

Linux-Foundation

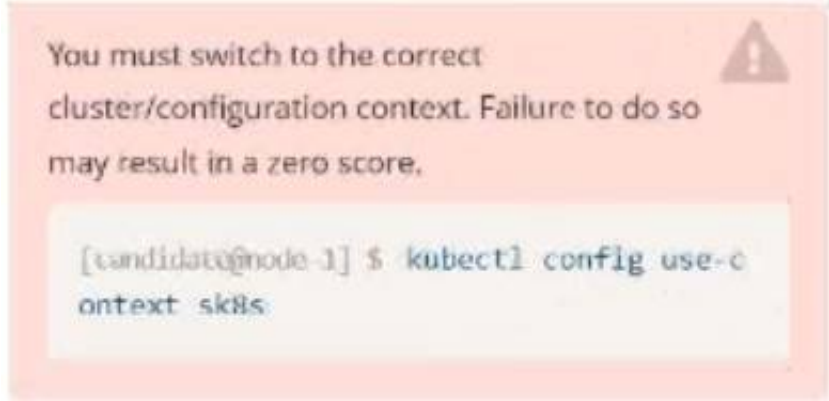
Exam Questions CKAD

Certified Kubernetes Application Developer (CKAD) Program



NEW QUESTION 1

Exhibit:

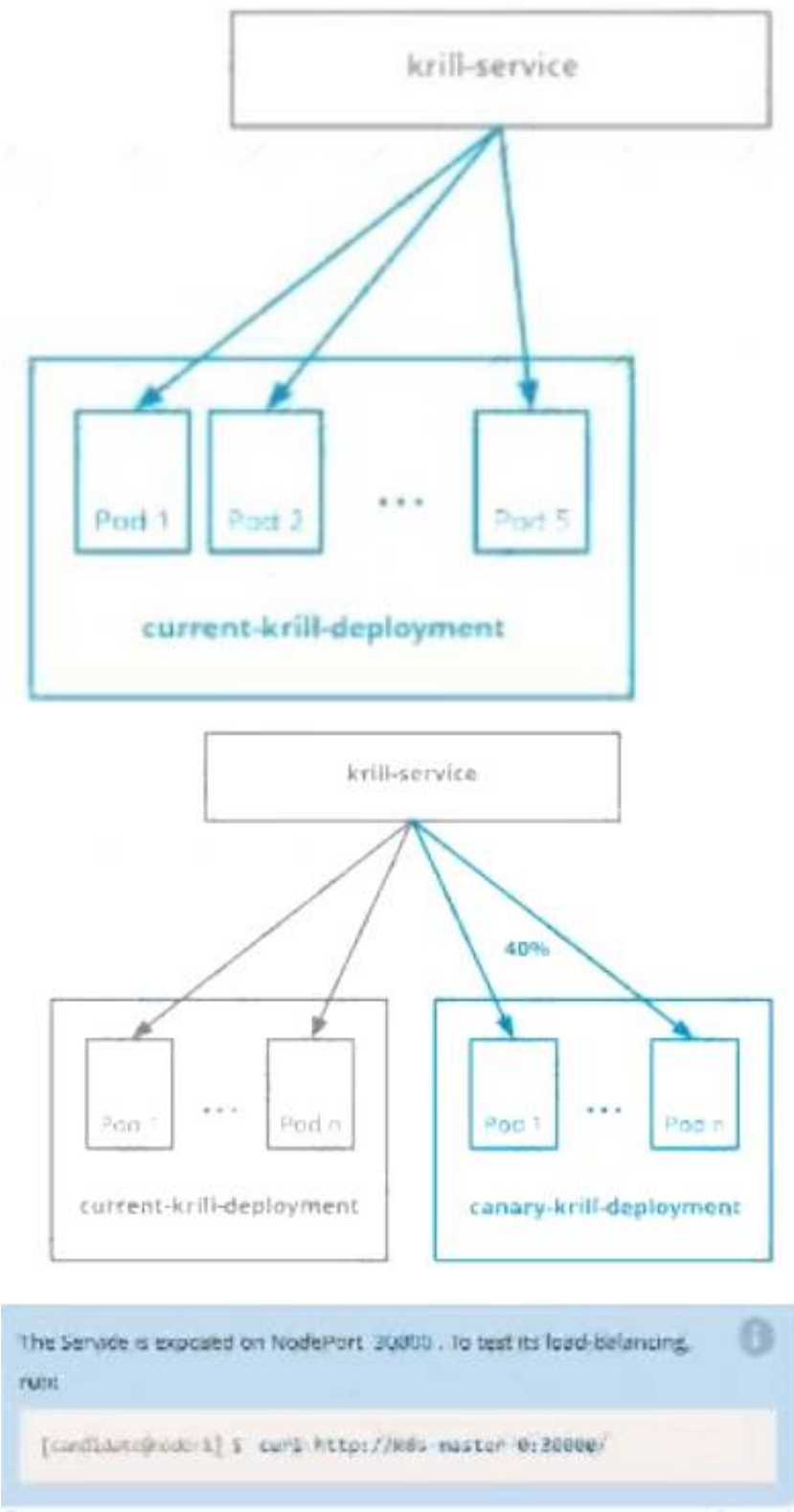


Context

You are asked to prepare a Canary deployment for testing a new application release.

Task:

A Service named krill-Service in the goshawk namespace points to 5 pod created by the Deployment named current-krill-deployment



- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:

```
candidate@node-1:~/humane-stork$ kubectl scale deploy canary-krill-deployment --replicas 4 -n goshawk
deployment.apps/canary-krill-deployment scaled
candidate@node-1:~/humane-stork$ kubectl get deploy -n goshawk
NAME                                READY  UP-TO-DATE  AVAILABLE  AGE
canary-krill-deployment             4/4    4           4          46s
current-krill-deployment            5/5    5           5          7h22m
candidate@node-1:~/humane-stork$ wget https://k8s.io/examples/
```

Text Description automatically generated

```
File Edit View Terminal Tabs Help
candidate@node-1:~/humane-storks$ wget https://k8s.io/examples/admin/resource/quota-pod.yaml
--2022-09-24 11:43:51-- https://k8s.io/examples/admin/resource/quota-pod.yaml
Resolving k8s.io (k8s.io)... 34.107.204.206, 2600:1901:0:26f3::
Connecting to k8s.io (k8s.io)|34.107.204.206|:443... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://kubernetes.io/examples/admin/resource/quota-pod.yaml [following]
--2022-09-24 11:43:52-- https://kubernetes.io/examples/admin/resource/quota-pod.yaml
Resolving kubernetes.io (kubernetes.io)... 147.75.40.148
Connecting to kubernetes.io (kubernetes.io)|147.75.40.148|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 90 [application/x-yaml]
Saving to: 'quota-pod.yaml'

quota-pod.yaml      100%[=====]          90  --.-KB/s   in 0s

2022-09-24 11:43:52 (15.0 MB/s) - 'quota-pod.yaml' saved [90/90]

candidate@node-1:~/humane-storks$ vim quota-pod.yaml

File Edit View Terminal Tabs Help

2022-09-24 11:43:52 (15.0 MB/s) - 'quota-pod.yaml' saved [90/90]

candidate@node-1:~/humane-storks$ vim quota-pod.yaml
candidate@node-1:~/humane-storks$ kubectl create -f quota-pod.yaml
resourcequota/pod-demo created
candidate@node-1:~/humane-storks$ kubectl get quota -n go
No resources found in go namespace.
candidate@node-1:~/humane-storks$ kubectl get quota -n goshawk
NAME      AGE   REQUEST   LIMIT
pod-demo  19s   pods: 9/10
candidate@node-1:~/humane-storks$ curl http://k8s-master-0:30000/
current-krill-deployment-fb7c7995c-kvtjr
app.kubernetes.io/name="current"
app.kubernetes.io/part-of="krill"
pod-template-hash="fb7c7995c"candidate@node-1:~/humane-storks$ curl http://k8s-master-0:30000/
current-krill-deployment-fb7c7995c-4whfm
app.kubernetes.io/name="current"
app.kubernetes.io/part-of="krill"
pod-template-hash="fb7c7995c"candidate@node-1:~/humane-storks$ curl http://k8s-master-0:30000/
canary-krill-deployment-5f78fd4786-dfk7l
app.kubernetes.io/name="canary"
app.kubernetes.io/part-of="krill"
pod-template-hash="5f78fd4786"candidate@node-1:~/humane-storks$ curl http://k8s-master-0:30000/
canary-krill-deployment-5f78fd4786-z5zrt
app.kubernetes.io/name="canary"
app.kubernetes.io/part-of="krill"
pod-template-hash="5f78fd4786"candidate@node-1:~/humane-storks$ curl http://k8s-master-0:30000/
canary-krill-deployment-5f78fd4786-2774b
app.kubernetes.io/name="canary"
app.kubernetes.io/part-of="krill"
pod-template-hash="5f78fd4786"candidate@node-1:~/humane-storks$
```

NEW QUESTION 2

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 Ifccncf/redis image with the 3.2 tag
- Expose port 6379


- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:

Readme
Web Terminal



```

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 3

Exhibit:



Context

You are tasked to create a secret and consume the secret in a pod using environment variables as follow:

Task

- Create a secret named another-secret with a key/value pair; key1/value4
- Start an nginx pod named nginx-secret using container image nginx, and add an environment variable exposing the value of the secret key key 1, using COOL_VARIABLE as the name for the environment variable inside the pod

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:


Solution:

```

student@node-1:~$ kubectl create secret generic some-secret --from-literal=key1=value4
secret/some-secret created
student@node-1:~$ kubectl get secret
NAME                                TYPE                                DATA   AGE
default-token-4kvr5                 kubernetes.io/service-account-token 3       2d11h
some-secret                         Opaque                             1       5s
student@node-1:~$ kubectl run nginx-secret --image=nginx --dry-run=client -o yaml > nginx_secret.yml
student@node-1:~$ vim nginx_secret.yml

```

Readme
Web Terminal




```

apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: nginx-secret
  name: nginx-secret
spec:
  containers:
  - image: nginx
    name: nginx-secret
    resources: {}
  dnsPolicy: ClusterFirst
  restartPolicy: Always
status: {}
~
~
~
~
~
~
~
~
~
~
"nginx_secret.yml" 15L, 253C
1,1
All

```


Readme
Web Terminal




```

apiVersion: v1
kind: Pod
metadata:
  labels:
    run: nginx-secret
    name: nginx-secret
spec:
  containers:
  - image: nginx
    name: nginx-secret
    env:
    - name: COOL_VARIABLE
      valueFrom:
        secretKeyRef:
          name: some-secret
          key: key1
  
```

-- INSERT --
16,20
All

Readme
Web Terminal



```

student@node-1:~$ kubectl get pods -n web
NAME      READY   STATUS    RESTARTS   AGE
cache     1/1     Running   0           9s
student@node-1:~$ kubectl create secret generic some-secret --from-literal=key1=value4
secret/some-secret created
student@node-1:~$ kubectl get secret
NAME                                TYPE                                DATA   AGE
default-token-4kvr5                 kubernetes.io/service-account-token 3       2d11h
some-secret                         Opaque                              1       5s
student@node-1:~$ kubectl run nginx-secret --image=nginx --dry-run=client -o yaml > nginx_secret.yml
student@node-1:~$ vim nginx_secret.yml
student@node-1:~$ kubectl create -f nginx_secret.yml
pod/nginx-secret created
student@node-1:~$ kubectl get pods
NAME            READY   STATUS             RESTARTS   AGE
liveness-http   1/1     Running            0           6h38m
nginx-101       1/1     Running            0           6h39m
nginx-secret    0/1     ContainerCreating  0           4s
poller          1/1     Running            0           6h39m
student@node-1:~$ kubectl get pods
NAME            READY   STATUS    RESTARTS   AGE
liveness-http   1/1     Running   0           6h38m
nginx-101       1/1     Running   0           6h39m
nginx-secret    1/1     Running   0           8s
poller          1/1     Running   0           6h39m
student@node-1:~$
  
```

NEW QUESTION 4

Exhibit:



Task

You are required to create a pod that requests a certain amount of CPU and memory, so it gets scheduled to-a node that has those resources available.

- Create a pod named nginx-resources in the pod-resources namespace that requests a minimum of 200m CPU and 1Gi memory for its container
- The pod should use the nginx image
- The pod-resources namespace has already been created

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:

[illegible]

Readme

Web Terminal

THE **LINUX** FOUNDATION

```
apiVersion: v1
kind: Pod
metadata:
  labels:
    run: nginx-resources
    name: nginx-resources
    namespace: pod-resources
spec:
  containers:
  - image: nginx
    name: nginx-resources
    resources:
      requests:
        cpu: 200m
        memory: "1Gi"
```

-- INSERT --

15,22

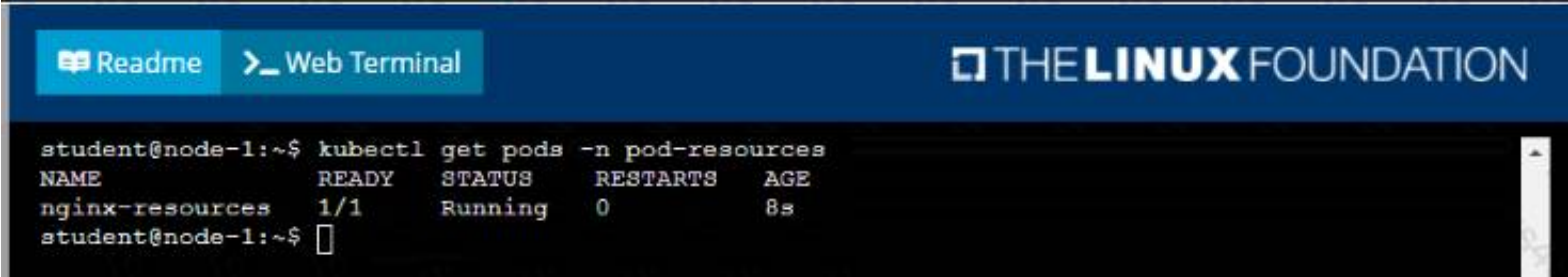
All

Readme

Web Terminal

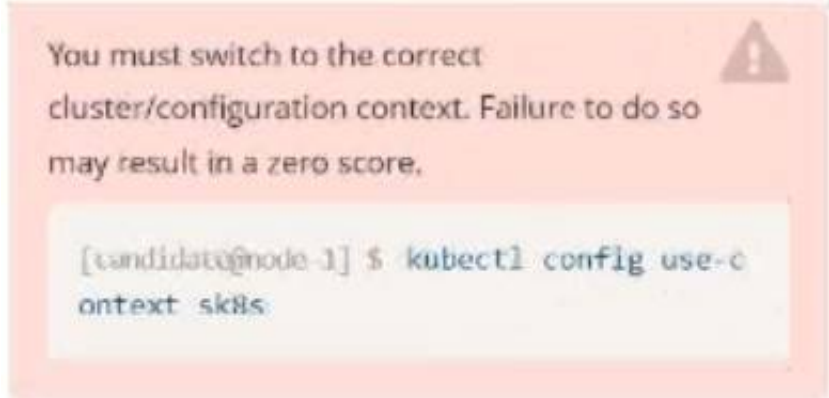
THE **LINUX** FOUNDATION

```
student@node-1:~$ kubectl run nginx-resources -n pod-resources --image=nginx --dry-run=client -o
yaml > nginx_resources.yml
student@node-1:~$ vim nginx_resources.yml
student@node-1:~$ kubectl create -g nginx_resources.yml
Error: unknown shorthand flag: 'g' in -g
See 'kubectl create --help' for usage.
student@node-1:~$ kubectl create -f nginx_resources.yml
pod/nginx-resources created
student@node-1:~$ kubectl get pods -n pod-re
```

NEW QUESTION 5

Exhibit:



Task:

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

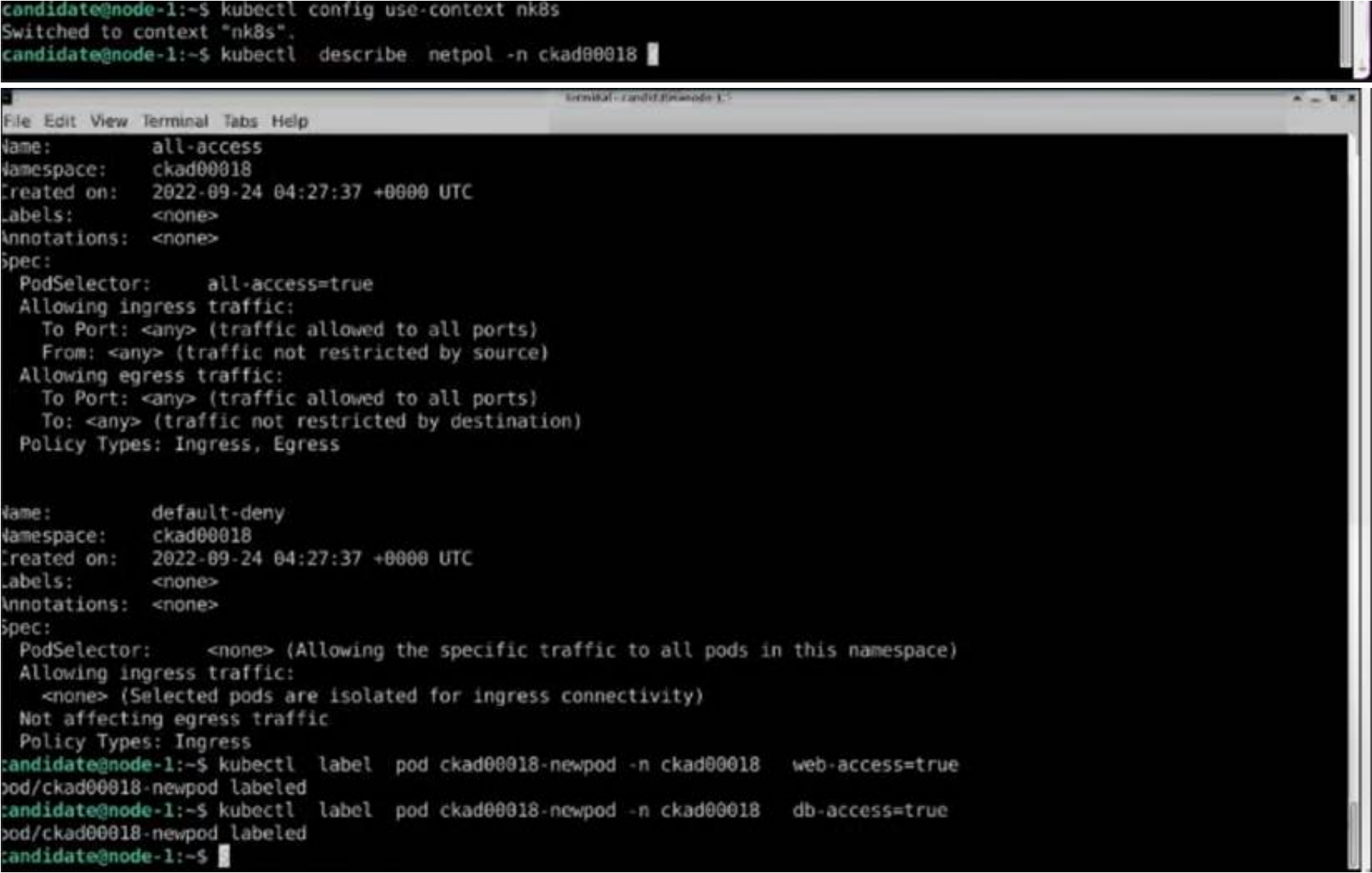


- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:



NEW QUESTION 6

Exhibit:



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

A. Mastered

B. Not Mastered

Answer: A

Explanation:

Solution:

apiVersion: v1 kind: Pod metadata: labels:

test: liveness

name: liveness-exec

spec: containers:

- name: liveness

image: k8s.gcr.io/busybox

args:

- /bin/sh

- -c

- touch /tmp/healthy; sleep 30; rm -rf /tmp/healthy; sleep 600

livenessProbe: exec: command:

- cat

- /tmp/healthy

initialDelaySeconds: 5

periodSeconds: 5

In the configuration file, you can see that the Pod has a single Container. The periodSeconds field specifies that the kubelet should perform a liveness probe every 5 seconds. The initialDelaySeconds field tells the kubelet that it should wait 5 seconds before performing the first probe. To perform a probe, the kubelet executes the command cat /tmp/healthy in the target container. If the command succeeds, it returns 0, and the kubelet considers the container to be alive and healthy. If the command returns a non-zero value, the kubelet kills the container and restarts it.

When the container starts, it executes this command:

/bin/sh -c "touch /tmp/healthy; sleep 30; rm -rf /tmp/healthy; sleep 600"

For the first 30 seconds of the container's life, there is a /tmp/healthy file. So during the first 30 seconds, the command cat /tmp/healthy returns a success code.

After 30 seconds, cat /tmp/healthy returns a failure co

Create the Pod:

kubectl apply -f <https://k8s.io/examples/pods/probe/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 "k8s.gcr.io/busybox" 23s 23s 1 {kubelet worker0} spec.containers{liveness}

Normal Pulled Successfully pulled image

"k8s.gcr.io/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 "k8s.gcr.io/busybox" 36s 36s 1 {kubelet worker0} spec.containers{liveness}

Normal Pulled Successfully pulled image

"k8s.gcr.io/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

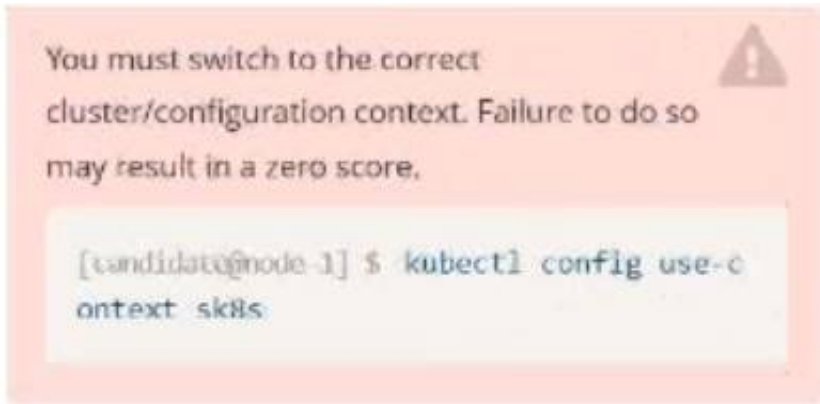
Wait another 30 seconds, and verify that the container has been restarted: kubectl get pod liveness-exec

The output shows that RESTARTS has been incremented: NAME READY STATUS RESTARTS AGE

liveness-exec 1/1 Running 1 1m

NEW QUESTION 7

Exhibit:



Task

A Deployment named backend-deployment in namespace staging runs a web application on port 8081.

✦ The Deployment's manifest files can be found at `~/spicy-pikachu/backend-deployment.yaml`.

Modify the Deployment specifying a readiness probe using path `/healthz`.

Set `initialDelaySeconds` to 8 and `periodSeconds` to 5.

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:

```
File Edit View Terminal Tabs Help
Warning: Permanently added '172.31.17.21' (ECDSA) to the list of known hosts.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

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

Text Description automatically generated

```
File Edit View Terminal Tabs Help
apiVersion: apps/v1
kind: Deployment
metadata:
  name: backend-deployment
  namespace: staging
spec:
  selector:
    matchLabels:
      app: nginx
  replicas: 3
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: nginx:1.14.2
          ports:
            - containerPort: 8081
          readinessProbe:
            initialDelaySeconds: 8
            periodSeconds: 5
            httpGet:
              path: /healthz
              port: 8081
          volumeMounts:
            - mountPath: /etc/nginx/conf.d/
              name: config
            - mountPath: /usr/share/nginx/html/
              name: www
-- INSERT --
```

```
File Edit View Terminal Tabs Help
Warning: Permanently added '172.31.17.21' (ECDSA) to the list of known hosts.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

candidate@node-1:~$ vi ~/spicy-pikachu/backend-deployment.yaml
candidate@node-1:~$ kubectl config use-context sk8s
Switched to context "sk8s".
candidate@node-1:~$ vim .vimrc
candidate@node-1:~$ vim ~/spicy-pikachu/backend-deployment.yaml
candidate@node-1:~$ kubectl apply -f ~/spicy-pikachu/backend-deployment.yaml
deployment.apps/backend-deployment configured
candidate@node-1:~$ kubectl get pods -n staging
NAME                                READY   STATUS    RESTARTS   AGE
backend-deployment-59d449b99d-cxct6 1/1     Running   0           20s
backend-deployment-59d449b99d-h2zjq 0/1     Running   0           9s
backend-deployment-78976f74f5-b8c85 1/1     Running   0           6h40m
backend-deployment-78976f74f5-flfsj 1/1     Running   0           6h40m
candidate@node-1:~$ kubectl get deploy -n staging
NAME             READY   UP-TO-DATE   AVAILABLE   AGE
backend-deployment 3/3     3             3           6h40m
candidate@node-1:~$ kubectl get deploy -n staging
NAME             READY   UP-TO-DATE   AVAILABLE   AGE
backend-deployment 3/3     3             3           6h41m
candidate@node-1:~$ vim ~/spicy-pikachu/backend-deployment.yaml
```

NEW QUESTION 8

Exhibit:



Context

Developers occasionally need to submit pods that run periodically. Task

Follow the steps below to create a pod that will start at a predetermined time and which runs to completion only once each time it is started:

- Create a YAML formatted Kubernetes manifest /opt/KDPD00301/periodic.yaml that runs the following shell command: date in a single busybox container. The command should run every minute and must complete within 22 seconds or be terminated by Kubernetes. The Cronjob name and container name should both be hello
- Create the resource in the above manifest and verify that the job executes successfully at least once

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:

```
Readme Web Terminal THE LINUX FOUNDATION

student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "*" * * * * --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
error: unable to match a printer suitable for the output format "yaml", allowed formats are: go-t
emplate, go-template-file, json, jsonpath, jsonpath-as-json, jsonpath-file, name, template, templatefile
, yaml
student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "*" * * * * --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
student@node-1:~$ vim /opt/KDPD00301/periodic.yaml
```


ReadmeWeb Terminal

THE LINUX FOUNDATION

```
apiVersion: batch/v1beta1
kind: CronJob
metadata:
  name: hello
spec:
  jobTemplate:
    metadata:
      name: hello
    spec:
      template:
        spec:
          containers:
            - image: busybox
              name: hello
              args: ["/bin/sh", "-c", "date"]
              restartPolicy: Never
  schedule: '*/* * * * *'
  startingDeadlineSeconds: 22
  concurrencyPolicy: Allow
```

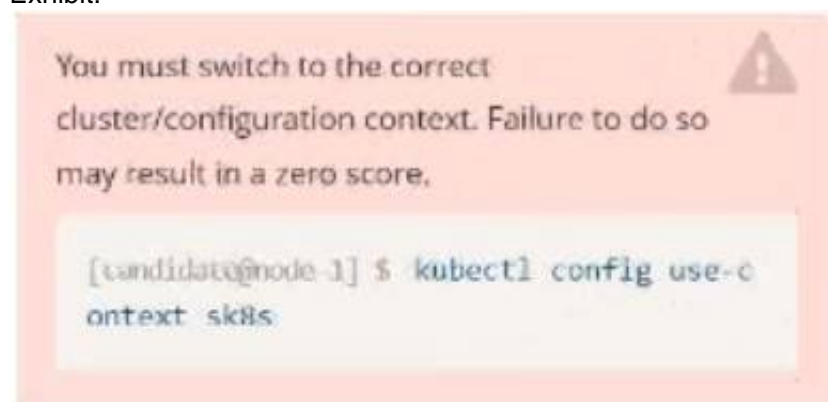
19,26All

ReadmeWeb Terminal

THE LINUX FOUNDATION

NEW QUESTION 9

Exhibit:



Task:

Create a Pod named nginx resources in the existing pod resources namespace. Specify a single container using nginx:stable image. Specify a resource request of 300m cpus and 1G1 of memory for the Pod's container.

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:

```
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ kubectl run nginx-resources -n pod-resources --image=nginx:stable --dry-run=client -o yaml > hw.yaml
candidate@node-1:~$ vim hw.yaml
```

Text Description automatically generated with medium confidence


```
File Edit View Terminal Tabs Help
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: nginx-resources
  name: nginx-resources
  namespace: pod-resources
spec:
  containers:
  - image: nginx:stable
    name: nginx-resources
    resources:
      requests:
        cpu: 300m
        memory: "1Gi"
```

Text Description automatically generated

```
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ kubectl run nginx-resources -n pod-resources --image=nginx:stable --dry-run=client -o yaml > hw.yaml
candidate@node-1:~$ vim hw.yaml
candidate@node-1:~$ kubectl create -f hw.yaml
pod/nginx-resources created
candidate@node-1:~$ kubectl get pods -n pod-resources
NAME          READY   STATUS    RESTARTS   AGE
nginx-resources 1/1     Running   0           13s
candidate@node-1:~$ kubectl describe pods -n pod-resources
```

Text Description automatically generated

```
File Edit View Terminal Tabs Help
memory:      1Gi
Environment: <none>
Mounts:
  /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-dmx9j (ro)
Conditions:
  Type             Status
  Initialized       True
  Ready             True
  ContainersReady   True
  PodScheduled      True
Volumes:
  kube-api-access-dmx9j:
    Type:              Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds: 3607
    ConfigMapName:       kube-root-ca.crt
    ConfigMapOptional:    <nil>
    DownwardAPI:         true
QoS Class:           Burstable
Node-Selectors:       <none>
Tolerations:          node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                      node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:
  Type    Reason      Age   From          Message
  ----    -
  Normal  Scheduled   20s   default-scheduler  Successfully assigned pod-resources/nginx-resources to k8s-node-0
  Normal  Pulling     19s   kubelet        Pulling image "nginx:stable"
  Normal  Pulled      13s   kubelet        Successfully pulled image "nginx:stable" in 6.55664052s
  Normal  Created     13s   kubelet        Created container nginx-resources
  Normal  Started     12s   kubelet        Started container nginx-resources
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ kubectl create deploy expose -n ckad00014 --image lfccncf/nginx:1.13.7 --dry-run=client -o yaml>
```

NEW QUESTION 10
 Exhibit:



Task:

Modify the existing Deployment named broker-deployment running in namespace quetzal so that its containers. The broker-deployment is manifest file can be found at:

```
~/daring-mocasin/broker-deployment.yaml
```

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:

```
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ vim
```

Text Description automatically generated

```
File Edit View Terminal Tabs Help
containers:
- name: broker
  image: redis:alpine
  ports:
  - containerPort: 6379
  securityContext:
    runAsUser: 30000
    privileged: false

candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ vim ~/daring-mocasin/broker-deployment.yaml
candidate@node-1:~$ kubectl apply -f ~/daring-mocasin/broker-deployment.yaml
deployment.apps/broker-deployment configured
candidate@node-1:~$ kubectl get pods -n quetzal
NAME                                READY   STATUS    RESTARTS   AGE
broker-deployment-65446d6d94-868p6  1/1     Running   0           30s
broker-deployment-65446d6d94-8dn7l  1/1     Running   0           32s
broker-deployment-65446d6d94-p4h4l  1/1     Running   0           31s
candidate@node-1:~$ kubectl get deploy -n quetzal
NAME            READY   UP-TO-DATE   AVAILABLE   AGE
broker-deployment  3/3     3             3           7h3m
candidate@node-1:~$
```

NEW QUESTION 10

Exhibit:



Task:

The pod for the Deployment named nosql in the crayfish namespace fails to start because its container runs out of resources. Update the nosql Deployment so that the Pod:

• The nosql Deployment's manifest file can be found at `~/chief-cardinal/nosql.yaml`.

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:

```
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ vim ~/chief-cardinal/nosql.yaml
```

```
File Edit View Terminal Tabs Help
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nosql
  namespace: crayfish
  labels:
    app.kubernetes.io/name: nosql
    app.kubernetes.io/component: backend
spec:
  selector:
    matchLabels:
      app.kubernetes.io/name: nosql
      app.kubernetes.io/component: backend
  replicas: 1
  template:
    metadata:
      labels:
        app.kubernetes.io/name: nosql
        app.kubernetes.io/component: backend
    spec:
      containers:
        - name: mongo
          image: mongo:4.2
          args:
            - --bind ip
            - 0.0.0.0
          ports:
            - containerPort: 27017
-- INSERT --
```



```
File Edit View Terminal Tabs Help
- name: mongo
  image: mongo:4.2
  args:
    - --bind_ip
    - 0.0.0.0
  ports:
    - containerPort: 27017
  resources:
    requests:
      memory: "160Mi"
    limits:
      memory: "320Mi"

:wq

File Edit View Terminal Tabs Help
To: <any> (traffic not restricted by destination)
Policy Types: Ingress, Egress

Name:      default-deny
Namespace: ckad00018
Created on: 2022-09-24 04:27:37 +0000 UTC
Labels:    <none>
Annotations: <none>
Spec:
  PodSelector: <none> (Allowing the specific traffic to all pods in this namespace)
  Allowing ingress traffic:
    <none> (Selected pods are isolated for ingress connectivity)
  Not affecting egress traffic
  Policy Types: Ingress
candidate@node-1:~$ kubectl label pod ckad00018-newpod -n ckad00018 web-access=true
pod/ckad00018-newpod labeled
candidate@node-1:~$ kubectl label pod ckad00018-newpod -n ckad00018 db-access=true
pod/ckad00018-newpod labeled
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ vim ~/chief-cardinal/nosql.yaml
candidate@node-1:~$ vim ~/chief-cardinal/nosql.yaml
candidate@node-1:~$ kubectl apply -f ~/chief-cardinal/nosql.yaml
deployment.apps/nosql configured
candidate@node-1:~$ kubectl get pods -n crayfish
NAME                                READY   STATUS    RESTARTS   AGE
nosql-74cccf7d64-lkqlg             1/1     Running   0           3m2s
candidate@node-1:~$ kubectl get deploy -n crayfish
NAME    READY   UP-TO-DATE   AVAILABLE   AGE
nosql   1/1     1            1           7h16m
candidate@node-1:~$
```

NEW QUESTION 12

Exhibit:



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 a deployment in namespace `kdsn00101` a service that accomplishes the following:

- Exposes the service on TCP port 8080
- is mapped to the pods defined by the specification of `kdsn00101-deployment`

- Is of type NodePort
- Has a name of cherry

A. Mastered
 B. Not Mastered

Answer: A

Explanation:

Solution:

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

Readme
Web Terminal



```

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:
    type: RollingUpdate
"/tmp/kubectl-edit-d4y5r.yaml" 70L, 1957C
1,1
Top

```

Readme
Web Terminal



```

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

```

NEW QUESTION 16

Exhibit:



Task

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

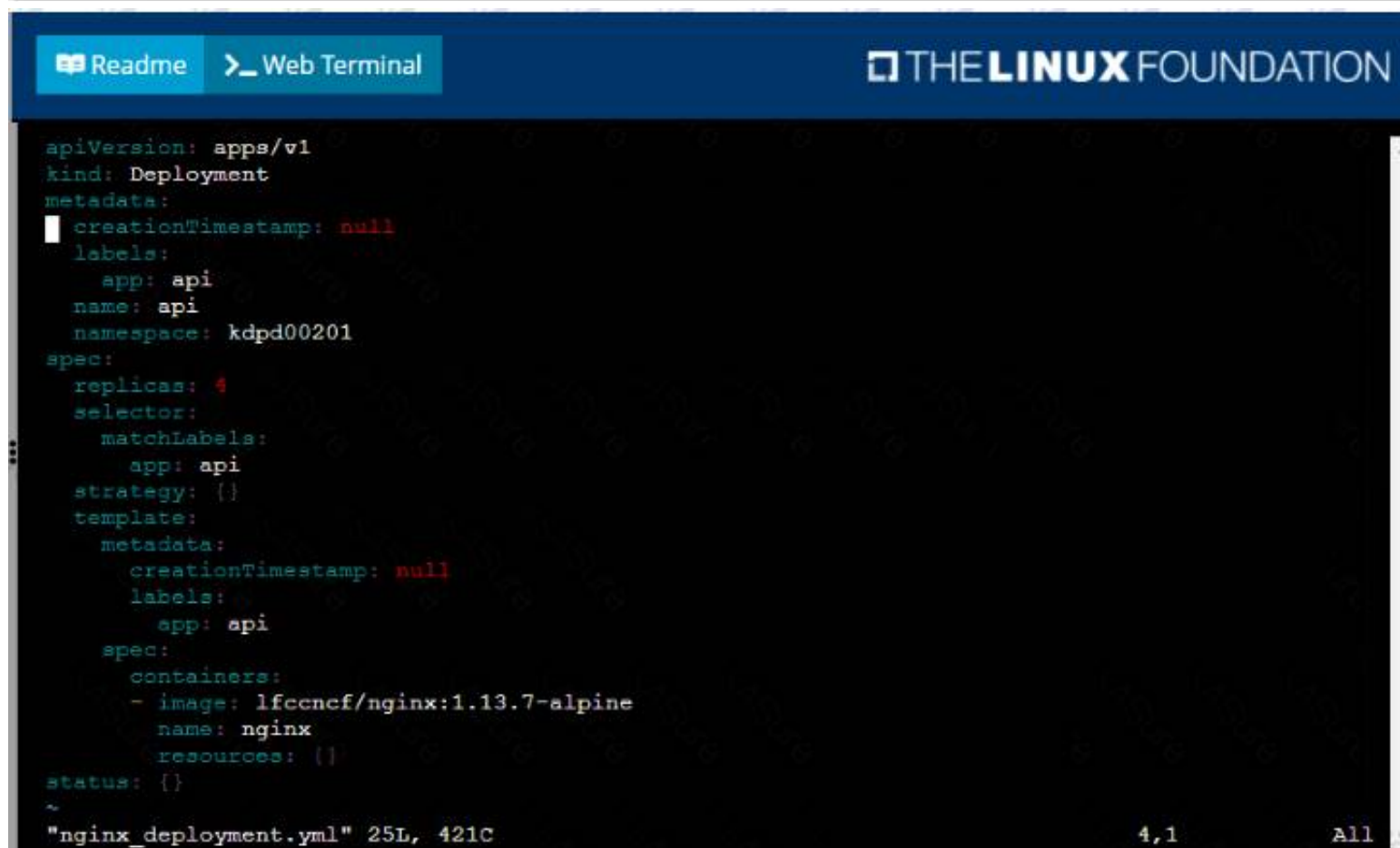
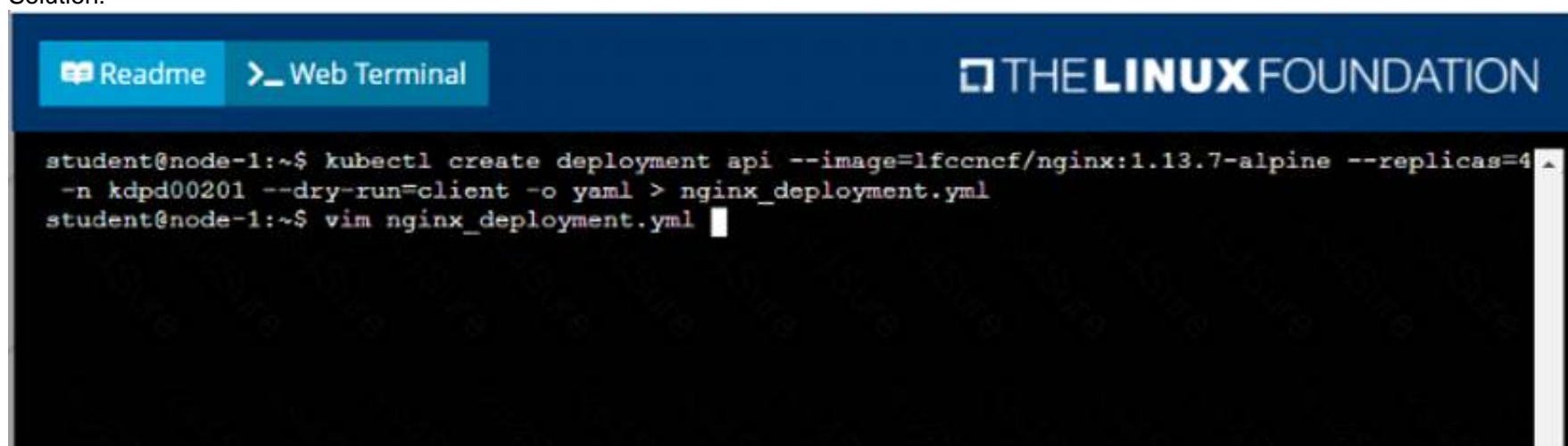
- Run the deployment in the kdpd00201 namespace. The namespace has already been created
- Name the deployment frontend and configure with 4 replicas
- Configure the pod with a container image of lfcncf/nginx:1.13.7
- Set an environment variable of NGINX PORT=8080 and also expose that port for the container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

```

NEW QUESTION 17

.....

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 First Try
- * CKAD Practice Test Questions in Multiple Choice Formats and Updates for 1 Year

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