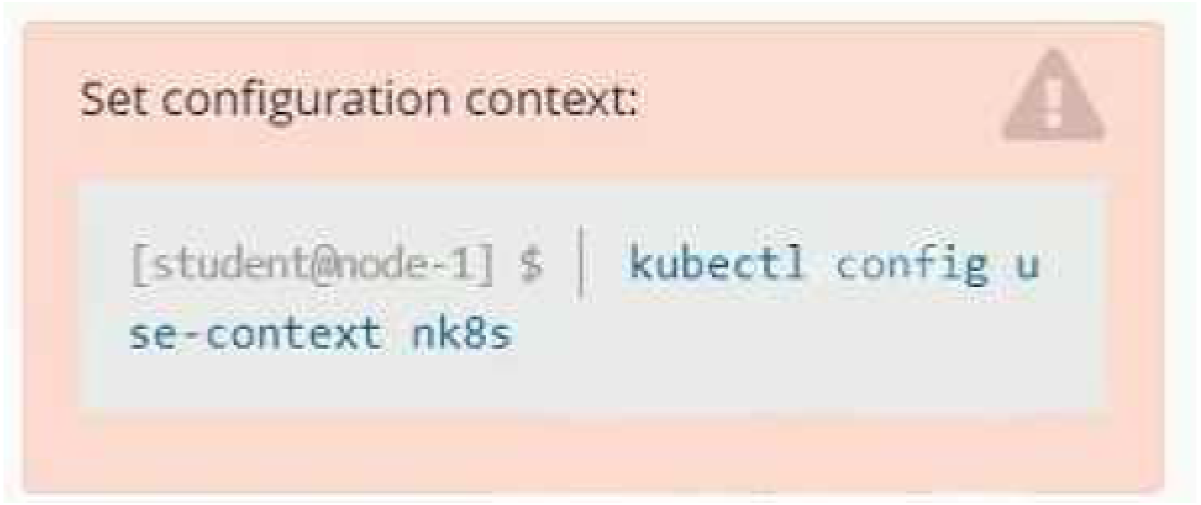
Free Questions for CKAD

Shared by Carrillo on 04-10-2024

Question 1

Question Type: **MultipleChoice**

Refer to Exhibit.



Set configuration context:

```
[student@node-1] $ | kubectl config use-context nk8s
```

Task

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

Options:

A- Explanation:

create deploy hello-deploy --image=nginx --dry-run=client -o yaml > hello-deploy.yaml
Update deployment image tonginx:1.17.4: kubectl set image deploy/hello-deploy nginx=nginx:1.17.4

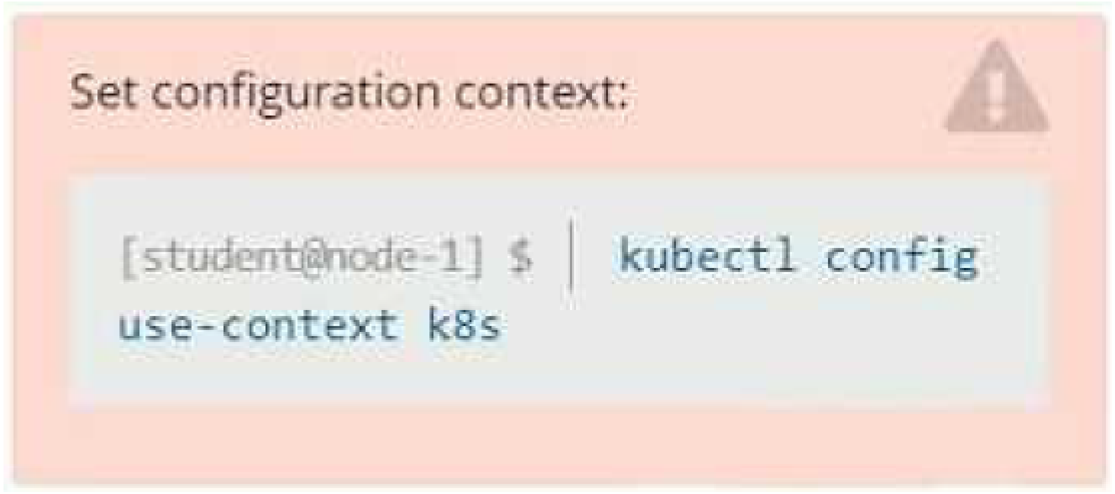
Answer:

A

Question 2

Question Type: **MultipleChoice**

Refer to 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:

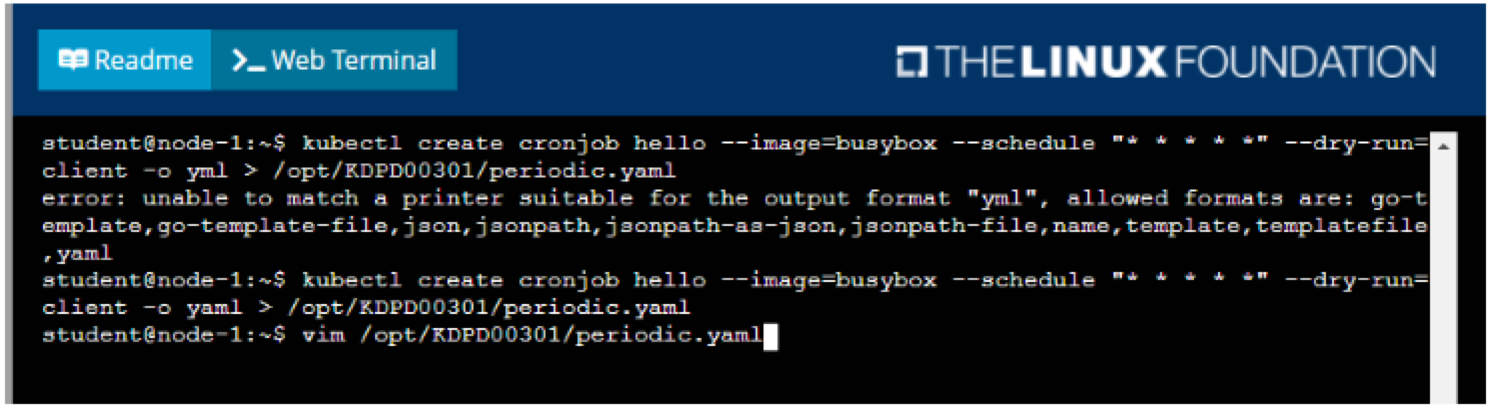
* Create a YAML formatted Kubernetes manifest /opt/KDPD00301/periodic.yaml that runs the following shell command: date in a single busybox container. The command should run every minute and must complete within 22 seconds or be terminated oy Kubernetes. The Cronjob namp and container name should both be hello

* Create the resource in the above manifest and verify that the job executes successfully at least once

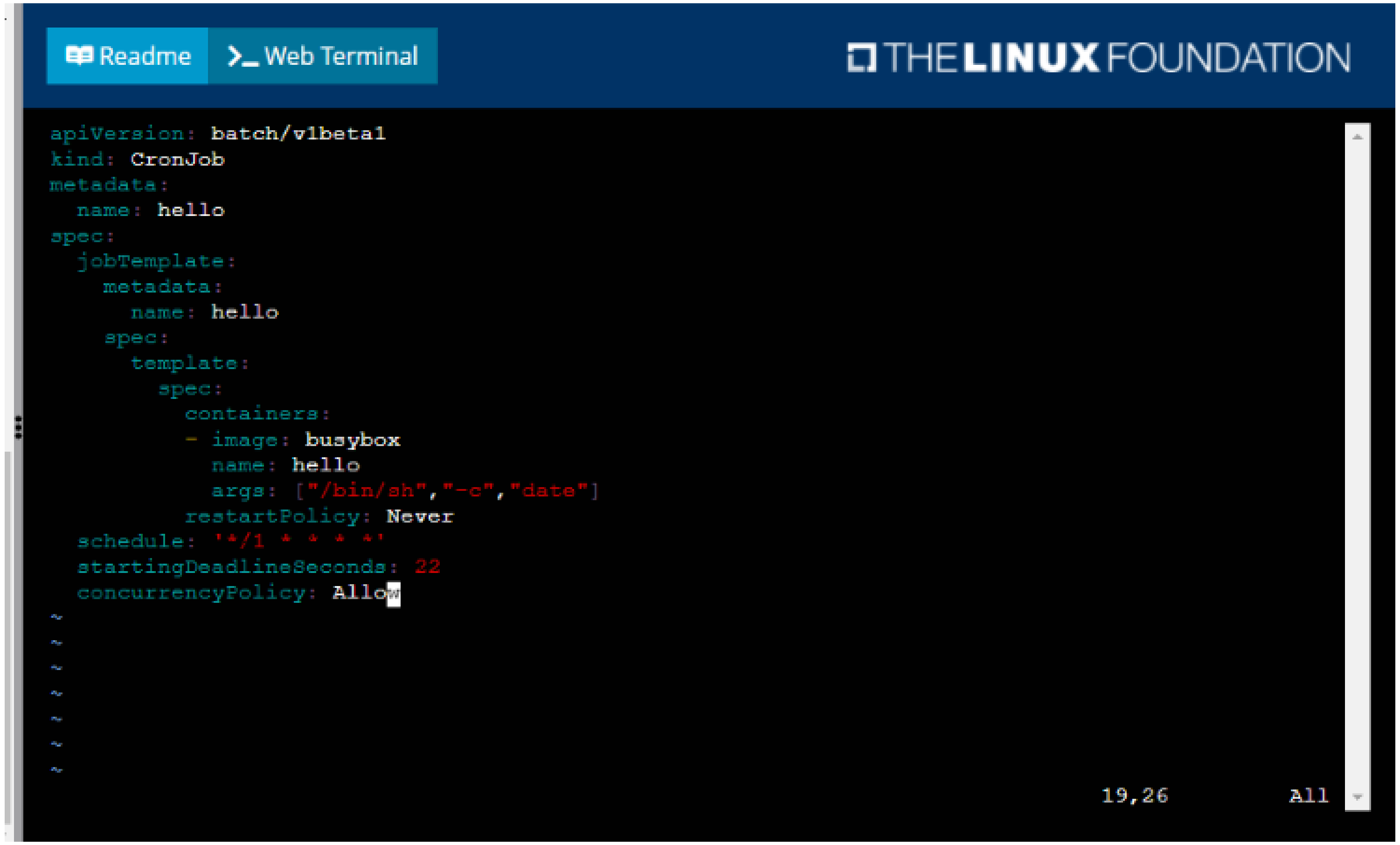
Options:

A- Explanation:

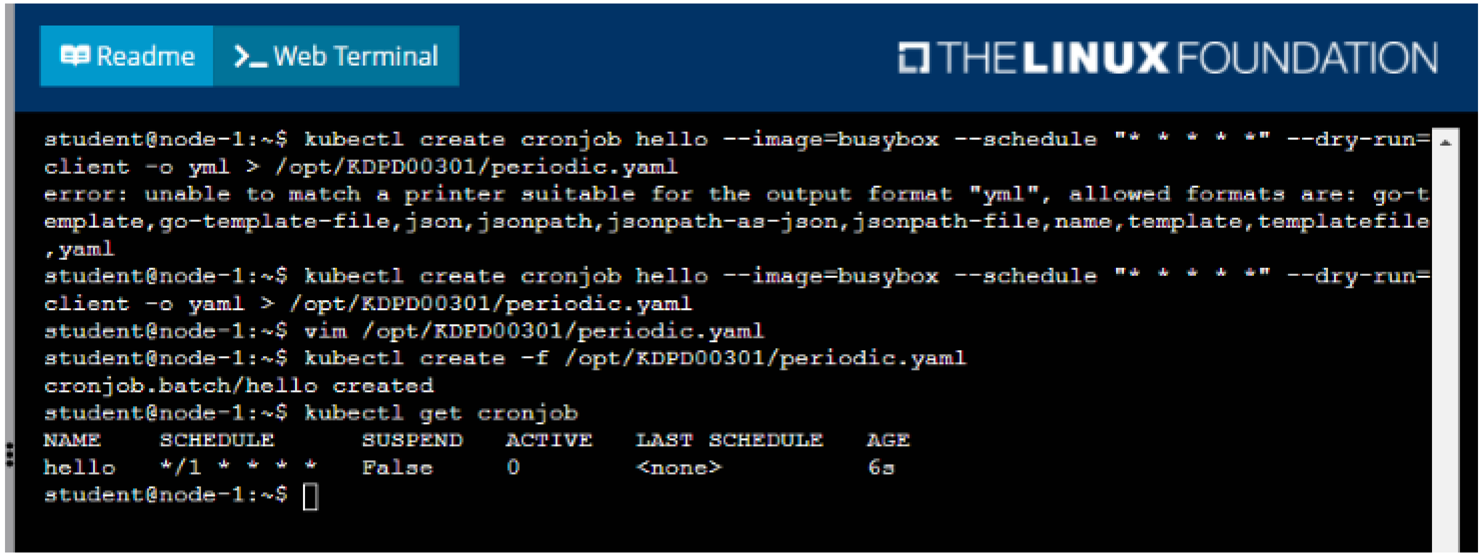
Solution:



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



```
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: '*/1 * * * *'
  startingDeadlineSeconds: 22
  concurrencyPolicy: Allow
~
~
~
~
~
~
~
                                                                  19,26         All
```



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

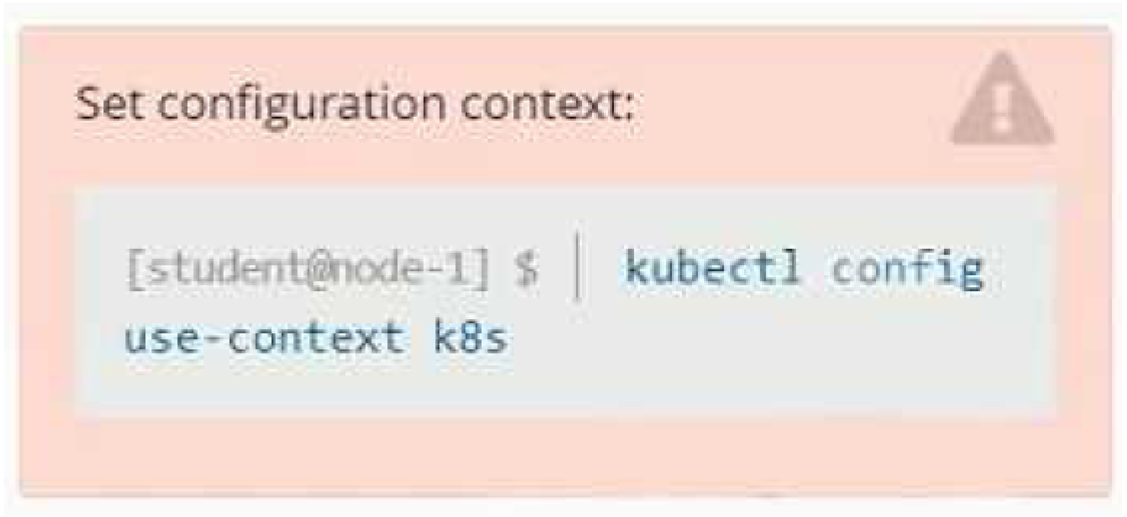
Answer:

A

Question 3

Question Type: MultipleChoice

Refer to Exhibit.


Set configuration context:

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

Set Configuration Context:

[student@node-1] $ | kubectl

Config use-context k8s

Context

A container within the poller pod is hard-coded to connect the nginxsvc service on port 90 . As this port changes to 5050 an additional container needs to be added to the poller pod which adapts the container to connect to this new port. This should be realized as an ambassador container within the pod.

Task

* Update the nginxsvc service to serve on port 5050.

* Add an HAproxy container named haproxy bound to port 90 to the poller pod and deploy the enhanced pod. Use the image haproxy and inject the configuration located at /opt/KDMC00101/haproxy.cfg, with a ConfigMap named haproxy-config, mounted into the container so that haproxy.cfg is available at /usr/local/etc/haproxy/haproxy.cfg. Ensure that you update the args of the poller container to connect to localhost instead of nginxsvc so that the connection is correctly proxied to the new service endpoint. You must not modify the port of the endpoint in poller's args . The spec file used to create the initial poller pod is available in /opt/KDMC00101/poller.yaml

Options:

A- Explanation:

Solution:

To update the nginxsvc service to serve on port 5050, you will need to edit the service's definition yaml file. You can use the kubectl edit command to edit the service in place.

kubectl edit svc nginxsvc

This will open the service definition yaml file in your default editor. Change the targetPort of the service to 5050 and save the file.

To add an HAproxy container named haproxy bound to port 90 to the poller pod, you will need to edit the pod's definition yaml file located at /opt/KDMC00101/poller.yaml.

You can add a new container to the pod's definition yaml file, with the following configuration:

```
containers:
- name: haproxy
image: haproxy
ports:
- containerPort: 90
volumeMounts:
- name: haproxy-config
```

mountPath: /usr/local/etc/haproxy/haproxy.cfg

subPath: haproxy.cfg

args: ['haproxy', '-f', '/usr/local/etc/haproxy/haproxy.cfg']

This will add the HAproxy container to the pod and configure it to listen on port 90. It will also mount the ConfigMap haproxy-config to the container, so that haproxy.cfg is available at /usr/local/etc/haproxy/haproxy.cfg.

To inject the configuration located at /opt/KDMC00101/haproxy.cfg to the container, you will need to create a ConfigMap using the following command:

kubectl create configmap haproxy-config --from-file=/opt/KDMC00101/haproxy.cfg

You will also need to update the args of the poller container so that it connects to localhost instead of nginxsvc. You can do this by editing the pod's definition yaml file and changing the args field to args: ['poller','--host=localhost'].

Once you have made these changes, you can deploy the updated pod to the cluster by running the following command:

kubectl apply -f /opt/KDMC00101/poller.yaml

This will deploy the enhanced pod with the HAproxy container to the cluster. The HAproxy container will listen on port 90 and proxy connections to the nginxsvc service on port 5050. The poller container will connect to localhost instead of nginxsvc, so that the connection is correctly proxied to the new service endpoint.

Please note that, this is a basic example and you may need to tweak the haproxy.cfg file and the args based on your use case.

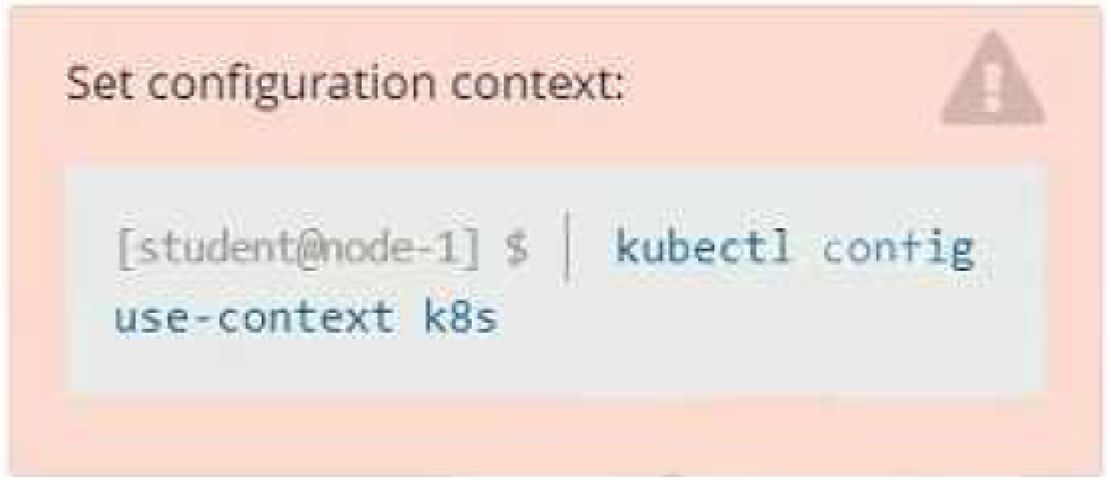
Answer:

A

Question 4

Question Type: **MultipleChoice**

Refer to Exhibit.


Set configuration context:

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

Context

You have been tasked with scaling an existing deployment for availability, and creating a service to expose the deployment within your infrastructure.

Task

Start with the deployment named kdsn00101-deployment which has already been deployed to the namespace kdsn00101 . Edit it to:

* Add the func=webFrontEnd key/value label to the pod template metadata to identify the pod for the service definition

* Have 4 replicas

Next, create ana deploy in namespace kdsn00l01 a service that accomplishes the following:

* Exposes the service on TCP port 8080

* is mapped to me pods defined by the specification of kdsn00l01-deployment

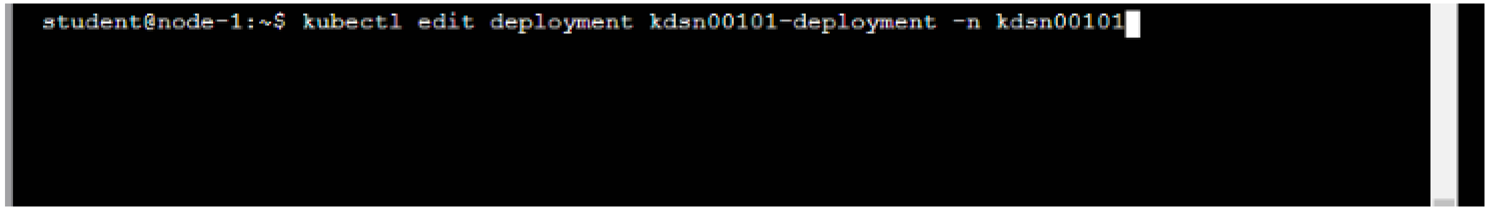
* Is of type NodePort

* Has a name of cherry

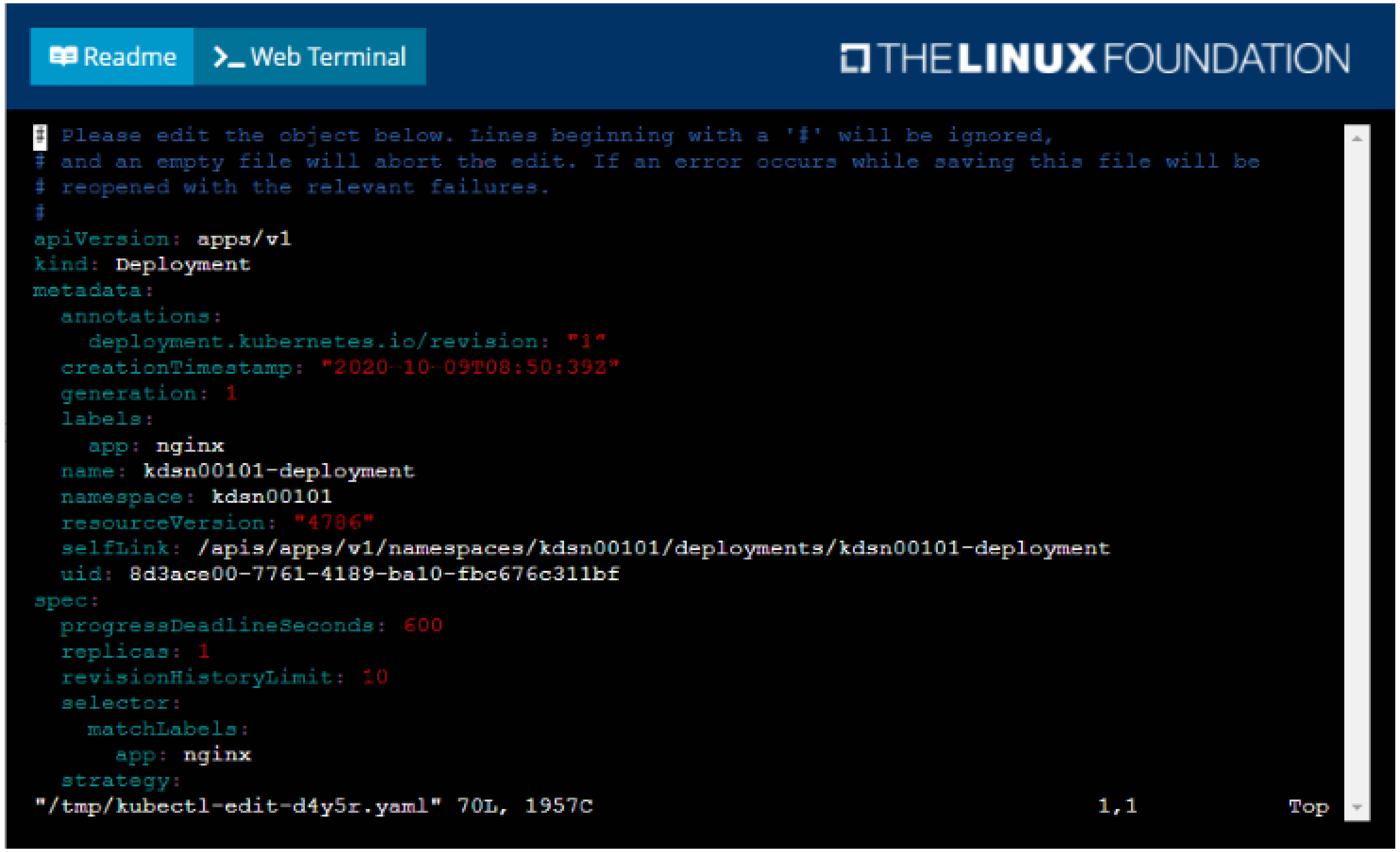
Options:

A- Explanation:

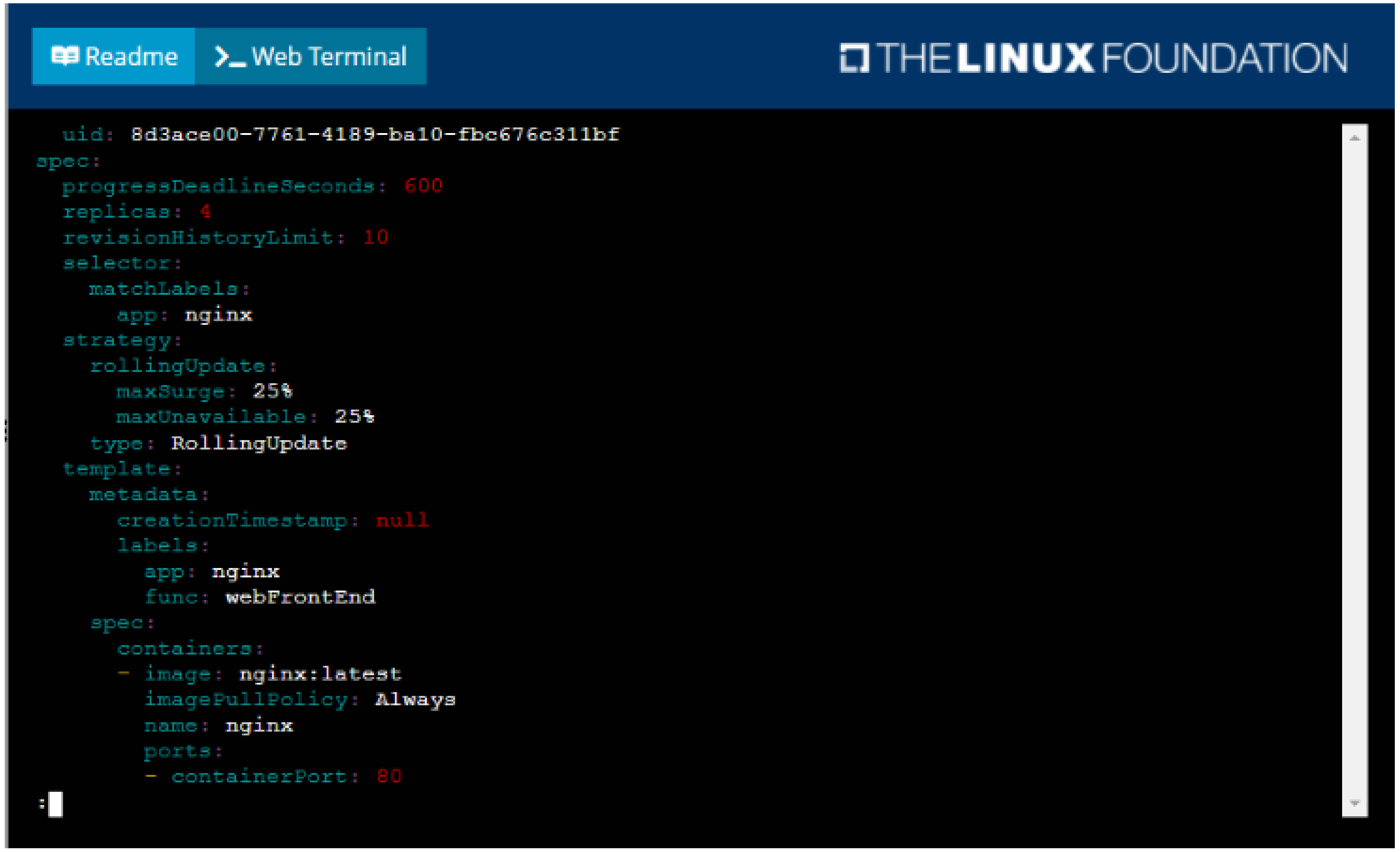
Solution:



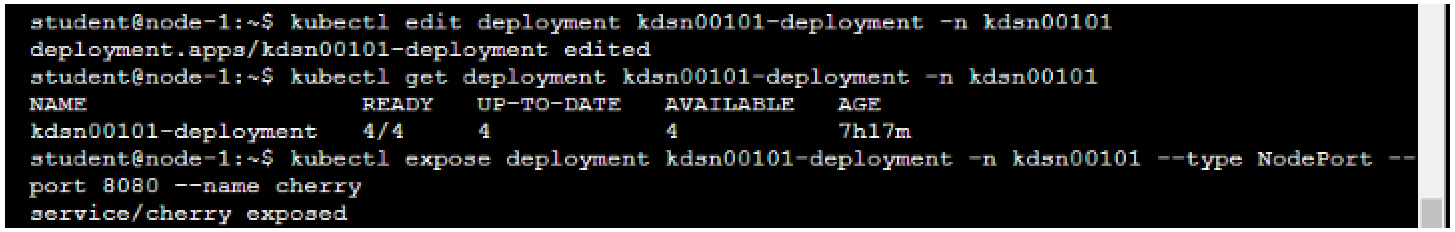
```
student@node-1:~$ kubectl edit deployment kdsn00101-deployment -n kdsn00101
```



```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.
#
apiVersion: apps/v1
kind: Deployment
metadata:
  annotations:
    deployment.kubernetes.io/revision: "1"
  creationTimestamp: "2020-10-09T08:50:39Z"
  generation: 1
  labels:
    app: nginx
  name: kdsn00101-deployment
  namespace: kdsn00101
  resourceVersion: "4786"
  selfLink: /apis/apps/v1/namespaces/kdsn00101/deployments/kdsn00101-deployment
  uid: 8d3ace00-7761-4189-ba10-fbc676c311bf
spec:
  progressDeadlineSeconds: 600
  replicas: 1
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app: nginx
  strategy:
"/tmp/kubectl-edit-d4y5r.yaml" 70L, 1957C                                   1,1           Top
```



```
  uid: 8d3ace00-7761-4189-ba10-fbc676c311bf
spec:
  progressDeadlineSeconds: 600
  replicas: 4
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app: nginx
  strategy:
    rollingUpdate:
      maxSurge: 25%
      maxUnavailable: 25%
    type: RollingUpdate
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: nginx
        func: webFrontEnd
    spec:
      containers:
      - image: nginx:latest
        imagePullPolicy: Always
        name: nginx
        ports:
        - containerPort: 80
:
```



```
student@node-1:~$ kubectl edit deployment kdsn00101-deployment -n kdsn00101
deployment.apps/kdsn00101-deployment edited
student@node-1:~$ kubectl get deployment kdsn00101-deployment -n kdsn00101
NAME                   READY   UP-TO-DATE   AVAILABLE   AGE
kdsn00101-deployment   4/4     4            4           7h17m
student@node-1:~$ kubectl expose deployment kdsn00101-deployment -n kdsn00101 --type NodePort --
port 8080 --name cherry
service/cherry exposed
```

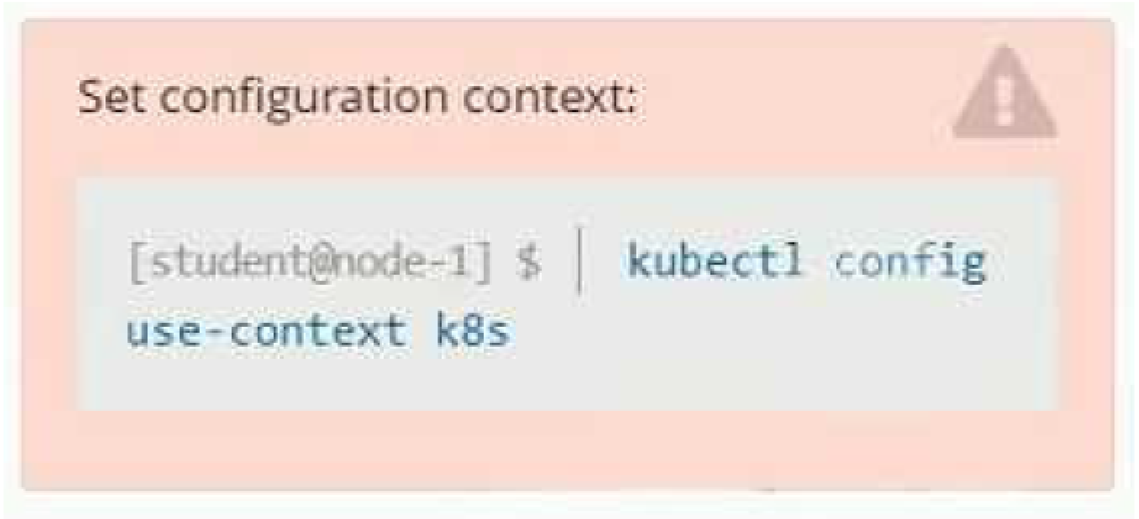
Answer:

A

Question 5

Question Type: MultipleChoice

Refer to Exhibit.



Set configuration context:

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

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%

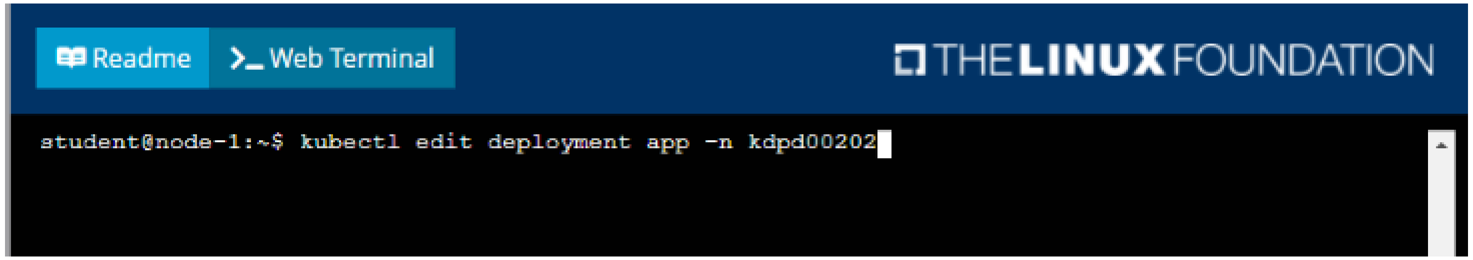
* Perform a rolling update of the web1 deployment, changing the Ifccncf/ngmx image version to 1.13

* Roll back the app deployment to the previous version

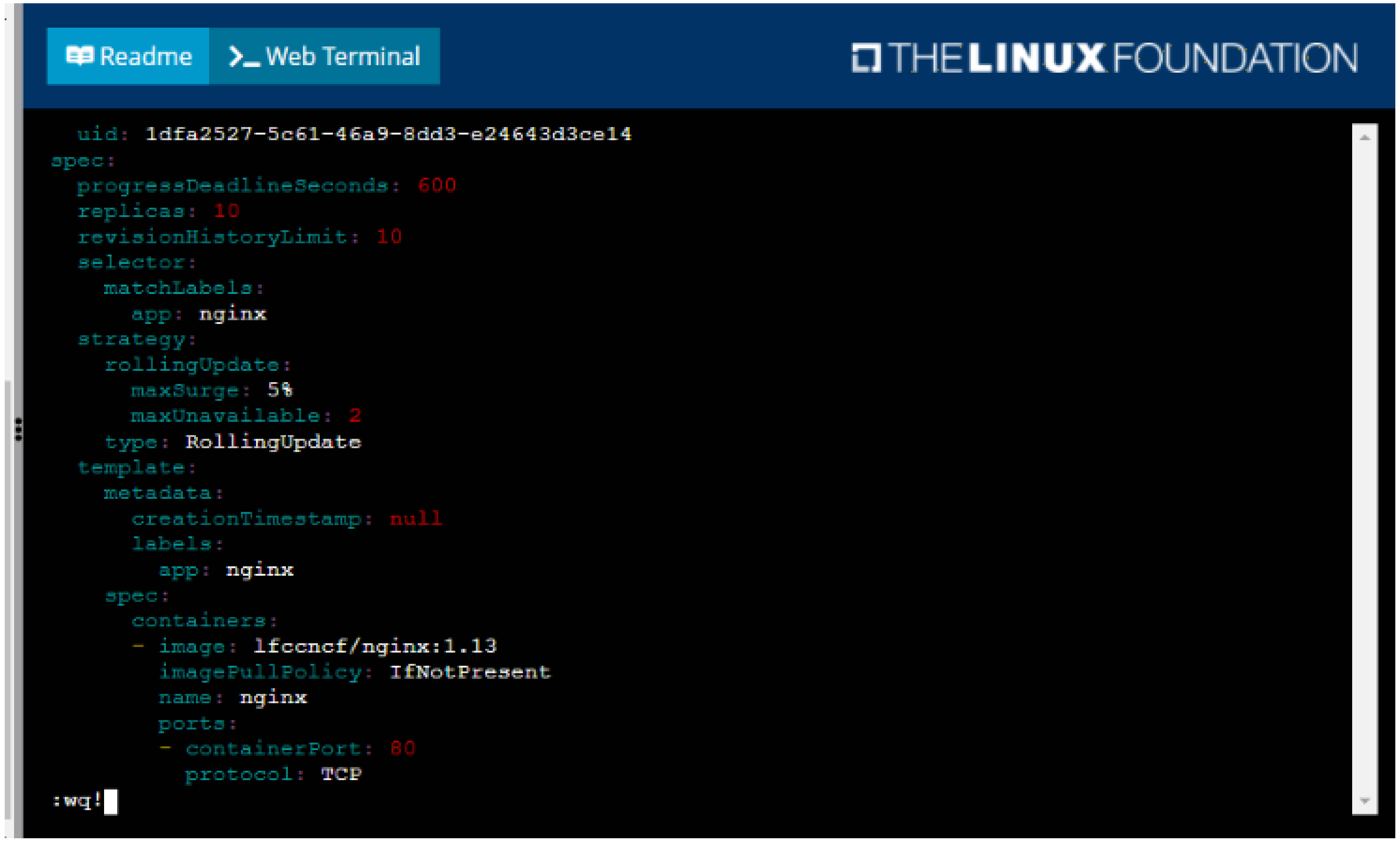
Options:

A- Explanation:

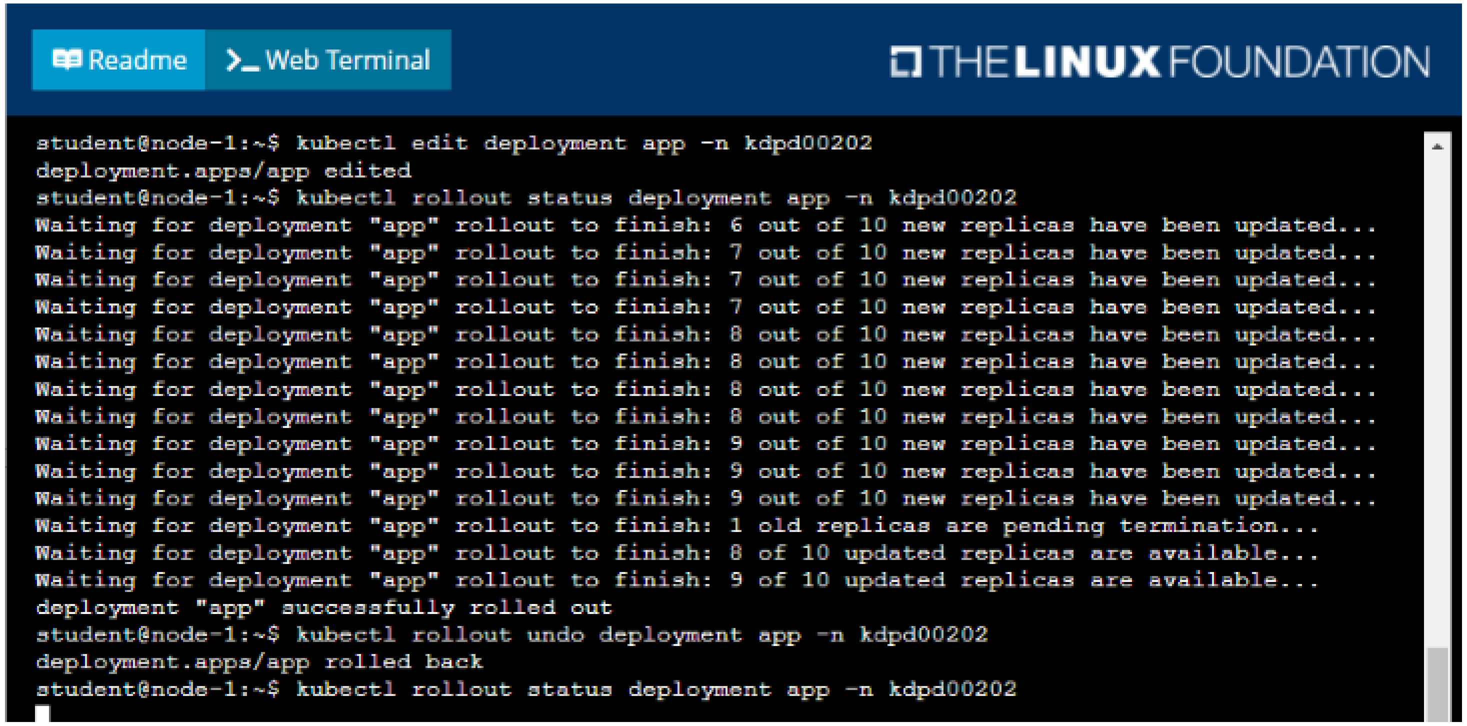
Solution:



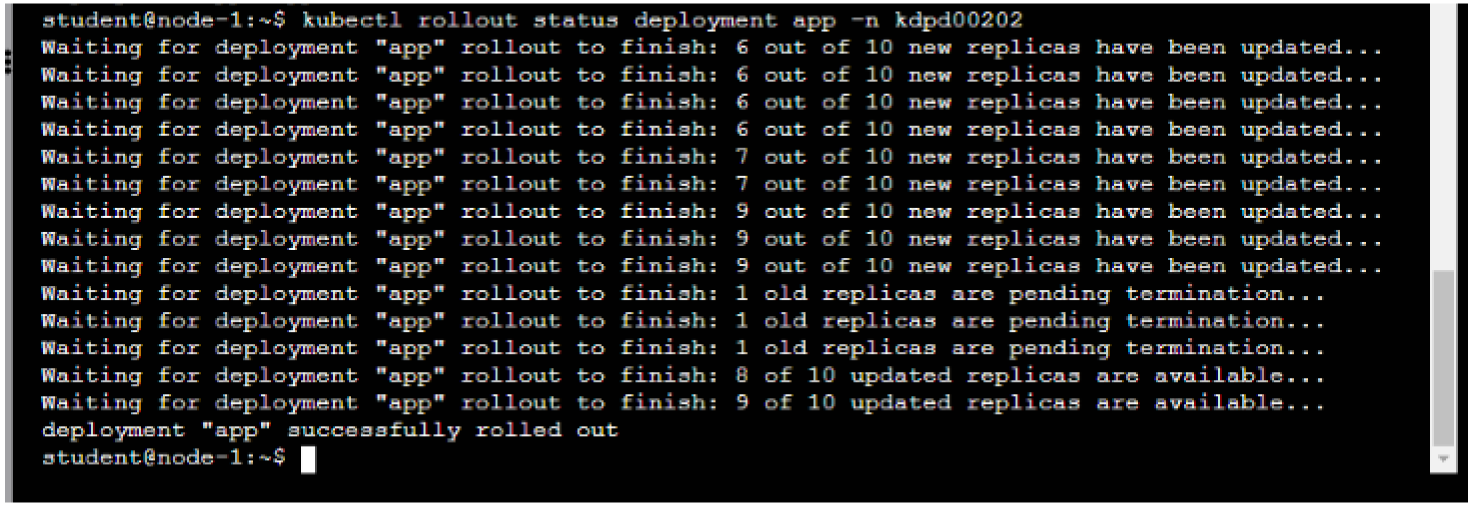
```
student@node-1:~$ kubectl edit deployment app -n kdpd00202
```



```
  uid: 1dfa2527-5c61-46a9-8dd3-e24643d3ce14
spec:
  progressDeadlineSeconds: 600
  replicas: 10
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app: nginx
  strategy:
    rollingUpdate:
      maxSurge: 5%
      maxUnavailable: 2
    type: RollingUpdate
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: nginx
    spec:
      containers:
      - image: lfccncf/nginx:1.13
        imagePullPolicy: IfNotPresent
        name: nginx
        ports:
        - containerPort: 80
          protocol: TCP
:wq!
```



```
student@node-1:~$ kubectl edit deployment app -n kdpd00202
deployment.apps/app edited
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: 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: 7 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 8 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 8 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 8 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 8 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: 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: 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:~$ kubectl rollout undo deployment app -n kdpd00202
deployment.apps/app rolled back
student@node-1:~$ kubectl rollout status deployment app -n kdpd00202
```



```
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: 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:~$
```

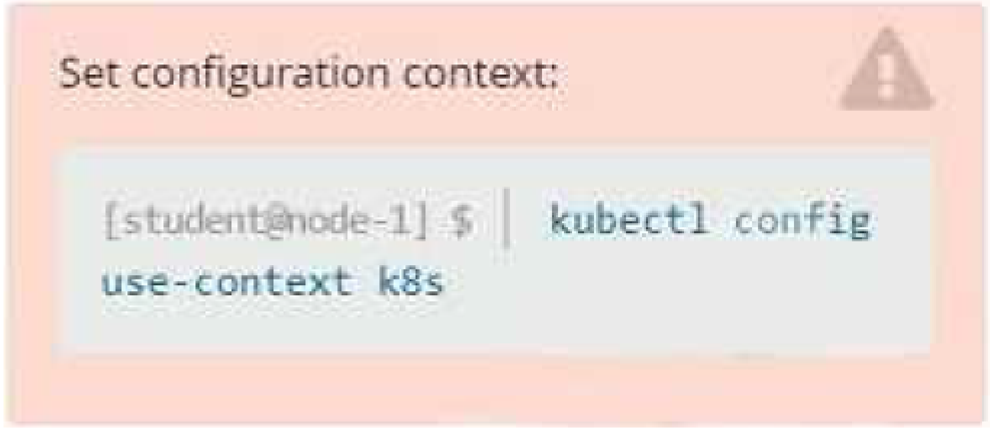
Answer:

A

Question 6

Question Type: MultipleChoice

Refer to 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

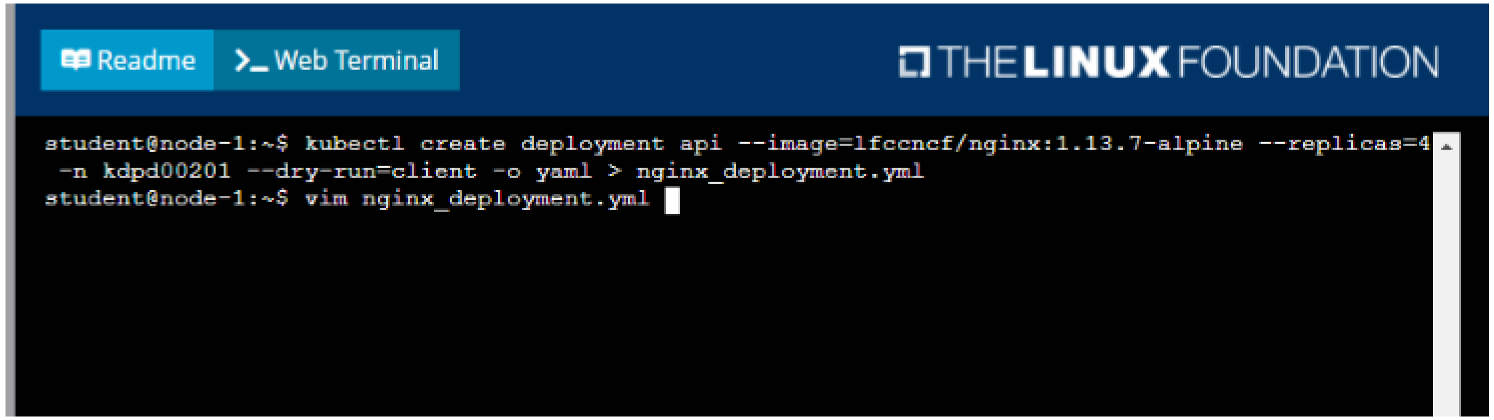
* Configure the pod with a container image of lfccncf/nginx:1.13.7

* Set an environment variable of NGINX__PORT=8080 and also expose that port for the container above

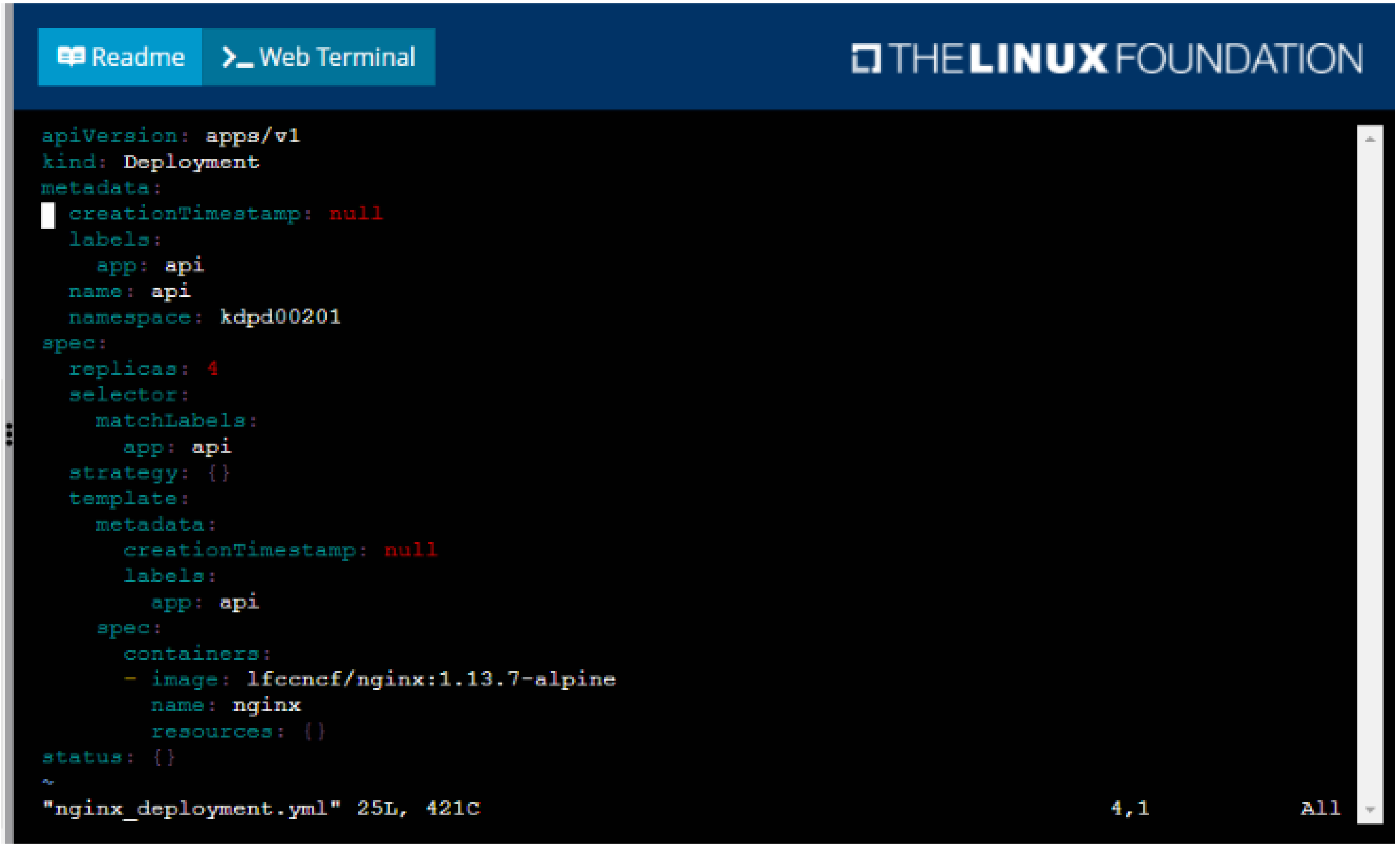
Options:

A- Explanation:

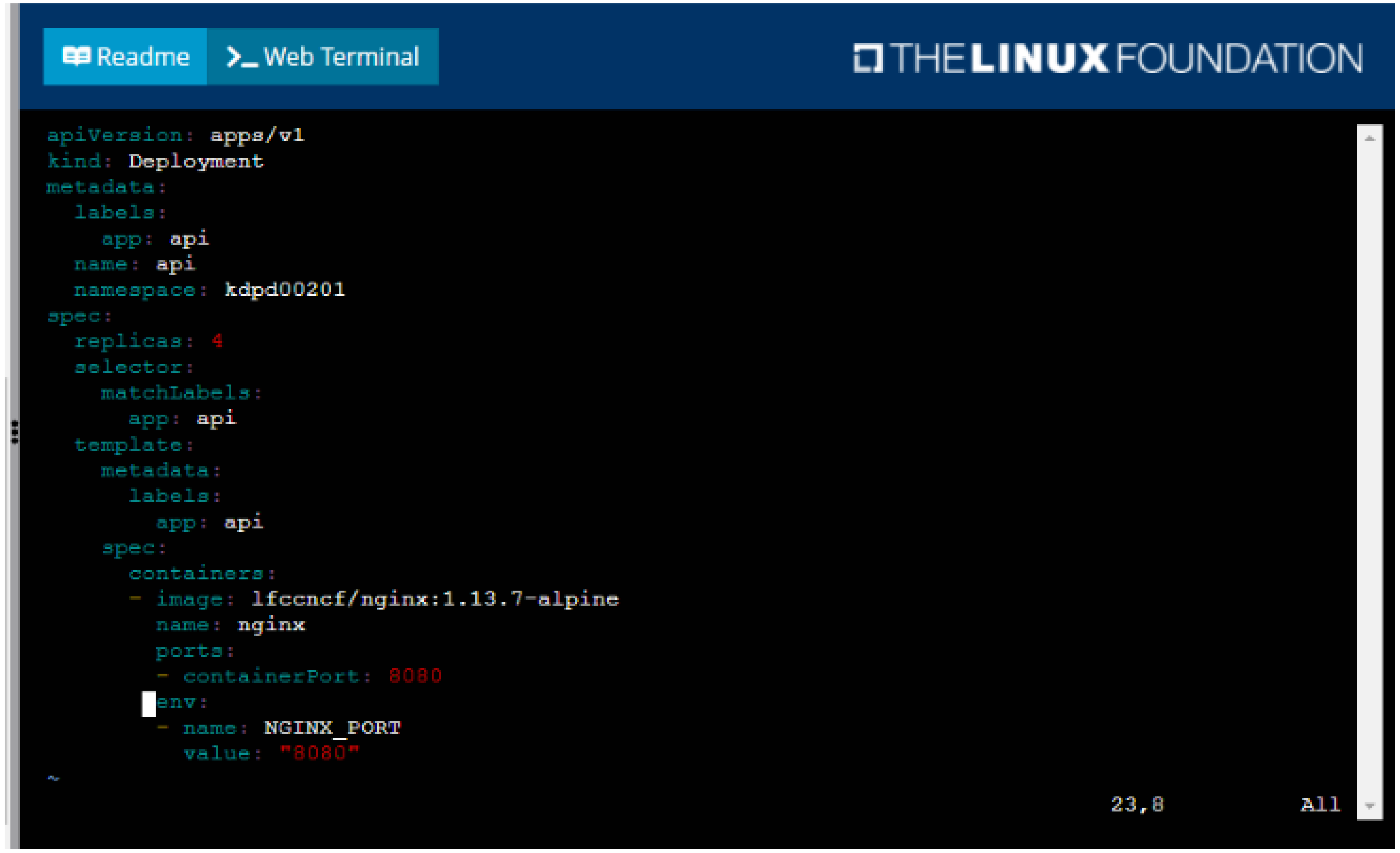
Solution:



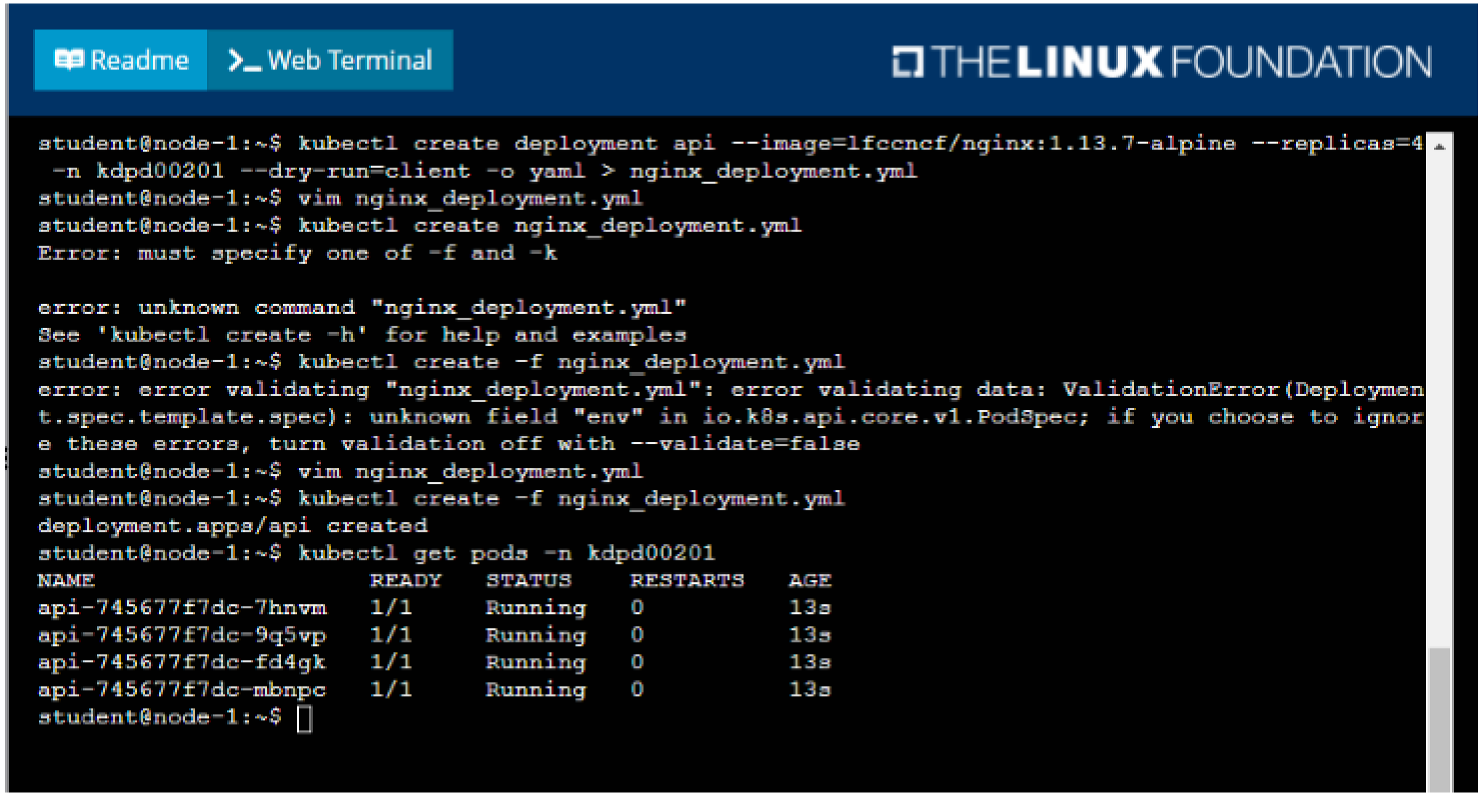
```
student@node-1:~$ kubectl create deployment api --image=lfccncf/nginx:1.13.7-alpine --replicas=4
 -n kdpd00201 --dry-run=client -o yaml > nginx_deployment.yml
student@node-1:~$ vim nginx_deployment.yml
```



```
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: lfccncf/nginx:1.13.7-alpine
        name: nginx
        resources: {}
status: {}
~
"nginx_deployment.yml" 25L, 421C                                4,1           All
```



```
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: lfccncf/nginx:1.13.7-alpine
        name: nginx
        ports:
        - containerPort: 8080
        env:
        - name: NGINX_PORT
          value: "8080"
~
                                                                              23,8          All
```



```
student@node-1:~$ kubectl create deployment api --image=lfccncf/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(Deploymen
t.spec.template.spec): unknown field "env" in io.k8s.api.core.v1.PodSpec; if you choose to ignor
e 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-7hnvm       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:~$
```

Answer:

A

Question 7

Question Type: **MultipleChoice**

Context

Anytime a team needs to run a container on Kubernetes they will need to define a pod within which to run the container.

Task

Please complete the following:

* Create a YAML formatted pod manifest

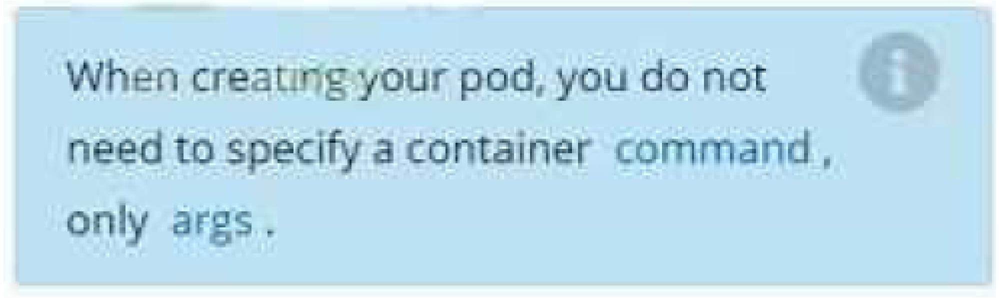
/opt/KDPD00101/podl.yml to create a pod named app1 that runs a container named app1cont using image Ifccncf/arg-output

with these command line arguments: -lines 56 -F

* Create the pod with the kubect1 command using the YAML file created in the previous step

* When the pod is running display summary data about the pod in JSON format using the kubect1 command and redirect the output to a file named /opt/KDPD00101/out1.json

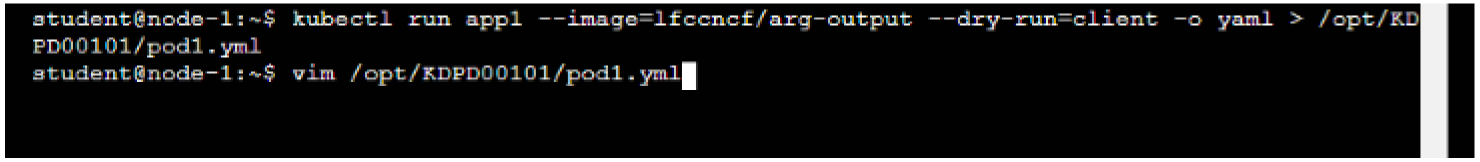
* All of the files you need to work with have been created, empty, for your convenience

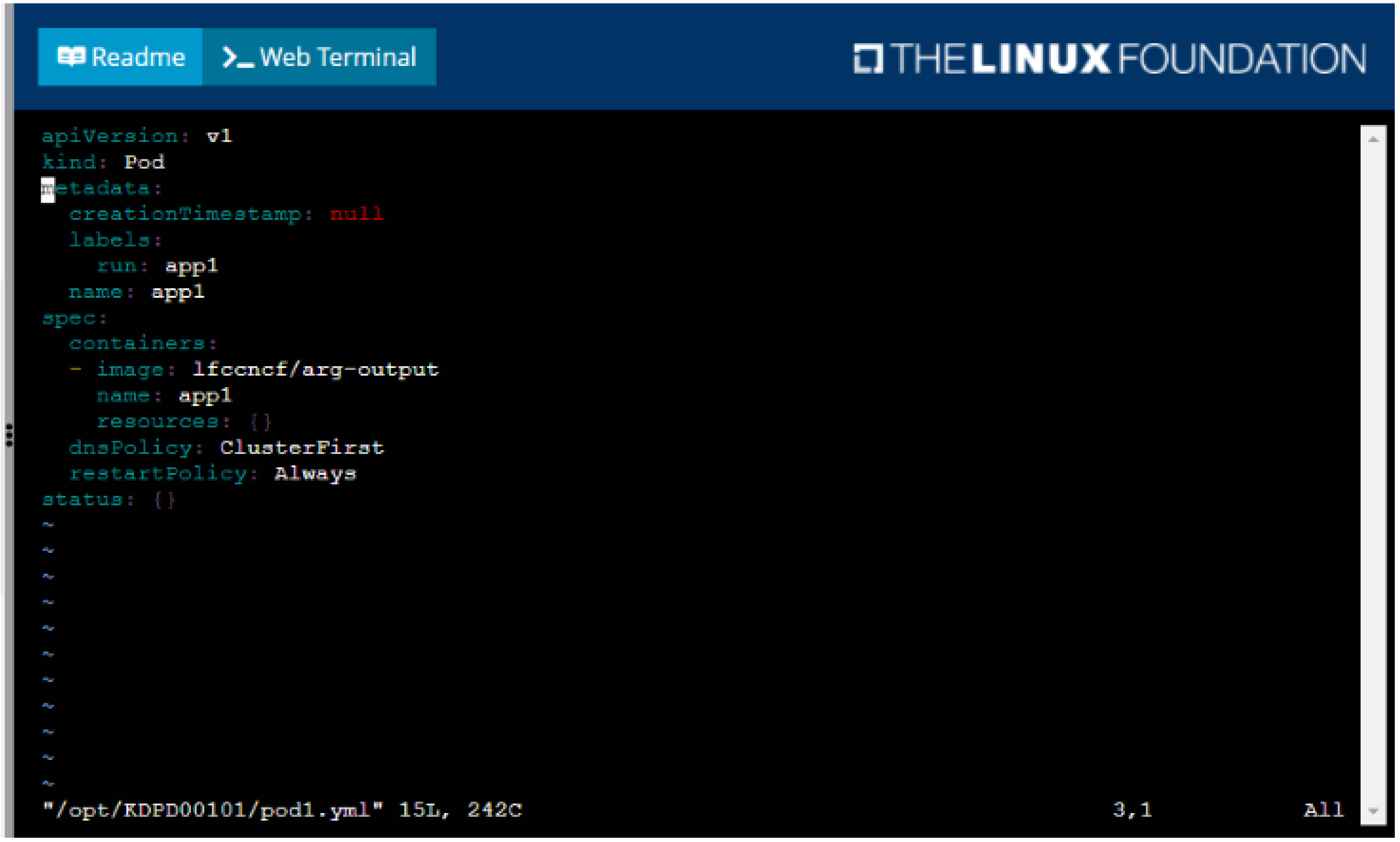

When creating your pod, you do not need to specify a container command, only args.

Options:

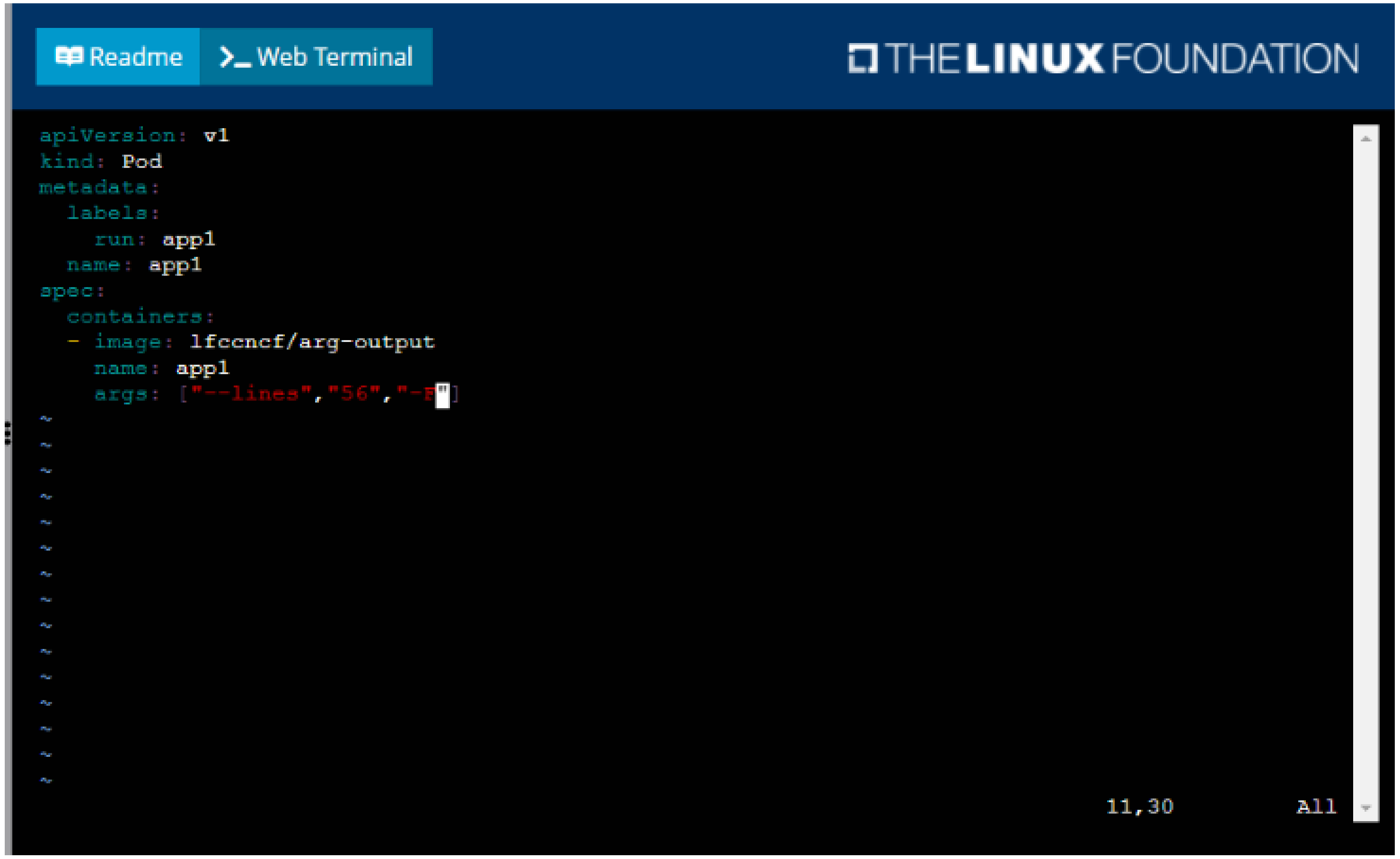
A- Explanation:

Solution:

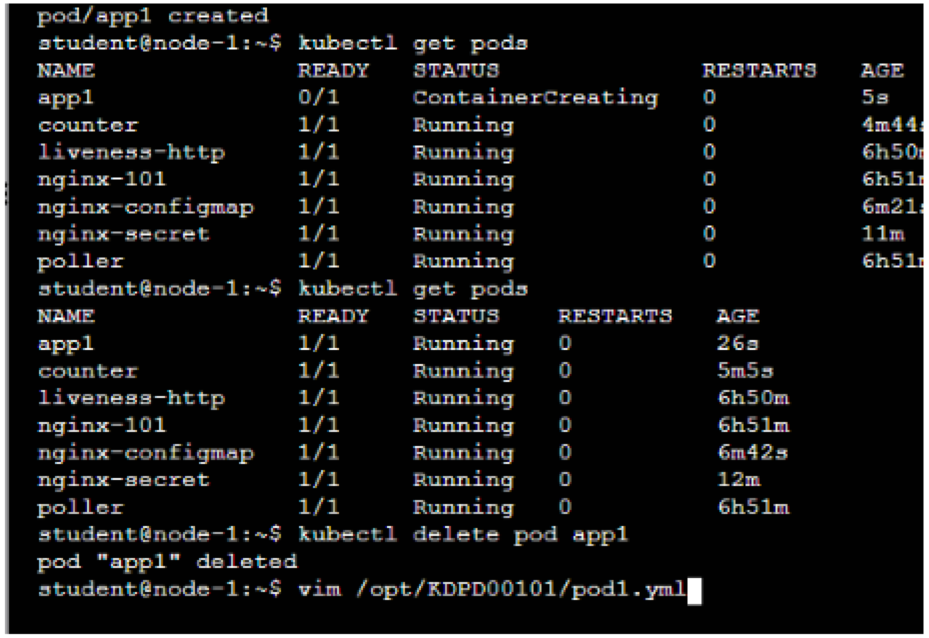

```
student@node-1:~$ kubectl run app1 --image=lfccncf/arg-output --dry-run=client -o yaml > /opt/KD
PD00101/pod1.yml
student@node-1:~$ vim /opt/KDPD00101/pod1.yml
```



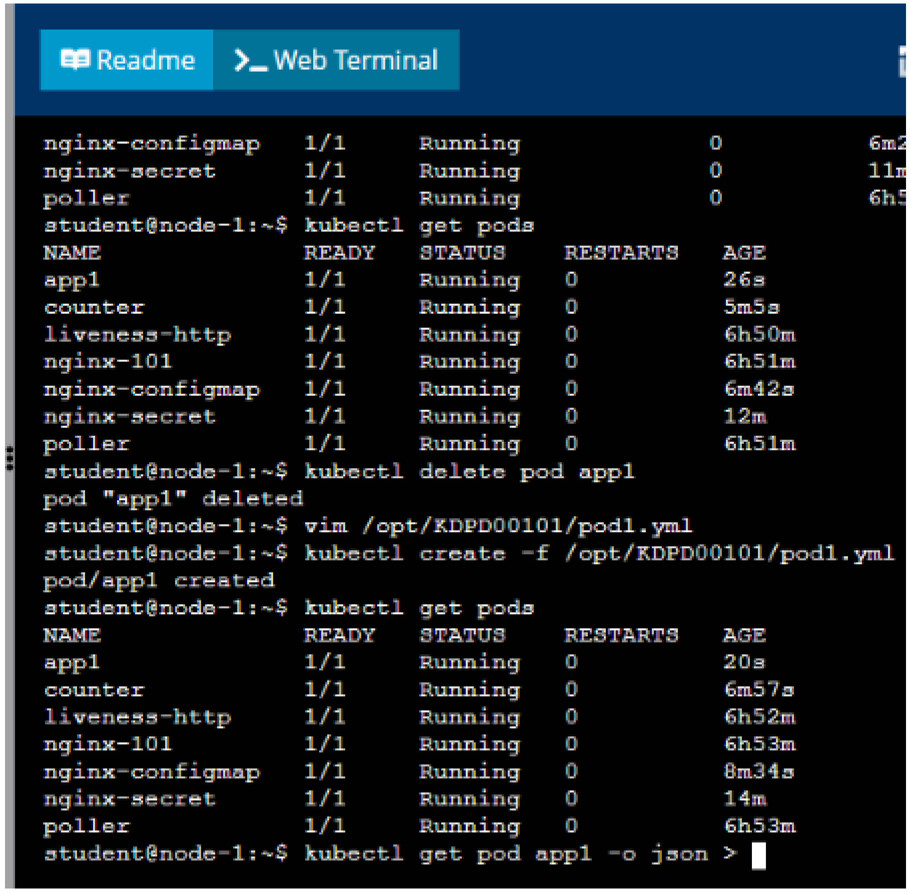
```
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: app1
  name: app1
spec:
  containers:
  - image: lfccncf/arg-output
    name: app1
    resources: {}
  dnsPolicy: ClusterFirst
  restartPolicy: Always
status: {}
~
~
~
~
~
~
~
~
~
~
~
"/opt/KDPD00101/pod1.yml" 15L, 242C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



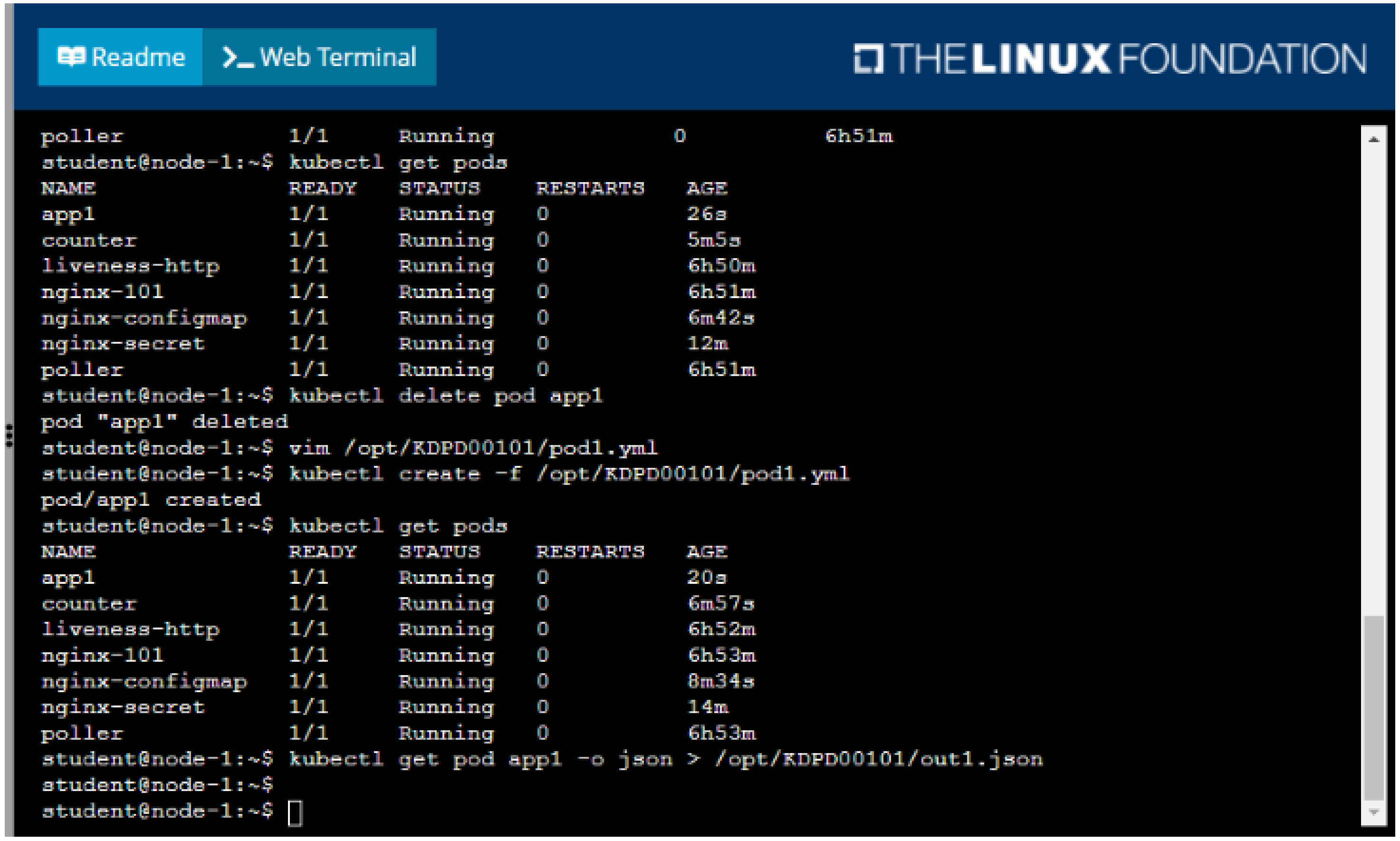
```
apiVersion: v1
kind: Pod
metadata:
  labels:
    run: app1
  name: app1
spec:
  containers:
  - image: lfccncf/arg-output
    name: app1
    args: ["--lines","56","-F"]
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
                                                                       11,30          All
```



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



```
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 >
```



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

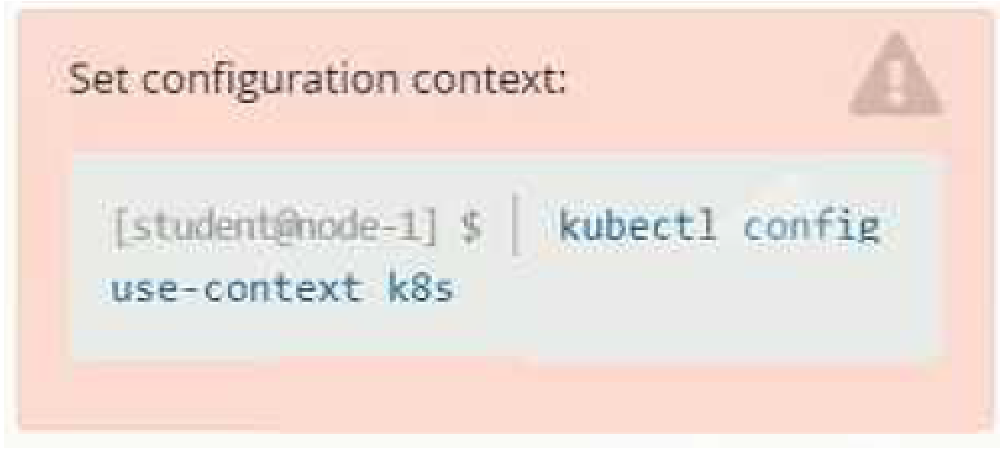
Answer:

A

Question 8

Question Type: MultipleChoice

Refer to Exhibit.


Set configuration context:

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

Context

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

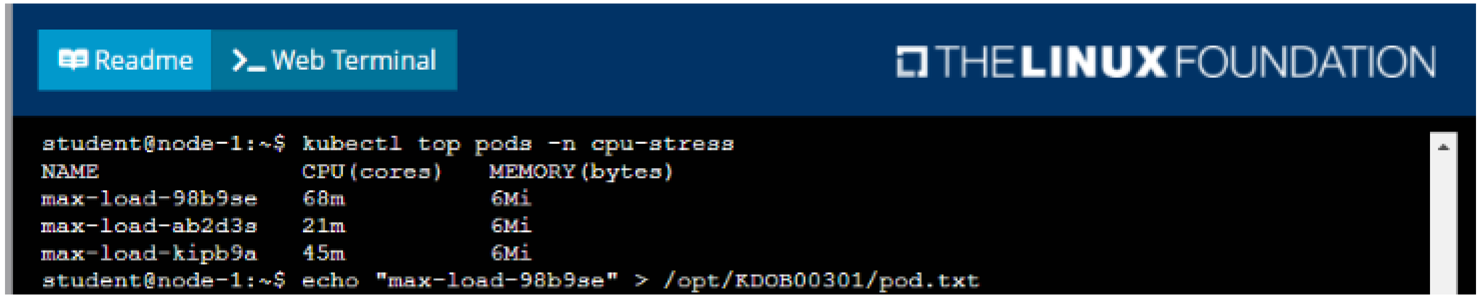
Task

* From the pods running in namespace cpu-stress , write the name only of the pod that is consuming the most CPU to file /opt/KDOBG030l/pod.txt, which has already been created.

Options:

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
```

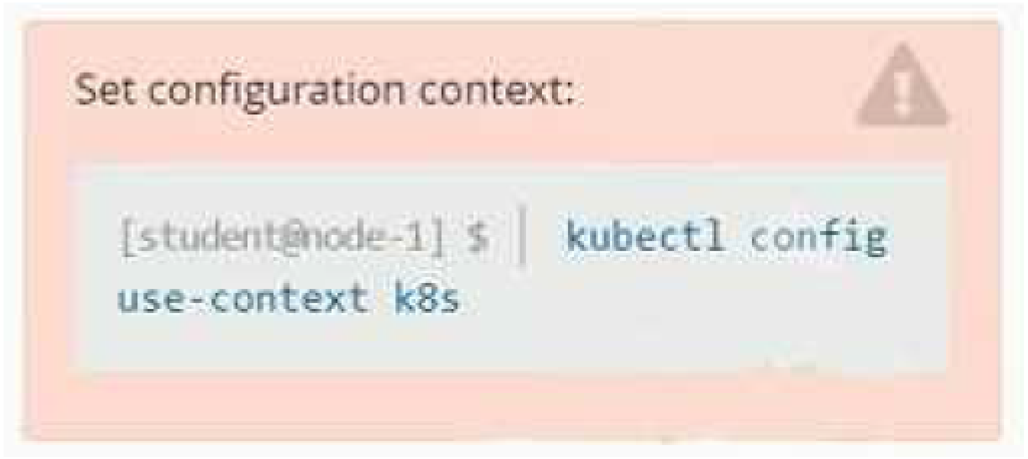
Answer:

A

Question 9

Question Type: **MultipleChoice**

Refer to Exhibit.


Set configuration context:

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

Set Configuration Context:

[student@node-1] $ | kubectl

Config use-context k8s

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/KDOB0020l/log_Output.txt, which has already been created

Options:

A- Explanation:

Solution:

To deploy the counter pod to the cluster using the provided YAML spec file, you can use the kubectl apply command. The apply command creates and updates resources in a cluster.

kubectl apply -f /opt/KDOB00201/counter.yaml

This command will create the pod in the cluster. You can use the kubectl get pods command to check the status of the pod and ensure that it is running.

kubectl get pods

To retrieve all currently available application logs from the running pod and store them in the file /opt/KDOB0020l/log_Output.txt, you can use the kubectl logs command. The logs command retrieves logs from a container in a pod.

kubectl logs -f > /opt/KDOB0020l/log_Output.txt

Replace with the name of the pod.

You can also use -f option to stream the logs.

kubectl logs -f > /opt/KDOB0020l/log_Output.txt &

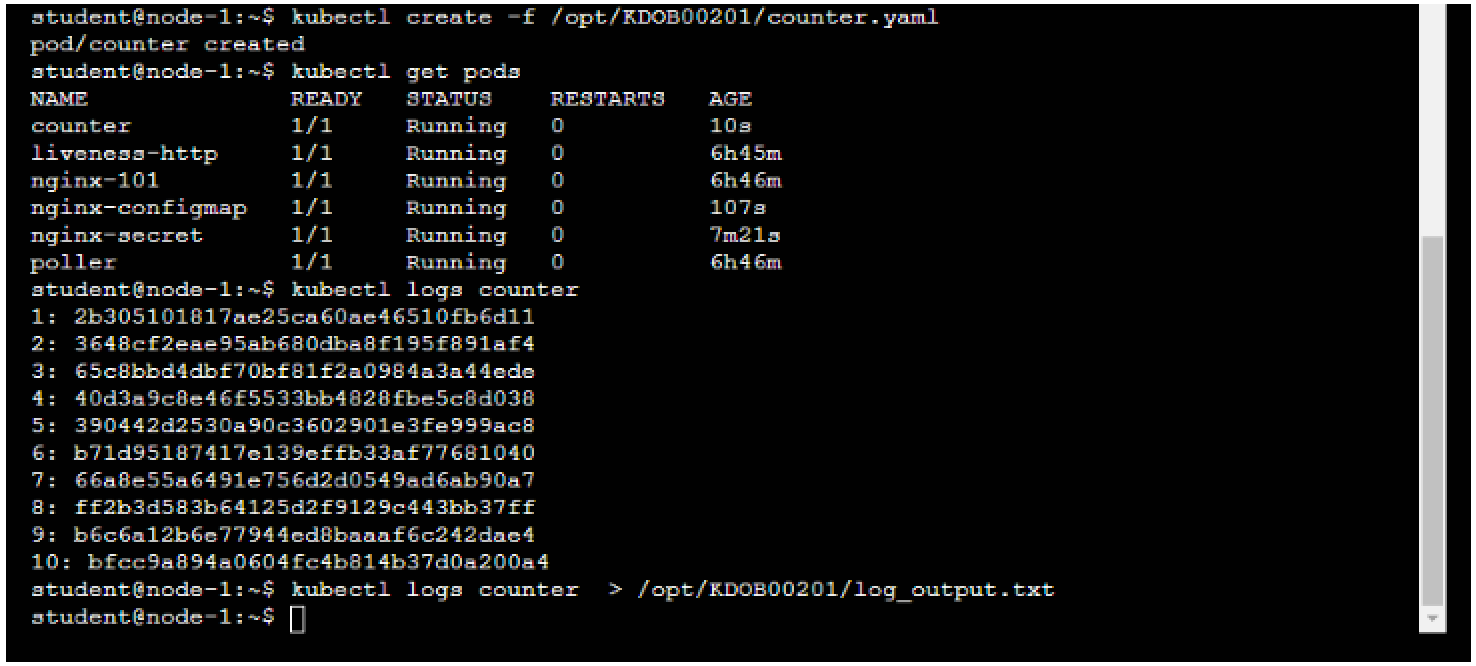
This command will retrieve the logs from the pod and write them to the /opt/KDOB0020l/log_Output.txt file.

Please note that the above command will retrieve all logs from the pod, including previous logs. If you want to retrieve only the new logs that are generated after running the command, you can add the --since flag to the kubectl logs command and specify a duration, for example --since=24h for logs generated in the last 24 hours.

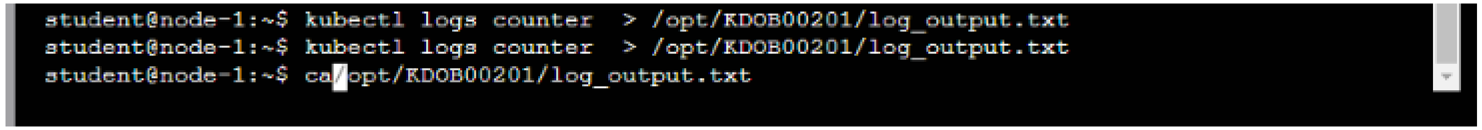
Also, please note that, if the pod has multiple containers, you need to specify the container name using -c option.

kubectl logs -f -c <container-name> > /opt/KDOB0020l/log_Output.txt

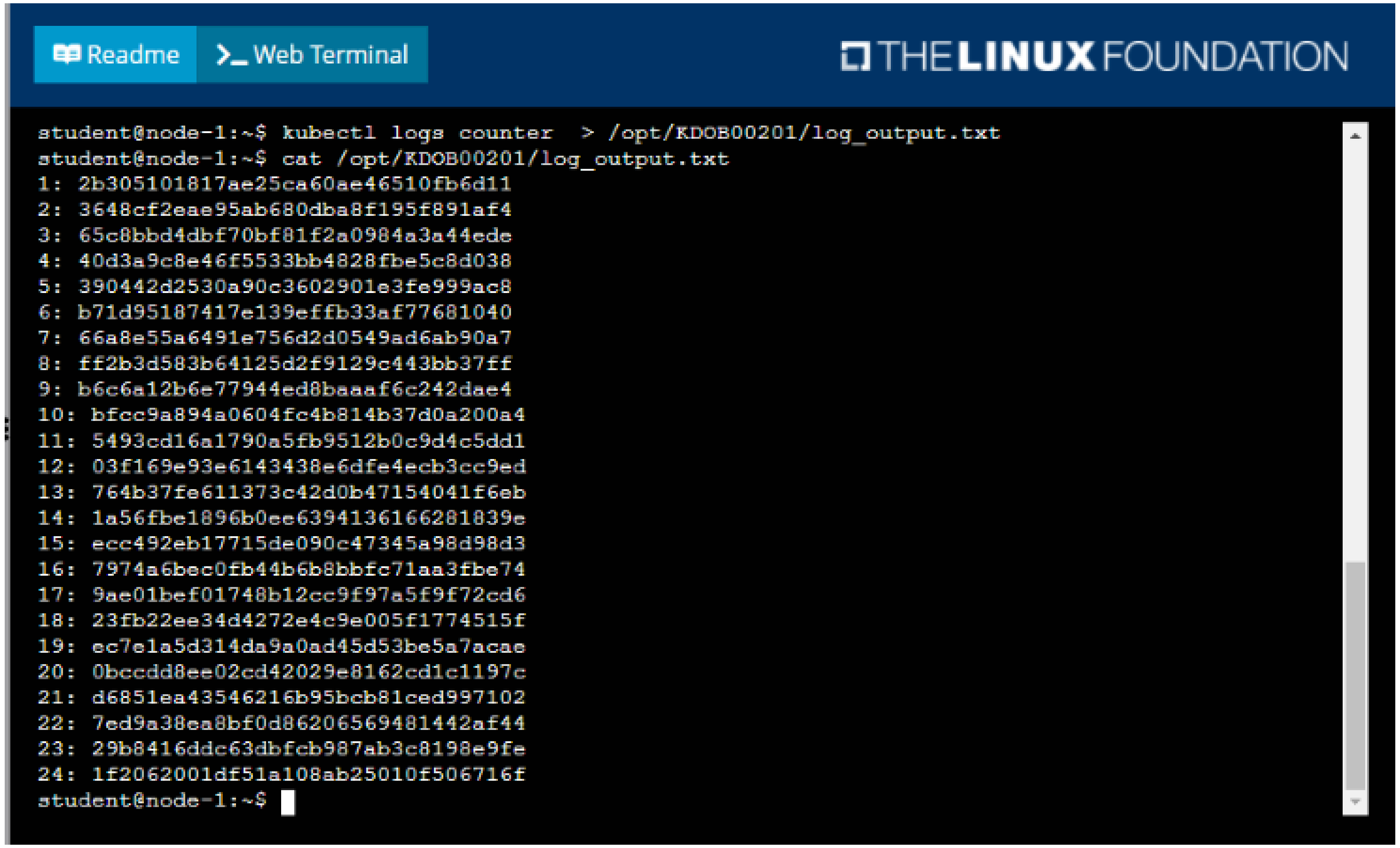
The above command will redirect the logs of the specified container to the file.



```
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:~$ ca/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
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: 1a56fbe1896b0ee6394136166281839e
15: ecc492eb17715de090c47345a98d98d3
16: 7974a6bec0fb44b6b8bbfc71aa3fbe74
17: 9ae01bef01748b12cc9f97a5f9f72cd6
18: 23fb22ee34d4272e4c9e005f1774515f
19: ec7e1a5d314da9a0ad45d53be5a7acae
20: 0bccdd8ee02cd42029e8162cd1c1197c
21: d6851ea43546216b95bcb81ced997102
22: 7ed9a38ea8bf0d86206569481442af44
23: 29b8416ddc63dbfcb987ab3c8198e9fe
24: 1f2062001df51a108ab25010f506716f
student@node-1:~$
```

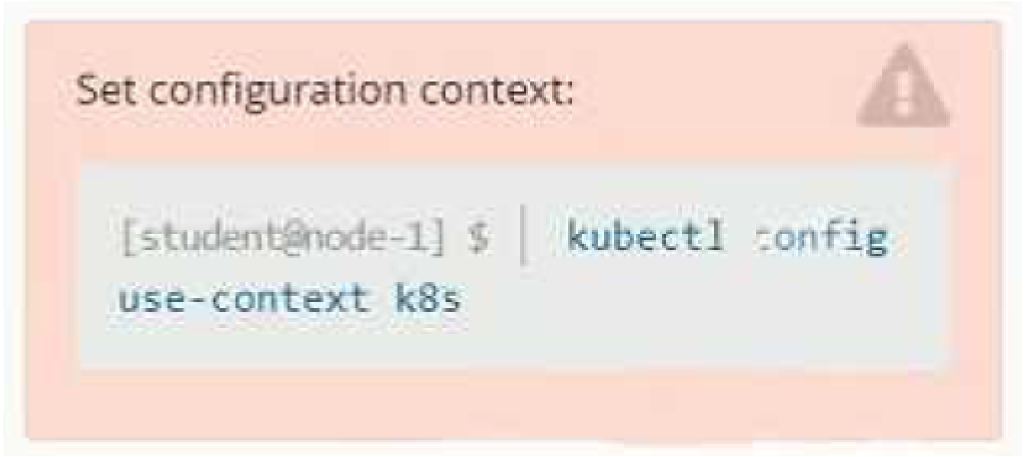
Answer:

A

Question 10

Question Type: **MultipleChoice**

Refer to Exhibit.



Set configuration context:

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

Set Configuration Context:

[student@node-1] $ | kubectl

Config use-context k8s

Context

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

Task

The desired behavior is to have Kubemetes restart the pod when an endpoint returns an HTTP 500 on the /healthz endpoint. The service, probe-pod, should never send traffic to the pod while it is failing. Please complete the following:

* The application has an endpoint, /started, that will indicate if it can accept traffic by returning an HTTP 200. If the endpoint returns an HTTP 500, the application has not yet finished initialization.

* The application has another endpoint /healthz that will indicate if the application is still working as expected by returning an HTTP 200. If the endpoint returns an HTTP 500 the application is no longer responsive.

* Configure the probe-pod pod provided to use these endpoints

* The probes should use port 8080

Options:

A- Explanation:

Solution:

To have Kubernetes automatically restart a pod when an endpoint returns an HTTP 500 on the /healthz endpoint, you will need to configure liveness and readiness probes on the pod.

First, you will need to create a livenessProbe and a readinessProbe in the pod's definition yaml file. The livenessProbe will check the /healthz endpoint, and if it returns an HTTP 500, the pod will be restarted. The readinessProbe will check the /started endpoint, and if it returns an HTTP 500, the pod will not receive traffic.

Here's an example of how you can configure the liveness and readiness probes in the pod definition yaml file:

```
apiVersion: v1
kind: Pod
metadata:
```

```
name: probe-pod
spec:
containers:
- name: probe-pod
image: <image-name>
ports:
- containerPort: 8080
livenessProbe:
httpGet:
path: /healthz
port: 8080
initialDelaySeconds: 15
periodSeconds: 10
failureThreshold: 3
readinessProbe:
httpGet:
path: /started
port: 8080
initialDelaySeconds: 15
periodSeconds: 10
failureThreshold: 3
```

The httpGet specifies the endpoint to check and the port to use. The initialDelaySeconds is the amount of time the pod will wait before starting the probe. periodSeconds is the amount of time between each probe check, and the failureThreshold is the number of failed probes before the pod is considered unresponsive.

You can use kubectl to create the pod by running the following command:

```
kubectl apply -f <filename>.yaml
```

Once the pod is created, Kubernetes will start monitoring it using the configured liveness and readiness probes. If the /healthz endpoint returns an HTTP 500, the pod will be restarted. If the /started endpoint returns an HTTP 500, the pod will not receive traffic.
Please note that if the failure threshold is set to 3, it means that if the probe fails 3 times consecutively it will be considered as a failure.
The above configuration assumes that the application is running on port 8080 and the endpoints are available on the same port.

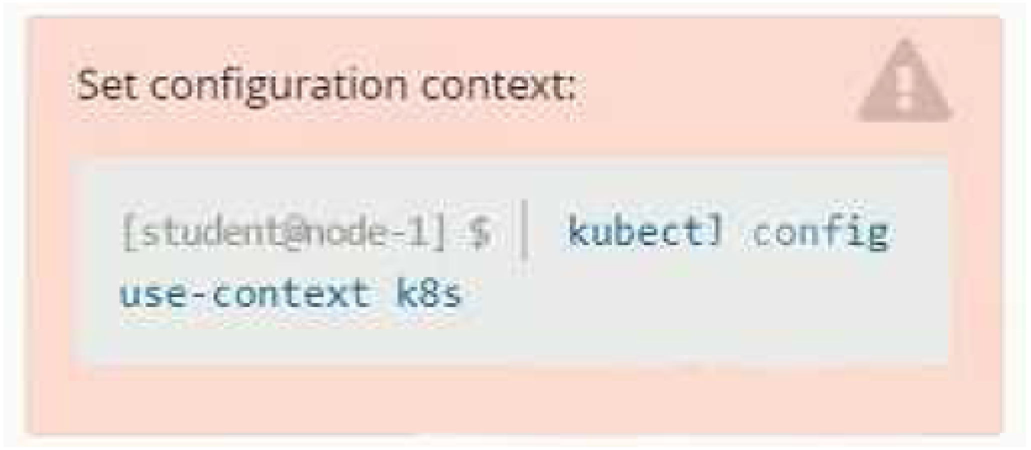
Answer:

A

Question 11

Question Type: **MultipleChoice**

Refer to Exhibit.


Set configuration context:

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

Context

Your application's namespace requires a specific service account to be used.

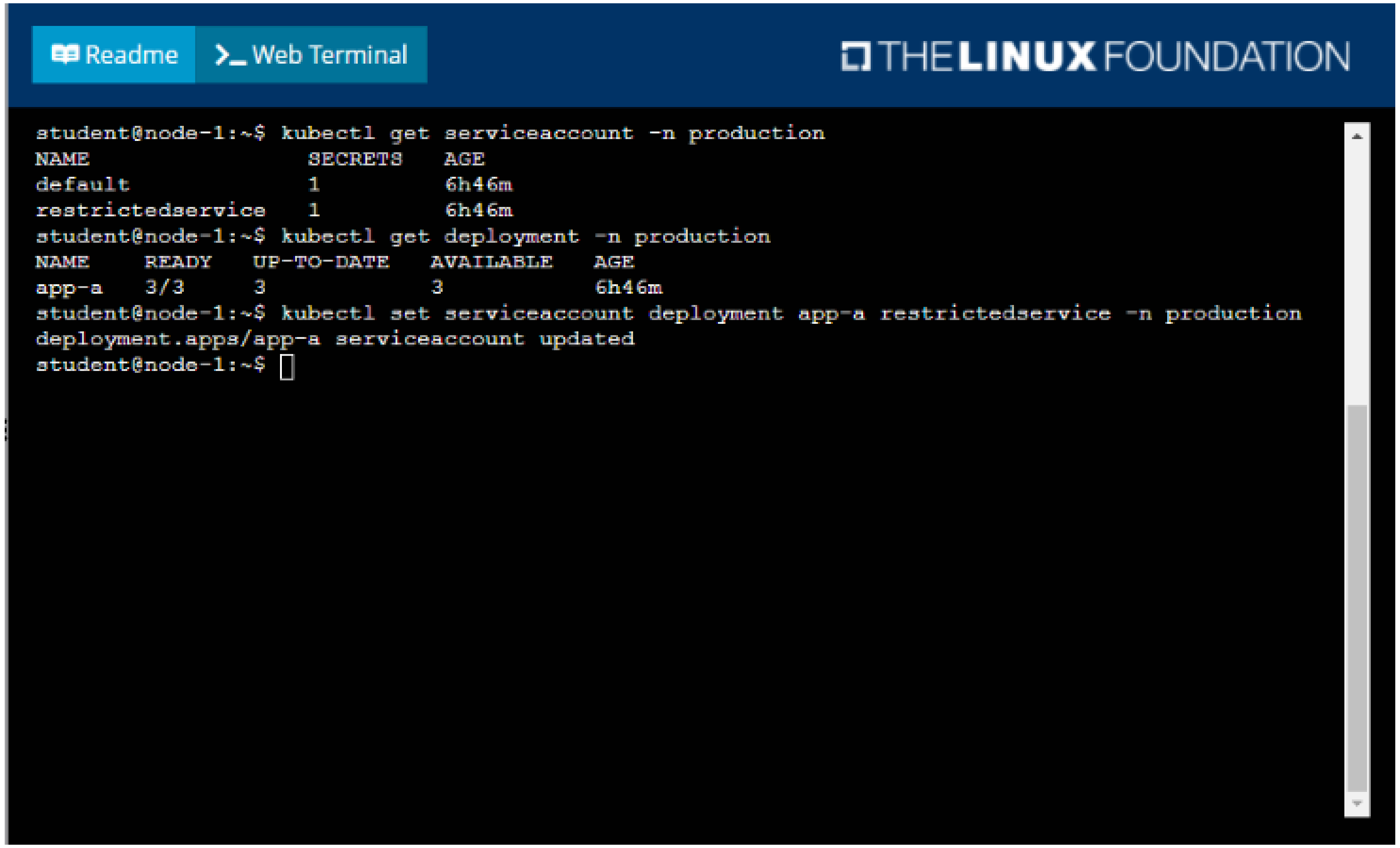
Task

Update the app-a deployment in the production namespace to run as the restrictedservice service account. The service account has already been created.

Options:

A- Explanation:

Solution:



```
student@node-1:~$ kubectl get serviceaccount -n production
NAME                 SECRETS   AGE
default              1         6h46m
restrictedservice    1         6h46m
student@node-1:~$ kubectl get deployment -n production
NAME    READY   UP-TO-DATE   AVAILABLE   AGE
app-a   3/3     3            3           6h46m
student@node-1:~$ kubectl set serviceaccount deployment app-a restrictedservice -n production
deployment.apps/app-a serviceaccount updated
student@node-1:~$ 
```

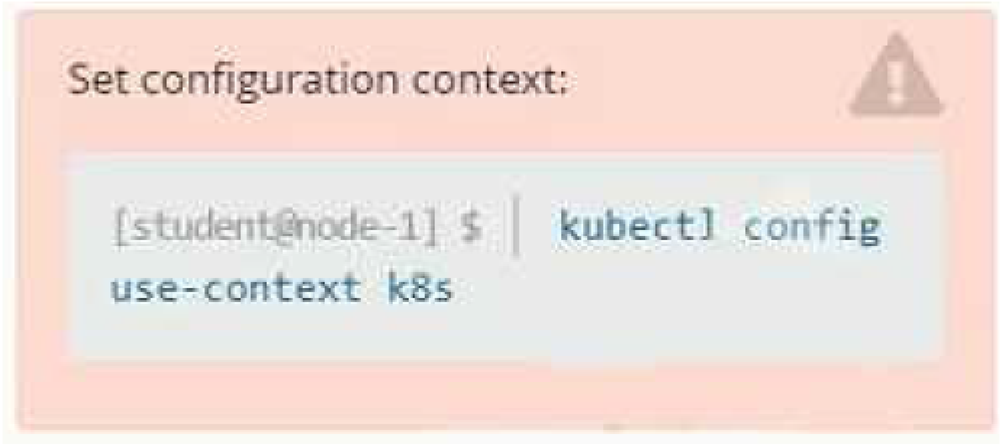
Answer:

A

Question 12

Question Type: MultipleChoice

Refer to 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

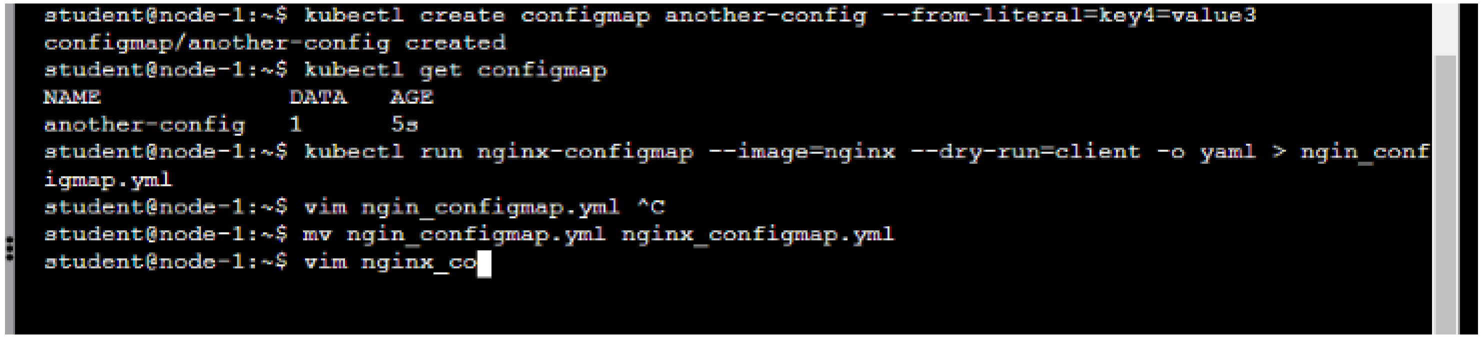
* start a pod named nginx-configmap containing a single container using the

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

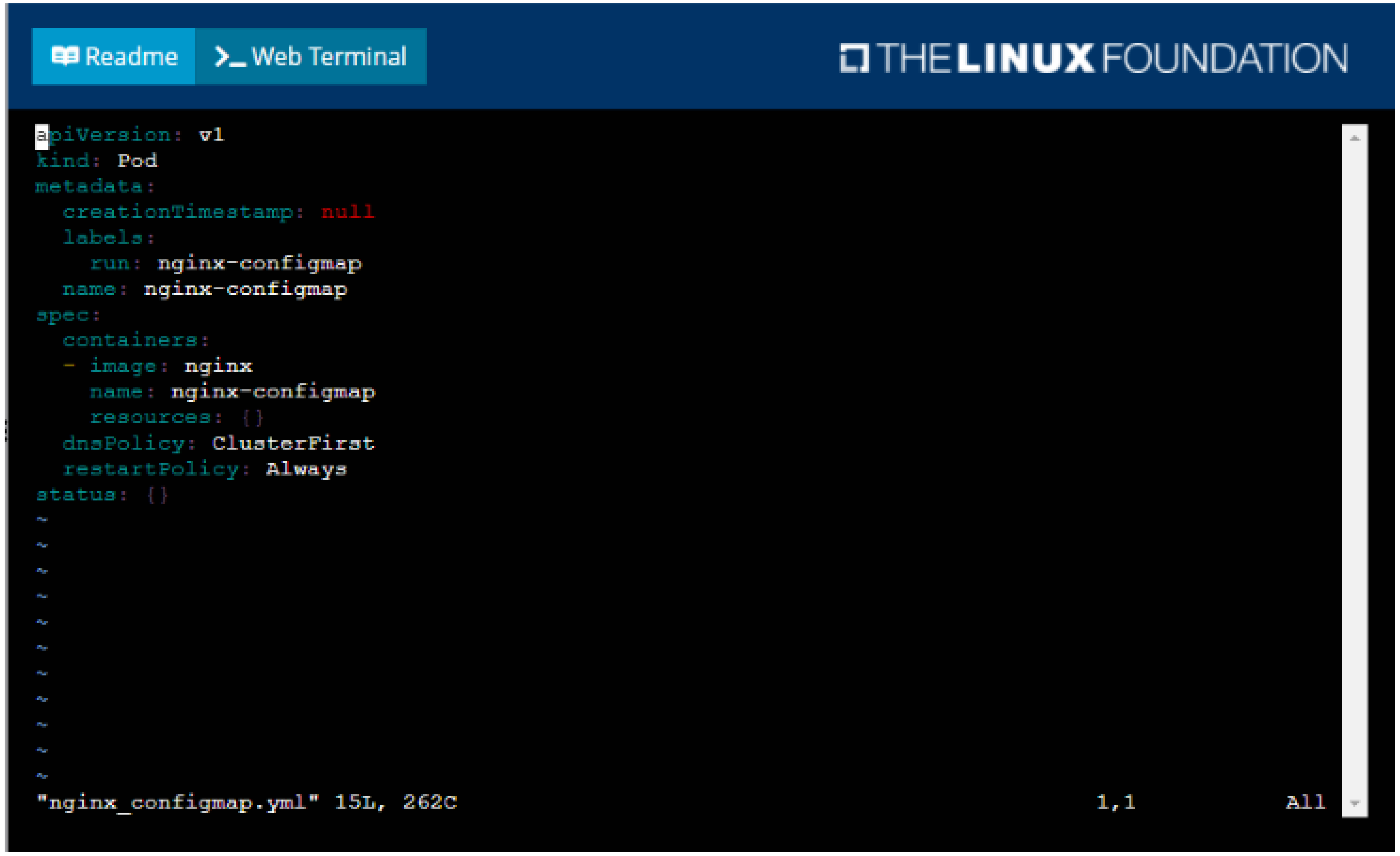
Options:

A- Explanation:

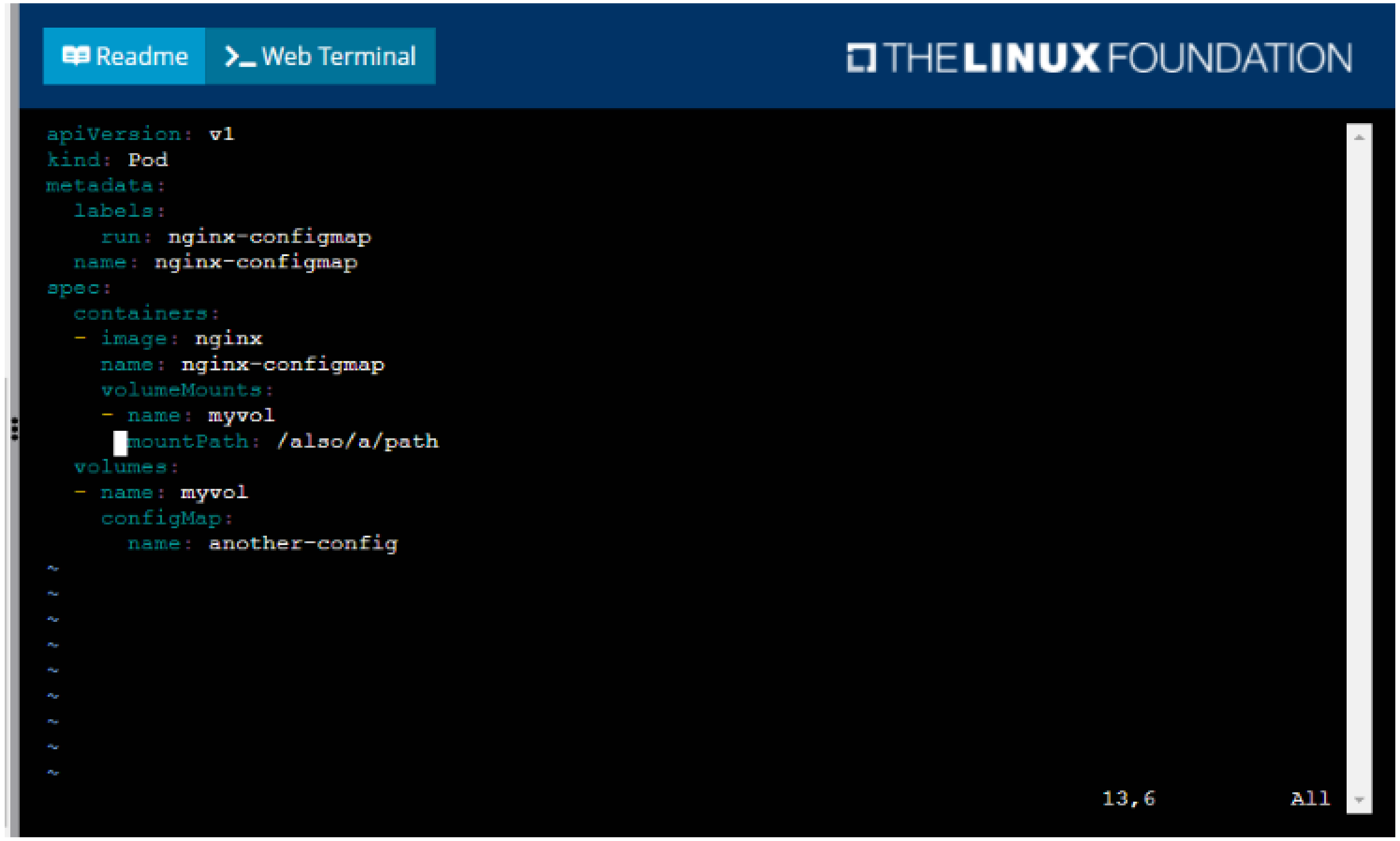
Solution:



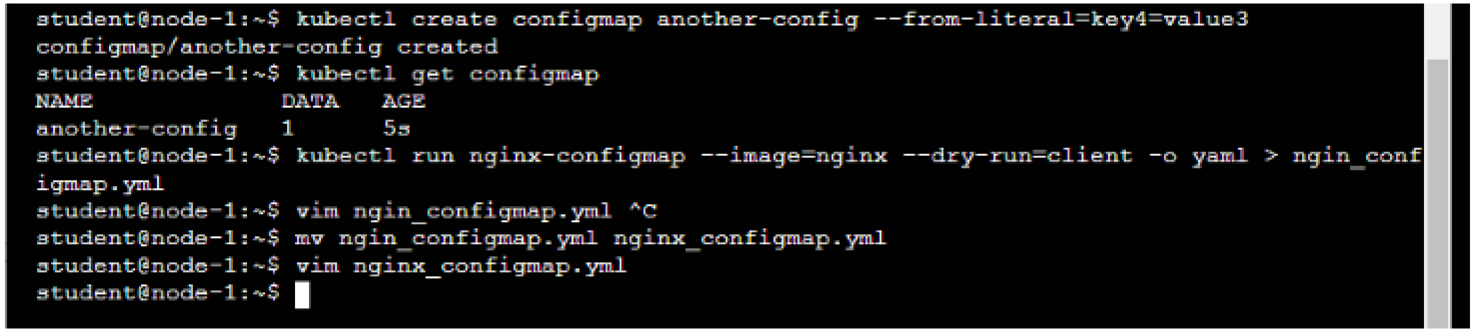
```
student@node-1:~$ kubectl create configmap another-config --from-literal=key4=value3
configmap/another-config created
student@node-1:~$ kubectl get configmap
NAME               DATA   AGE
another-config     1      5s
student@node-1:~$ kubectl run nginx-configmap --image=nginx --dry-run=client -o yaml > 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_co
```



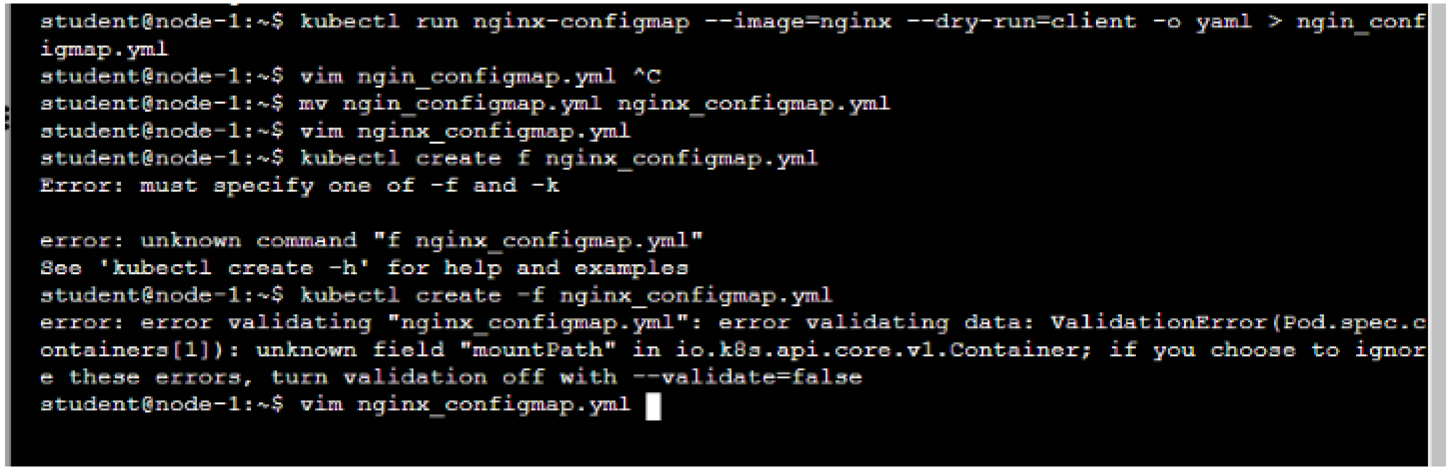
```
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: nginx-configmap
  name: nginx-configmap
spec:
  containers:
  - image: nginx
    name: nginx-configmap
    resources: {}
  dnsPolicy: ClusterFirst
  restartPolicy: Always
status: {}
~
~
~
~
~
~
~
~
~
~
~
"nginx_configmap.yml"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



```
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
```

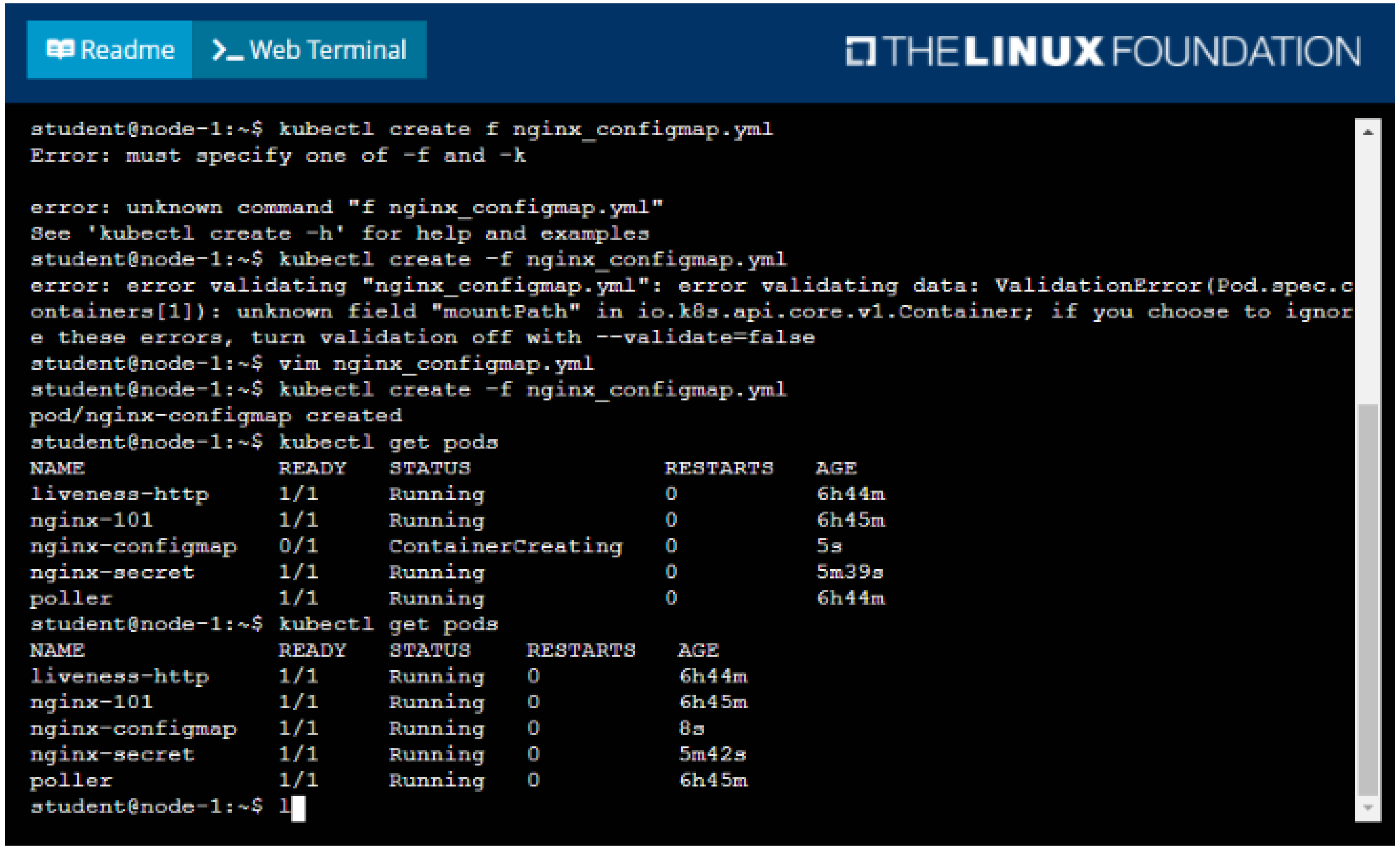


```
student@node-1:~$ kubectl create configmap another-config --from-literal=key4=value3
configmap/another-config created
student@node-1:~$ kubectl get configmap
NAME             DATA   AGE
another-config   1      5s
student@node-1:~$ kubectl run nginx-configmap --image=nginx --dry-run=client -o yaml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



```
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:~$ 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.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.yml 
```



```
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.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.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
```

Answer:

A