

Exam Questions CKAD

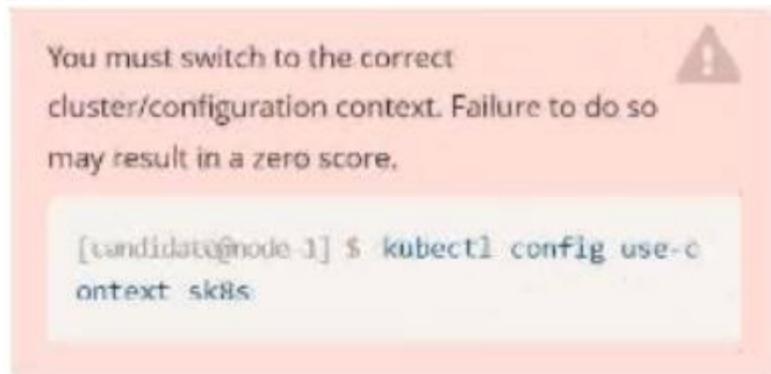
Certified Kubernetes Application Developer (CKAD) Program

<https://www.2passeasy.com/dumps/CKAD/>



NEW QUESTION 1

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

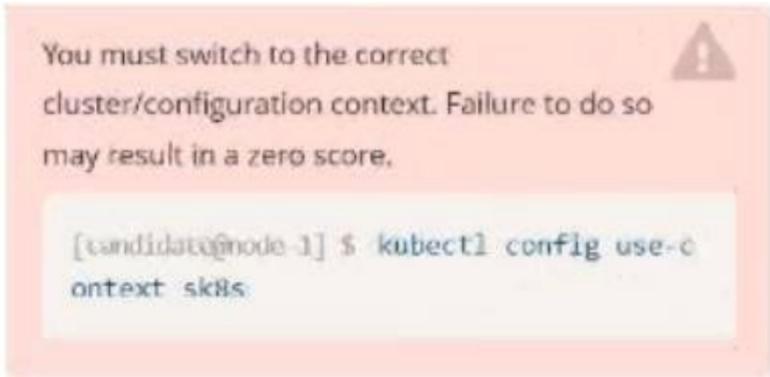
```
File Edit View Terminal Tabs Help
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: nginx-secret
  name: nginx-secret
  namespace: default
spec:
  containers:
  - image: nginx:stable
    name: nginx-secret
    env:
    - name: BEST_VARIABLE
      valueFrom:
        secretKeyRef: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

Text Description automatically generated

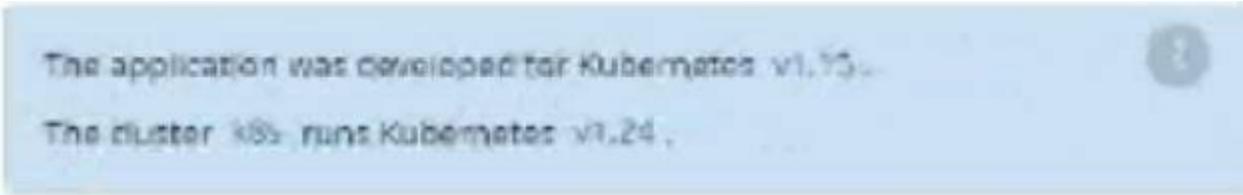
```
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ kubectl create secret generic app-secret -n default --from-literal=key3=value1
secret/app-secret created
candidate@node-1:~$ kubectl get secrets
NAME          TYPE      DATA   AGE
app-secret    Opaque   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 2

Exhibit:



Task:



- 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 ~/credible-mite/www.yaml
```

Text Description automatically generated

```
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: {}
```

Text Description automatically generated

```
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:~$
```

NEW QUESTION 3

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/pod1.yml to create a pod named app1 that runs a container named app1cont using image lfcncf/arg-output with these command line arguments: -lines 56 -F
- Create the pod with the kubectl 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 kubectl 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. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:

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



```

Readme Web Terminal
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 >

```

```

Readme Web Terminal THE LINUX FOUNDATION
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:~$

```

NEW QUESTION 4

Exhibit:



Task

A deployment is falling on the cluster due to an incorrect image being specified. Locate the deployment, and fix the problem.

- A. Mastered
- B. Not Mastered

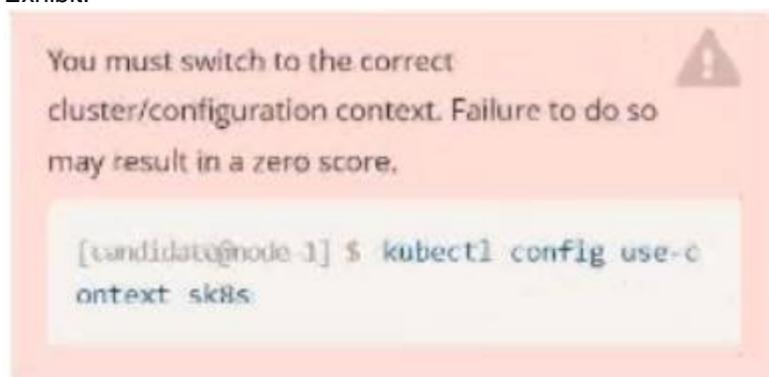
Answer: A

Explanation:

create deploy hello-deploy --image=nginx --dry-run=client -o yaml > hello-deploy.yaml
 Update deployment image to nginx:1.17.4: kubec
 nginx=nginx:1.17.4

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 --
```

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:~$
```

```

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 ckad00017
ckad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad00017
ckad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad00017
ckad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad00017
ckad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad00017
ckad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad00017
ckad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad00017
ckad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad00017
ckad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -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:~$

```

```

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:



Task:

A Dockerfile has been prepared at `~/human-stork/build/Dockerfile`

Multiple image builders and tools have been pre-installed at the base system, including: docker, skopeo, buildah, img, and podman.

Please do not push the built image to a registry, run a container, or otherwise consume it.

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:

```
candidate@node-1:~$ cd humane-stork/build/
candidate@node-1:~/humane-stork/build$ ls -l
total 16
-rw-r--r-- 1 candidate candidate 201 Sep 24 04:21 Dockerfile
-rw-r--r-- 1 candidate candidate 644 Sep 24 04:21 text1.html
-rw-r--r-- 1 candidate candidate 813 Sep 24 04:21 text2.html
-rw-r--r-- 1 candidate candidate 383 Sep 24 04:21 text3.html
candidate@node-1:~/humane-stork/build$ sudo docker build -t macaque:3.0 .
Sending build context to Docker daemon 6.144kB
Step 1/5 : FROM docker.io/lfcncf/nginx:mainline
--> ea335eea17ab
Step 2/5 : ADD text1.html /usr/share/nginx/html/
--> 8967ee9ee5d0
Step 3/5 : ADD text2.html /usr/share/nginx/html/
--> cb0554422f26
Step 4/5 : ADD text3.html /usr/share/nginx/html/
--> 62e879ab821e
Step 5/5 : COPY text2.html /usr/share/nginx/html/index.html
--> 331c8a94372c
Successfully built 331c8a94372c
Successfully tagged macaque:3.0
candidate@node-1:~/humane-stork/build$ sudo docker save macaque:3.0 > ~/humane-stork/macaque-3.0.tar
candidate@node-1:~/humane-stork/build$ cd ..
candidate@node-1:~/humane-stork$ ls -l
total 142532
drwxr-xr-x 2 candidate candidate 4096 Sep 24 04:21 build
-rw-rw-r-- 1 candidate candidate 145948672 Sep 24 11:39 macaque-3.0.tar
candidate@node-1:~/humane-stork$
```

NEW QUESTION 7

Exhibit:

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:

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

All required NetworkPolicies have already been created.

You must not create, modify or delete any NetworkPolicy while working on this task. You may only use existing NetworkPolicies.

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:

```

candidate@node-1:~$ kubectl config use-context nk8s
Switched to context "nk8s".
candidate@node-1:~$ kubectl describe netpol -n ckad00018

Name:          all-access
Namespace:     ckad00018
Created on:    2022-09-24 04:27:37 +0000 UTC
Labels:        <none>
Annotations:   <none>
Spec:
  PodSelector:  all-access=true
  Allowing ingress traffic:
    To Port: <any> (traffic allowed to all ports)
    From: <any> (traffic not restricted by source)
  Allowing egress traffic:
    To Port: <any> (traffic allowed to all ports)
    To: <any> (traffic not restricted by destination)
  Policy Types: Ingress, Egress

Name:          default-deny
Namespace:     ckad00018
Created on:    2022-09-24 04:27:37 +0000 UTC
Labels:        <none>
Annotations:   <none>
Spec:
  PodSelector:  <none> (Allowing the specific traffic to all pods in this namespace)
  Allowing ingress traffic:
    <none> (Selected pods are isolated for ingress connectivity)
  Not affecting egress traffic
  Policy Types: Ingress
candidate@node-1:~$ kubectl label pod ckad00018-newpod -n ckad00018 web-access=true
pod/ckad00018-newpod labeled
candidate@node-1:~$ kubectl label pod ckad00018-newpod -n ckad00018 db-access=true
pod/ckad00018-newpod labeled
candidate@node-1:~$

```

NEW QUESTION 8

Exhibit:

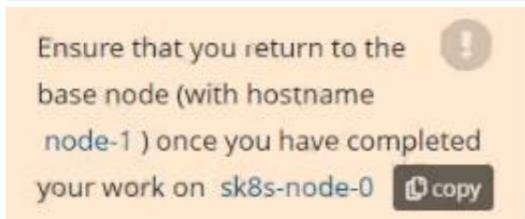
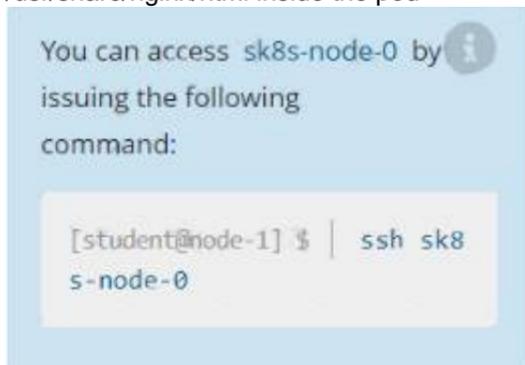


Context

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:

```

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

```

THE LINUX FOUNDATION
Readme Web Terminal
* 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:~$
    
```

```

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

```

THE LINUX FOUNDATION
Readme Web Terminal
-- 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 9

Exhibit:

```

Set configuration context:
[student@node-1] $ | kubectl config
use-context k8s

```

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:

```

Readme Web Terminal THE LINUX FOUNDATION
student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "*" * * * * --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
error: unable to match a printer suitable for the output format "yaml", 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

```

```

Readme Web Terminal THE LINUX FOUNDATION
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: '*/* * * * *'
  startingDeadlineSeconds: 22
  concurrencyPolicy: Allow

```

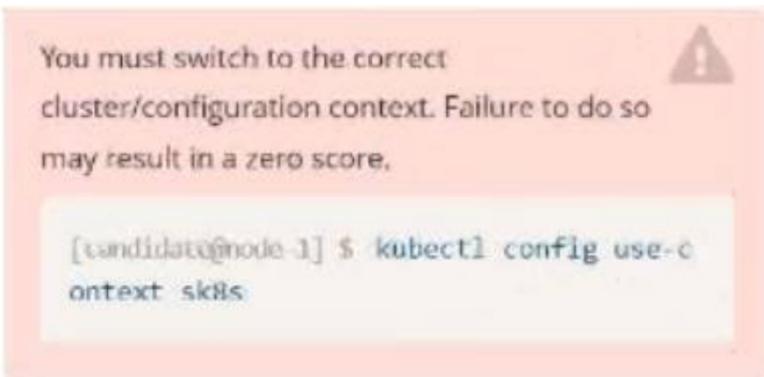
```

Readme Web Terminal THE LINUX FOUNDATION
student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "* * * * *" --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
error: unable to match a printer suitable for the output format "yaml", 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: "1Gi"

```

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:



Context

A user has reported an application is unteachable due to a failing livenessProbe . Task

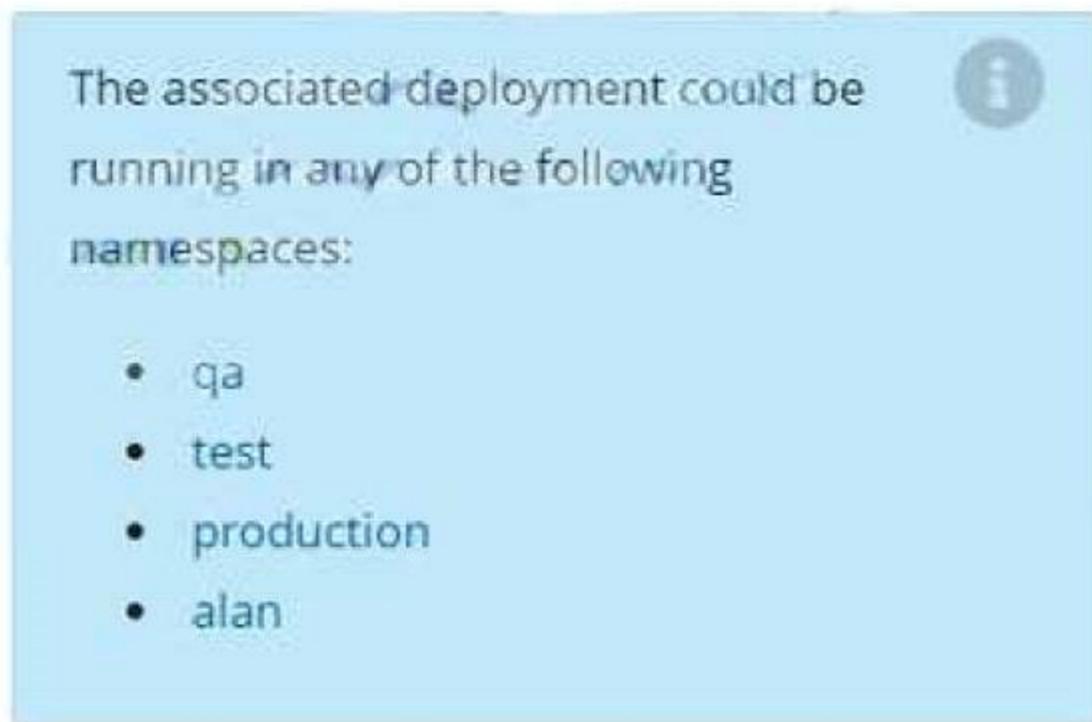
Perform the following tasks:

- Find the broken pod and store its name and namespace to /opt/KDOB00401/broken.txt in the format:



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.



- A. Mastered
- B. Not Mastered

Answer: A

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 SubjectPath 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 SubjectPath 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 13

Exhibit:



```
Set configuration context: [!]  
[student@node-1] $ | kubectl config  
use-context k8s
```

Task

Create a new deployment for running nginx with the following parameters;

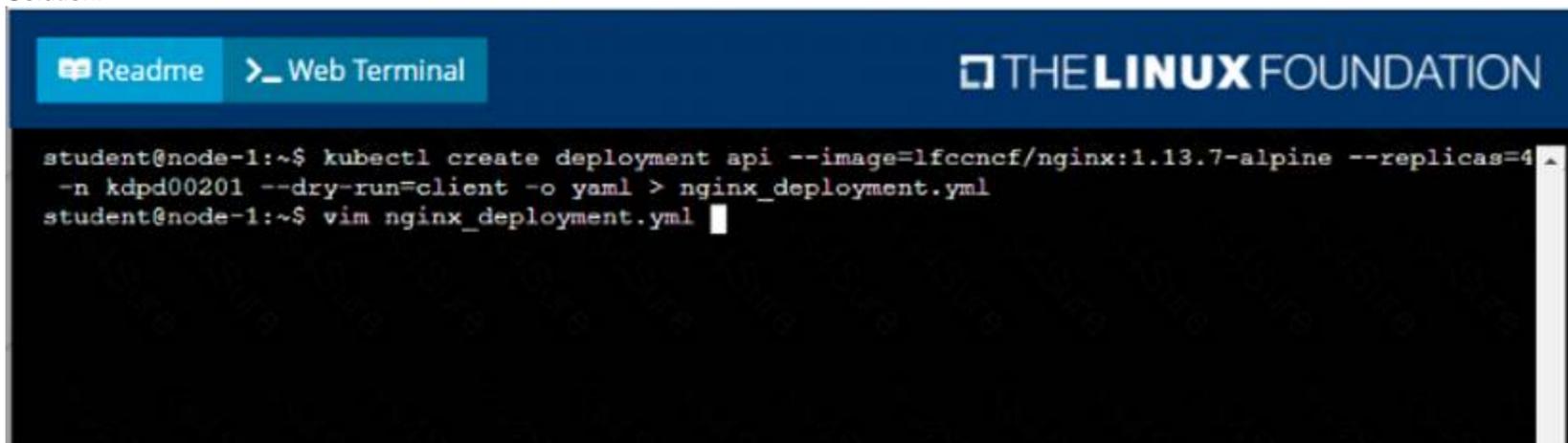
- Run the deployment in the kdpd00201 namespace. The namespace has already been created
- Name the deployment frontend and configure with 4 replicas
- Configure the pod with a container image of lfcncf/nginx:1.13.7
- Set an environment variable of NGINX_PORT=8080 and also expose that port for the container above

- A. Mastered
- B. Not Mastered

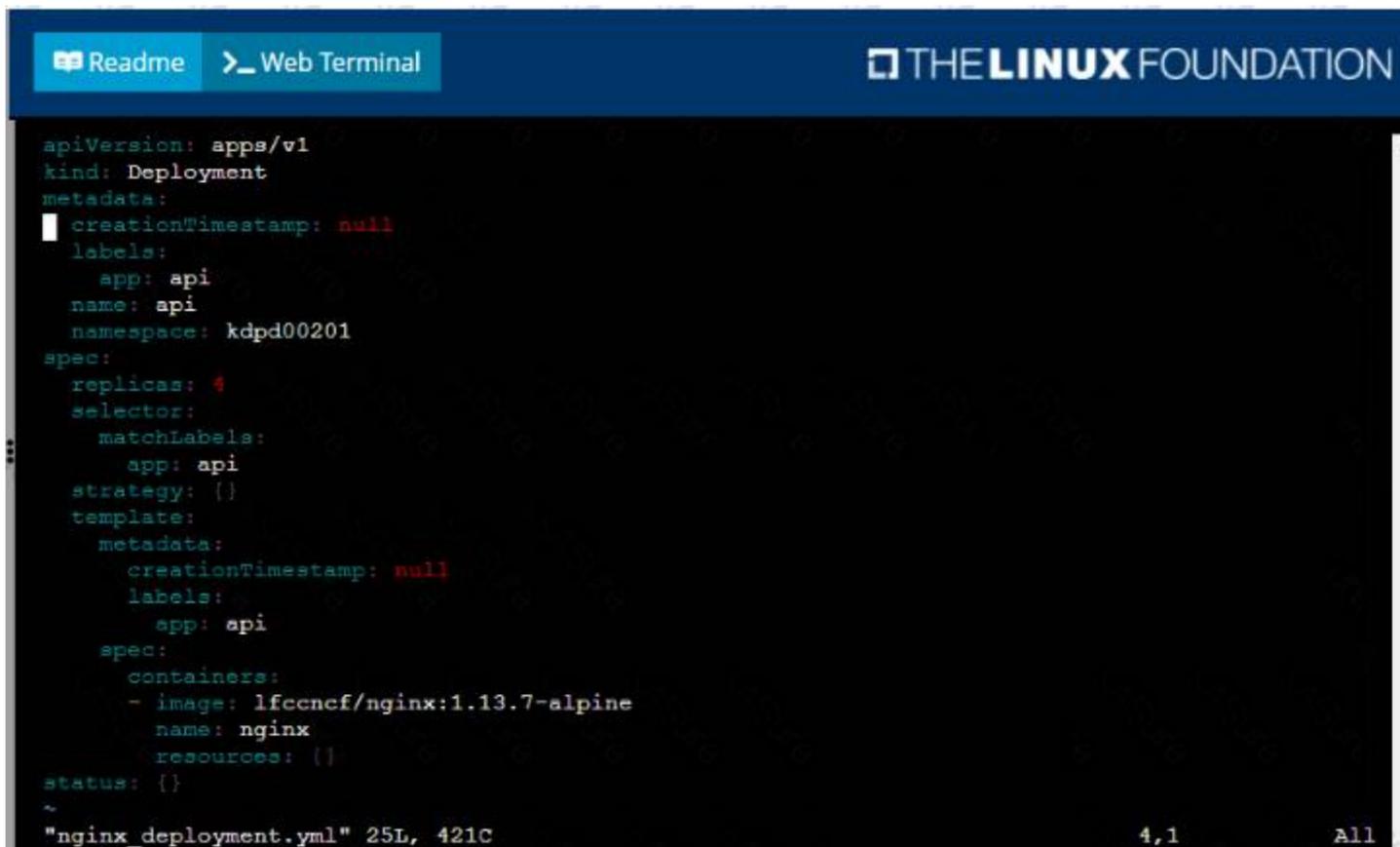
Answer: A

Explanation:

Solution:



```
student@node-1:~$ kubectl create deployment api --image=lfcncf/nginx:1.13.7-alpine --replicas=4  
-n kdpd00201 --dry-run=client -o yaml > nginx_deployment.yml  
student@node-1:~$ vim nginx_deployment.yml
```



```
apiVersion: apps/v1  
kind: Deployment  
metadata:  
  creationTimestamp: null  
  labels:  
    app: api  
  name: api  
  namespace: kdpd00201  
spec:  
  replicas: 4  
  selector:  
    matchLabels:  
      app: api  
  strategy: {}  
  template:  
    metadata:  
      creationTimestamp: null  
      labels:  
        app: api  
    spec:  
      containers:  
      - image: lfcncf/nginx:1.13.7-alpine  
        name: nginx  
        resources: {}  
status: {}  
~  
"nginx_deployment.yml" 25L, 421C 4,1 All
```

```

Readme Web Terminal THE LINUX FOUNDATION
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: api
  name: api
  namespace: kdpd00201
spec:
  replicas: 4
  selector:
    matchLabels:
      app: api
  template:
    metadata:
      labels:
        app: api
    spec:
      containers:
      - image: lfcncf/nginx:1.13.7-alpine
        name: nginx
        ports:
        - containerPort: 8080
      env:
      - name: NGINX_PORT
        value: "8080"
23,8 All

```

```

Readme Web Terminal THE LINUX FOUNDATION
student@node-1:~$ kubectl create deployment api --image=lfcncf/nginx:1.13.7-alpine --replicas=4
-n kdpd00201 --dry-run=client -o yaml > nginx_deployment.yml
student@node-1:~$ vim nginx_deployment.yml
student@node-1:~$ kubectl create nginx_deployment.yml
Error: must specify one of -f and -k

error: unknown command "nginx_deployment.yml"
See 'kubectl create -h' for help and examples
student@node-1:~$ kubectl create -f nginx_deployment.yml
error: error validating "nginx_deployment.yml": error validating data: ValidationError(Deployment.spec.template.spec): unknown field "env" in io.k8s.api.core.v1.PodSpec; if you choose to ignore these errors, turn validation off with --validate=false
student@node-1:~$ vim nginx_deployment.yml
student@node-1:~$ kubectl create -f nginx_deployment.yml
deployment.apps/api created
student@node-1:~$ kubectl get pods -n kdpd00201
NAME          READY   STATUS    RESTARTS   AGE
api-745677f7dc-7hnmv  1/1     Running   0           13s
api-745677f7dc-9q5vp  1/1     Running   0           13s
api-745677f7dc-fd4gk  1/1     Running   0           13s
api-745677f7dc-mbnpc  1/1     Running   0           13s
student@node-1:~$

```

NEW QUESTION 16

Exhibit:



Given a container that writes a log file in format A and a container that converts log files from format A to format B, create a deployment that runs both containers such that the log files from the first container are converted by the second container, emitting logs in format B.

Task:

- Create a deployment named deployment-xyz in the default namespace, that:
 - Includes a primary lfcncf/busybox:1 container, named logger-dev
 - includes a sidecar lfcncf/fluentd:v0.12 container, named adapter-zen
 - Mounts a shared volume /tmp/log on both containers, which does not persist when the pod is deleted
 - Instructs the logger-dev container to run the command

```

while true; do
  echo "i luv cncf" >> /
  tmp/log/input.log;
  sleep 10;
done

```

which should output logs to /tmp/log/input.log in plain text format, with example values:

```
i luv cncf
i luv cncf
i luv cncf
```

• The adapter-zen sidecar container should read /tmp/log/input.log and output the data to /tmp/log/output.* in Fluentd JSON format. Note that no knowledge of Fluentd is required to complete this task: all you will need to achieve this is to create the ConfigMap from the spec file provided at /opt/KDMC00102/fluentd-configmap.p.yaml , and mount that ConfigMap to /fluentd/etc in the adapter-zen sidecar container

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:

```
student@node-1:~$ kubectl create deployment deployment-xyz --image=lfcncf/busybox:1 --dry-run=c
lient -o yaml > deployment_xyz.yml
student@node-1:~$ vim deployment_xyz.yml
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  creationTimestamp: null
  labels:
    app: deployment-xyz
  name: deployment-xyz
spec:
  replicas: 1
  selector:
    matchLabels:
      app: deployment-xyz
  strategy: {}
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: deployment-xyz
    spec:
      containers:
      - image: lfcncf/busybox:1
        name: busybox
        resources: {}
status: {}
```

```
kind: Deployment
metadata:
  labels:
    app: deployment-xyz
  name: deployment-xyz
spec:
  replicas: 1
  selector:
    matchLabels:
      app: deployment-xyz
  template:
    metadata:
      labels:
        app: deployment-xyz
    spec:
      volumes:
      - name: myvoll
        emptyDir: {}
      containers:
      - image: lfcncf/busybox:1
        name: logger-dev
        volumeMounts:
        - name: myvoll
          mountPath: /tmp/log
      - image: lfcncf/fluentd:v0.12
        name: adapterzen
```

Readme Web Terminal THE LINUX FOUNDATION

```

replicas: 1
selector:
  matchLabels:
    app: deployment-xyz
template:
  metadata:
    labels:
      app: deployment-xyz
  spec:
    volumes:
      - name: myvol1
        emptyDir: {}
    containers:
      - image: lfcncf/busybox:1
        name: logger-dev
        command: ["/bin/sh", "-c", "while [ true ]; do echo 'i luv cnf' >> /tmp/log/input.log; sl
sleep 10; done"]
        volumeMounts:
          - name: myvol1
            mountPath: /tmp/log
      - image: lfcncf/fluentd:v0.12
        name: adapter-zen
        command: ["/bin/sh", "-c", "tail -f /tmp/log/input.log >> /tmp/log/output.log"]
        volumeMounts:
          - name: myvol1
            mountPath: /tmp/log
    
```

29,83 Bot

Readme Web Terminal THE LINUX FOUNDATION

```

metadata:
  labels:
    app: deployment-xyz
spec:
  volumes:
    - name: myvol1
      emptyDir: {}
    - name: myvol2
      configMap:
        name: logconf
  containers:
    - image: lfcncf/busybox:1
      name: logger-dev
      command: ["/bin/sh", "-c", "while [ true ]; do echo 'i luv cnf' >> /tmp/log/input.log; sl
sleep 10; done"]
      volumeMounts:
        - name: myvol1
          mountPath: /tmp/log
    - image: lfcncf/fluentd:v0.12
      name: adapter-zen
      command: ["/bin/sh", "-c", "tail -f /tmp/log/input.log >> /tmp/log/output.log"]
      volumeMounts:
        - name: myvol1
          mountPath: /tmp/log
        - name: myvol2
          mountPath: /fluentd/et
    
```

37,33 Bot

```

student@node-1:~$ kubectl create -f deployment_xyz.yml
deployment.apps/deployment-xyz created
student@node-1:~$ kubectl get deployment
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
deployment-xyz 0/1     1             0           5s
student@node-1:~$ kubectl get deployment
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
deployment-xyz 0/1     1             0           9s
student@node-1:~$ kubectl get deployment
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
deployment-xyz 1/1     1             1           12s
student@node-1:~$
    
```

NEW QUESTION 19

Exhibit:



Context

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

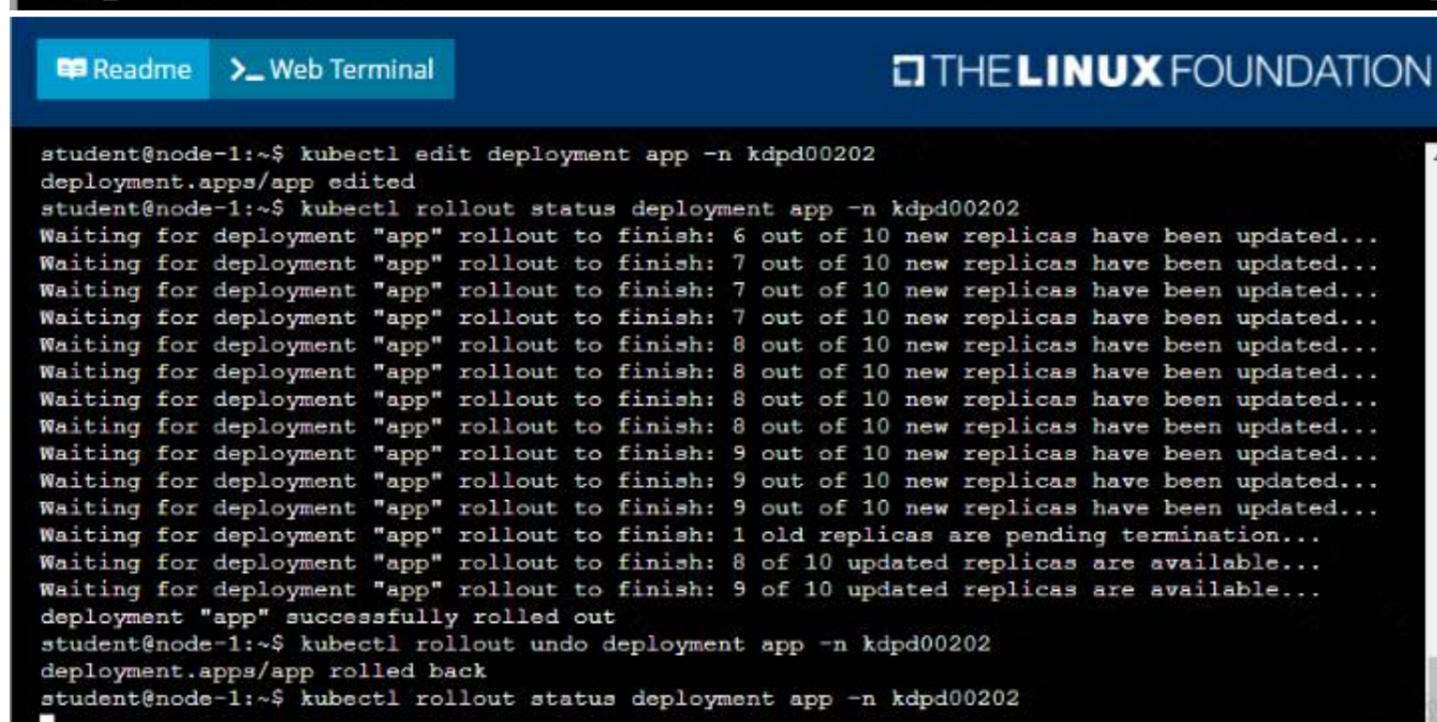
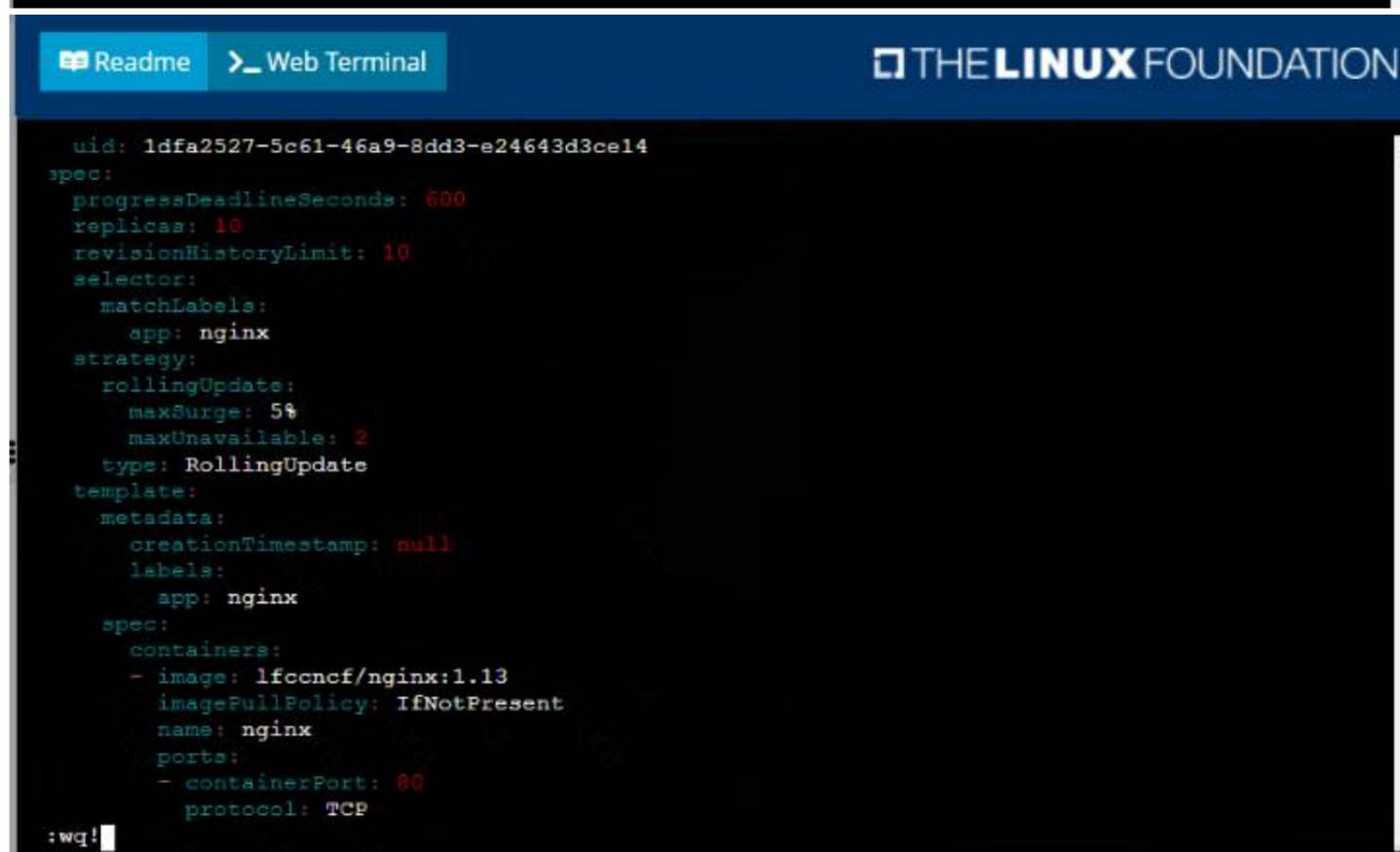
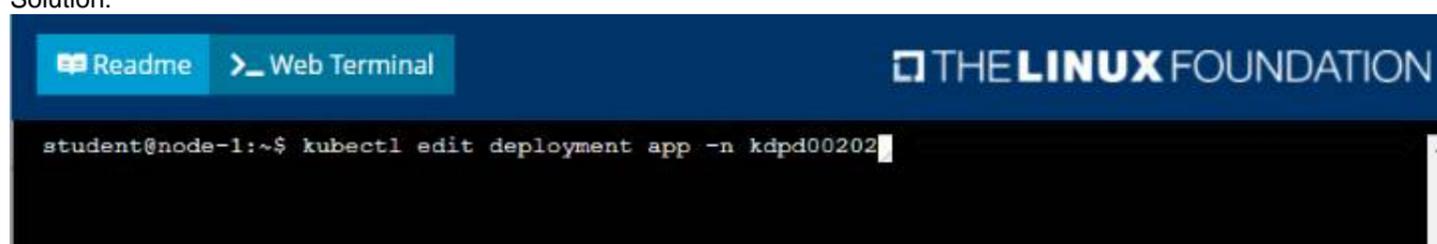
- Update the app deployment in the kdpd00202 namespace with a maxSurge of 5% and a maxUnavailable of 2%
- Perform a rolling update of the web1 deployment, changing the lfcncf/nginx image version to 1.13
- Roll back the app deployment to the previous version

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:



```
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 24

.....

THANKS FOR TRYING THE DEMO OF OUR PRODUCT

Visit Our Site to Purchase the Full Set of Actual CKAD Exam Questions With Answers.

We Also Provide Practice Exam Software That Simulates Real Exam Environment And Has Many Self-Assessment Features. Order the CKAD Product From:

<https://www.2passeasy.com/dumps/CKAD/>

Money Back Guarantee

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