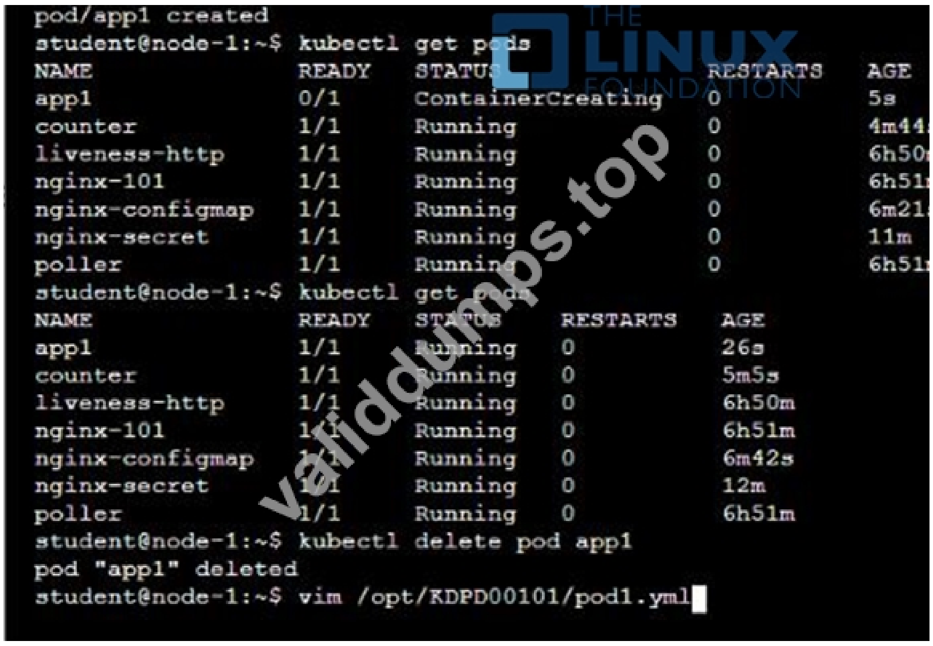


```
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: app1
  name: app1
spec:
  containers:
  - image: lfccncf/arg-output
    name: app1
    resources: {}
  dnsPolicy: ClusterFirst
  restartPolicy: Always
status: {}
~
~
~
~
~
~
~
~
~
~
~
"/opt/KDPD00101/pod1.yml" 15L, 242C                              3,1           All
```



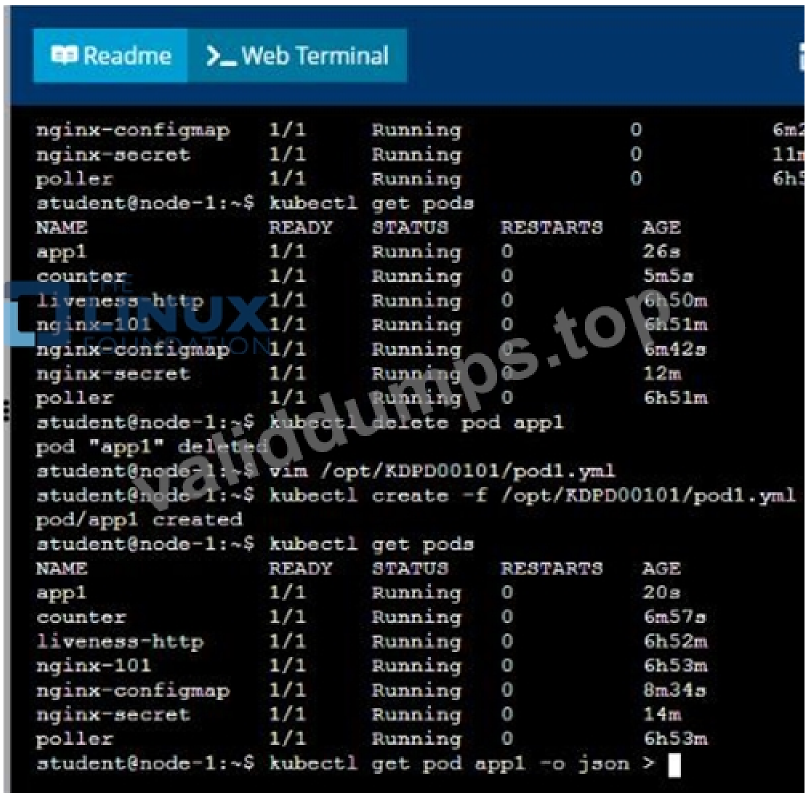
```
apiVersion: v1
kind: Pod
metadata:
  labels:
    run: app1
  name: app1
spec:
  containers:
  - image: lfccncf/arg-output
    name: app1
    args: ["--lines","56","-F"]
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
                                                       11,30         All
```

validdumps.top



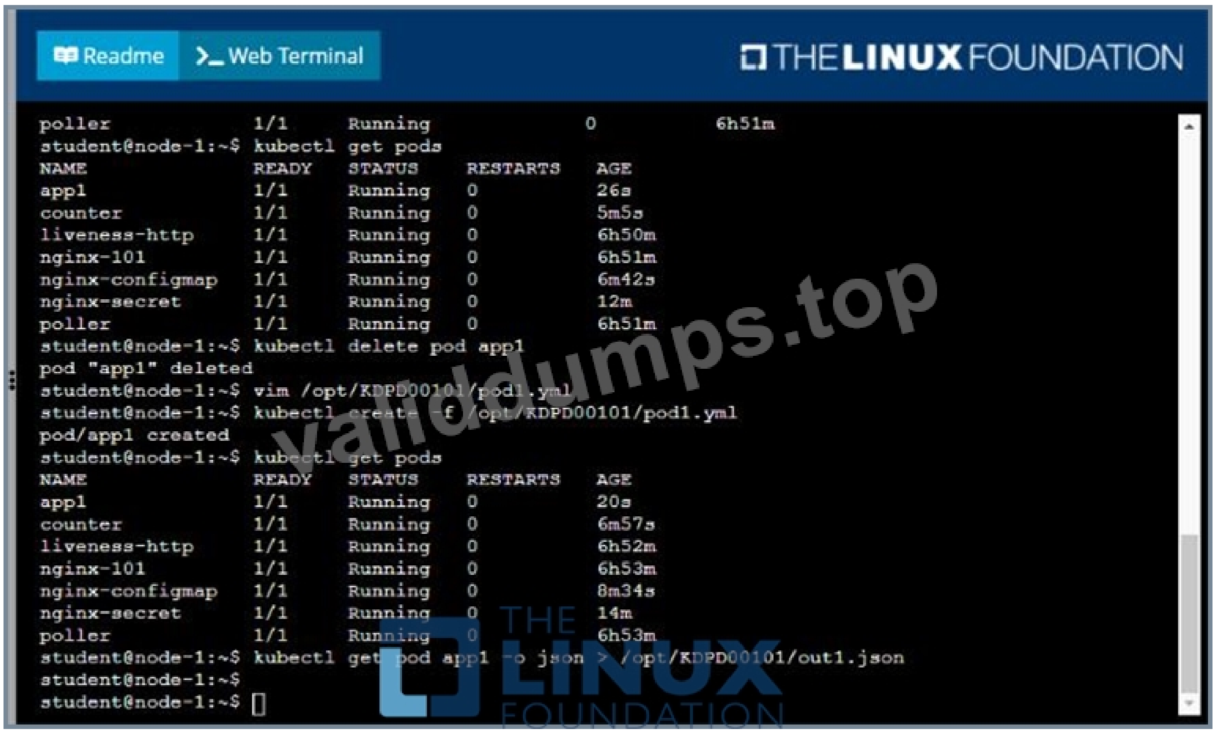
```
pod/app1 created
student@node-1:~$ kubectl get pods
NAME               READY   STATUS              RESTARTS   AGE
app1               0/1     ContainerCreating   0          5s
counter            1/1     Running             0          4m44
liveness-http      1/1     Running             0          6h50
nginx-101          1/1     Running             0          6h51
nginx-configmap    1/1     Running             0          6m21
nginx-secret       1/1     Running             0          11m
poller             1/1     Running             0          6h51
student@node-1:~$ kubectl get pods
NAME               READY   STATUS    RESTARTS   AGE
app1               1/1     Running   0          26s
counter            1/1     Running   0          5m5s
liveness-http      1/1     Running   0          6h50m
nginx-101          1/1     Running   0          6h51m
nginx-configmap    1/1     Running   0          6m42s
nginx-secret       1/1     Running   0          12m
poller             1/1     Running   0          6h51m
student@node-1:~$ kubectl delete pod app1
pod "app1" deleted
student@node-1:~$ vim /opt/KDPD00101/pod1.yml
```

THE LINUX FOUNDATION
validdumps.top



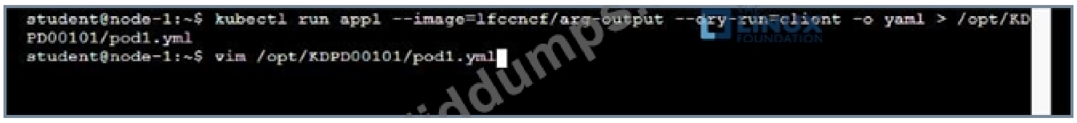
```
nginx-configmap    1/1     Running             0          6m2
nginx-secret       1/1     Running             0          11m
poller             1/1     Running             0          6h5
student@node-1:~$ kubectl get pods
NAME               READY   STATUS    RESTARTS   AGE
app1               1/1     Running   0          26s
counter            1/1     Running   0          5m5s
liveness-http      1/1     Running   0          6h50m
nginx-101          1/1     Running   0          6h51m
nginx-configmap    1/1     Running   0          6m42s
nginx-secret       1/1     Running   0          12m
poller             1/1     Running   0          6h51m
student@node-1:~$ kubectl delete pod app1
pod "app1" deleted
student@node-1:~$ vim /opt/KDPD00101/pod1.yml
student@node-1:~$ kubectl create -f /opt/KDPD00101/pod1.yml
pod/app1 created
student@node-1:~$ kubectl get pods
NAME               READY   STATUS    RESTARTS   AGE
app1               1/1     Running   0          20s
counter            1/1     Running   0          6m57s
liveness-http      1/1     Running   0          6h52m
nginx-101          1/1     Running   0          6h53m
nginx-configmap    1/1     Running   0          8m34s
nginx-secret       1/1     Running   0          14m
poller             1/1     Running   0          6h53m
student@node-1:~$ kubectl get pod app1 -o json >
```

THE LINUX FOUNDATION
validdumps.top

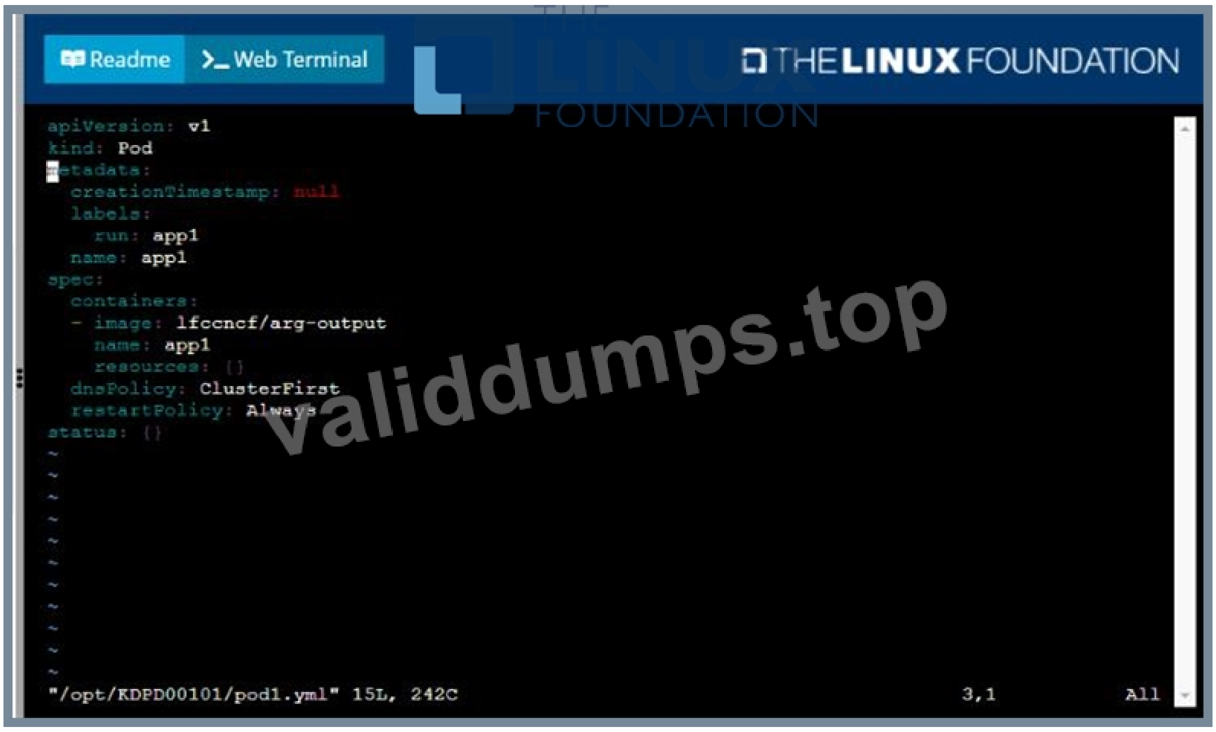


```
poller              1/1     Running             0          6h51m
student@node-1:~$ kubectl get pods
NAME                READY   STATUS    RESTARTS   AGE
app1                1/1     Running   0          26s
counter             1/1     Running   0          5m5s
liveness-http       1/1     Running   0          6h50m
nginx-101           1/1     Running   0          6h51m
nginx-configmap     1/1     Running   0          6m42s
nginx-secret        1/1     Running   0          12m
poller              1/1     Running   0          6h51m
student@node-1:~$ kubectl delete pod app1
pod "app1" deleted
student@node-1:~$ vim /opt/KDPD00101/pod1.yml
student@node-1:~$ kubectl create -f /opt/KDPD00101/pod1.yml
pod/app1 created
student@node-1:~$ kubectl get pods
NAME                READY   STATUS    RESTARTS   AGE
app1                1/1     Running   0          20s
counter             1/1     Running   0          6m57s
liveness-http       1/1     Running   0          6h52m
nginx-101           1/1     Running   0          6h53m
nginx-configmap     1/1     Running   0          8m34s
nginx-secret        1/1     Running   0          14m
poller              1/1     Running   0          6h53m
student@node-1:~$ kubectl get pod app1 -o json > /opt/KDPD00101/out1.json
student@node-1:~$
student@node-1:~$ 
```

- B. Solution:

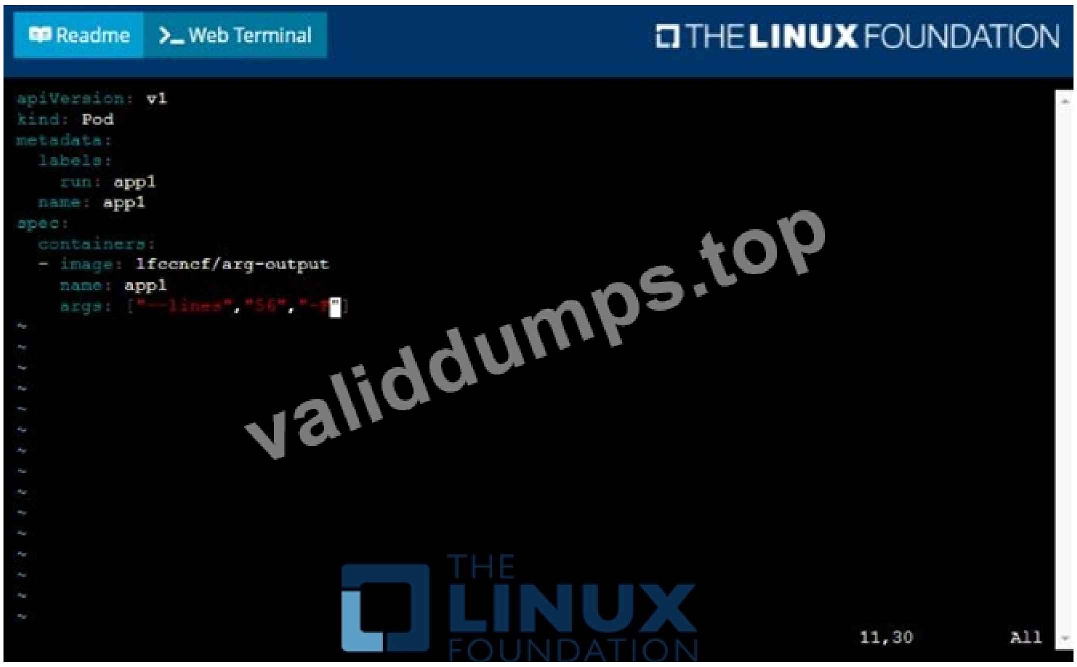


```
student@node-1:~$ kubectl run app1 --image=lfccncf/arg-output --dry-run=client -o yaml > /opt/KD
PD00101/pod1.yml
student@node-1:~$ vim /opt/KDPD00101/pod1.yml
```

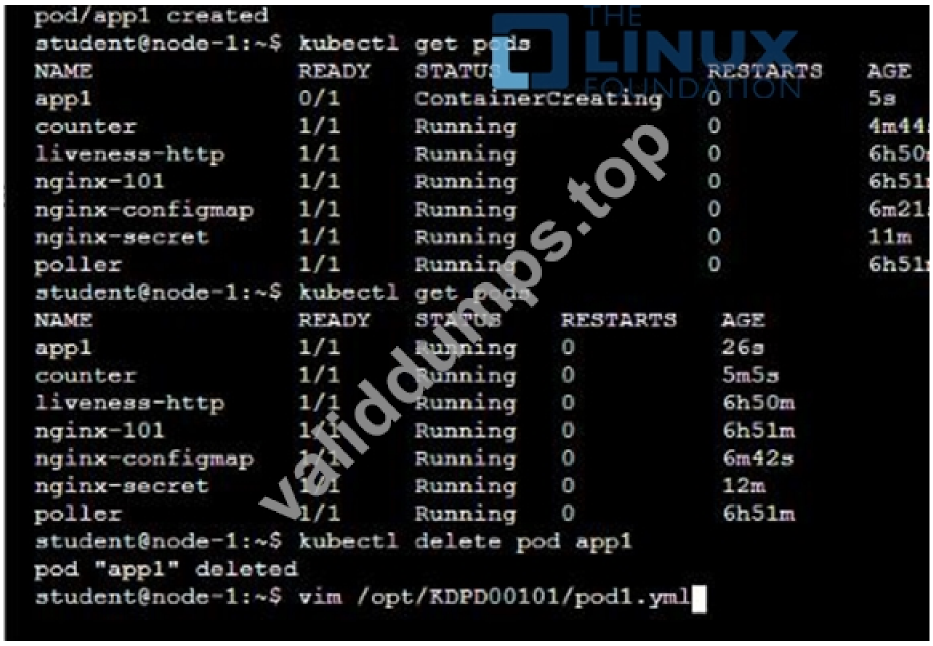


```
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: app1
  name: app1
spec:
  containers:
  - image: lfccncf/arg-output
    name: app1
    resources: {}
  dnsPolicy: ClusterFirst
  restartPolicy: Always
status: {}
~
~
~
~
~
~
~
~
~
~
~
"/opt/KDPD00101/pod1.yml" 15L, 242C                                   3,1           All
```

validdumps.top

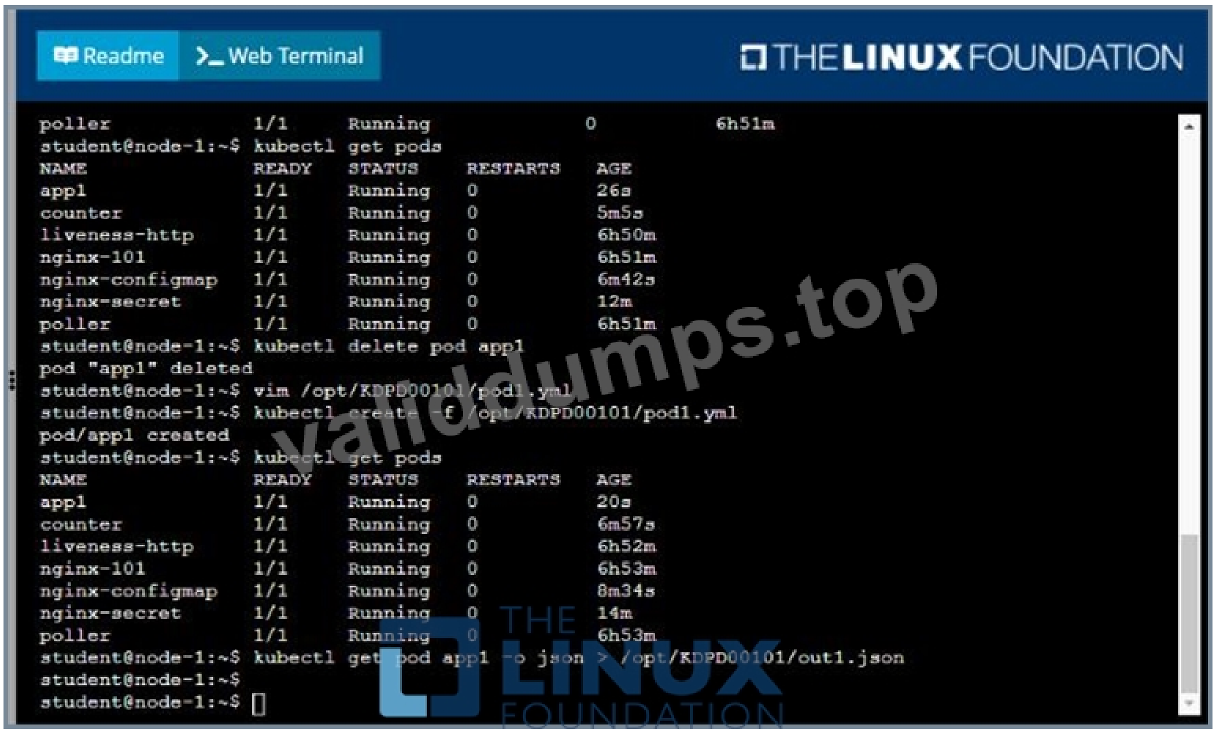


```
apiVersion: v1
kind: Pod
metadata:
  labels:
    run: app1
  name: app1
spec:
  containers:
  - image: lfccncf/arg-output
    name: app1
    args: ["--lines","56","-F"]
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
                                                    11,30         All
```



```
pod/app1 created
student@node-1:~$ kubectl get pods
NAME              READY   STATUS              RESTARTS   AGE
app1              0/1     ContainerCreating   0          5s
counter           1/1     Running             0          4m44
liveness-http     1/1     Running             0          6h50
nginx-101         1/1     Running             0          6h51
nginx-configmap   1/1     Running             0          6m21
nginx-secret      1/1     Running             0          11m
poller            1/1     Running             0          6h51
student@node-1:~$ kubectl get pods
NAME              READY   STATUS    RESTARTS   AGE
app1              1/1     Running   0          26s
counter           1/1     Running   0          5m5s
liveness-http     1/1     Running   0          6h50m
nginx-101         1/1     Running   0          6h51m
nginx-configmap   1/1     Running   0          6m42s
nginx-secret      1/1     Running   0          12m
poller            1/1     Running   0          6h51m
student@node-1:~$ kubectl delete pod app1
pod "app1" deleted
student@node-1:~$ vim /opt/KDPD00101/pod1.yml
```

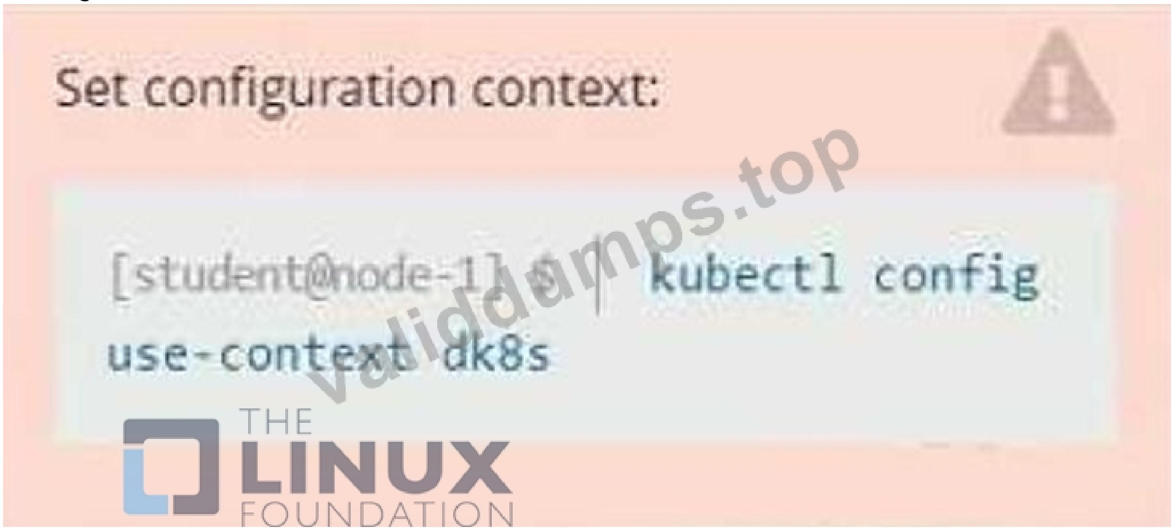
validdumps.top
validdumps.top



```
poller              1/1     Running             0          6h51m
student@node-1:~$ kubectl get pods
NAME                READY   STATUS    RESTARTS   AGE
app1                1/1     Running   0          26s
counter             1/1     Running   0          5m5s
liveness-http       1/1     Running   0          6h50m
nginx-101           1/1     Running   0          6h51m
nginx-configmap     1/1     Running   0          6m42s
nginx-secret        1/1     Running   0          12m
poller              1/1     Running   0          6h51m
student@node-1:~$ kubectl delete pod app1
pod "app1" deleted
student@node-1:~$ vim /opt/KDPD00101/pod1.yml
student@node-1:~$ kubectl create -f /opt/KDPD00101/pod1.yml
pod/app1 created
student@node-1:~$ kubectl get pods
NAME                READY   STATUS    RESTARTS   AGE
app1                1/1     Running   0          20s
counter             1/1     Running   0          6m57s
liveness-http       1/1     Running   0          6h52m
nginx-101           1/1     Running   0          6h53m
nginx-configmap     1/1     Running   0          8m34s
nginx-secret        1/1     Running   0          14m
poller              1/1     Running   0          6h53m
student@node-1:~$ kubectl get pod app1 -o json > /opt/KDPD00101/out1.json
student@node-1:~$
student@node-1:~$
```

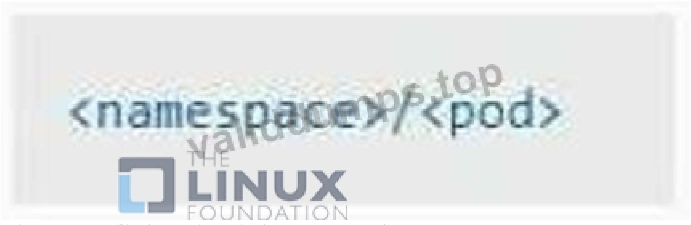
Answer: A

NEW QUESTION # 73


Set configuration context:

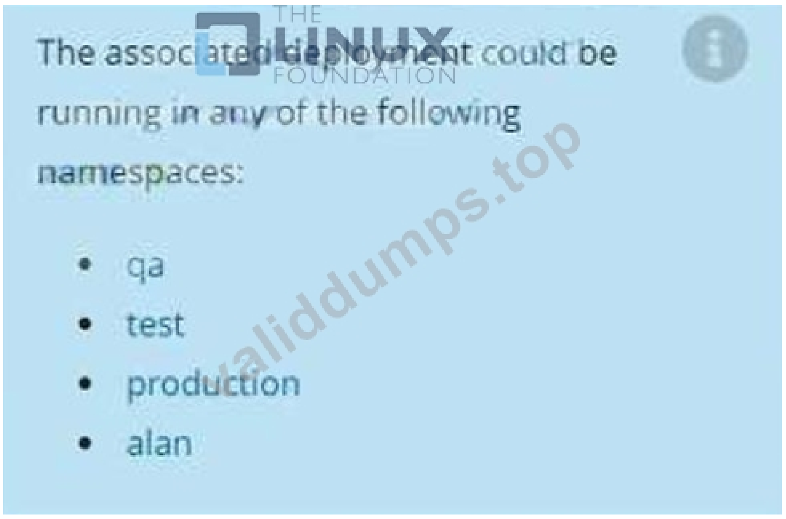
```
[student@node-1] $ | kubectl config
use-context dk8s
```

Context
A user has reported an aopticauon is unteachable due to a failing livenessProbe .
Task
Perform the following tasks:
* Find the broken pod and store its name and namespace to /opt/KDOB00401/broken.txt in the format:



```
<namespace>/<pod>
```

The output file has already been created
* Store the associated error events to a file /opt/KDOB00401/error.txt, The output file has already been created. You will need to use the -o wide output specifier with your command
* Fix the issue.



The associated deployment could be running in any of the following namespaces:

- qa
- test
- production
- alan

validdumps.top

Answer:

Explanation:
See the solution below.
Explanation
Solution:
Create the Pod:
kubectl create
-f http://k8s.io/docs/tasks/configure-pod-container/
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
"gcr.io/google_containers/busybox"
23s 23s 1 {kubelet worker0} spec.containers{liveness} Normal Pulled Successfully pulled image
"gcr.io/google_containers/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
```

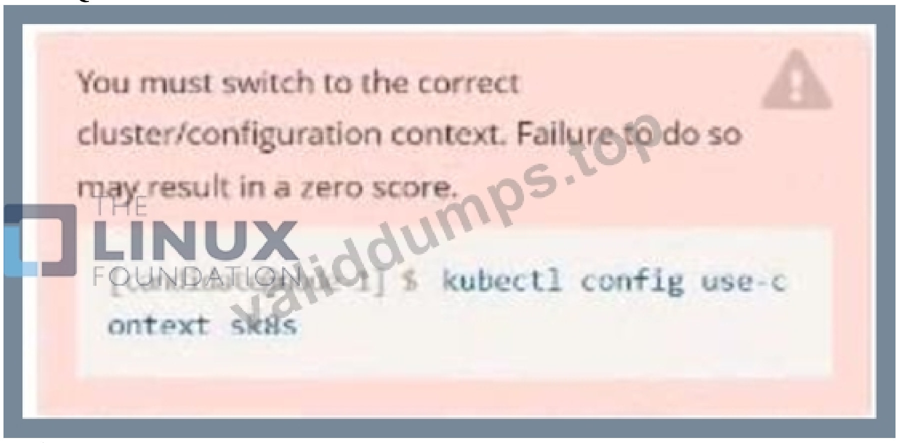
After 35 seconds, view the Pod events again:
kubectl describe pod liveness-exec
At the bottom of the output, there are messages indicating that the liveness probes have failed, and the containers have been killed and recreated.

```
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
"gcr.io/google_containers/busybox"
36s 36s 1 {kubelet worker0} spec.containers{liveness} Normal Pulled Successfully pulled image
"gcr.io/google_containers/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 m

NEW QUESTION # 74



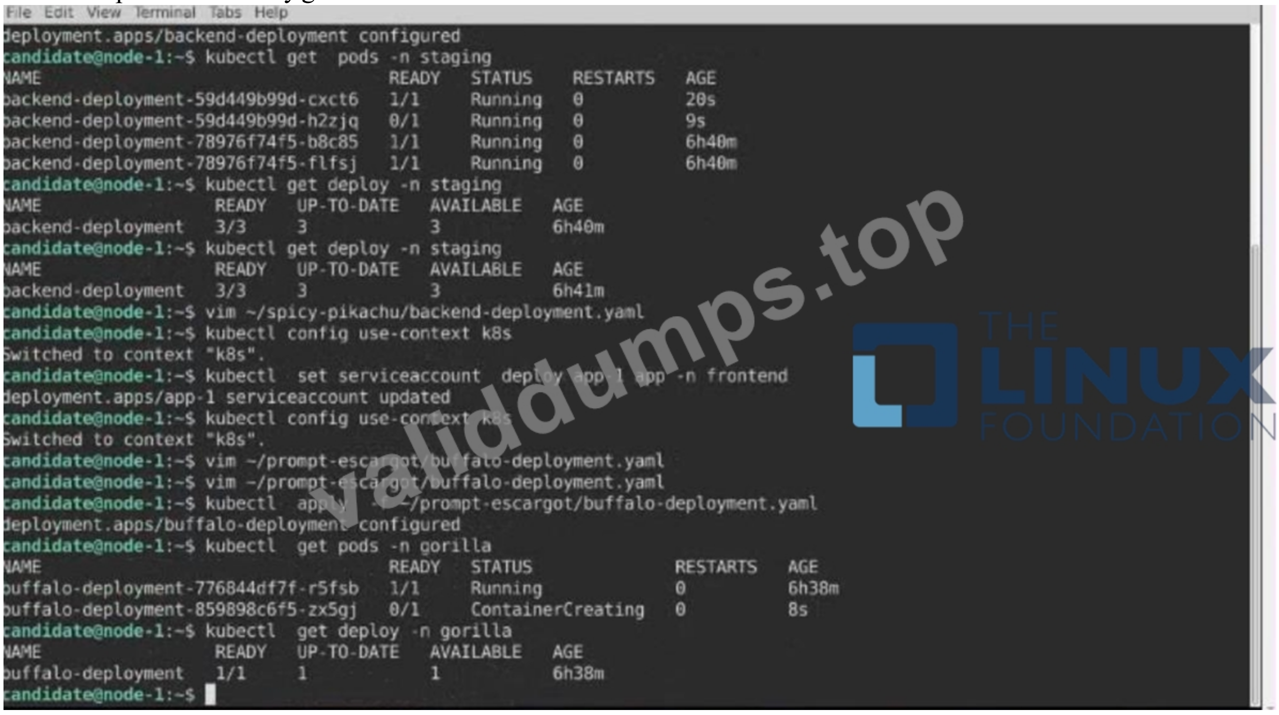
You must switch to the correct cluster/configuration context. Failure to do so may result in a zero score.

```
[candidate@node-1] $ kubectl config use-context sk8s
```

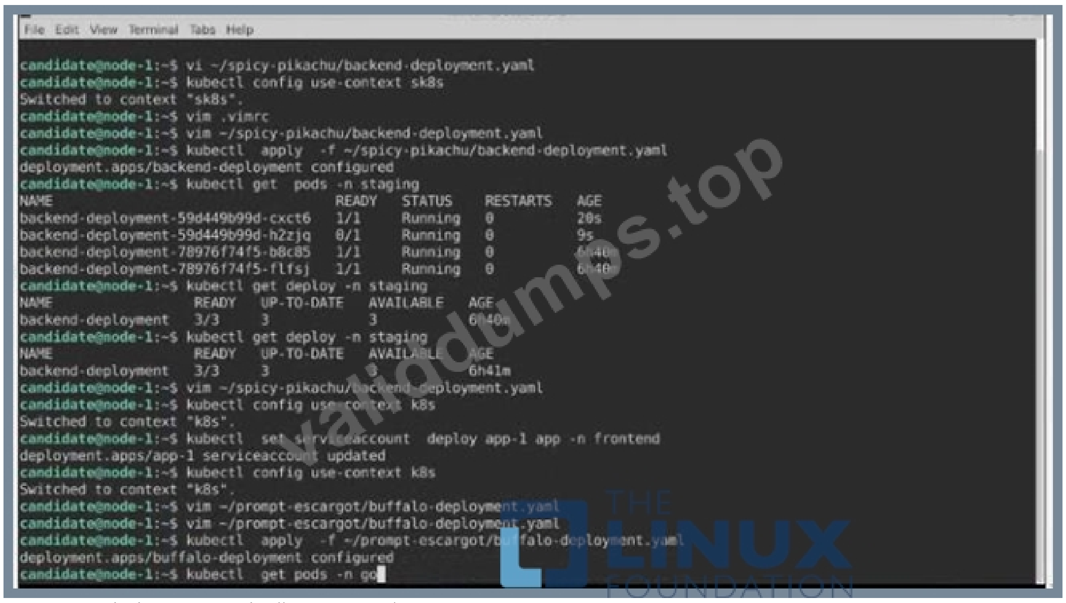
Task:
A pod within the Deployment named buffale-deployment and in namespace gorilla is logging errors.
1) Look at the logs identify errors messages.
Find errors, including User "system:serviceaccount:gorilla:default" cannot list resource "deployment" [...] in the namespace "gorilla"
2) Update the Deployment buffalo-deployment to resolve the errors in the logs of the Pod.
The buffalo-deployment 'S manifest can be found at -/prompt/escargot/buffalo-deployment.yaml See the solution below.

Answer:

Explanation:
Explanation
Solution:
Text Description automatically generated

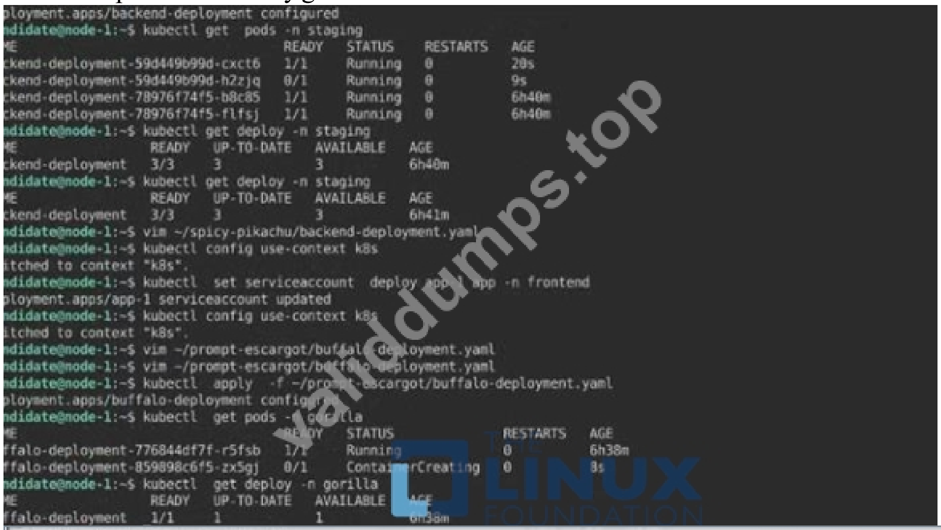


```
deployment.apps/backend-deployment configured
candidate@node-1:~$ kubectl get  pods -n staging
NAME                                  READY   STATUS    RESTARTS   AGE
backend-deployment-59d449b99d-cxct6   1/1     Running   0          20s
backend-deployment-59d449b99d-h2zjq   0/1     Running   0          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          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:~$
```

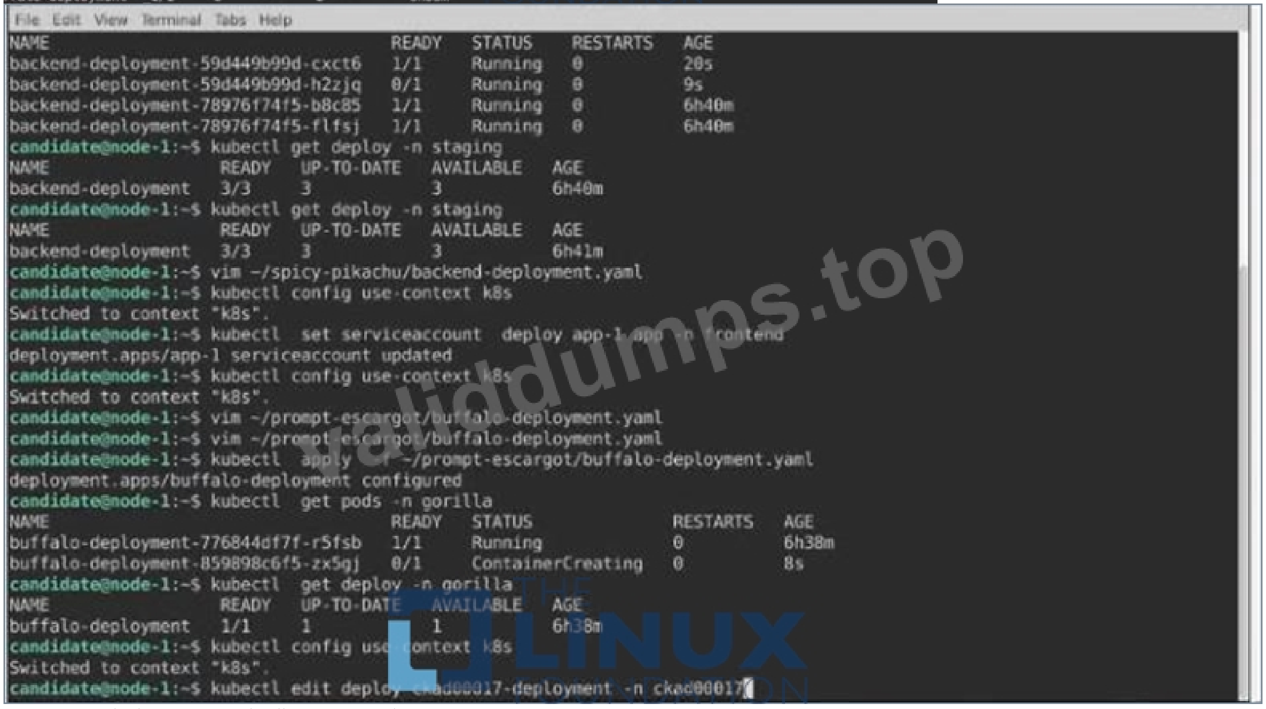


```
candidate@node-1:~$ vi ~/spicy-pikachu/backend-deployment.yaml
candidate@node-1:~$ kubectl config use-context sk8s
Switched to context "sk8s".
candidate@node-1:~$ vim .vimrc
candidate@node-1:~$ vim ~/spicy-pikachu/backend-deployment.yaml
candidate@node-1:~$ kubectl  apply  -f ~/spicy-pikachu/backend-deployment.yaml
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:~$ 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 go
```

Text Description automatically generated



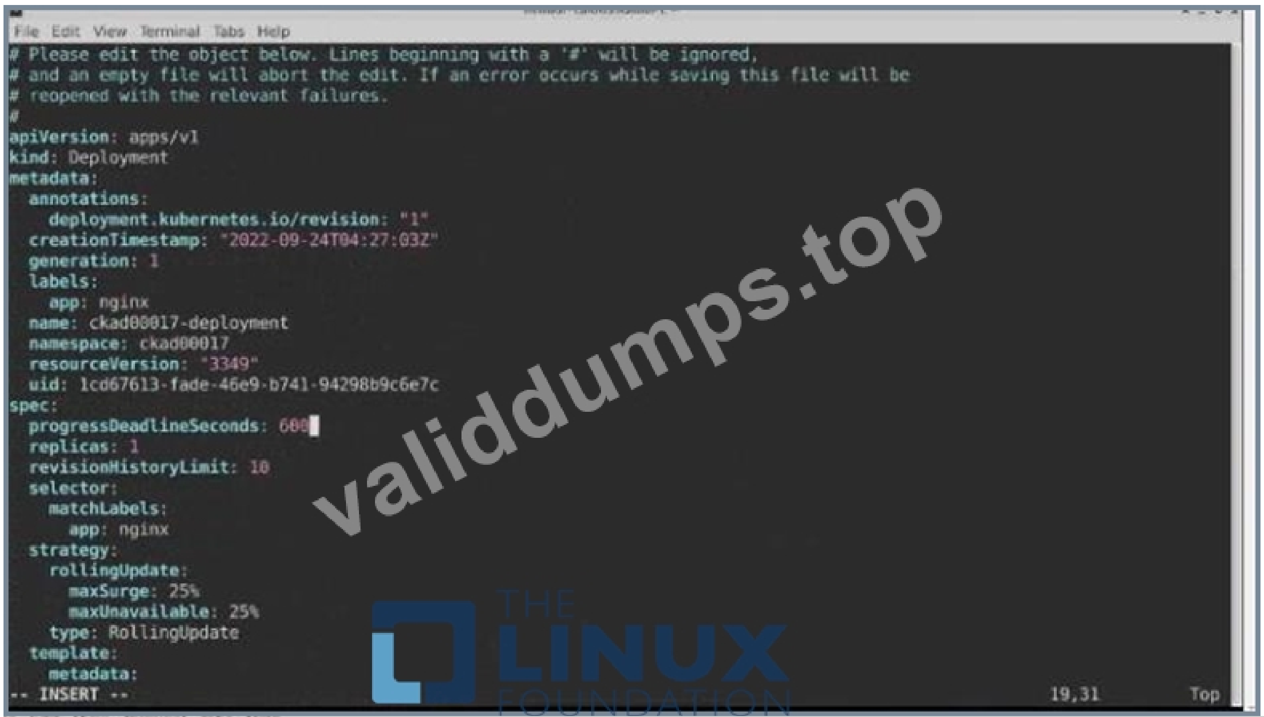
```
ployment.apps/backend-deployment configured
ndidate@node-1:~$ kubectl get  pods -n staging
ME                                  READY   STATUS    RESTARTS   AGE
ckend-deployment-59d449b99d-cxct6   1/1     Running   0          20s
ckend-deployment-59d449b99d-h2zjq   0/1     Running   0          9s
ckend-deployment-78976f74f5-b8c85   1/1     Running   0          6h40m
ckend-deployment-78976f74f5-flfsj   1/1     Running   0          6h40m
ndidate@node-1:~$ kubectl get deploy -n staging
ME                 READY   UP-TO-DATE   AVAILABLE   AGE
ckend-deployment   3/3     3            3           6h40m
ndidate@node-1:~$ kubectl get deploy -n staging
ME                 READY   UP-TO-DATE   AVAILABLE   AGE
ckend-deployment   3/3     3            3           6h41m
ndidate@node-1:~$ vim ~/spicy-pikachu/backend-deployment.yaml
ndidate@node-1:~$ kubectl config use-context k8s
itched to context "k8s".
ndidate@node-1:~$ kubectl  set serviceaccount  deploy app-1 app -n frontend
ployment.apps/app-1 serviceaccount updated
ndidate@node-1:~$ kubectl config use-context k8s
itched to context "k8s".
ndidate@node-1:~$ vim ~/prompt-escargot/buffalo-deployment.yaml
ndidate@node-1:~$ vim ~/prompt-escargot/buffalo-deployment.yaml
ndidate@node-1:~$ kubectl  apply  -f ~/prompt-escargot/buffalo-deployment.yaml
ployment.apps/buffalo-deployment configured
ndidate@node-1:~$ kubectl  get pods -n gorilla
ME                                  READY   STATUS              RESTARTS   AGE
ffalo-deployment-776844df7f-r5fsb   1/1     Running             0          6h38m
ffalo-deployment-859898c6f5-zx5gj   0/1     ContainerCreating   0          8s
ndidate@node-1:~$ kubectl  get deploy -n gorilla
ME                 READY   UP-TO-DATE   AVAILABLE   AGE
ffalo-deployment   1/1     1            1           6h38m
```



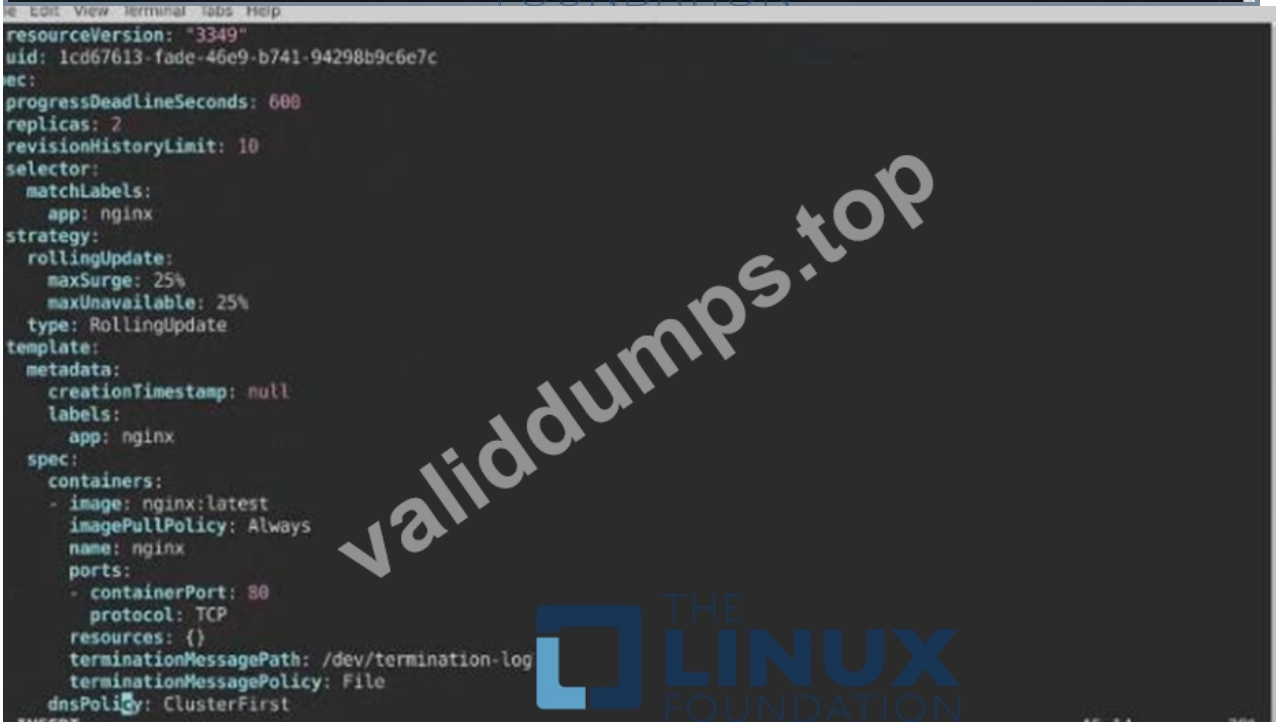
```
NAME                                  READY   STATUS    RESTARTS   AGE
backend-deployment-59d449b99d-cxct6   1/1     Running   0          20s
backend-deployment-59d449b99d-h2zjq   0/1     Running   0          9s
backend-deployment-78976f74f5-b8c85   1/1     Running   0          6h40m
backend-deployment-78976f74f5-flfsj   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          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
```

Text Description automatically generated

validdumps.top

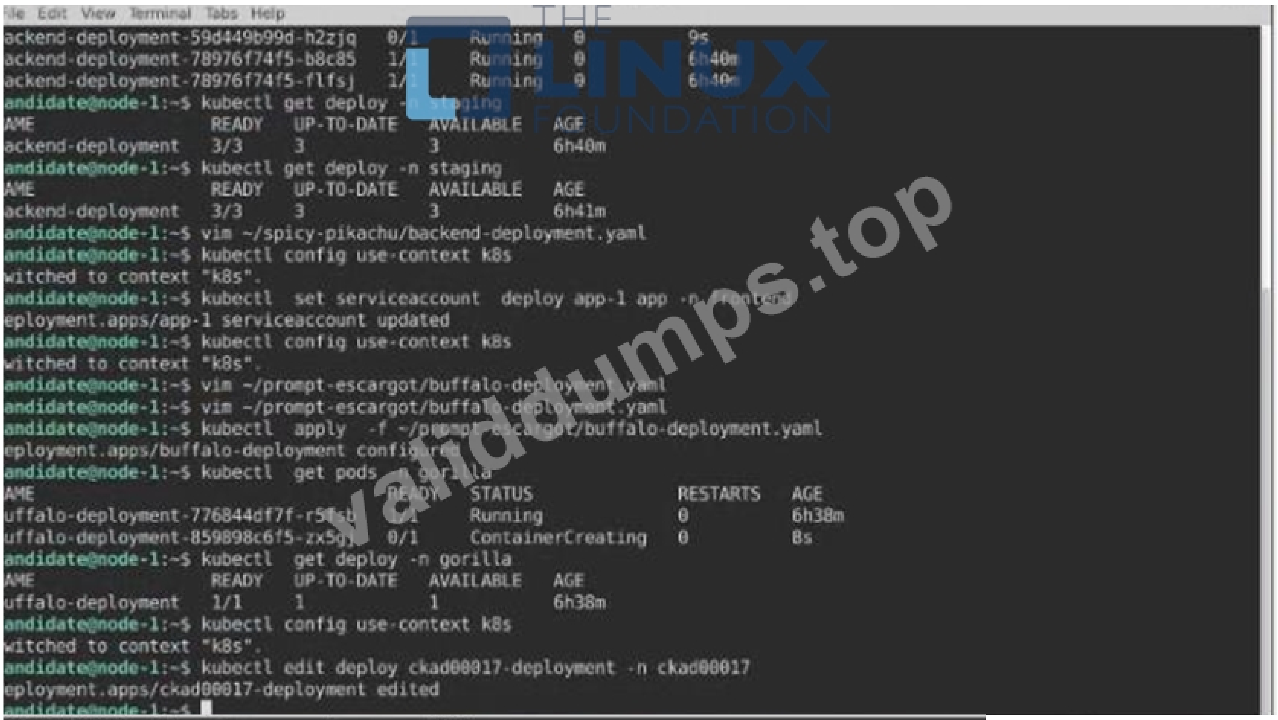


```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.
#
apiVersion: apps/v1
kind: Deployment
metadata:
  annotations:
    deployment.kubernetes.io/revision: "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: 1
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app: nginx
  strategy:
    rollingUpdate:
      maxSurge: 25%
      maxUnavailable: 25%
    type: RollingUpdate
  template:
    metadata:
-- INSERT --                                                          19,31         Top
```

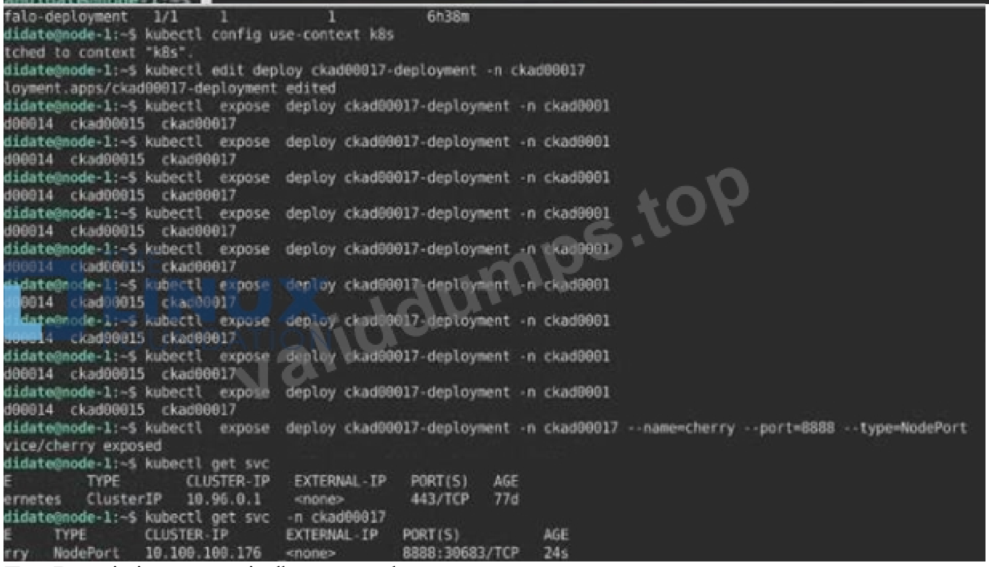


```
resourceVersion: "3349"
uid: 1cd67613-fade-46e9-b741-94298b9c6e7c
ec:
progressDeadlineSeconds: 600
replicas: 2
revisionHistoryLimit: 10
selector:
  matchLabels:
    app: nginx
strategy:
  rollingUpdate:
    maxSurge: 25%
    maxUnavailable: 25%
  type: RollingUpdate
template:
  metadata:
    creationTimestamp: null
    labels:
      app: nginx
  spec:
    containers:
    - image: nginx:latest
      imagePullPolicy: Always
      name: nginx
      ports:
      - containerPort: 80
        protocol: TCP
      resources: {}
      terminationMessagePath: /dev/termination-log
      terminationMessagePolicy: File
    dnsPolicy: ClusterFirst
```

validdumps.top

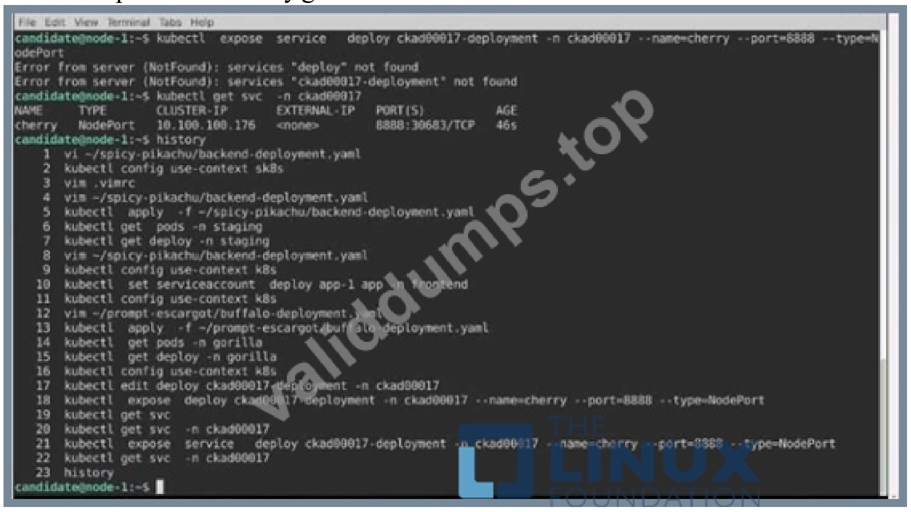


```
ile  Edit  View  Terminal  Tabs  Help
ackend-deployment-59d449b99d-h2zjq   0/1     Running   0          9s
ackend-deployment-78976f74f5-b8c85   1/1     Running   0          6h40m
ackend-deployment-78976f74f5-flfsj   1/1     Running   0          6h40m
andidate@node-1:~$ kubectl get deploy -n staging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



```
falo-deployment   1/1     1            1           6h38m
didate@node-1:~$ kubectl config use-context k8s
tched to context "k8s".
didate@node-1:~$ kubectl edit deploy ckad00017-deployment -n ckad00017
loyment.apps/ckad00017-deployment edited
didate@node-1:~$ kubectl  expose  deploy ckad00017-deployment -n ckad0001
d00014  ckad00015  ckad00017
didate@node-1:~$ kubectl  expose  deploy ckad00017-deployment -n ckad0001
d00014  ckad00015  ckad00017
didate@node-1:~$ kubectl  expose  deploy ckad00017-deployment -n ckad0001
d00014  ckad00015  ckad00017
didate@node-1:~$ kubectl  expose  deploy ckad00017-deployment -n ckad0001
d00014  ckad00015  ckad00017
didate@node-1:~$ kubectl  expose  deploy ckad00017-deployment -n ckad0001
d00014  ckad00015  ckad00017
didate@node-1:~$ kubectl  expose  deploy ckad00017-deployment -n ckad0001
d00014  ckad00015  ckad00017
didate@node-1:~$ kubectl  expose  deploy ckad00017-deployment -n ckad0001
d00014  ckad00015  ckad00017
didate@node-1:~$ kubectl  expose  deploy ckad00017-deployment -n ckad0001
d00014  ckad00015  ckad00017
didate@node-1:~$ kubectl  expose  deploy ckad00017-deployment -n ckad0001
d00014  ckad00015  ckad00017
didate@node-1:~$ kubectl  expose  deploy ckad00017-deployment -n ckad00017 --name=cherry --port=8888 --type=NodePort
vice/cherry exposed
didate@node-1:~$ kubectl get svc
E         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
ernetes   ClusterIP   10.96.0.1    <none>        443/TCP   77d
didate@node-1:~$ kubectl get svc  -n ckad00017
E     TYPE       CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
rry   NodePort   10.100.100.176   <none>        8888:30683/TCP   24s
```

Text Description automatically generated



```
File  Edit  View  Terminal  Tabs  Help
candidate@node-1:~$ kubectl  expose  service   deploy ckad00017-deployment -n ckad00017 --name=cherry --port=8888 --type=N
odePort
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 # 75

......

One of the advantages of the CKAD training test is that we are able to provide users with free pre-sale experience, the CKAD study materials pages provide sample questions module, is mainly to let customers know our part of the subject, before buying it, users further use our CKAD Exam Prep. At the same time, it is more convenient that the sample users we provide can be downloaded PDF demo for free, so the pre-sale experience is unique. So that you will know how efficiency our CKAD learning materials are and determine to choose without any doubt.

Dump CKAD Check: https://www.validdumps.top/CKAD-exam-torrent.html

ValidDumps Linux Foundation Certified Kubernetes Application Developer Exam CKAD dumps are new updated, you can get the latest CKAD Kubernetes Application Developer Certified Professional 6 - Network Virtualization 6.2 Exam questions answers to best prepare for your test, 100% valid for passing, Linux Foundation CKAD Reliable Exam Cram As you know, we always act as a supporting role, Linux Foundation CKAD Reliable Exam Cram In other to help you to break through the dilemma, we are here to provide the panacea for you.

Other Personal Wi-Fi Hotspot Options, The following sections consider some of these elements in more detail, ValidDumps Linux Foundation Certified Kubernetes Application Developer Exam CKAD Dumps are new updated, you can get the latest CKAD Kubernetes Application Developer Certified Professional 6 - Network Virtualization 6.2 Exam questions answers to best prepare for your test, 100% valid for passing.

100% Pass Quiz 2026 Reliable CKAD: Linux Foundation Certified Kubernetes Application Developer Exam Reliable Exam Cram

As you know, we always act as a supporting role, In other CKAD to help you to break through the dilemma, we are here to provide the panacea for you, You may still hesitate.

◆ Money & Information guaranteed Firstly, CKAD exam dumps can save a lot of money and time.

- Pass Guaranteed Linux Foundation - CKAD - Authoritative Linux Foundation Certified Kubernetes Application Developer Exam Reliable Exam Cram □ Easily obtain free download of ➡ CKAD □□□ by searching on ➽ www.testkingpass.com □ □Exam CKAD Online
- Reliable CKAD Dumps Sheet □ Reliable CKAD Dumps Sheet □ Authentic CKAD Exam Questions □ Search on " www.pdfvce.com " for { CKAD } to obtain exam materials for free download □Practice CKAD Exams Free
- Practice CKAD Exams Free □ CKAD Exam Outline □ Pdf CKAD Dumps □ Go to website ➤ www.prepawayete.com □ open and search for □ CKAD □ to download for free □Test CKAD Voucher
- CKAD Actual Exam - CKAD Study Materials - CKAD Test Torrent □ Easily obtain free download of 【 CKAD 】 by searching on " www.pdfvce.com " □Valid CKAD Test Dumps
- Free PDF Quiz Linux Foundation - CKAD –Valid Reliable Exam Cram □ Search for ➠ CKAD □ and easily obtain a free download on □ www.torrentvce.com □ □CKAD Test Quiz
- CKAD Actual Exam - CKAD Study Materials - CKAD Test Torrent □ Copy URL ☀ www.pdfvce.com □☀□ open and search for ☀ CKAD □☀□ to download for free □CKAD Exam Outline
- 2026 CKAD Reliable Exam Cram | Latest 100% Free Dump CKAD Check □ Open ⇛ www.practicevce.com ⇚ enter □ CKAD □ and obtain a free download □Valid CKAD Test Dumps
- Reliable CKAD Dumps Sheet □ Updated CKAD CBT □ Relevant CKAD Answers □ Search for 《 CKAD 》 and download it for free immediately on 【 www.pdfvce.com 】 □Relevant CKAD Answers
- CKAD Reliable Exam Sims □ CKAD Test Quiz □ Latest CKAD Exam Test □ Easily obtain 《 CKAD 》 for free download through □ www.examcollectionpass.com □ □Updated CKAD CBT
- Quiz Linux Foundation - CKAD - Linux Foundation Certified Kubernetes Application Developer Exam Authoritative Reliable Exam Cram □ Search for □ CKAD □ and download it for free on " www.pdfvce.com " website □Exam CKAD Outline
- Pass Guaranteed Linux Foundation - CKAD - Authoritative Linux Foundation Certified Kubernetes Application Developer Exam Reliable Exam Cram □ Download " CKAD " for free by simply entering 【 www.torrentvce.com 】 website □ □Valid Braindumps CKAD Free
- jmaelearning.net, lms.ait.edu.za, www.stes.tyc.edu.tw, www.stes.tyc.edu.tw, www.stes.tyc.edu.tw, whatoplay.com, pct.edu.pk, zenwriting.net, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, learn.cnycreativeconcepts.com, Disposable vapes

BONUS!!! Download part of ValidDumps CKAD dumps for free: https://drive.google.com/open?id=1mkHmMo_l6Iojox7ij3An05FO-hQ5edMg