

# Linux-Foundation

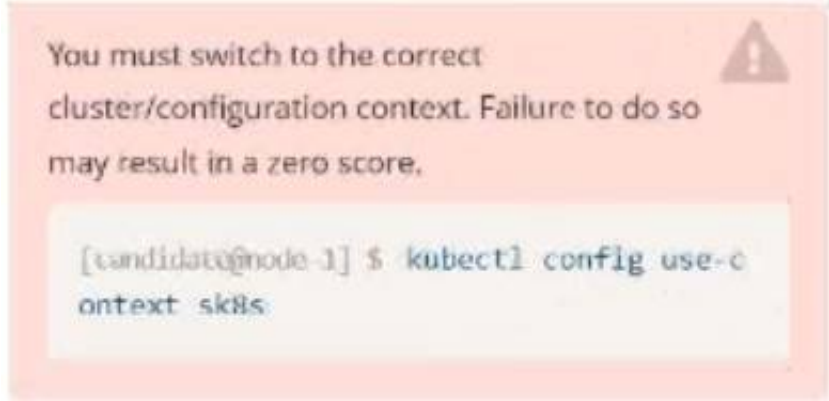
## Exam Questions CKAD

Certified Kubernetes Application Developer (CKAD) Program



NEW QUESTION 1

Exhibit:



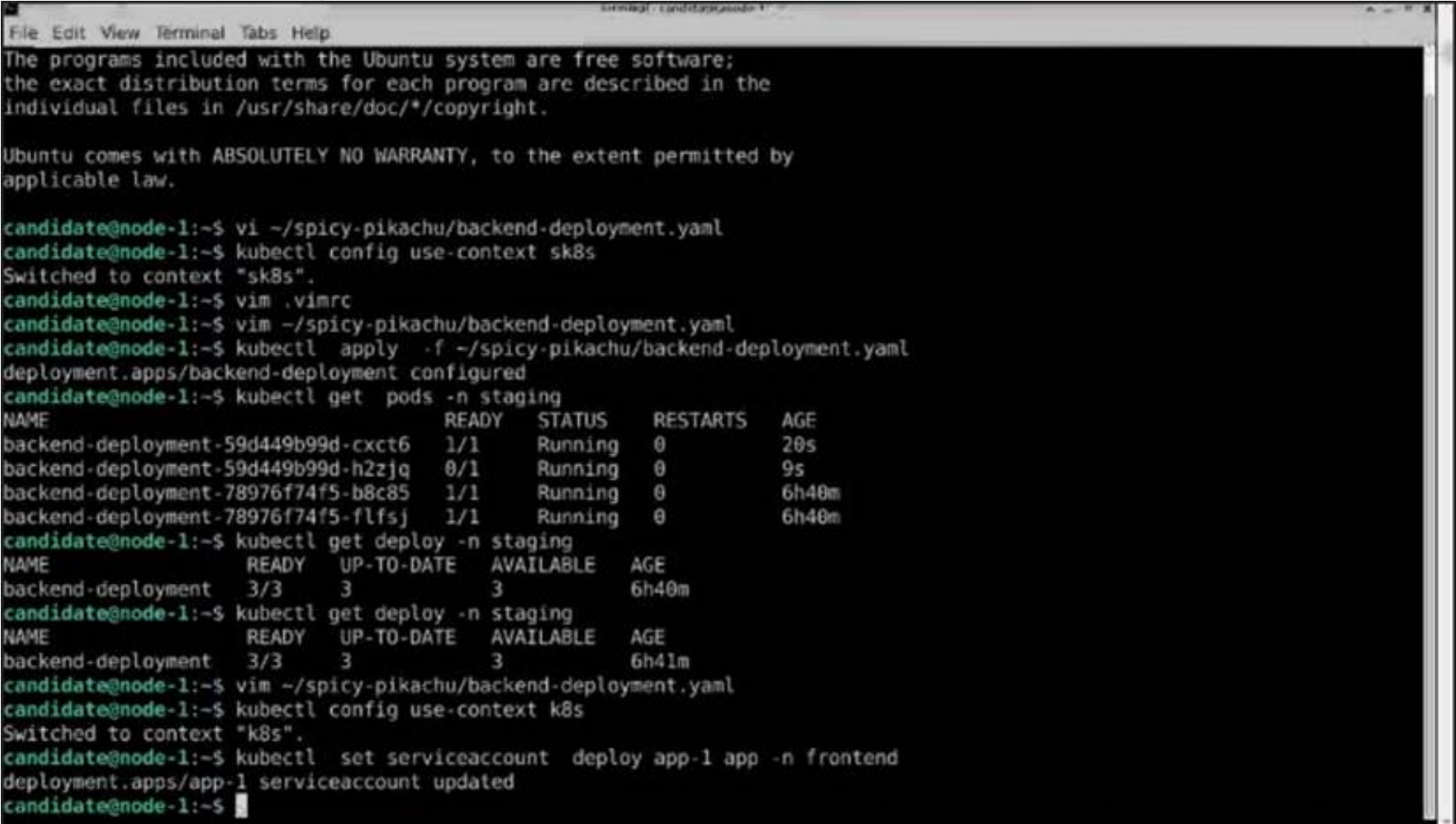
Task:  
Update the Deployment app-1 in the frontend namespace to use the existing ServiceAccount app.

- A. Mastered
- B. Not Mastered

Answer: A

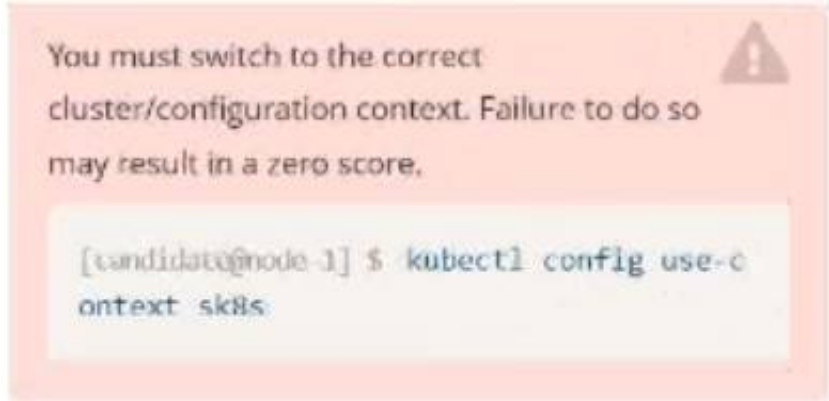
Explanation:

Solution:  
Text Description automatically generated



NEW QUESTION 2

Exhibit:



Task:  
Key3: value1  
Add an environment variable named BEST\_VARIABLE consuming the value of the secret key3.

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:

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

Text Description automatically generated

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

Text Description automatically generated

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

NEW QUESTION 3

Exhibit:



Context

A web application requires a specific version of redis to be used as a cache. Task  
 Create a pod with the following characteristics, and leave it running when complete:

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

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:

Readme

> Web Terminal

THE **LINUX** FOUNDATION

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

## NEW QUESTION 4

## 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 lfccncf/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 `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 appl --image=lfcncf/arg-output --dry-run=client -o yaml > /opt/KDPD00101/pod1.yml
student@node-1:~$ vim /opt/KDPD00101/pod1.yml
```

Readme

> Web Terminal

THE LINUX FOUNDATION

```
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
```



11,30	All
-------	-----

[Readme](#) [Web Terminal](#)[Readme](#) [Web Terminal](#)

Readme
Web Terminal



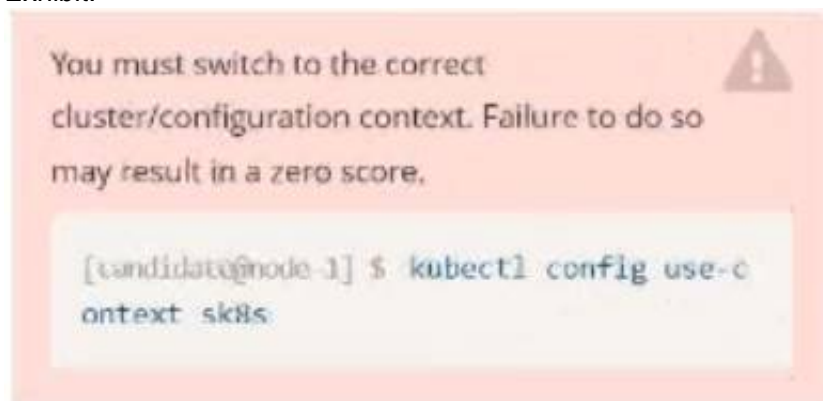
```

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 5

Exhibit:



Task:

- 1- Update the Propertunel scaling configuration of the Deployment web1 in the ckad00015 namespace setting maxSurge to 2 and maxUnavailable to 59
- 2- Update the web1 Deployment to use version tag 1.13.7 for the lfconf/nginx container image.
- 3- Perform a rollback of the web1 Deployment to its previous version

- 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 edit deploy web1 -n ckad00015

```

Text Description automatically generated



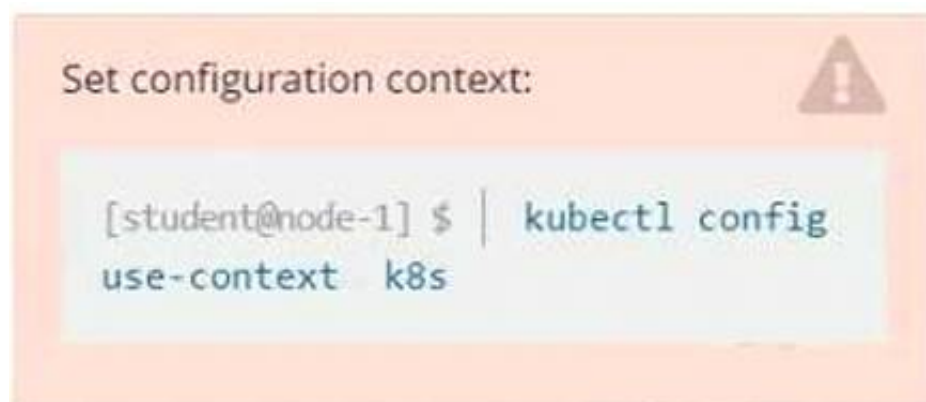
```
File Edit View Terminal Tabs Help
app: nginx
strategy:
  rollingUpdate:
    maxSurge: 2%
    maxUnavailable: 5%
  type: RollingUpdate
template:
  metadata:
    creationTimestamp: null
    labels:
      app: nginx
  spec:
    containers:
      - image: lfccncf/nginx:1.13.7
        imagePullPolicy: IfNotPresent
        name: nginx
        ports:
          - containerPort: 80
            protocol: TCP
        resources: {}
        terminationMessagePath: /dev/termination-log
        terminationMessagePolicy: File
    dnsPolicy: ClusterFirst
    restartPolicy: Always
    schedulerName: default-scheduler
    securityContext: {}
    terminationGracePeriodSeconds: 30
status:
  availableReplicas: 2
  conditions:
    - lastTransitionTime: "2022-09-24T04:26:41Z"
```

```
File Edit View Terminal Tabs Help
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:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ kubectl edit deploy web1 -n ckad00015
deployment.apps/web1 edited
candidate@node-1:~$ kubectl rollout status deploy web1 -n ckad00015
deployment "web1" successfully rolled out
candidate@node-1:~$ kubectl rollout undo deploy web1 -n ckad00015
deployment.apps/web1 rolled back
candidate@node-1:~$ kubectl rollout history deploy web1 -n ckad00015
deployment.apps/web1
REVISION  CHANGE-CAUSE
2         <none>
3         <none>

candidate@node-1:~$ kubectl get rs -n ckad00015
NAME                DESIRED   CURRENT   READY   AGE
web1-56f98bcb79      0         0         0       63s
web1-85775b6b79      2         2         2       6h53m
candidate@node-1:~$
```

## NEW QUESTION 6

Exhibit:



Context

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

Please complete the following;

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

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:

```
student@node-1:~$ kubectl create -f /opt/KDOB00201/counter.yaml
pod/counter created
student@node-1:~$ kubectl get pods
NAME                READY   STATUS    RESTARTS   AGE
counter             1/1     Running   0           10s
liveness-http       1/1     Running   0           6h45m
nginx-101           1/1     Running   0           6h46m
nginx-configmap     1/1     Running   0           107s
nginx-secret        1/1     Running   0           7m21s
poller              1/1     Running   0           6h46m
student@node-1:~$ kubectl logs counter
1: 2b305101817ae25ca60ae46510fb6d11
2: 3648cf2eae95ab680dba8f195f891af4
3: 65c8bbd4dbf70bf81f2a0984a3a44ede
4: 40d3a9c8e46f5533bb4828fbe5c8d038
5: 390442d2530a90c3602901e3fe999ac8
6: b71d95187417e139effb33af77681040
7: 66a8e55a6491e756d2d0549ad6ab90a7
8: ff2b3d583b64125d2f9129c443bb37ff
9: b6c6a12b6e77944ed8baaaf6c242dae4
10: bfcc9a894a0604fc4b814b37d0a200a4
student@node-1:~$ kubectl logs counter > /opt/KDOB00201/log_output.txt
student@node-1:~$
```

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

Readme Web Terminal

THE LINUX FOUNDATION

```
student@node-1:~$ kubectl logs counter > /opt/KDOB00201/log_output.txt
student@node-1:~$ cat /opt/KDOB00201/log_output.txt
1: 2b305101817ae25ca60ae46510fb6d11
2: 3648cf2eae95ab680dba8f195f891af4
3: 65c8bbd4dbf70bf81f2a0984a3a44ede
4: 40d3a9c8e46f5533bb4828fbe5c8d038
5: 390442d2530a90c3602901e3fe999ac8
6: b71d95187417e139effb33af77681040
7: 66a8e55a6491e756d2d0549ad6ab90a7
8: ff2b3d583b64125d2f9129c443bb37ff
9: b6c6a12b6e77944ed8baaaf6c242dae4
10: bfcc9a894a0604fc4b814b37d0a200a4
11: 5493cd16a1790a5fb9512b0c9d4c5dd1
12: 03f169e93e6143438e6dfe4ecb3cc9ed
13: 764b37fe611373c42d0b47154041f6eb
14: 1a56fbc1896b0ee6394136166281839e
15: ecc492eb17715de090c47345a98d98d3
16: 7974a6bec0fb44b6b8bbfc71aa3fbe74
17: 9ae01bef01748b12cc9f97a5f9f72cd6
18: 23fb22ee34d4272e4c9e005f1774515f
19: ec7e1a5d314da9a0ad45d53be5a7acae
20: 0bccdd8ee02cd42029e8162cd1c1197c
21: d6851ea43546216b95bcb81ced997102
22: 7ed9a38ea8bf0d86206569481442af44
23: 29b8416ddc63dbfcb987ab3c8198e9fe
24: 1f2062001df51a108ab25010f506716f
student@node-1:~$
```

## NEW QUESTION 7

Exhibit:



Context

It is always useful to look at the resources your applications are consuming in a cluster. Task

- From the pods running in namespace cpu-stress , write the name only of the pod that is consuming the most CPU to file /opt/KDOBG030/pod.txt, which has already been created.

A. Mastered

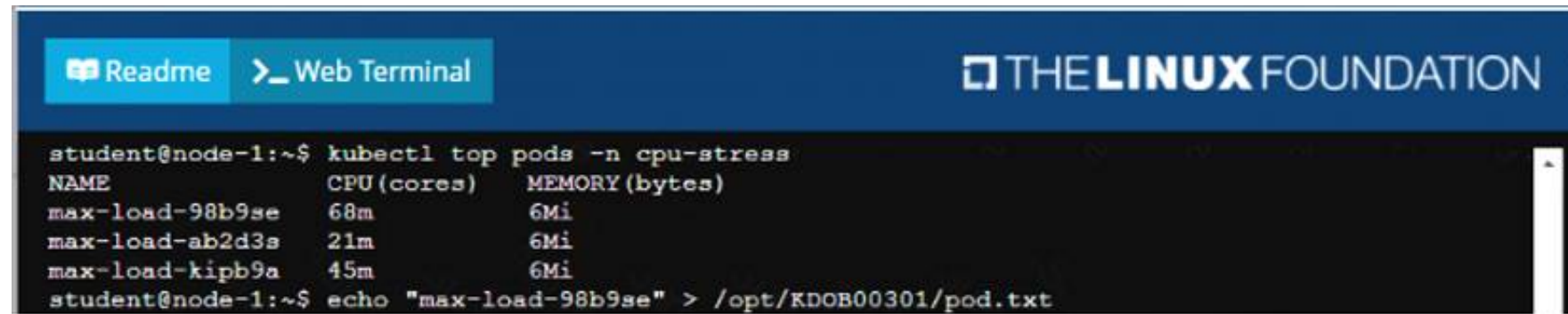


B. Not Mastered

Answer: A

Explanation:

Solution:



```
student@node-1:~$ kubectl top pods -n cpu-stress
NAME                CPU(cores)   MEMORY(bytes)
max-load-98b9se     68m          6Mi
max-load-ab2d3s     21m          6Mi
max-load-kipb9a     45m          6Mi
student@node-1:~$ echo "max-load-98b9se" > /opt/KDOB00301/pod.txt
```

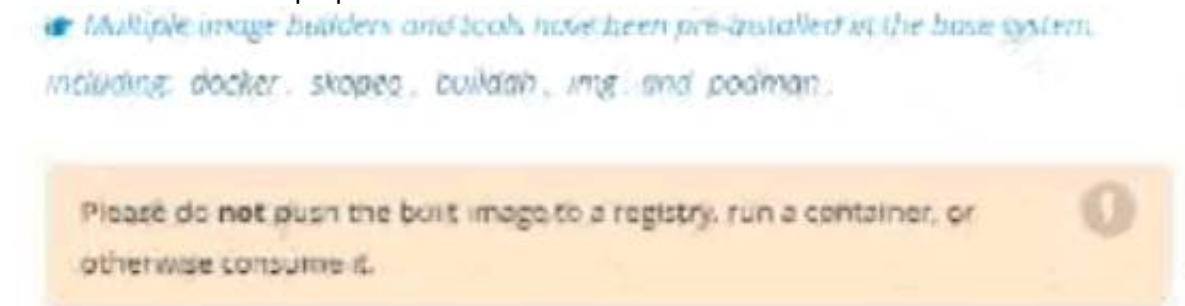
## NEW QUESTION 8

Exhibit:



Task:

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



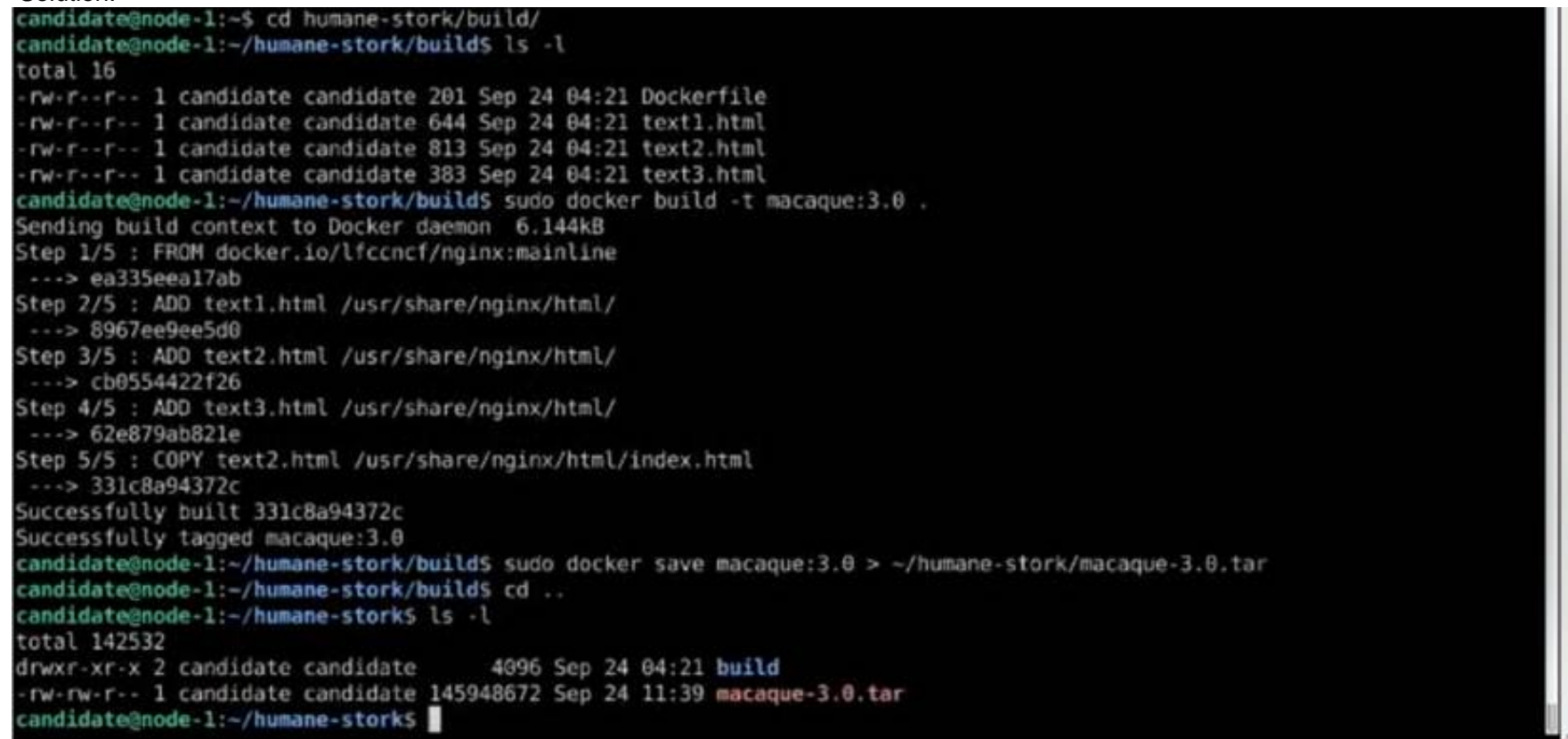
A. Mastered

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/lfccncf/nginx:mainline
--> ea335eeal7ab
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 9

Exhibit:



Context

You are tasked to create a ConfigMap and consume the ConfigMap in a pod using a volume mount. Task

Please complete the following:

- Create a ConfigMap named another-config containing the key/value pair: key4/value3
- start a pod named nginx-configmap containing a single container using the nginx image, and mount the key you just created into the pod under directory /also/a/path

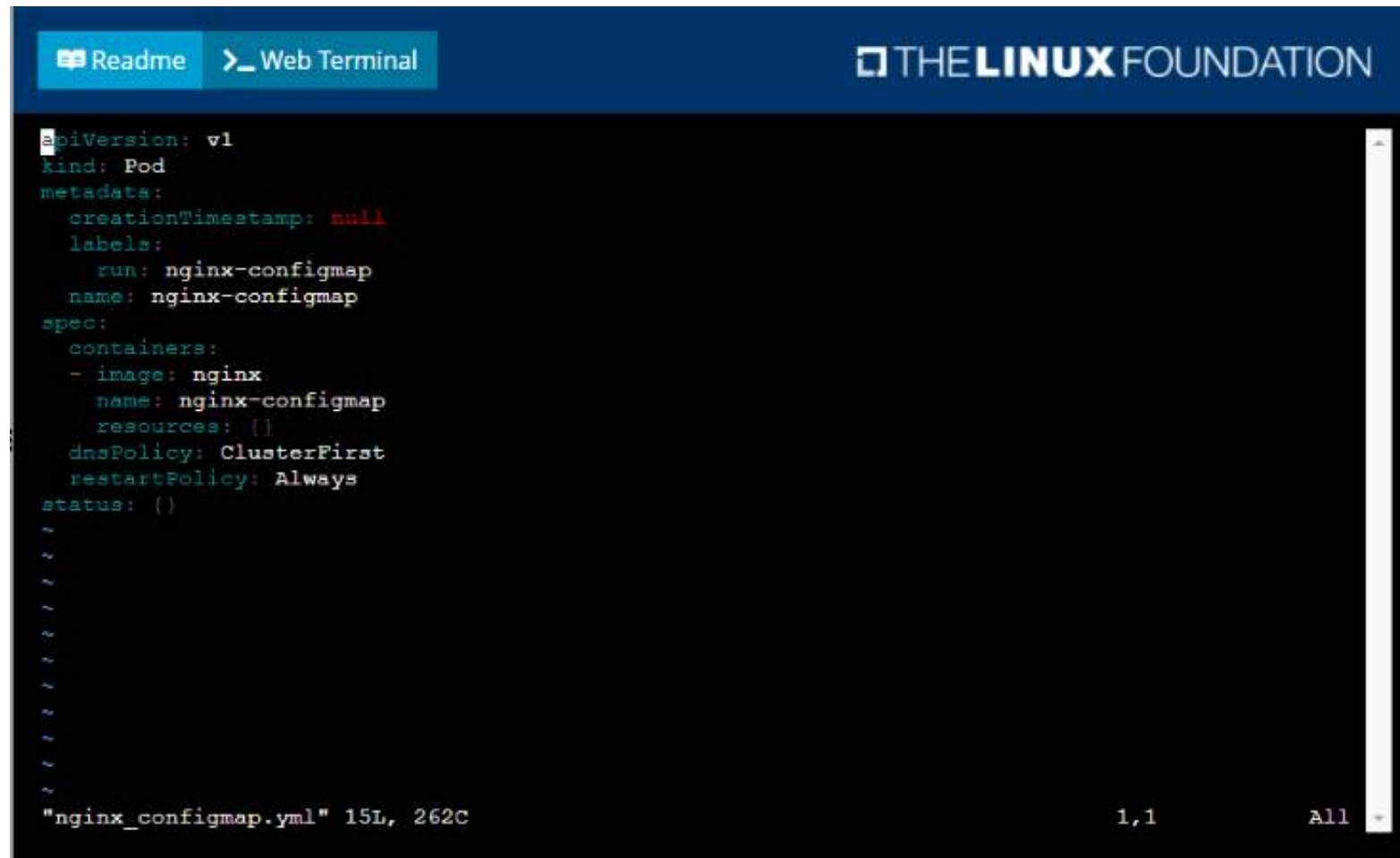
- A. Mastered
- B. Not Mastered

**Answer: A**

**Explanation:**

Solution:

```
student@node-1:~$ kubectl create configmap another-config --from-literal=key4=value3
configmap/another-config created
student@node-1:~$ kubectl get configmap
NAME          DATA   AGE
another-config 1       5s
student@node-1:~$ kubectl run nginx-configmap --image=nginx --dry-run=client -o yaml > nginx_configmap.yml
student@node-1:~$ vim nginx_configmap.yml ^C
student@node-1:~$ mv nginx_configmap.yml nginx_configmap.yml
student@node-1:~$ vim nginx_co
```





Readme
Web Terminal
THE LINUX FOUNDATION

```

apiVersion: v1
kind: Pod
metadata:
  labels:
    run: nginx-configmap
  name: nginx-configmap
spec:
  containers:
  - image: nginx
    name: nginx-configmap
    volumeMounts:
    - name: myvol
      mountPath: /also/a/path
  volumes:
  - name: myvol
    configMap:
      name: another-config
~
~
~
~
~
~
~
~
~
~
13,6 All

```

```

student@node-1:~$ kubectl create configmap another-config --from-literal=key4=value3
configmap/another-config created
student@node-1:~$ kubectl get configmap
NAME          DATA      AGE
another-config 1          5s
student@node-1:~$ kubectl run nginx-configmap --image=nginx --dry-run=client -o yaml > nginx_conf
igmap.yaml
student@node-1:~$ vim nginx_configmap.yaml ^C
student@node-1:~$ mv nginx_configmap.yaml nginx_configmap.yaml
student@node-1:~$ vim nginx_configmap.yaml
student@node-1:~$

```

```

student@node-1:~$ kubectl run nginx-configmap --image=nginx --dry-run=client -o yaml > nginx_conf
igmap.yaml
student@node-1:~$ vim nginx_configmap.yaml ^C
student@node-1:~$ mv nginx_configmap.yaml nginx_configmap.yaml
student@node-1:~$ vim nginx_configmap.yaml
student@node-1:~$ kubectl create f nginx_configmap.yaml
Error: must specify one of -f and -k

error: unknown command "f nginx_configmap.yaml"
See 'kubectl create -h' for help and examples
student@node-1:~$ kubectl create -f nginx_configmap.yaml
error: error validating "nginx_configmap.yaml": error validating data: ValidationError(Pod.spec.c
ontainers[1]): unknown field "mountPath" in io.k8s.api.core.v1.Container; if you choose to ignor
e these errors, turn validation off with --validate=false
student@node-1:~$ vim nginx_configmap.yaml

```

Readme
Web Terminal
THE LINUX FOUNDATION

```

student@node-1:~$ kubectl create f nginx_configmap.yaml
Error: must specify one of -f and -k

error: unknown command "f nginx_configmap.yaml"
See 'kubectl create -h' for help and examples
student@node-1:~$ kubectl create -f nginx_configmap.yaml
error: error validating "nginx_configmap.yaml": error validating data: ValidationError(Pod.spec.c
ontainers[1]): unknown field "mountPath" in io.k8s.api.core.v1.Container; if you choose to ignor
e these errors, turn validation off with --validate=false
student@node-1:~$ vim nginx_configmap.yaml
student@node-1:~$ kubectl create -f nginx_configmap.yaml
pod/nginx-configmap created
student@node-1:~$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
liveness-http 1/1     Running   0           6h44m
nginx-101     1/1     Running   0           6h45m
nginx-configmap 0/1     ContainerCreating 0           5s
nginx-secret  1/1     Running   0           5m39s
poller        1/1     Running   0           6h44m
student@node-1:~$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
liveness-http 1/1     Running   0           6h44m
nginx-101     1/1     Running   0           6h45m
nginx-configmap 1/1     Running   0           8s
nginx-secret  1/1     Running   0           5m42s
poller        1/1     Running   0           6h45m
student@node-1:~$ l

```



## NEW QUESTION 10

Exhibit:

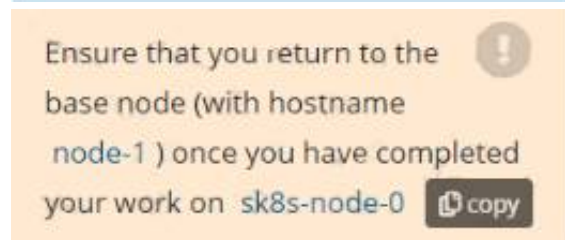
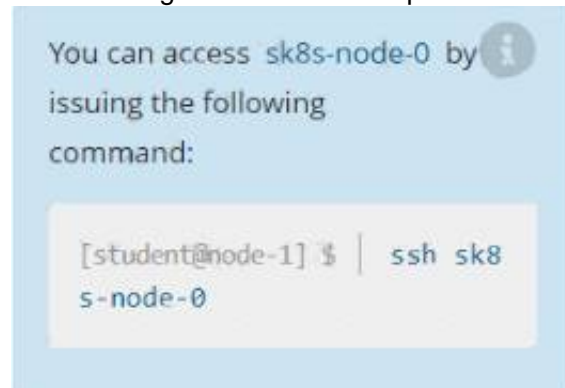


Context

A project that you are working on has a requirement for persistent data to be available. Task

To facilitate this, perform the following tasks:

- Create a file on node sk8s-node-0 at /opt/KDSP00101/data/index.html with the content Acct=Finance
- Create a PersistentVolume named task-pv-volume using hostPath and allocate 1Gi to it, specifying that the volume is at /opt/KDSP00101/data on the cluster's node. The configuration should specify the access mode of ReadWriteOnce . It should define the StorageClass name exam for the PersistentVolume , which will be used to bind PersistentVolumeClaim requests to this PersistentVolume.
- Create a PersistentVolumeClaim named task-pv-claim that requests a volume of at least 100Mi and specifies an access mode of ReadWriteOnce
- Create a pod that uses the PersistentVolumeClaim as a volume with a label app: my-storage-app mounting the resulting volume to a mountPath /usr/share/nginx/html inside the pod

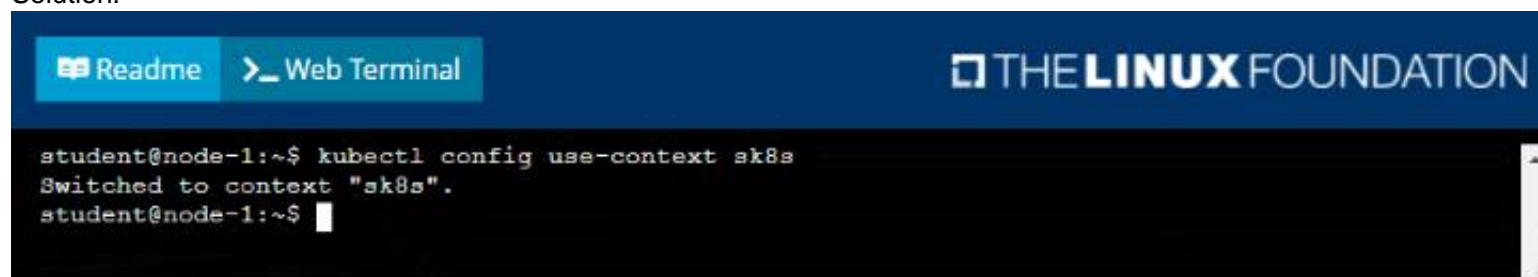


- A. Mastered
- B. Not Mastered

**Answer: A**

**Explanation:**

Solution:



Readme
Web Terminal
THE **LINUX** FOUNDATION

```

* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:        https://ubuntu.com/advantage

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

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

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

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

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

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

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

student@sk8s-node-0:~$

```

Readme
Web Terminal
THE **LINUX** FOUNDATION

```

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

```

Readme
Web Terminal
THE **LINUX** FOUNDATION

```

-- INSERT --
0,1 All

```

Readme
Web Terminal
THE **LINUX** FOUNDATION

```

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

```

ReadmeWeb 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              storageclass  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
```

ReadmeWeb 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
~
~
~
~
~
~
~
~
~
~
17,32  All
```

```
student@sk8s-node-0:~$ kubectl create -f pod.yml
pod/mypod created
student@sk8s-node-0:~$ kubectl get
```

ReadmeWeb 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 10

Exhibit:





#### Context

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

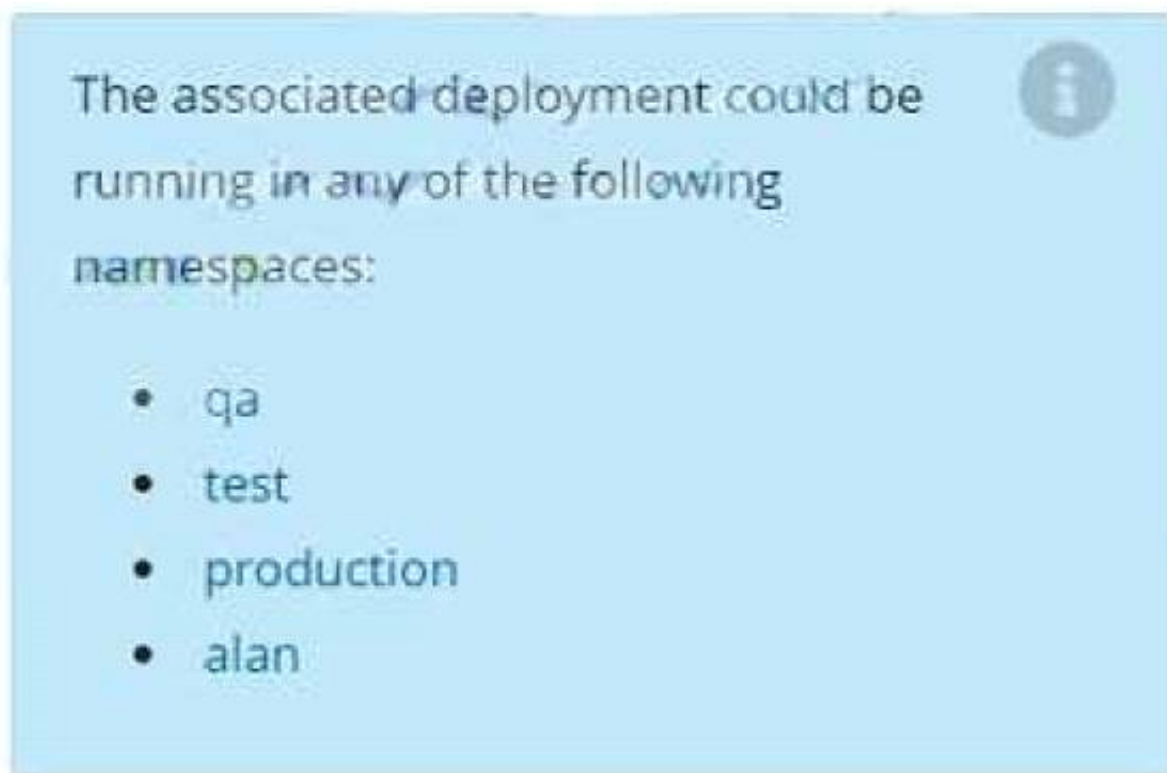
Perform the following tasks:

- 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 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 11

Exhibit:



Task

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

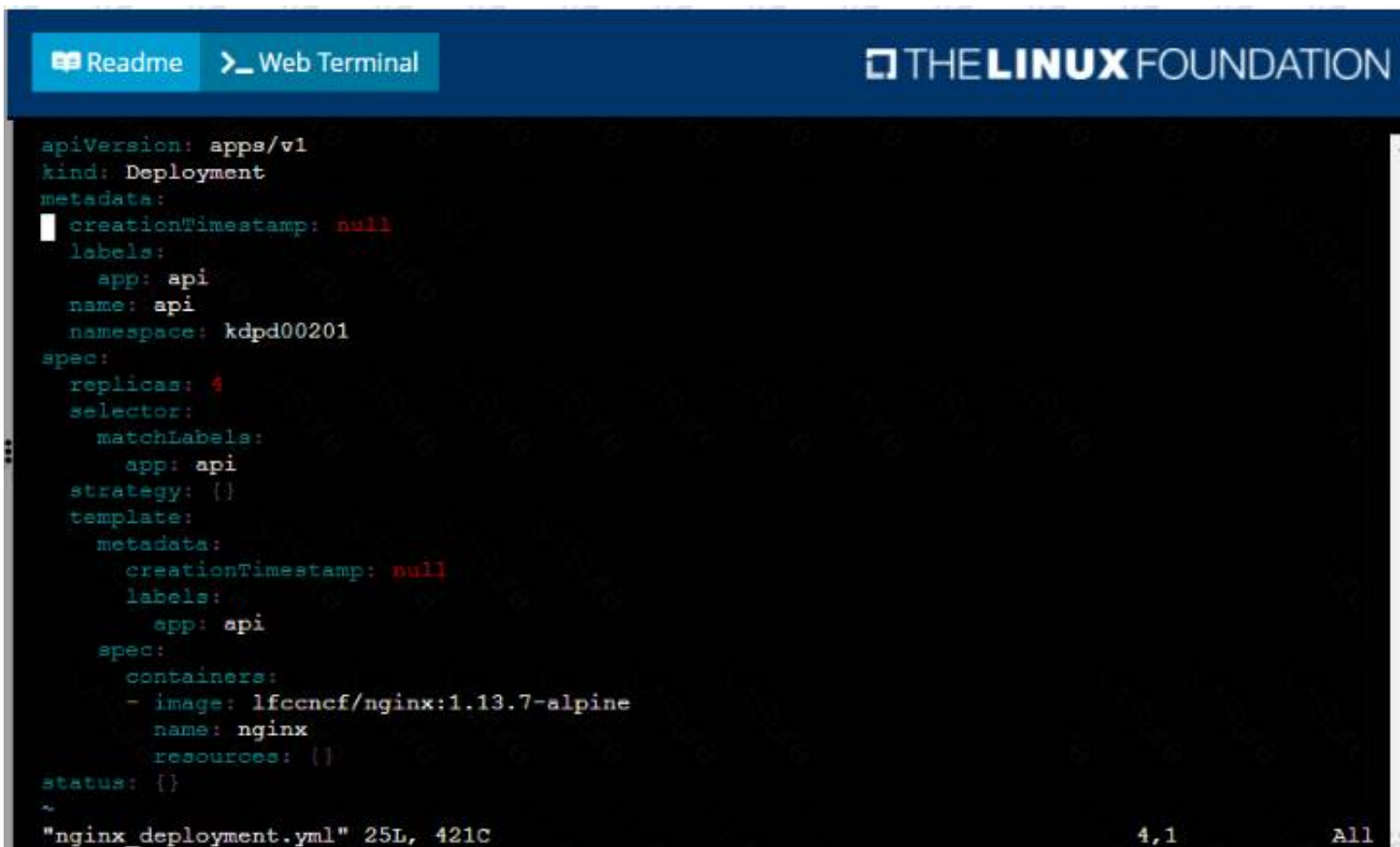
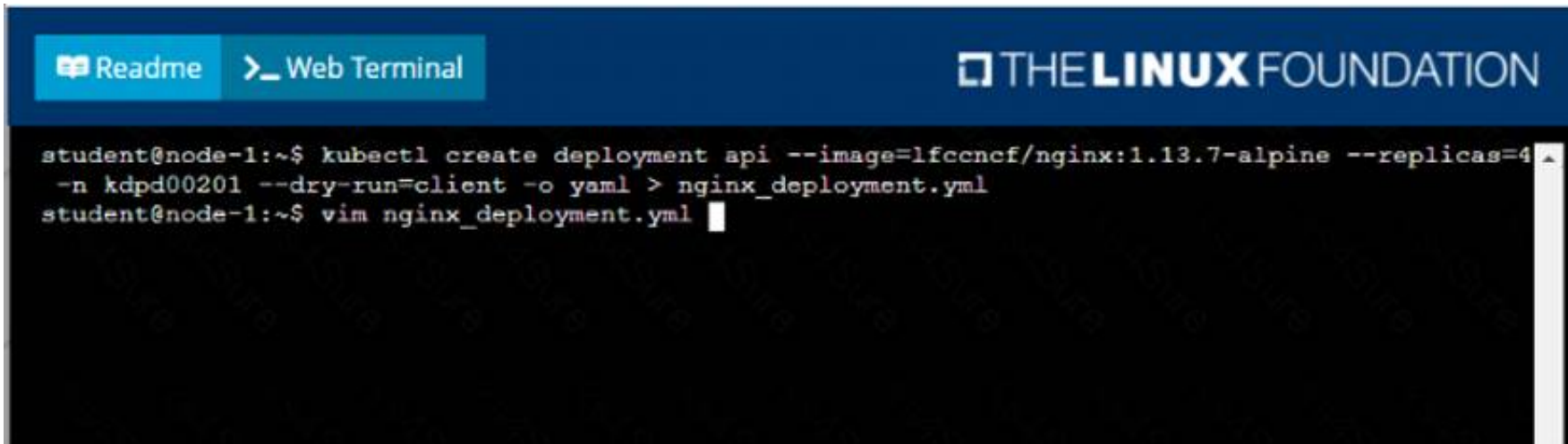
- Run the deployment in the kdpd00201 namespace. The namespace has already been created
- Name the deployment frontend and configure with 4 replicas
- Configure the pod with a container image of lfcncf/nginx:1.13.7
- Set an environment variable of NGINX PORT=8080 and also expose that port for the container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

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:



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 15

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
```

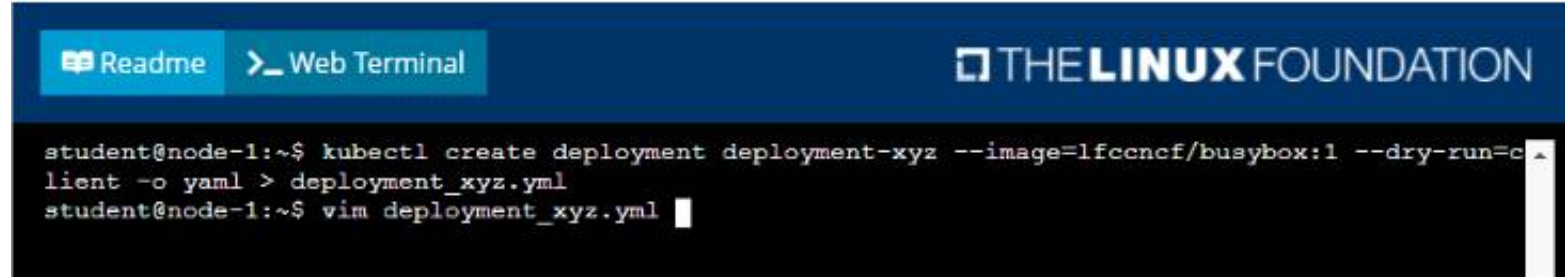
- The adapter-zen sidecar container should read /tmp/log/input.log and output the data to /tmp/log/output.\* in Fluentd JSON format. Note that no knowledge of Fluentd is required to complete this task: all you will need to achieve this is to create the ConfigMap from the spec file provided at /opt/KDMC00102/fluentd-configmap.p.yaml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

A. Mastered  
 B. Not Mastered

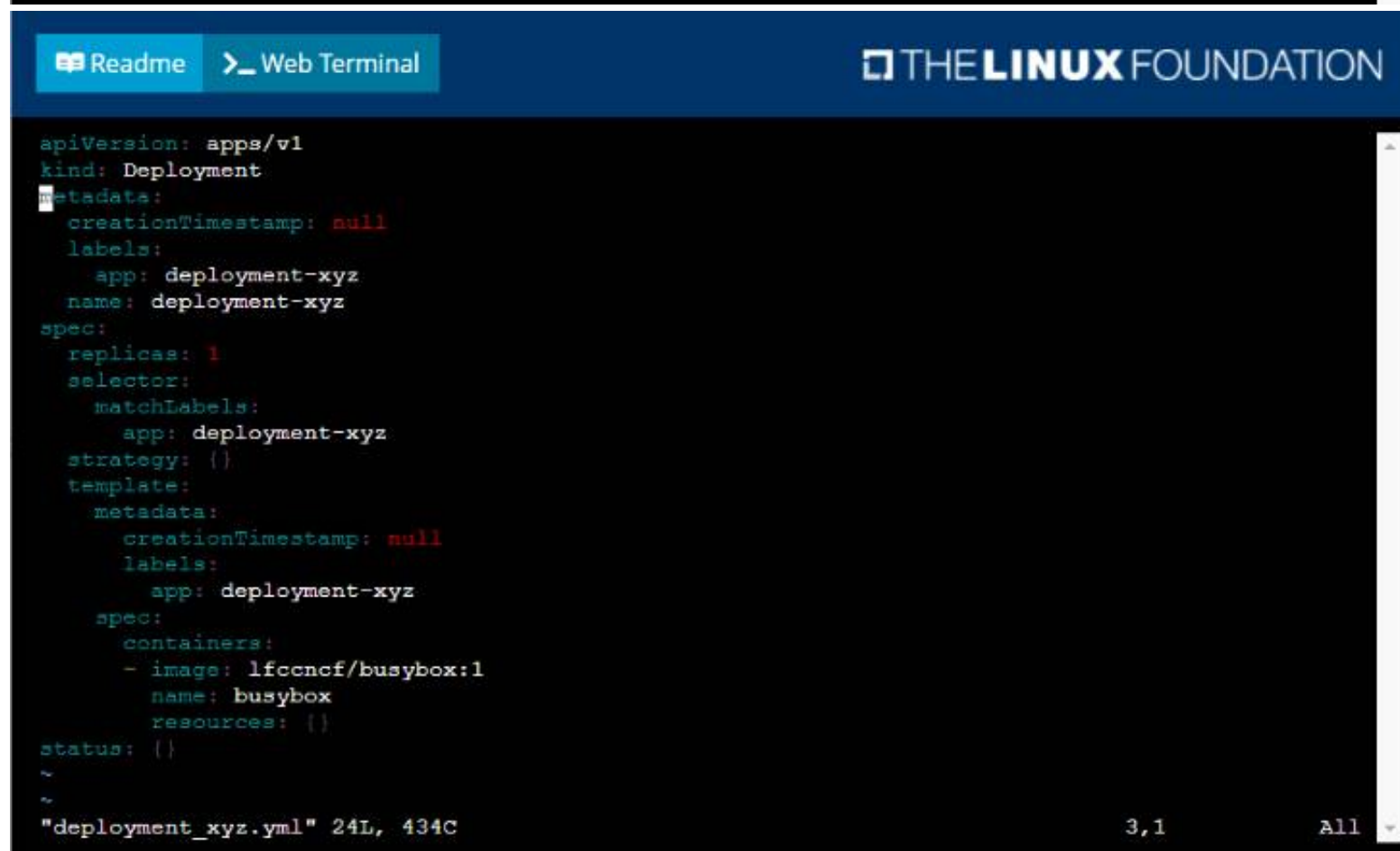
**Answer:** A

**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: {}
~
~
"deployment_xyz.yml" 24L, 434C 3,1 All
```



```
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: myvol1
        emptyDir: {}
      containers:
      - image: lfcncf/busybox:1
        name: logger-dev
        volumeMounts:
        - name: myvol1
          mountPath: /tmp/log
      - image: lfcncf/fluentd:v0.12
        name: adapter-zen
3 lines yanked 27,22 Bot
```

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: lfccncf/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: lfccncf/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: lfccncf/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: lfccncf/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/etc

```

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:~$

```

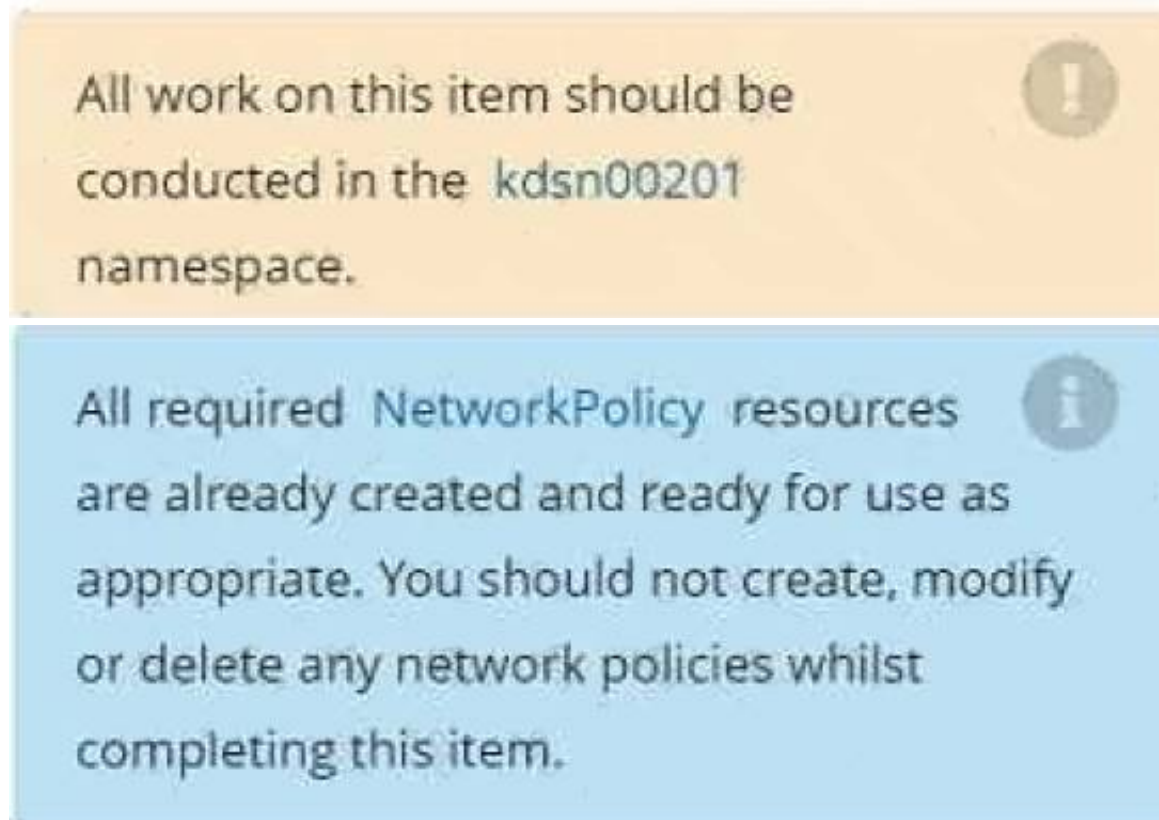
## NEW QUESTION 20

Exhibit:



#### Task

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



- A. Mastered
- B. Not Mastered

**Answer:** A

#### Explanation:

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

#### NEW QUESTION 22

.....



## Thank You for Trying Our Product

### We offer two products:

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

2nd - Questions and Answers in PDF Format

### CKAD Practice Exam Features:

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

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