



Linux Foundation CKA

Certified Kubernetes Administrator

Thank You for Downloading CKA Updated Exam Questions

<https://certs4sale.com/linux-foundation/cka-pdf-exam-dumps>

Version: 11.0

Question: 1

SIMULATION

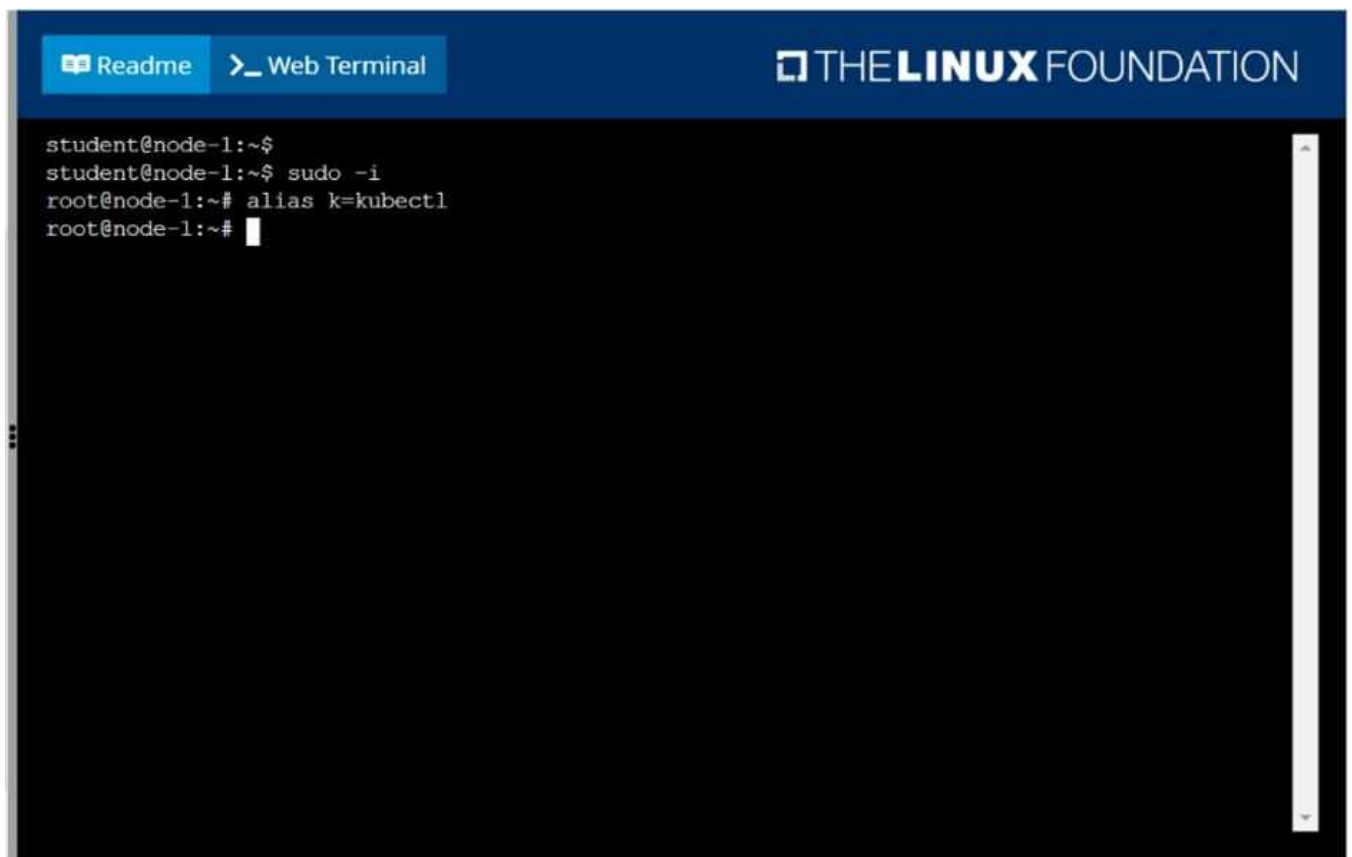
Monitor the logs of pod foo and:
Extract log lines corresponding to error
unable-to-access-website
Write them to/opt/KULM00201/foo



Answer: See the solution below.

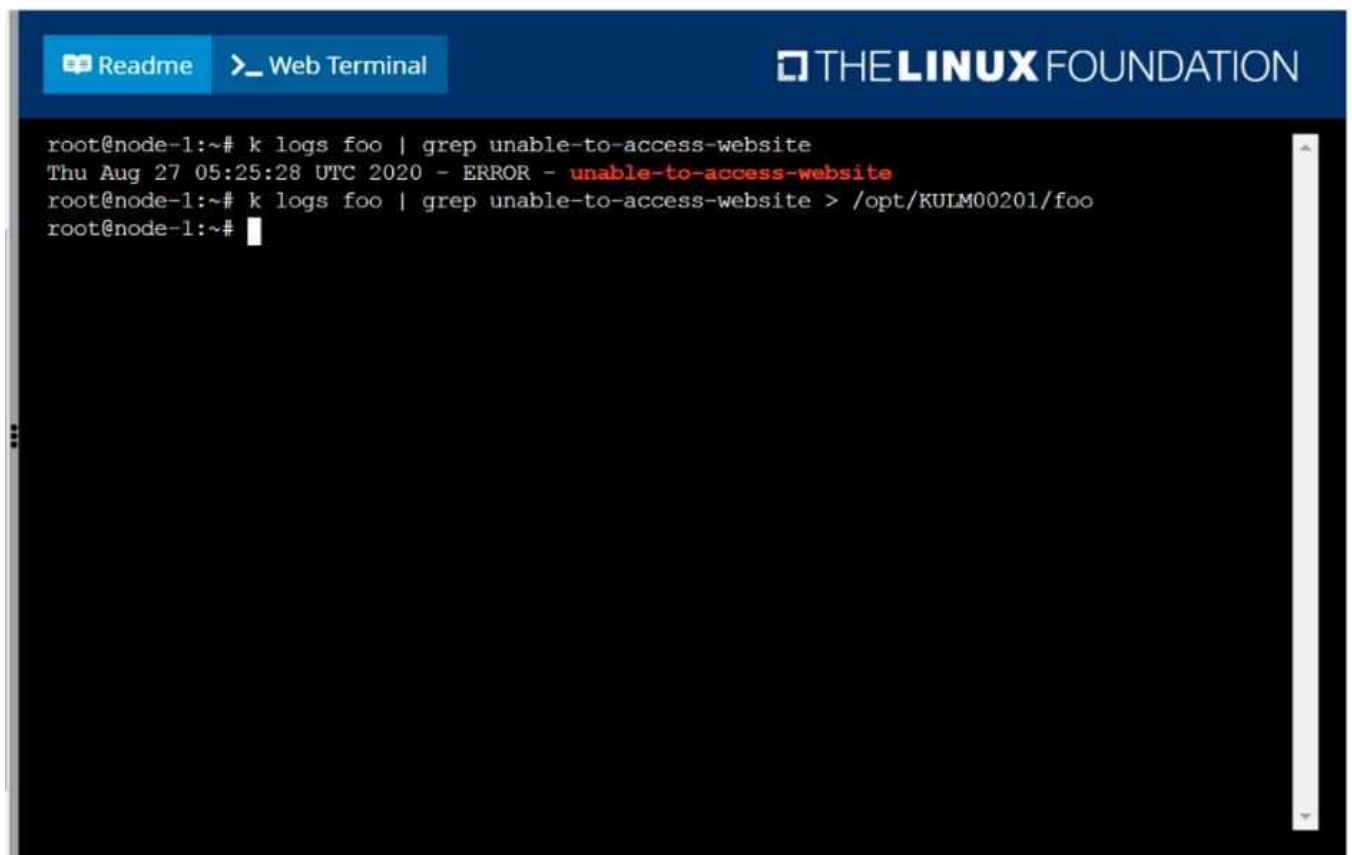
Explanation:

solution



```
student@node-1:~$  
student@node-1:~$ sudo -i  
root@node-1:~# alias k=kubectl  
root@node-1:~#
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\1 B.JPG



```
root@node-1:~# k logs foo | grep unable-to-access-website
Thu Aug 27 05:25:28 UTC 2020 - ERROR - unable-to-access-website
root@node-1:~# k logs foo | grep unable-to-access-website > /opt/KULM00201/foo
root@node-1:~#
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\1 C.JPG

Step 0: Set the correct Kubernetes context

If you're given a specific context (k8s in this case), you must switch to it:

```
kubectl config use-context k8s
```

⚠ Skipping this can cause you to work in the wrong cluster/namespace and cost you marks.

Step 1: Identify the namespace of the pod foo

First, check if foo is running in a specific namespace or in the default namespace.

```
kubectl get pods --all-namespaces | grep foo
```

Assume the pod is in the default namespace if no namespace is mentioned.

Step 2: Confirm pod foo exists and is running

```
kubectl get pod foo
```

You should get output similar to:

```
NAME READY STATUS RESTARTS AGE
```

```
foo 1/1 Running 0 1h
```

If the pod is not running, logs may not be available.

Step 3: View logs and filter specific error lines

We're looking for log lines that contain:

unable-to-access-website

Command:

```
kubectl logs foo | grep "unable-to-access-website"
```

Step 4: Write the filtered log lines to a file

Redirect the output to the required path:

```
kubectl logs foo | grep "unable-to-access-website" > /opt/KULM00201/foo
```

✓ This creates or overwrites the file /opt/KULM00201/foo with the filtered logs.

You may need sudo if /opt requires elevated permissions. But in most exam environments, you're already the root or privileged user.

Step 5: Verify the output file (optional but smart)

Check that the file was created and has the correct content:

```
cat /opt/KULM00201/foo
```

✓ Final Answer Summary:

```
kubectl config use-context k8s
```

```
kubectl logs foo | grep "unable-to-access-website" > /opt/KULM00201/foo
```

Question: 2

SIMULATION

List all persistent volumes sorted by capacity, saving the full kubectl output to /opt/KUCC00102/volume_list. Use kubectl 's own functionality for sorting the output, and do not manipulate it any further.

Answer: See the solution below.

Explanation:

solution

Readme
Web Terminal

```

77d
pv0007 7Gi RWO Recycle Available slow
77d
pv0006 8Gi RWO Recycle Available slow
77d
pv0003 10Gi RWO Recycle Available slow
77d
pv0002 11Gi RWO Recycle Available slow
77d
pv0010 13Gi RWO Recycle Available slow
77d
pv0011 14Gi RWO Recycle Available slow
77d
pv0001 16Gi RWO Recycle Available slow
77d
pv0009 17Gi RWO Recycle Available slow
77d
pv0005 18Gi RWO Recycle Available slow
77d
pv0008 19Gi RWO Recycle Available slow
77d
pv0000 21Gi RWO Recycle Available slow
77d
root@node-1:~# k get pv --sort-by=.spec.capacity.storage > /opt/KUCC00102/volume_list
root@node-1:~#

```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\2 C.JPG

Question: 3

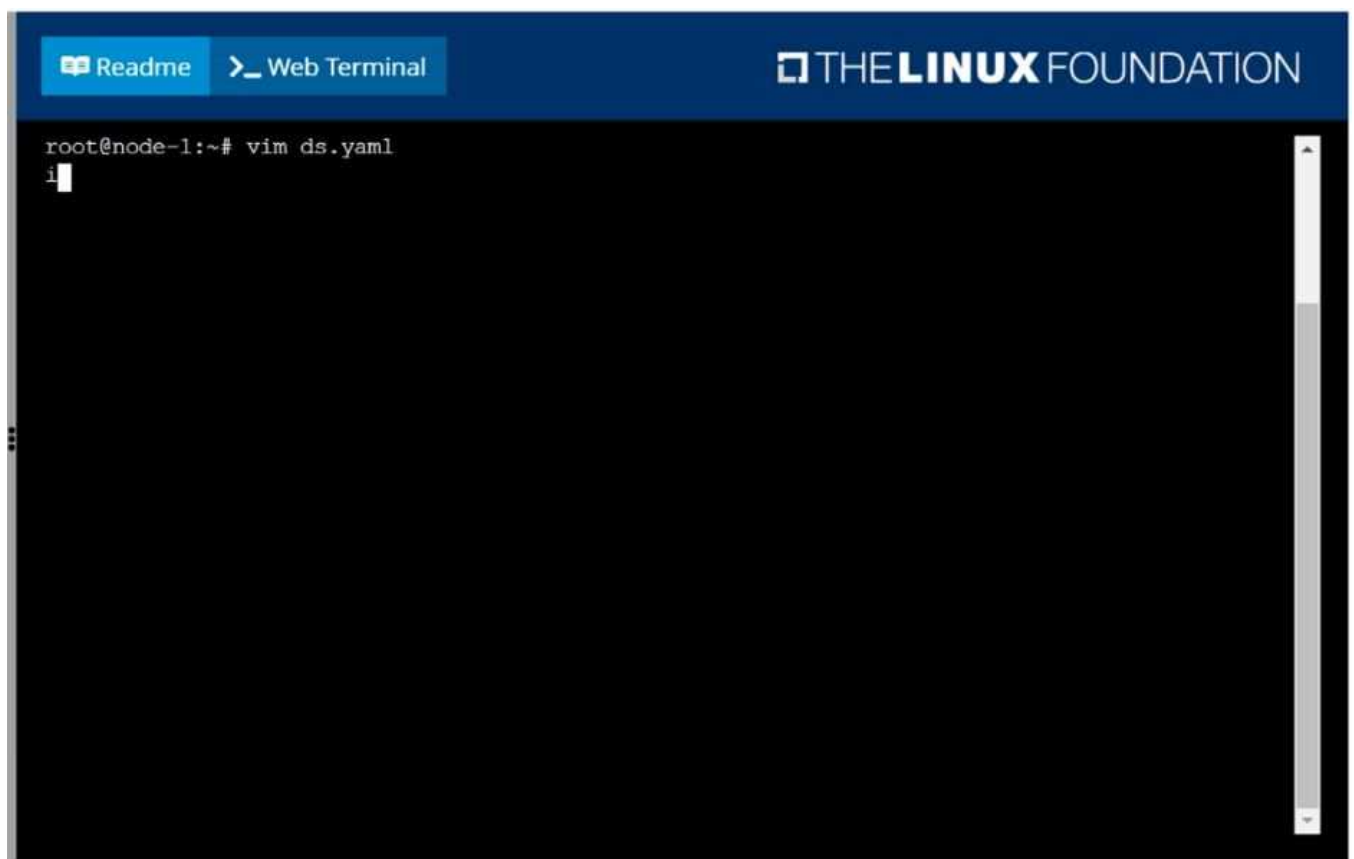
SIMULATION

Ensure a single instance of pod nginx is running on each node of the Kubernetes cluster where nginx also represents the Image name which has to be used. Do not override any taints currently in place. Use DaemonSet to complete this task and use ds-kusc00201 as DaemonSet name.

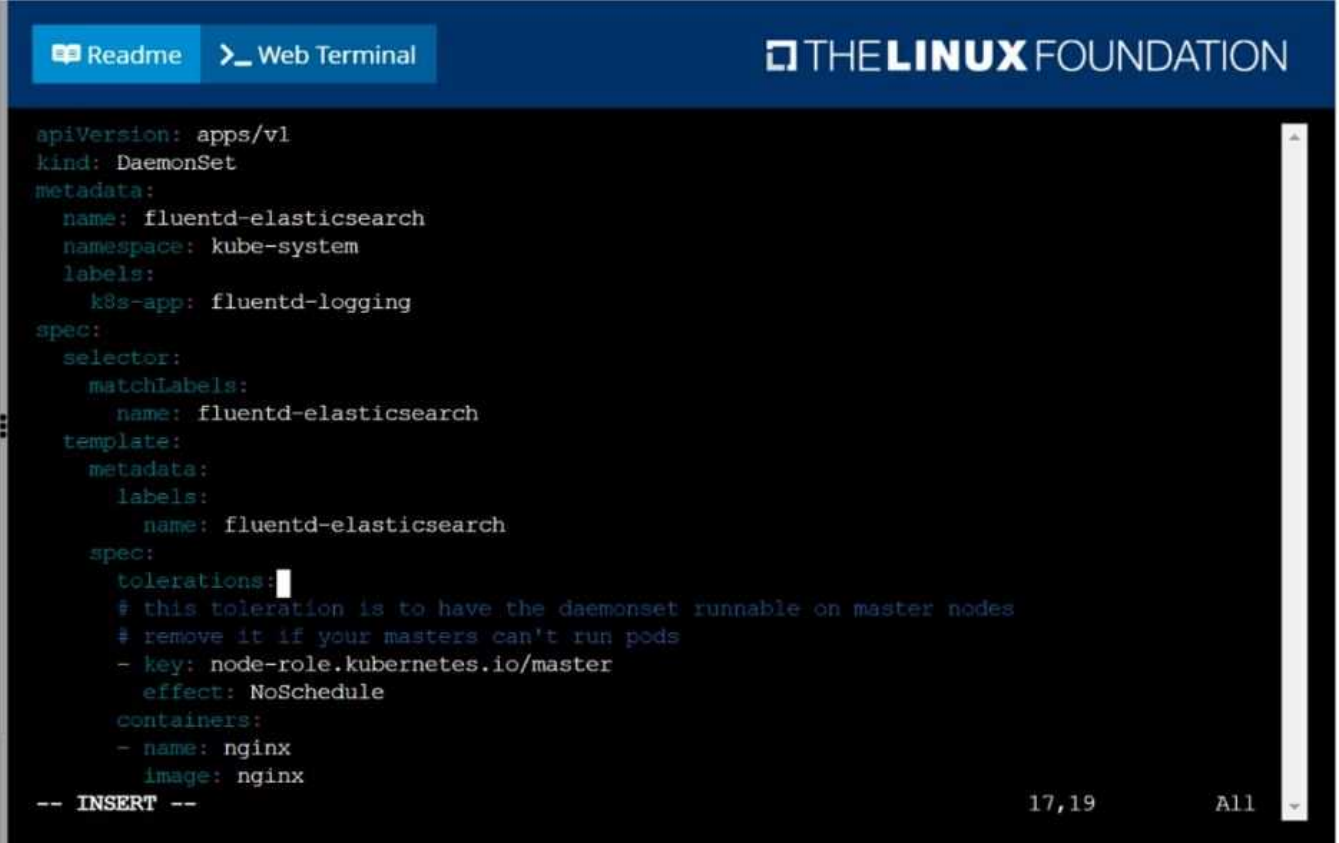
Answer: See the solution below.

Explanation:

solution



F:\Work\Data Entry Work\Data Entry\20200827\CKA\3 B.JPG

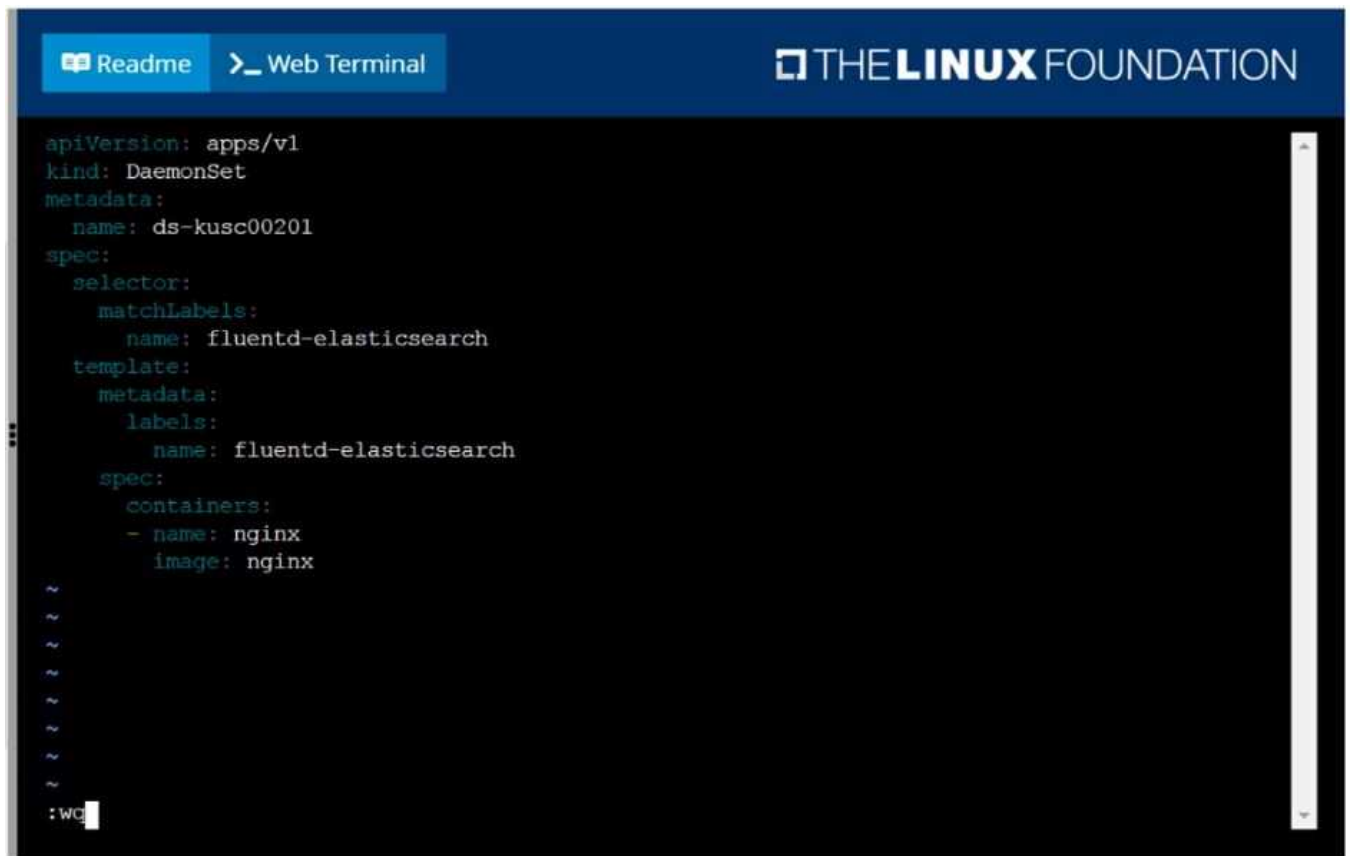


The screenshot shows a web terminal interface with a dark background. At the top, there is a blue header bar with a 'Readme' button and a 'Web Terminal' button. To the right of the header is the 'THE LINUX FOUNDATION' logo. The main area of the terminal displays a YAML configuration for a DaemonSet. The configuration includes metadata (name, namespace, labels), a selector (matchLabels), a template (metadata, labels, spec), and a tolerations section. The spec section includes a container named 'nginx' with the image 'nginx'. At the bottom of the terminal, there is a status bar showing '17,19' and 'All'.

```
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: fluentd-elasticsearch
  namespace: kube-system
  labels:
    k8s-app: fluentd-logging
spec:
  selector:
    matchLabels:
      name: fluentd-elasticsearch
  template:
    metadata:
      labels:
        name: fluentd-elasticsearch
    spec:
      tolerations:
        # this toleration is to have the daemonset runnable on master nodes
        # remove it if your masters can't run pods
        - key: node-role.kubernetes.io/master
          effect: NoSchedule
      containers:
        - name: nginx
          image: nginx
-- INSERT --
```

17,19 All

F:\Work\Data Entry Work\Data Entry\20200827\CKA\3 C.JPG

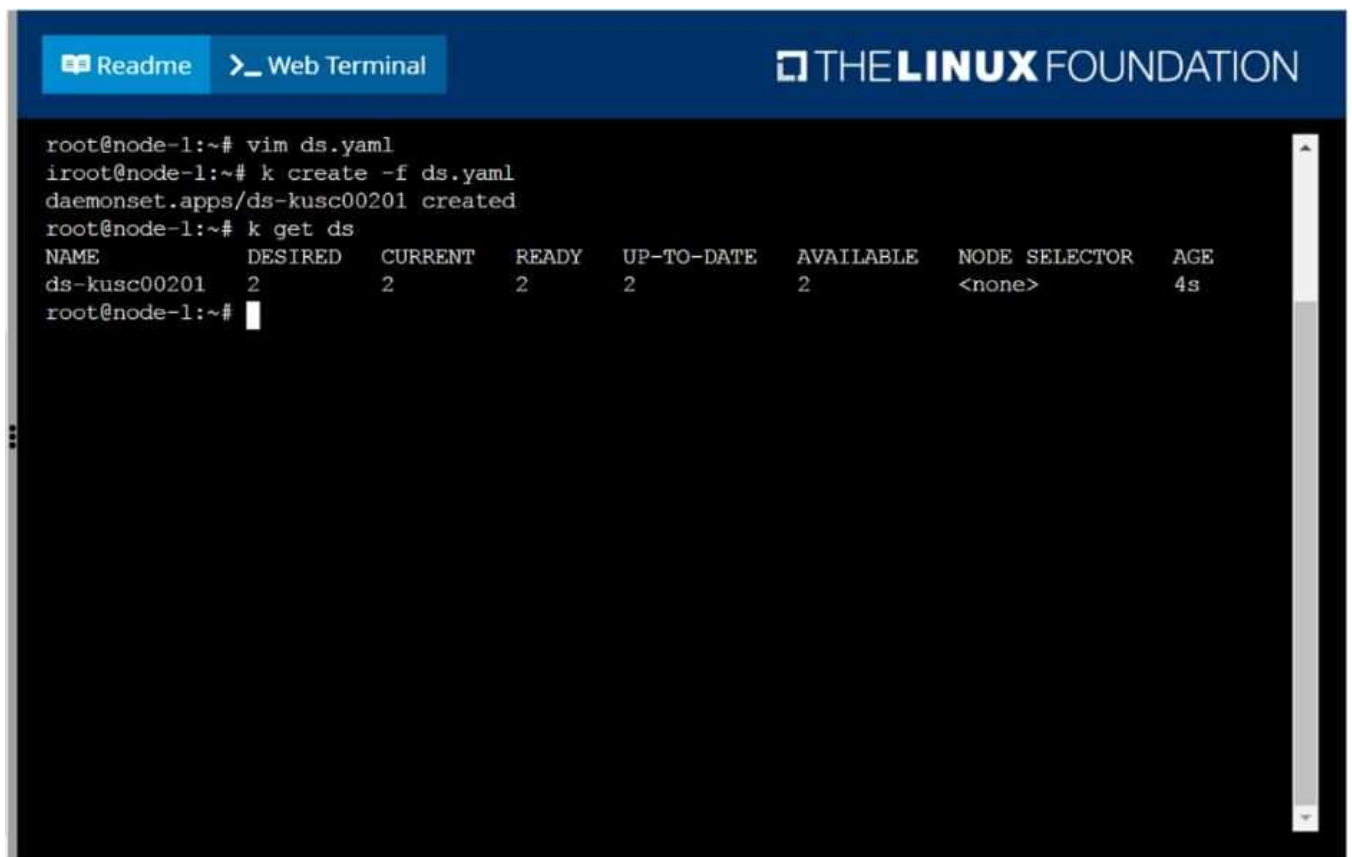


The screenshot shows a web terminal interface with a dark background. At the top, there is a blue header bar with a "Readme" button and a "Web Terminal" button. To the right of the header is the "THE LINUX FOUNDATION" logo. The terminal displays the following YAML configuration:

```
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: ds-kusc00201
spec:
  selector:
    matchLabels:
      name: fluentd-elasticsearch
  template:
    metadata:
      labels:
        name: fluentd-elasticsearch
    spec:
      containers:
      - name: nginx
        image: nginx
```

Below the configuration, there are several tilde (~) characters and a prompt :wq.

F:\Work\Data Entry Work\Data Entry\20200827\CKA\3 D.JPG



```
root@node-1:~# vim ds.yaml
root@node-1:~# k create -f ds.yaml
daemonset.apps/ds-kusc00201 created
root@node-1:~# k get ds
NAME          DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR   AGE
ds-kusc00201  2         2         2       2           2           <none>          4s
root@node-1:~#
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\3 E.JPG

Question: 4

SIMULATION

Perform the following tasks:

Add an init container to hungry-bear (which has been defined in spec file /opt/KUCC00108/pod-spec-KUCC00108.yaml)

The init container should create an empty file named /workdir/calm.txt

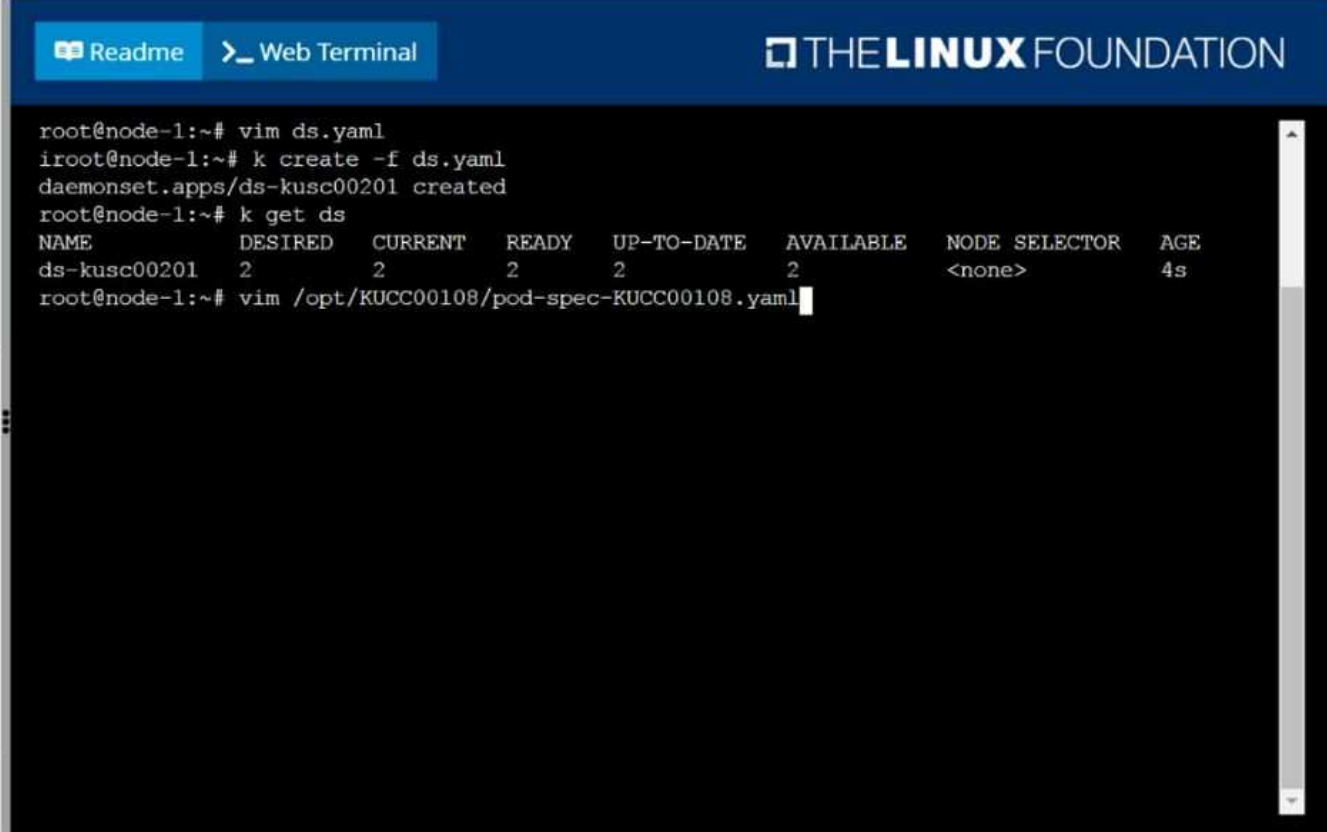
If /workdir/calm.txt is not detected, the pod should exit

Once the spec file has been updated with the init container definition, the pod should be created

**Answer: See the
solution below.**

Explanation:

solution



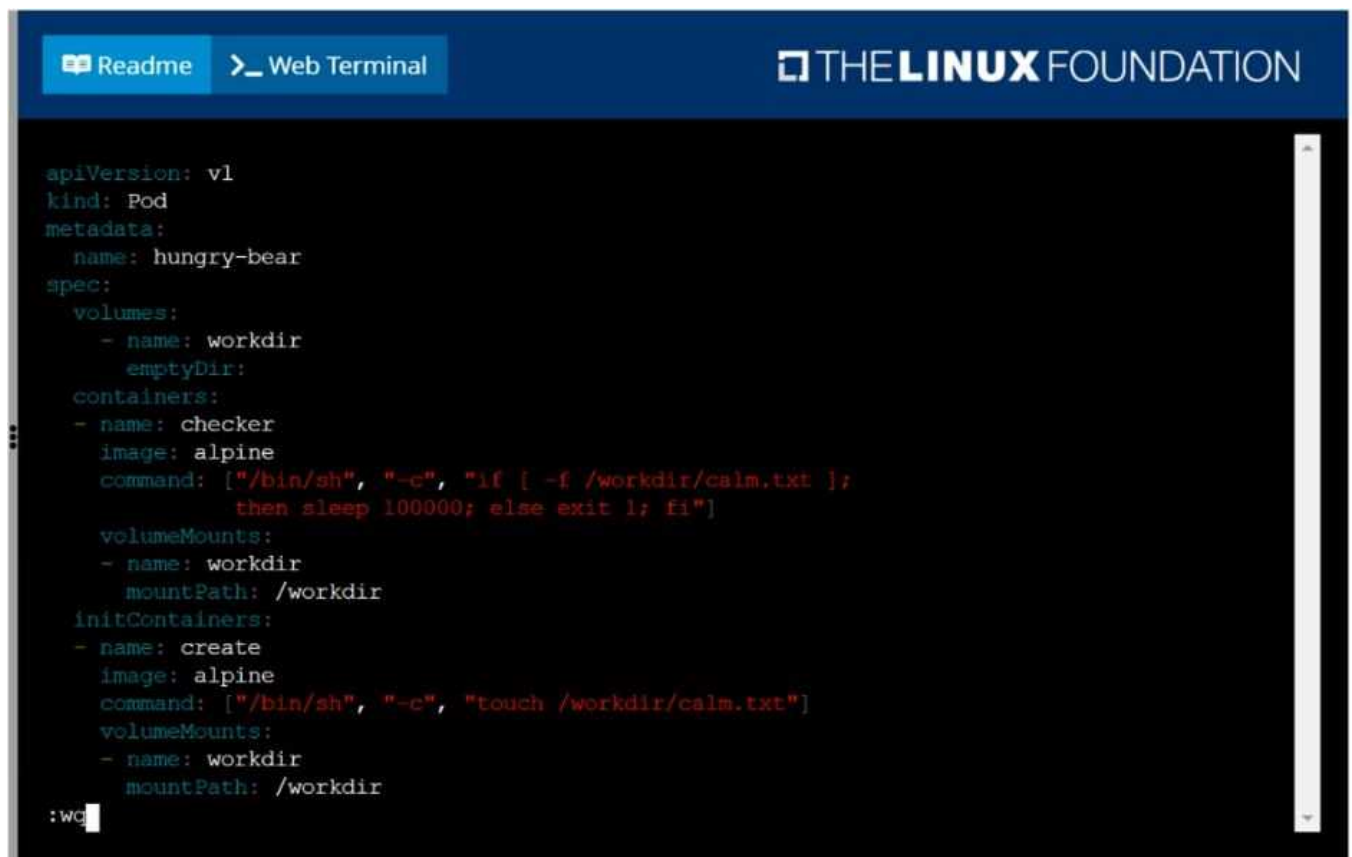
The screenshot shows a web terminal window with a dark blue header. On the left, there are two buttons: 'Readme' and 'Web Terminal'. On the right, the text 'THE LINUX FOUNDATION' is displayed. The terminal content shows a series of commands and their outputs:

```
root@node-1:~# vim ds.yaml
iroot@node-1:~# k create -f ds.yaml
daemonset.apps/ds-kusc00201 created
root@node-1:~# k get ds
```

NAME	DESIRED	CURRENT	READY	UP-TO-DATE	AVAILABLE	NODE SELECTOR	AGE
ds-kusc00201	2	2	2	2	2	<none>	4s

```
root@node-1:~# vim /opt/KUCC00108/pod-spec-KUCC00108.yaml
```

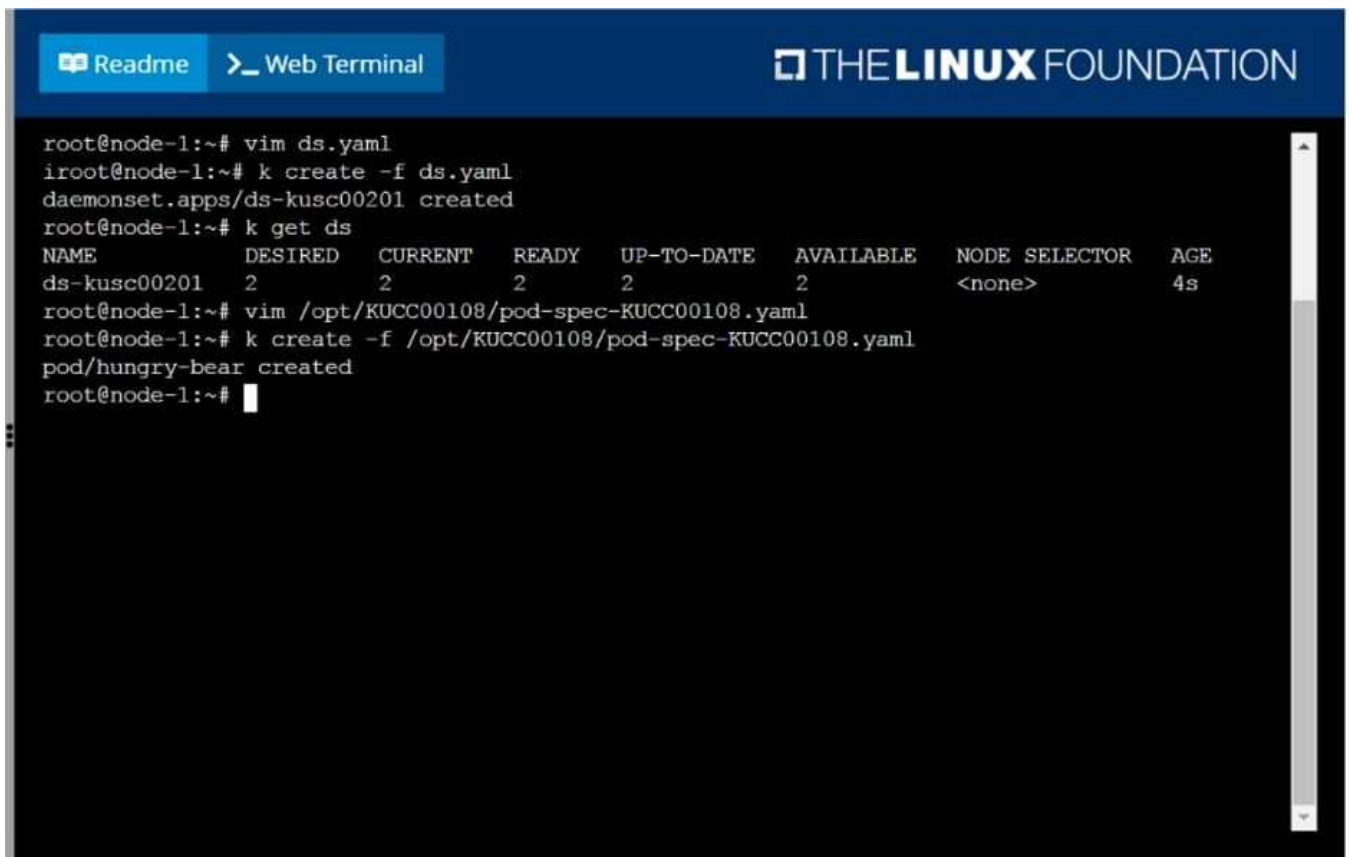
F:\Work\Data Entry Work\Data Entry\20200827\CKA\4 B.JPG



The screenshot shows a web terminal interface with a dark blue header. On the left, there are two buttons: 'Readme' and 'Web Terminal'. On the right, the 'THE LINUX FOUNDATION' logo is displayed. The main area of the terminal is black with white and red text, showing a Kubernetes manifest for a pod named 'hungry-bear'. The manifest includes details about the pod's API version, kind, metadata, and specifications for volumes, containers, and init containers. The 'checker' container runs a command that checks for the existence of a file and sleeps if it exists. The 'create' container runs a command to create the file. The terminal also shows a prompt ':wc' at the bottom left.

```
apiVersion: v1
kind: Pod
metadata:
  name: hungry-bear
spec:
  volumes:
    - name: workdir
      emptyDir: {}
  containers:
    - name: checker
      image: alpine
      command: ["/bin/sh", "-c", "if [ -f /workdir/calm.txt ];
        then sleep 100000; else exit 1; fi"]
      volumeMounts:
        - name: workdir
          mountPath: /workdir
  initContainers:
    - name: create
      image: alpine
      command: ["/bin/sh", "-c", "touch /workdir/calm.txt"]
      volumeMounts:
        - name: workdir
          mountPath: /workdir
:wc
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\4 C.JPG



```
root@node-1:~# vim ds.yaml
root@node-1:~# k create -f ds.yaml
daemonset.apps/ds-kusc00201 created
root@node-1:~# k get ds
NAME          DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR   AGE
ds-kusc00201  2         2         2       2           2          <none>          4s
root@node-1:~# vim /opt/KUCC00108/pod-spec-KUCC00108.yaml
root@node-1:~# k create -f /opt/KUCC00108/pod-spec-KUCC00108.yaml
pod/hungry-bear created
root@node-1:~#
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\4 D.JPG

Question: 5

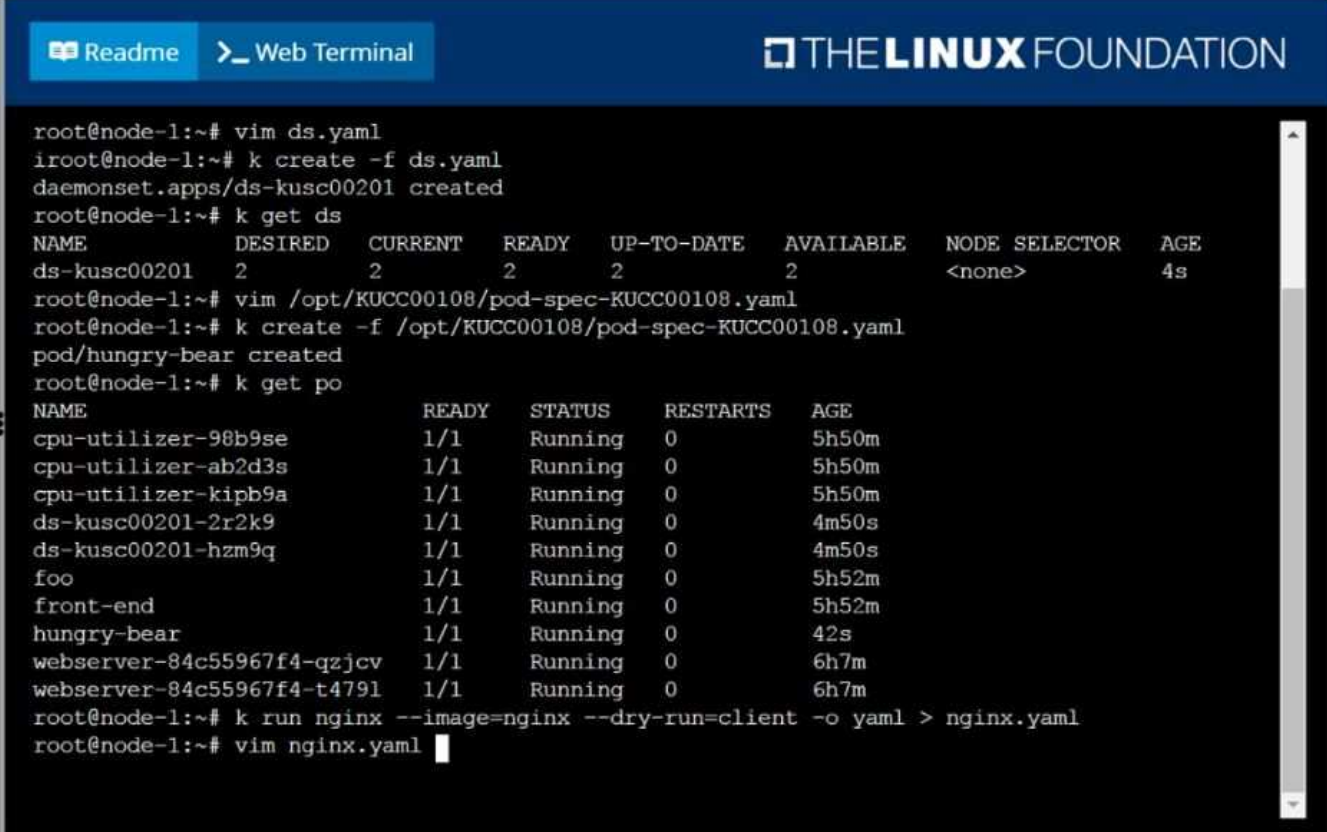
SIMULATION

Create a pod named kucc8 with a single app container for each of the following images running inside (there may be between 1 and 4 images specified):
nginx + redis + memcached.

**Answer: See the
solution below.**

Explanation:

solution



The screenshot shows a terminal window with a dark background. At the top, there is a blue header bar with 'Readme' and 'Web Terminal' buttons on the left, and 'THE LINUX FOUNDATION' logo on the right. The terminal content shows a series of commands and their outputs:

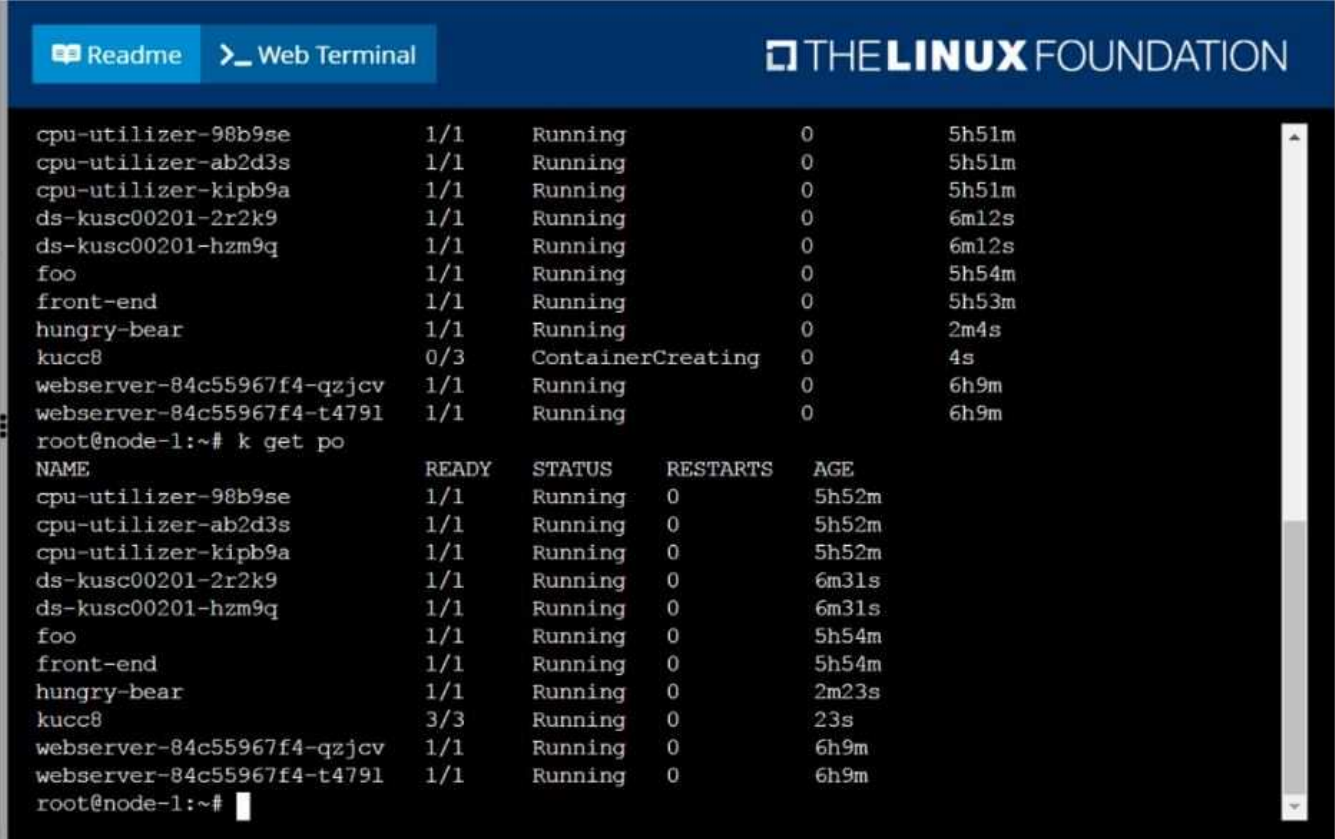
```

root@node-1:~# vim ds.yaml
iroot@node-1:~# k create -f ds.yaml
daemonset.apps/ds-kusc00201 created
root@node-1:~# k get ds
NAME                DESIRED    CURRENT    READY    UP-TO-DATE    AVAILABLE    NODE SELECTOR    AGE
ds-kusc00201         2          2          2        2             2            <none>           4s
root@node-1:~# vim /opt/KUCC00108/pod-spec-KUCC00108.yaml
root@node-1:~# k create -f /opt/KUCC00108/pod-spec-KUCC00108.yaml
pod/hungry-bear created
root@node-1:~# k get po
NAME                READY    STATUS    RESTARTS    AGE
cpu-utilizer-98b9se 1/1      Running   0            5h50m
cpu-utilizer-ab2d3s 1/1      Running   0            5h50m
cpu-utilizer-kipb9a 1/1      Running   0            5h50m
ds-kusc00201-2r2k9  1/1      Running   0            4m50s
ds-kusc00201-hzm9q  1/1      Running   0            4m50s
foo                 1/1      Running   0            5h52m
front-end           1/1      Running   0            5h52m
hungry-bear         1/1      Running   0            42s
webserver-84c55967f4-qzjcv 1/1      Running   0            6h7m
webserver-84c55967f4-t479l 1/1      Running   0            6h7m
root@node-1:~# k run nginx --image=nginx --dry-run=client -o yaml > nginx.yaml
root@node-1:~# vim nginx.yaml

```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\5 B.JPG





```

Readme  Web Terminal  THE LINUX FOUNDATION

cpu-utilizer-98b9se      1/1    Running    0        5h51m
cpu-utilizer-ab2d3s      1/1    Running    0        5h51m
cpu-utilizer-kipb9a      1/1    Running    0        5h51m
ds-kusc00201-2r2k9       1/1    Running    0        6m12s
ds-kusc00201-hzm9q       1/1    Running    0        6m12s
foo                      1/1    Running    0        5h54m
front-end                1/1    Running    0        5h53m
hungry-bear              1/1    Running    0        2m4s
kucc8                    0/3    ContainerCreating 0        4s
webserver-84c55967f4-qzjcv 1/1    Running    0        6h9m
webserver-84c55967f4-t479l 1/1    Running    0        6h9m
root@node-1:~# k get po
NAME                  READY   STATUS    RESTARTS   AGE
cpu-utilizer-98b9se   1/1     Running   0           5h52m
cpu-utilizer-ab2d3s   1/1     Running   0           5h52m
cpu-utilizer-kipb9a   1/1     Running   0           5h52m
ds-kusc00201-2r2k9    1/1     Running   0           6m31s
ds-kusc00201-hzm9q    1/1     Running   0           6m31s
foo                   1/1     Running   0           5h54m
front-end             1/1     Running   0           5h54m
hungry-bear           1/1     Running   0           2m23s
kucc8                 3/3     Running   0           23s
webserver-84c55967f4-qzjcv 1/1     Running   0           6h9m
webserver-84c55967f4-t479l 1/1     Running   0           6h9m
root@node-1:~#

```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\5 D.JPG

THANK YOU FOR DOWNLOADING CKA UPDATED EXAM QUESTIONS

Note: Thanks For Trying The Demo Of Our CKA Exam Product

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

Money Back Guarantee



Click The Link Below

<https://certs4sale.com/linux-foundation/cka-pdf-exam-dumps>