Linux Foundation

CKAD

Certified Kubernetes Application Developer

QUESTION & ANSWERS

QUESTION 1

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

lfccncf/busybox:1 container, named logger-dev

* includes a sidecar lfccncf/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-configma p.yaml , and mount that ConfigMap to /fluentd/etc in the adapter-

zen sidecar container

A.

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 cncf' >> /tmp/log/input.log; sl
eep 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:~$
```

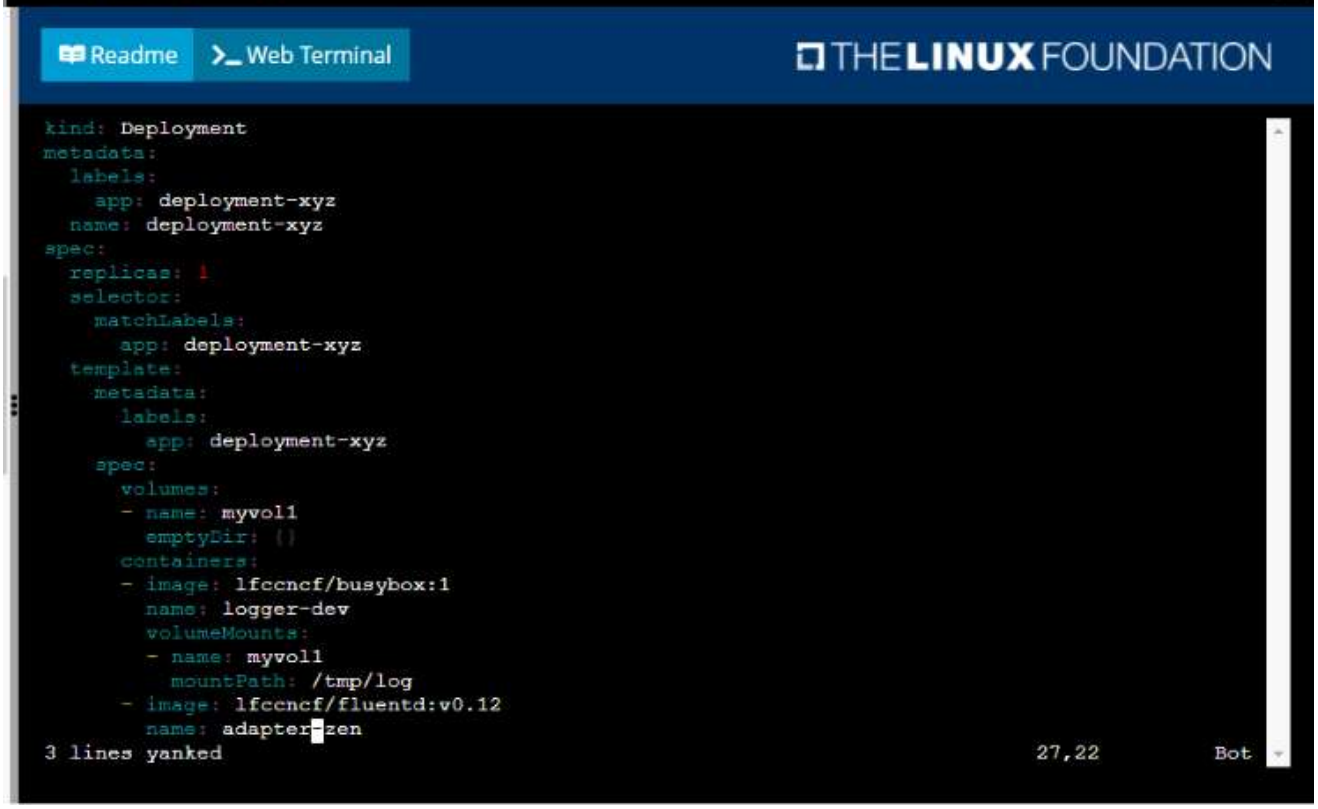

```
student@node-1:~$ kubectl create -f deployment_xyz.yml
deployment.apps/deployment-xyz created
student@node-1:~$ kubectl get deployment
NAME             READY   UP-TO-DATE   AVAILABLE   AGE
deployment-xyz   0/1     1            0           5s
student@node-1:~$ kubectl get deployment
NAME         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

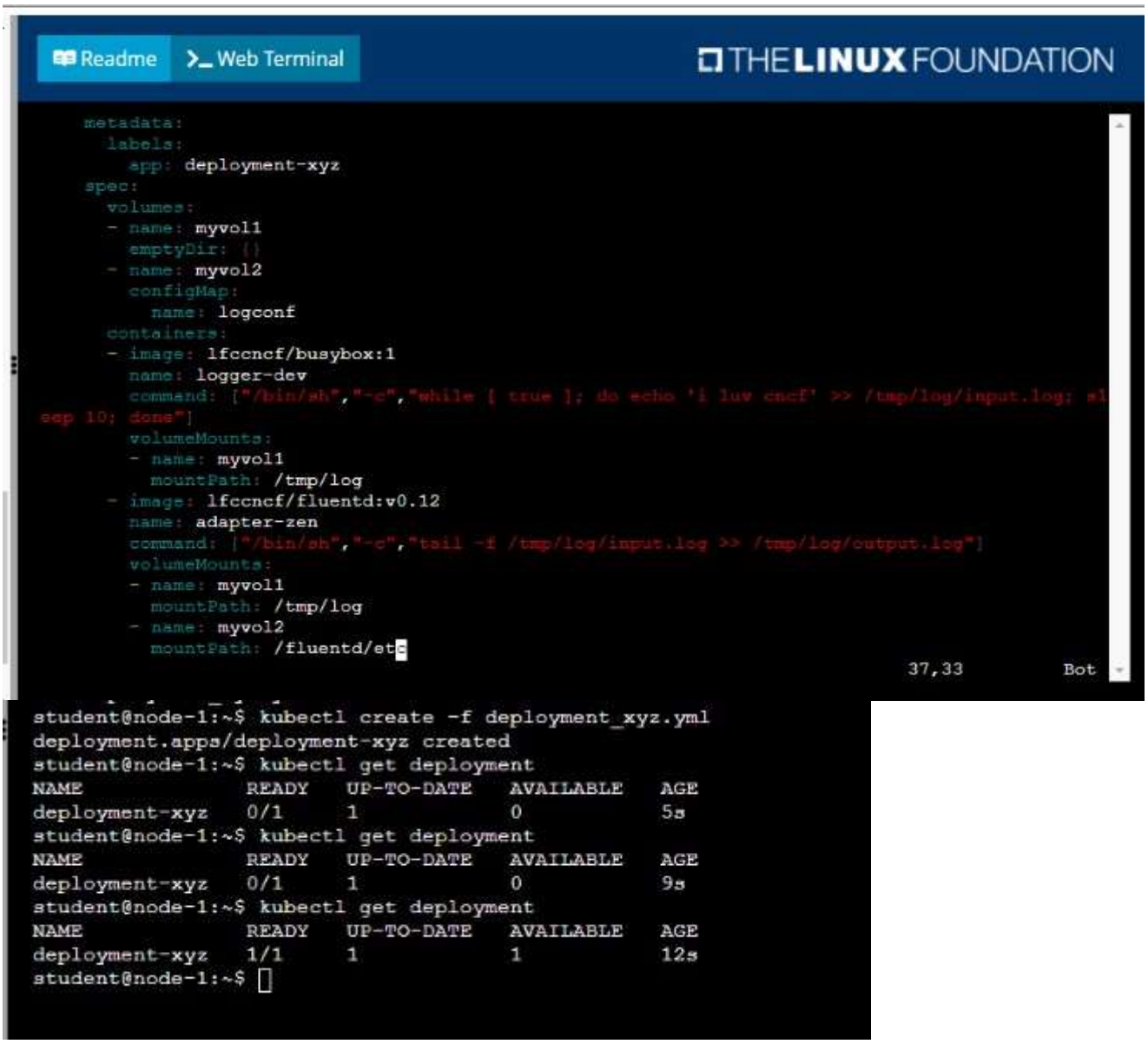

B.

Readme | Web Terminal — THE LINUX FOUNDATION


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


Readme
Web Terminal
THE LINUX FOUNDATION
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: lfccncf/busybox:1
name: busybox
resources: {}
status: {}
~
~
"deployment_xyz.yml" 24L, 434C
3,1
All


Readme
Web Terminal
THE LINUX FOUNDATION
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: lfccncf/busybox:1
name: logger-dev
volumeMounts:
- name: myvol1
mountPath: /tmp/log
- image: lfccncf/fluentd:v0.12
name: adapter-zen
3 lines yanked
27,22
Bot



Correct Answer: A

QUESTION 2

Exhibit:

Set configuration context:

```
[student@node-1] $ kubectl config use-context k8s
```

Context

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

Task

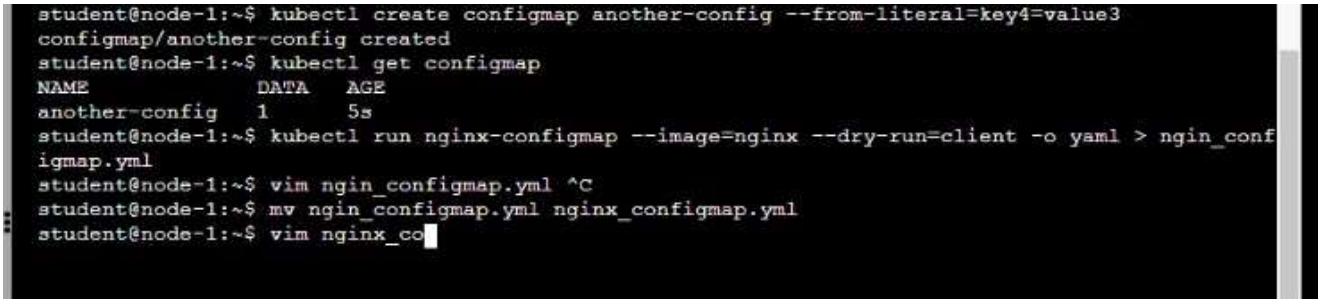
Please complete the following:

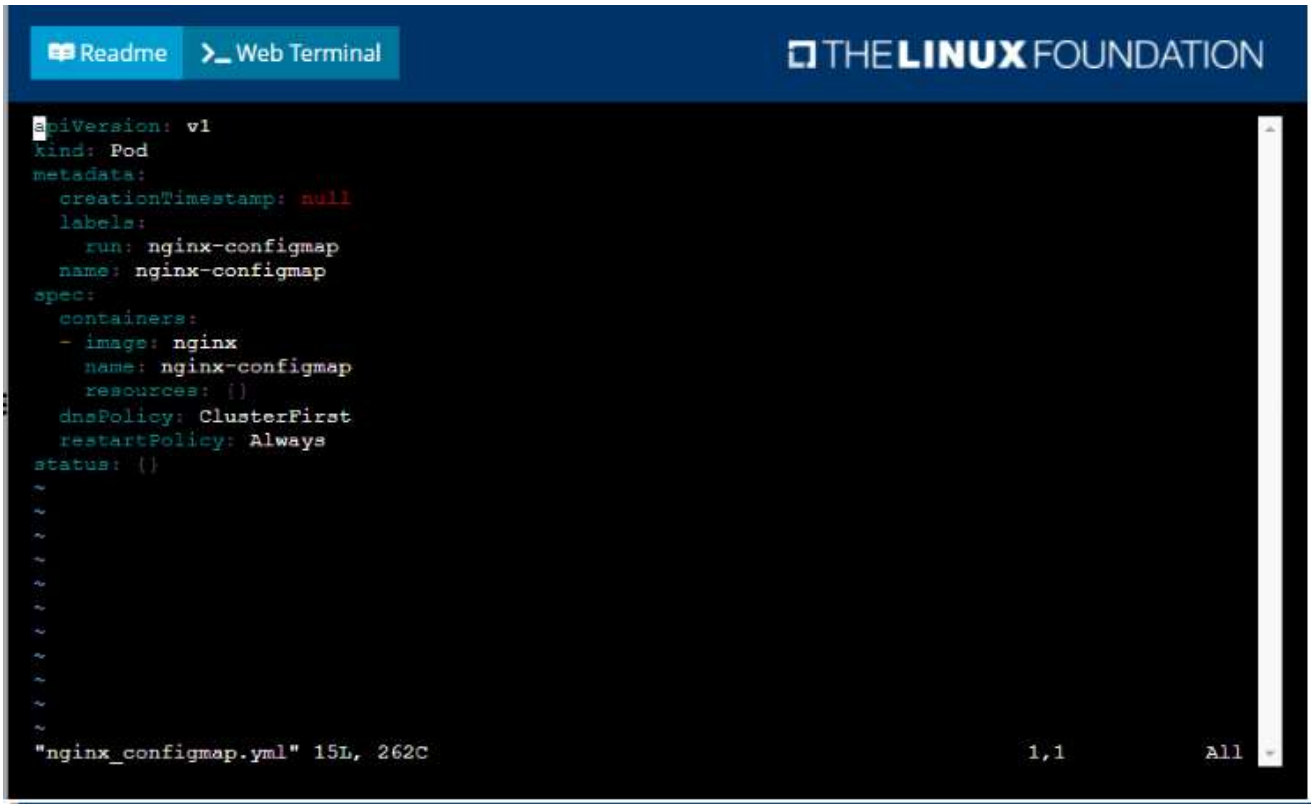
* Create a ConfigMap named another-config containing the key/value pair: key4/value3

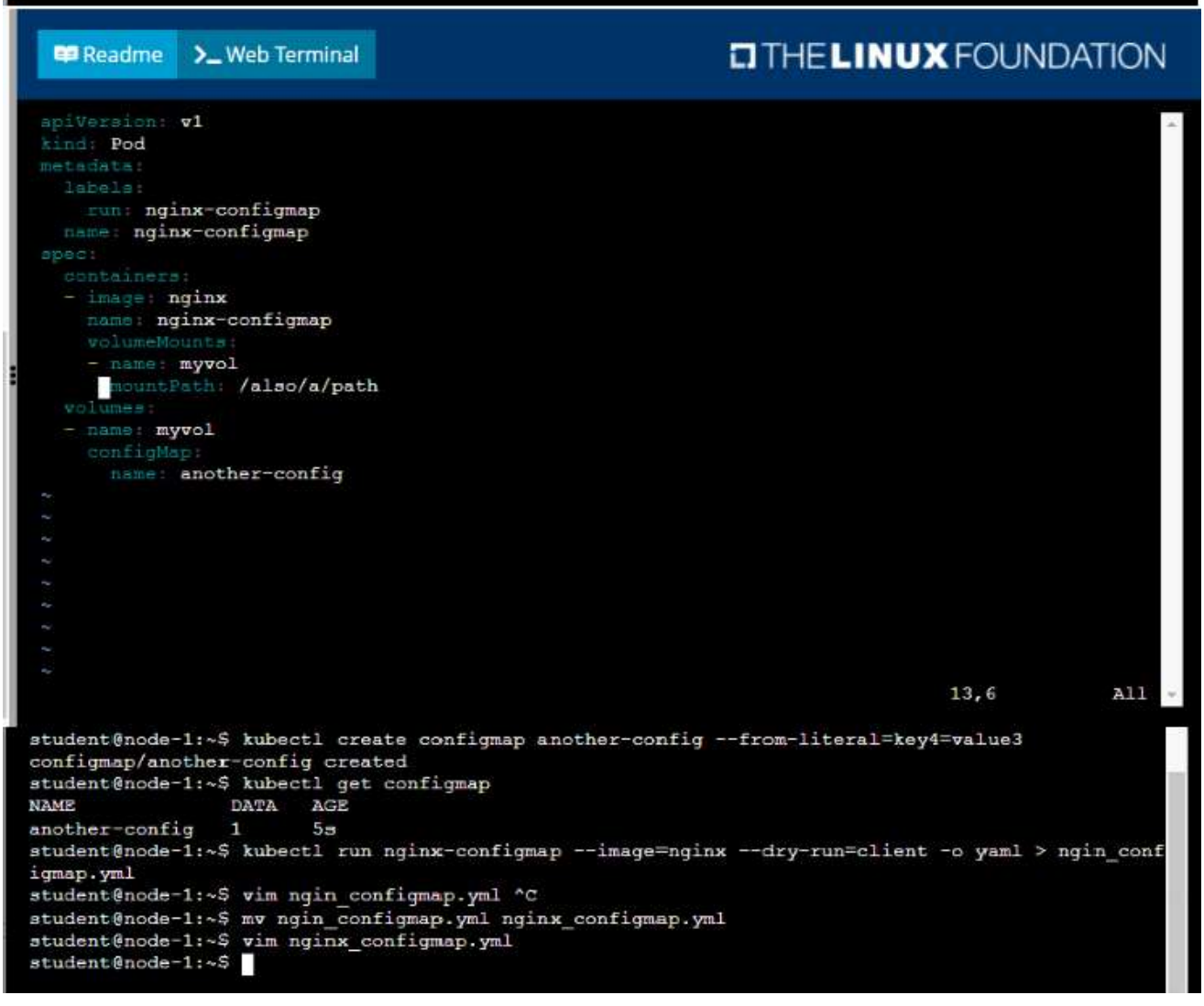
* start a pod named nginx-configmap containing a single container using the

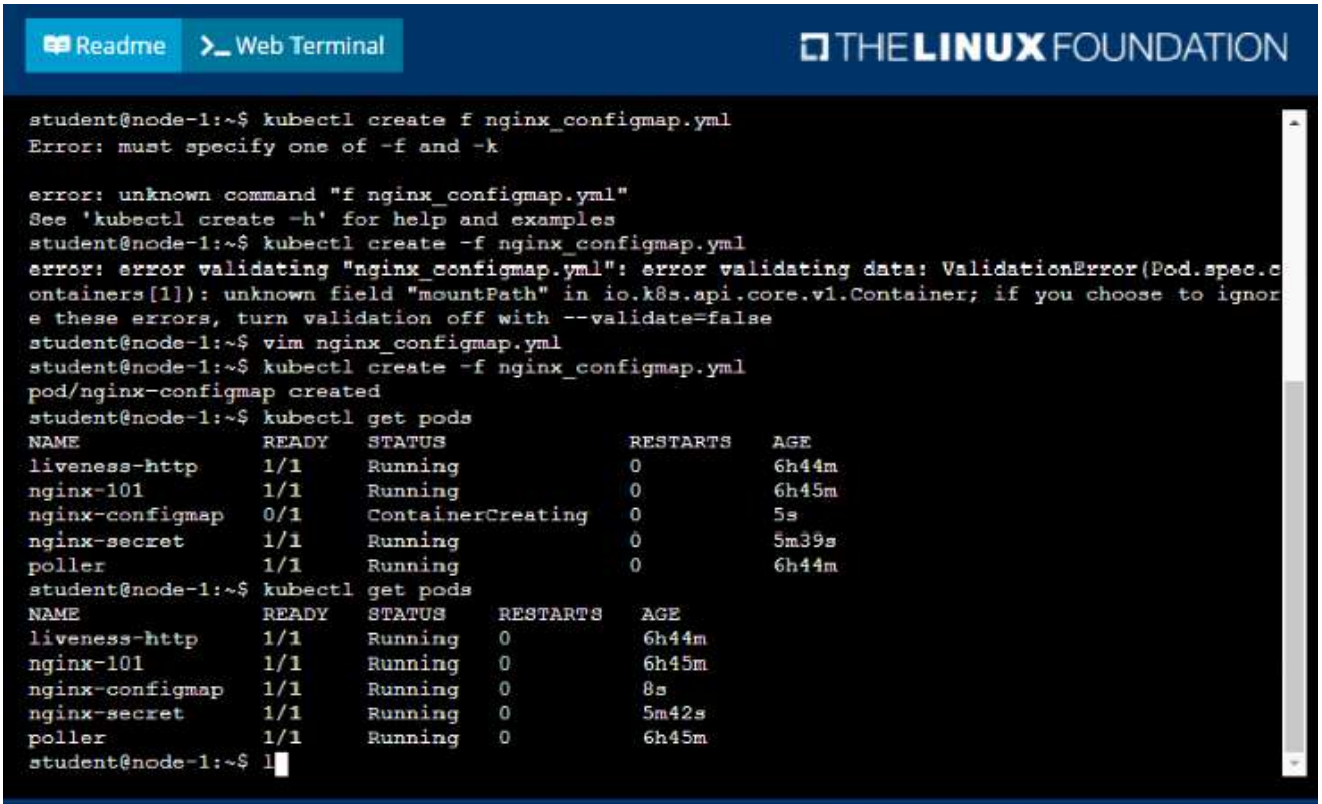
nginx image, and mount the key you just created into the pod under directory /also/a/path

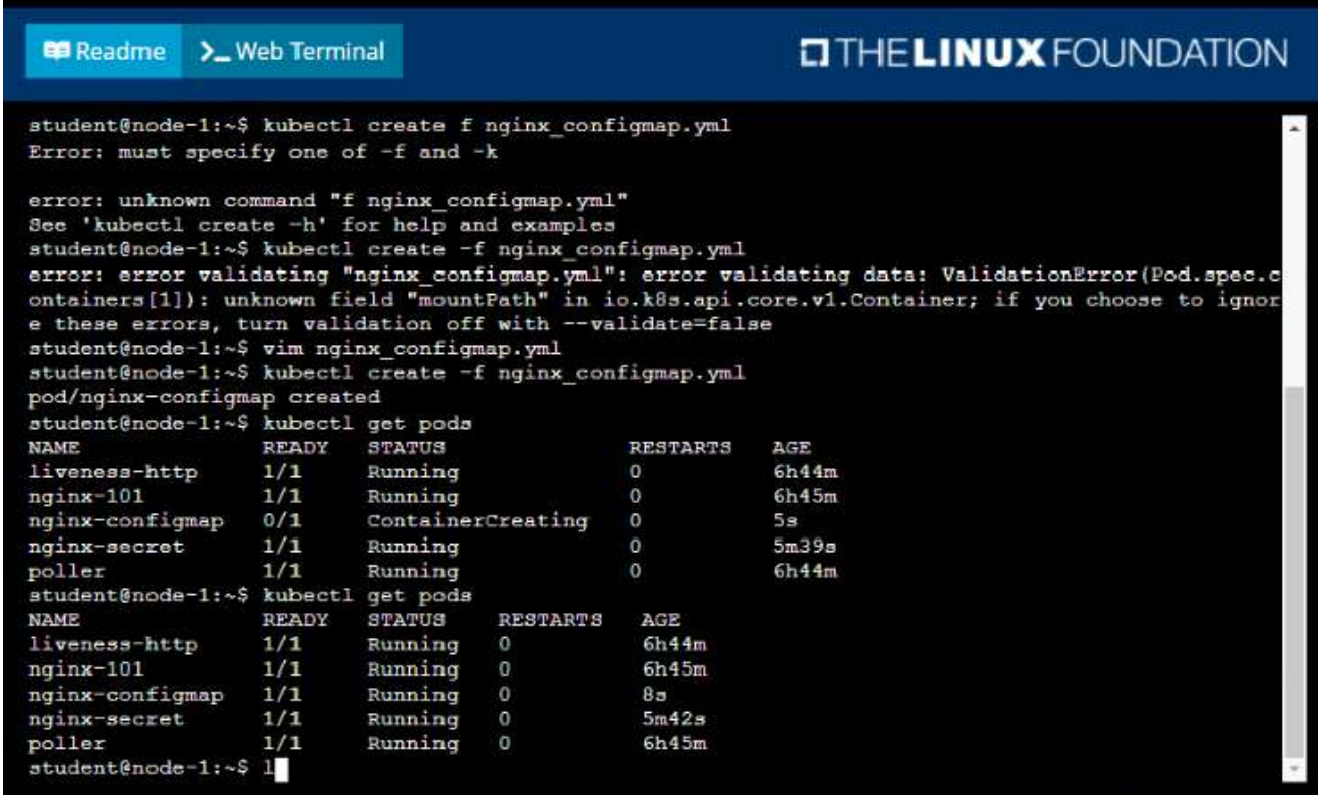
A.



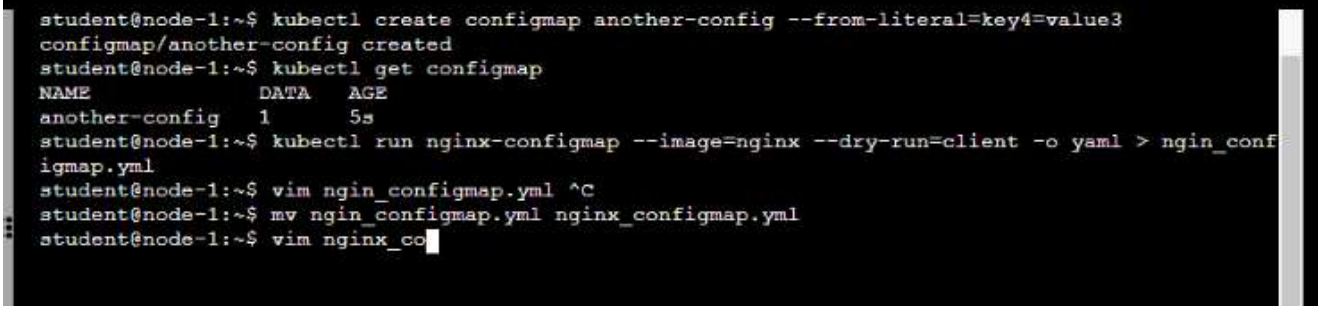

Readme
Web Terminal
THE LINUX FOUNDATION
apiVersion: v1
kind: Pod
metadata:
creationTimestamp: null
labels:
run: nginx-configmap
name: nginx-configmap
spec:
containers:
- image: nginx
name: nginx-configmap
resources: {}
dnsPolicy: ClusterFirst
restartPolicy: Always
status: {}
"nginx_configmap.yml" 15L, 262C
1,1
All

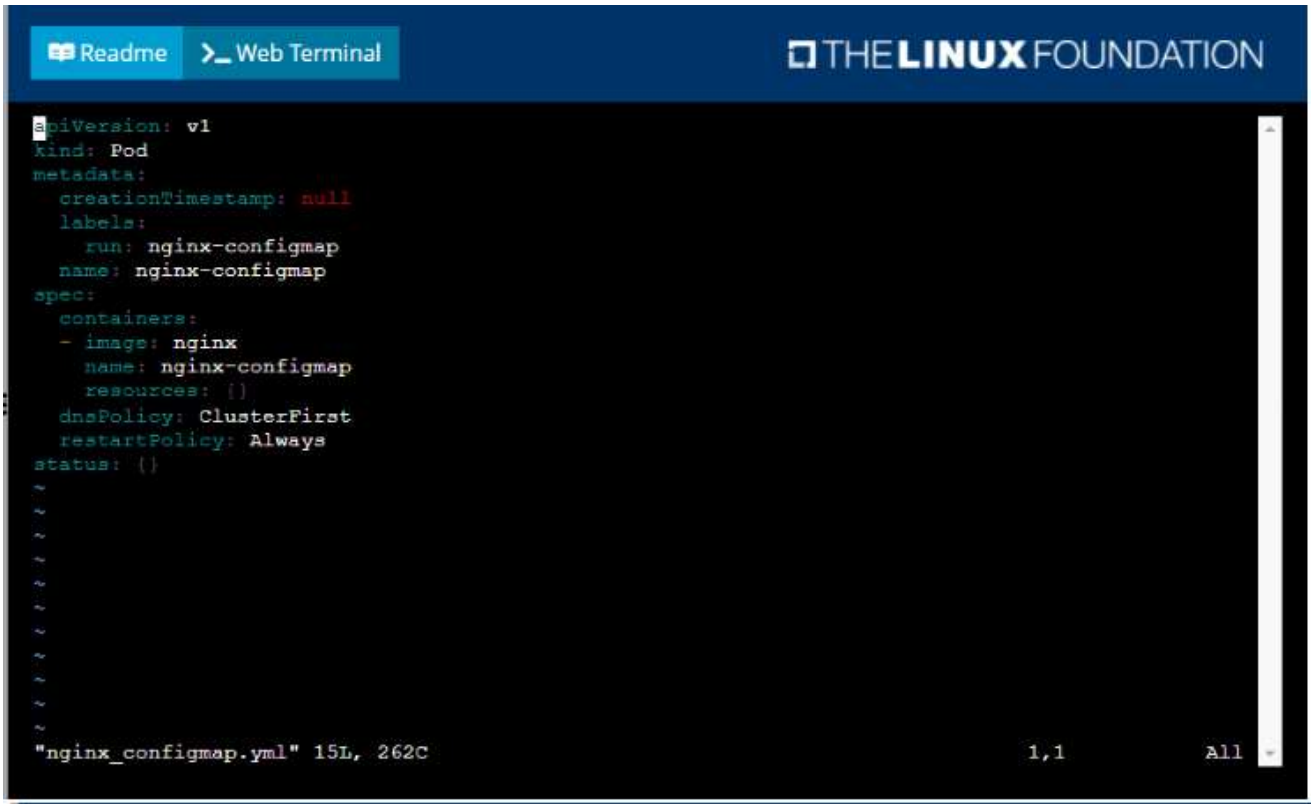

Readme
Web Terminal
THE LINUX FOUNDATION
apiVersion: v1
kind: Pod
metadata:
labels:
run: nginx-configmap
name: nginx-configmap
spec:
containers:
- image: nginx
name: nginx-configmap
volumeMounts:
- name: myvol
mountPath: /also/a/path
volumes:
- name: myvol
configMap:
name: another-config
13,6
All
student@node-1:~$ 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 > ngin_conf
igmap.yml
student@node-1:~$ vim ngin_configmap.yml ^C
student@node-1:~$ mv ngin_configmap.yml nginx_configmap.yml
student@node-1:~$ vim nginx_configmap.yml
student@node-1:~$

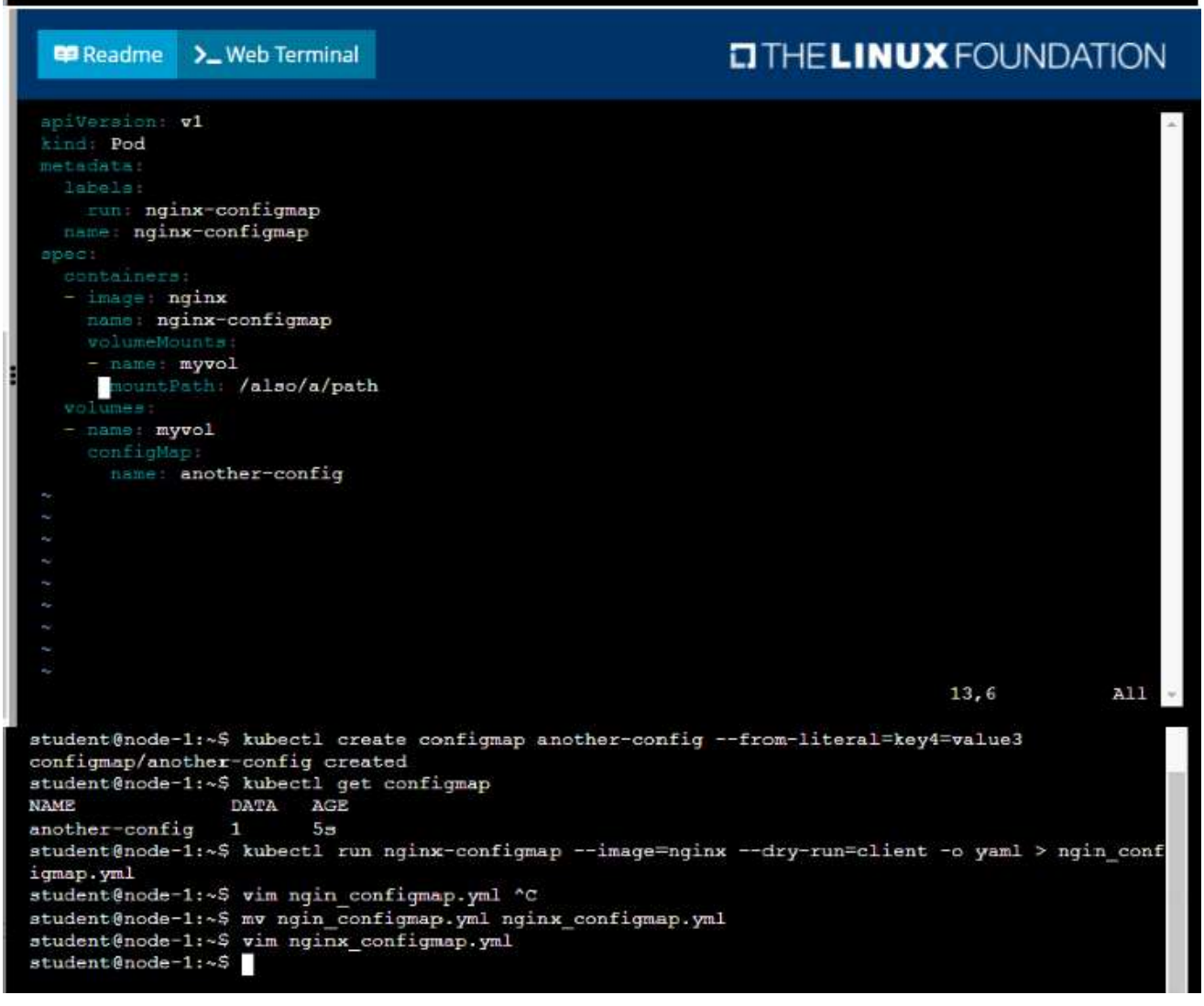

Readme
Web Terminal
THE LINUX FOUNDATION
student@node-1:~$ kubectl create f nginx_configmap.yml
Error: must specify one of -f and -k
error: unknown command "f nginx_configmap.yml"
See 'kubectl create -h' for help and examples
student@node-1:~$ kubectl create -f nginx_configmap.yml
error: error validating "nginx_configmap.yml": error validating data: ValidationError(Pod.spec.containers[1]): unknown field "mountPath" in io.k8s.api.core.v1.Container; if you choose to ignore these errors, turn validation off with --validate=false
student@node-1:~$ vim nginx_configmap.yml
student@node-1:~$ kubectl create -f nginx_configmap.yml
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

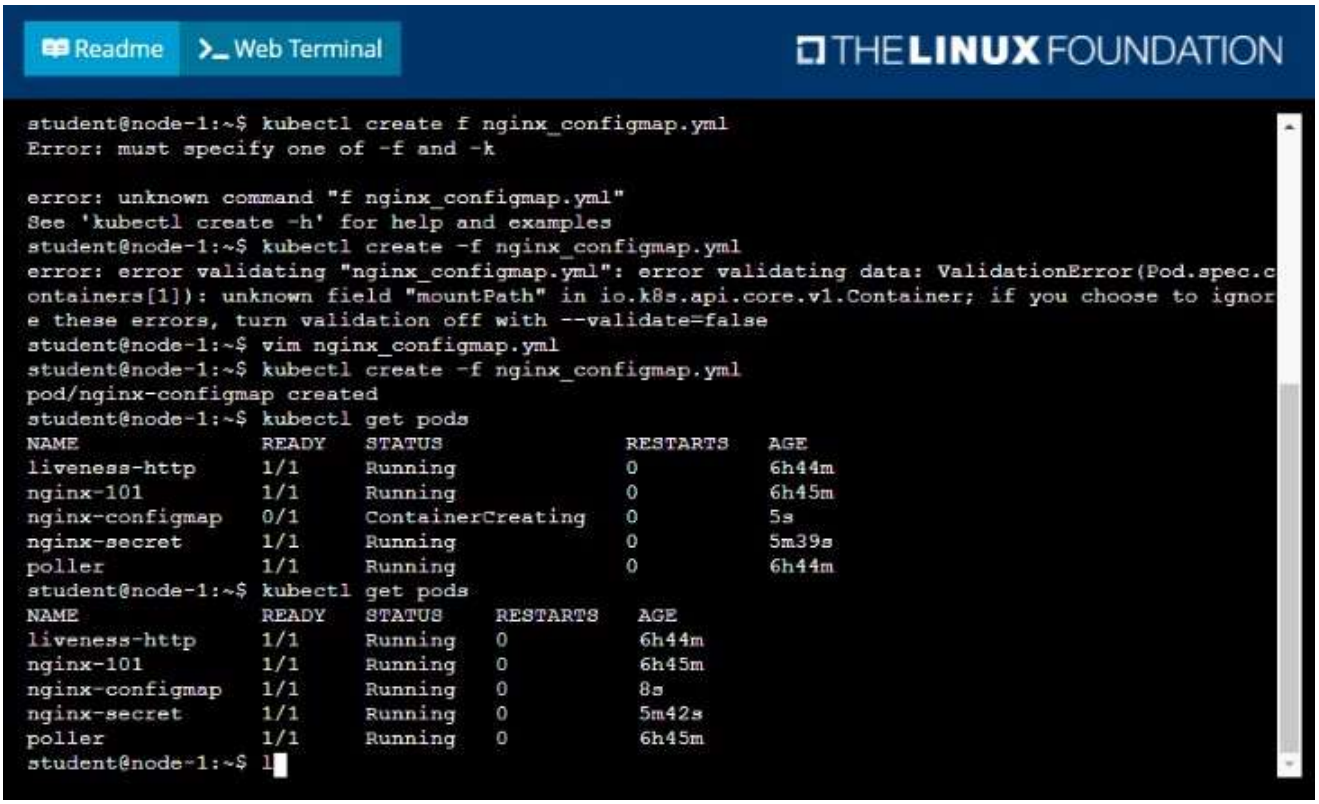

Readme
Web Terminal
THE LINUX FOUNDATION
student@node-1:~$ kubectl create f nginx_configmap.yml
Error: must specify one of -f and -k
error: unknown command "f nginx_configmap.yml"
See 'kubectl create -h' for help and examples
student@node-1:~$ kubectl create -f nginx_configmap.yml
error: error validating "nginx_configmap.yml": error validating data: ValidationError(Pod.spec.containers[1]): unknown field "mountPath" in io.k8s.api.core.v1.Container; if you choose to ignore these errors, turn validation off with --validate=false
student@node-1:~$ vim nginx_configmap.yml
student@node-1:~$ kubectl create -f nginx_configmap.yml
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

B.


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 > ngin_configmap.yml
student@node-1:~$ vim ngin_configmap.yml ^C
student@node-1:~$ mv ngin_configmap.yml nginx_configmap.yml
student@node-1:~$ vim nginx_co


Readme
Web Terminal
THE LINUX FOUNDATION
apiVersion: v1
kind: Pod
metadata:
creationTimestamp: null
labels:
run: nginx-configmap
name: nginx-configmap
spec:
containers:
- image: nginx
name: nginx-configmap
resources: {}
dnsPolicy: ClusterFirst
restartPolicy: Always
status: {}
"nginx_configmap.yml" 15L, 262C
1,1
All


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 > ngin_conf
igmap.yml
student@node-1:~$ vim ngin_configmap.yml ^C
student@node-1:~$ mv ngin_configmap.yml nginx_configmap.yml
student@node-1:~$ vim nginx_configmap.yml
student@node-1:~$



Correct Answer: A

QUESTION 3

Exhibit:

Context

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

Task

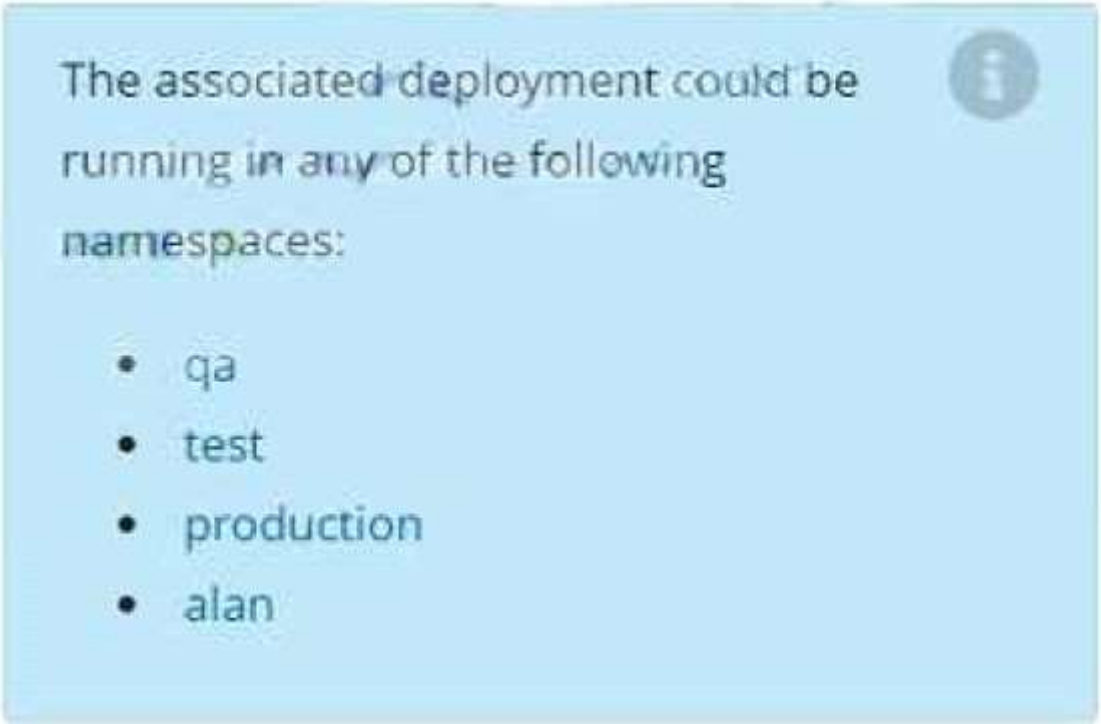
Perform the following tasks:

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

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. Solution:Create the Pod:kubectl create -f http://k8s.io/docs/tasks/configure-pod-container/exec-liveness.yamlWithin 30 seconds, view the Pod events:kubectl describe pod liveness-execThe 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 worker023s 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 86849c15382eAfter 35 seconds, view the Pod events again:kubectl describe pod liveness-execAt 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 worker036s 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 86849c15382e2s 2s 1 {kubelet worker0} spec.containers{liveness} Warning Unhealthy Liveness probe failed: cat: can't open '/tmp/healthy': No such file or directoryWait another 30 seconds, and verify that the Container has been restarted:kubectl get pod liveness-execThe output shows thatRESTARTShas been incremented:NAME READY STATUS RESTARTS AGEliveness-exec 1/1 Running 1 m

B. Solution:Create the Pod:kubectl create -f http://k8s.io/docs/tasks/configure-pod-container/exec-liveness.yamlWithin 30 seconds, view the Pod events:kubectl describe pod liveness-execThe 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 worker023s 23s 1 {kubelet worker0} spec.containers{liveness} Normal Pulling pulling image 'gcr.io/google_containers/busybox'kubectl describe pod liveness-execAt 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 worker036s 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 86849c15382e2s 2s 1 {kubelet worker0} spec.containers{liveness} Warning Unhealthy Liveness probe failed: cat: can't open '/tmp/healthy': No such file or directoryWait another 30 seconds, and verify that the Container has been restarted:kubectl get pod liveness-execThe output shows thatRESTARTShas been incremented:NAME READY STATUS RESTARTS AGEliveness-exec 1/1 Running 1 m

Correct Answer: A