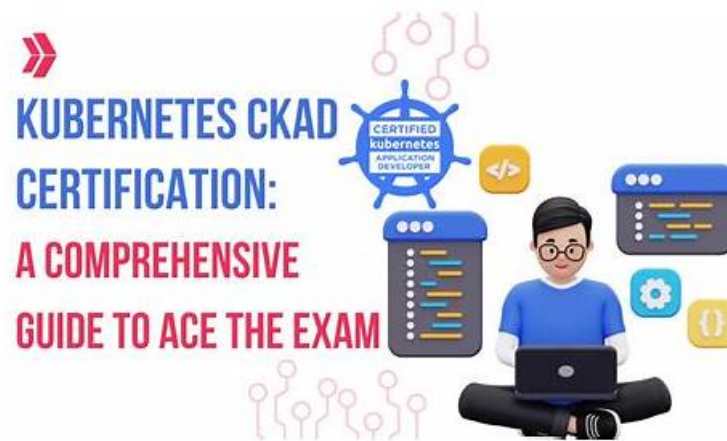


Test CKAD Guide & CKAD Latest Test Discount



What's more, part of that Braindumpsqa CKAD dumps now are free: <https://drive.google.com/open?id=1ChucDbuAO8LW9BXAGOiOdcqYIWtyg-OQ>

If you try on our CKAD exam braindumps, you will be very satisfied with its content and design. Trust me, you can't find anything better than our CKAD study materials. If you think I am exaggerating, you can try it for yourself. We can provide you with a free trial version. If you try another version and feel that our CKAD practice quiz are not bad, you can apply for another version of the learning materials again and choose the version that suits you best!

Linux Foundation Certified Kubernetes Application Developer (CKAD) exam is a certification program designed to evaluate and certify the skills and knowledge of developers in using Kubernetes to develop cloud-native applications. The CKAD Certification is recognized worldwide as a benchmark for Kubernetes application development expertise, and it validates the ability of developers to create and deploy cloud-native applications using Kubernetes.

>> Test CKAD Guide <<

Valid Test CKAD Guide Help You Clear Your CKAD: Linux Foundation Certified Kubernetes Application Developer Exam Exam Surely

Are you concerned for the training material for CKAD certification exam? So, your search is ended as you have got to the place where you can catch the finest CKAD certification exam dumps. Those entire applicants who put efforts in CKAD certification exam want to achieve their goal, but there are diverse means of preparing CKAD exams. Everyone might have their own approach to discover, how to associate CKAD Certified professional. It really doesn't matter how you concoct for the CKAD certification exam, you'd need some provision to make things calmer. Some candidates like to take help of their friends or tutors, while some simply rely on CKAD books. However, the easiest way to prepare the certification exam is to go through the study.

Linux Foundation Certified Kubernetes Application Developer (CKAD) Certification Exam is a rigorous test designed to evaluate the expertise of developers working with Kubernetes. Kubernetes is an open-source container orchestration system that automates the deployment, scaling, and management of containerized applications. It is widely used by organizations to manage their cloud-native applications, and the demand for certified Kubernetes developers is increasing day by day. The CKAD Certification Exam validates the skills and knowledge of developers in designing, building, configuring, and deploying cloud-native applications on Kubernetes.

Linux Foundation Certified Kubernetes Application Developer Exam Sample Questions (Q10-Q15):

NEW QUESTION # 10

You need to schedule a job to run every day at 10:00 AM to clean up old container images in your Kubernetes cluster. These images are tagged with "app=my-app" and have been created in the last 7 days. How would you implement this using a CronJob?

Answer:

Explanation:

See the solution below with Step by Step Explanation.

Explanation:

Solution (Step by Step) :

1. Create a CronJob YAML file:

```
apiVersion: batch/v1
kind: CronJob
metadata:
  name: image-cleanup-cronjob
spec:
  schedule: "0 10 * * * # Run at 10:00 AM every day"
  jobTemplate:
    spec:
      template:
        spec:
          containers:
            - name: image-cleanup
              image: alpine/kubectrl:latest # Use a container with kubectrl
              command: ["sh", "-c"]
              args:
                - |
                  kubectrl get images --selector "app=my-app" --output=jsonpath='{.items[].name}' |
                  while IFS= read -r image; do
                    kubectrl get images $image -o jsonpath='{.created}' | date -d @%s +%s > /tmp/image_age.txt
                    image_age=$(cat /tmp/image_age.txt)
                    current_time=$(date +%s)
                    if (( (current_time - image_age) / 86400 > 7 )); then
                      echo "Deleting image: $image"
                      kubectrl delete image $image
                    fi
                  done
          restartPolicy: OnFailure
```

2. Apply the Cronjob: `bash kubectrl apply -f image-cleanup-cronjob.yaml` 3. Verify the CronJob: `bash kubectrl get cronjobs` -

Schedule: The 'schedule' field defines the cron expression, which triggers the job every day at 10:00 AM. - Job Template: - The

'jobTemplate' defines the actual job that Will be executed. - Container: - The 'image' field specifies the container image to use. In this

case, it's a container with 'kubectrl pre-installed' - The 'command' and 'args' fields define the command to run in the container. The

command uses 'kubectrl' to list images With the specified label and then iterates through them, checking their creation date. If an

image is older than 7 days, it's deleted. - RestartPolicy: The 'restartPolicy' is set to 'OnFailure' to ensure the job restarts if it fails.

Important Note: - Make sure the container image you choose has the necessary tools (like 'kubectrl') to interact with your

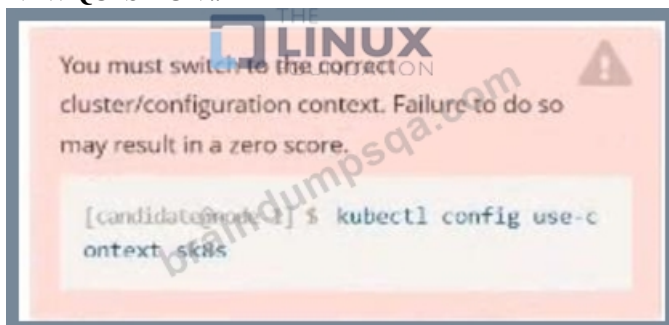
Kubernetes cluster. - This solution assumes you have the necessary permissions to delete images. If not, you may need to modify the

'kubectrl delete image' command to use appropriate RBAC roles. - This solution doesn't consider images used by running pods. You

should adjust the script to exclude images that are currently in use. This Cronjob will automatically run every day, cleaning up old

container images and maintaining a clean environment in your cluster.,

NEW QUESTION # 11



Task:

1- Update the Propertunel scaling configuration of the Deployment web1 in the ckad00015 namespace setting maxSurge to 2 and maxUnavailable to 59

2- Update the web1 Deployment to use version tag 1.13.7 for the Ifconf/nginx container image.

3- Perform a rollback of the web1 Deployment to its previous version

Answer:

Explanation:

See the solution below.

Explanation

Solution:

```
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ kubectl edit deploy web1 -n ckad00015
```

Text Description automatically generated

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

```
File Edit View Terminal Tabs Help
Switched to context "k8s".
candidate@node-1:~$ kubectl create secret generic app-secret -n default --from-literal=key3=value1
secret/app-secret created
candidate@node-1:~$ kubectl get secrets
NAME      TYPE      DATA   AGE
app-secret Opaque    1       4s
candidate@node-1:~$ kubectl run nginx-secret -n default --image=nginx:stable --dry-run=client -o yaml > sec.yaml
candidate@node-1:~$ vim sec.yaml
candidate@node-1:~$ kubectl create -f sec.yaml
pod/nginx-secret created
candidate@node-1:~$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
nginx-secret  1/1     Running   0          7s
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ kubectl edit deploy web1 -n ckad00015
deployment.apps/web1 edited
candidate@node-1:~$ kubectl rollout status deploy web1 -n ckad00015
deployment "web1" successfully rolled out
candidate@node-1:~$ kubectl rollout undo deploy web1 -n ckad00015
deployment.apps/web1 rolled back
candidate@node-1:~$ kubectl rollout history deploy web1 -n ckad00015
deployment.apps/web1
REVISION  CHANGE-CAUSE
2          <none>
3          <none>

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

NEW QUESTION # 12

Context



Context

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

Task

Perform the following tasks:

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



The output file has already been created

* Store the associated error events to a file /opt/KDOB00401/error.txt, The output file has already been created. You will need to use the -o wide output specifier with your command

* Fix the issue.



Answer:

Explanation:

Solution:

Create the Pod:

kubectl create -f <http://k8s.io/docs/tasks/configure-pod-container/exec-liveness.yaml> Within 30 seconds, view the Pod events:

kubectl describe pod liveness-exec

The output indicates that no liveness probes have failed yet:

FirstSeen LastSeen Count From SubobjectPath Type Reason Message

```
-----
24s 24s 1 {default-scheduler} Normal Scheduled Successfully assigned liveness-exec to worker0
23s 23s 1 {kubelet worker0} spec.containers{liveness} Normal Pulling pulling image "gcr.io/google_containers/busybox"
23s 23s 1 {kubelet worker0} spec.containers{liveness} Normal Pulled Successfully pulled image
"gcr.io/google_containers/busybox"
23s 23s 1 {kubelet worker0} spec.containers{liveness} Normal Created Created container with docker id 86849c15382e;
Security:[seccomp=unconfined]
```


23s 23s 1 {kubelet worker0} spec.containers{liveness} Normal Started Started container with docker id 86849c15382e After 35 seconds, view the Pod events again:

kubectl describe pod liveness-exec

At the bottom of the output, there are messages indicating that the liveness probes have failed, and the containers have been killed and recreated.

FirstSeen LastSeen Count From SubobjectPath Type Reason Message

```
-----
37s 37s 1 {default-scheduler} Normal Scheduled Successfully assigned liveness-exec to worker0
36s 36s 1 {kubelet worker0} spec.containers{liveness} Normal Pulling pulling image "gcr.io/google_containers/busybox"
36s 36s 1 {kubelet worker0} spec.containers{liveness} Normal Pulled Successfully pulled image
"gcr.io/google_containers/busybox"
36s 36s 1 {kubelet worker0} spec.containers{liveness} Normal Created Created container with docker id 86849c15382e;
Security:[seccomp=unconfined]
36s 36s 1 {kubelet worker0} spec.containers{liveness} Normal Started Started container with docker id 86849c15382e
2s 2s 1 {kubelet worker0} spec.containers{liveness} Warning Unhealthy Liveness probe failed: cat: can't open '/tmp/healthy': No
such file or directory Wait another 30 seconds, and verify that the Container has been restarted:
```

kubectl get pod liveness-exec

The output shows that RESTARTS has been incremented:

NAME READY STATUS RESTARTS AGE

liveness-exec 1/1 Running 1 m

NEW QUESTION # 13

You're working on a Kubernetes application that involves retrieving data from a database. You have a Deployment With multiple pods, each accessing the database directly. To improve the application's performance and reliability, you want to implement an adapter pattern that introduces a service layer to handle database interactions. This layer should be responsible for connection pooling, caching, and error handling, making the application more resilient to database outages.

Answer:

Explanation:

See the solution below with Step by Step Explanation.

Explanation:

Solution (Step by Step) :

1. Create a Service Account:

- Create a service account for the application. This will be used by the service layer to access the database.

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: db-service-account
```

- Apply the service account to the cluster: 'kubectl apply -f db-service-account.yaml' 2. Create a Role and RoleBinding: - Create a role that grants the necessary permissions to access the database.

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: db-access-role
  namespace:
rules:
- apiGroups: ["database.example.com"] # Replace with actual API group
  resources: ["databases", "databases/credentials"]
  verbs: ["get", "list", "create", "update", "delete", "watch", "patch"]
```

- Create a role binding that associates the role with the service account

```

apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: db-access-binding
  namespace:
subjects:
- kind: ServiceAccount
  name: db-service-account
  namespace:
roleRef:
  kind: Role
  name: db-access-role
apiGroup: rbac.authorization.k8s.io

```

- Apply the role and role binding to the cluster: - 'kubectl apply -f db-access-role.yaml' - 'kubectl apply -f db-access-binding.yaml'

3. Create the Service Layer Deployment: - Deploy the service layer component. This can be a containerized application that handles database interactions.

```

apiVersion: apps/v1
kind: Deployment
metadata:
  name: db-service
spec:
  replicas: 1 # Adjust as needed
  selector:
    matchLabels:
      app: db-service
  template:
    metadata:
      labels:
        app: db-service
    spec:
      serviceAccountName: db-service-account
      containers:
      - name: db-service
        image: # Replace with your service image
        ports:
        - containerPort: 8080 # Adjust based on your service's port
        env:
        - name: DATABASE_HOST
          value: # Replace with your database host
        - name: DATABASE_PORT
          value: # Replace with your database port
        - name: DATABASE_USER
          value: # Replace with your database user
        - name: DATABASE_PASSWORD
          valueFrom:
            secretKeyRef:
              name: db-credentials
              key: password # Replace with the key for your database password

```

- Apply the deployment: 'kubectl apply -f db-service.yaml' 4. Create a Secret for Database Credentials: - Create a secret to store sensitive database credentials.

```

apiVersion: v1
kind: Secret
metadata:
  name: db-credentials
stringData:
  password: # Replace with your database password

```

- Apply the secret 'kubectl apply -f db-credentials.yaml' 5. Create a Service for the Service Layer: - Create a service to expose the service layer to the application pods.

```

apiVersion: v1
kind: Service
metadata:
  name: db-service
spec:
  selector:
    app: db-service
  ports:
  - protocol: TCP
    port: 8080 # Adjust based on your service's port
    targetPort: 8080 # Match the containerPort in the Deployment

```

- Apply the service: 'kubectl apply -f db-service.yaml' 6. Update the Application Deployment: - Update the Deployment for your main application to use the service layer.

```

apiVersion: apps/v1
kind: Deployment
metadata:
  name:
spec:
  ...
  template:
    ...
    spec:
      containers:
        - name:
          image:
          env:
            - name: DB_SERVICE_HOST
              value: db-service # Use the service name
            - name: DB_SERVICE_PORT
              value: "8080" # Use the port of the service
          ...

```

T Test and Verify' - Verify the changes: - Check the logs for both the service layer and the application. - Test your application's functionality. Note: - Ensure to replace placeholders like ". ". ". ". ". ". and with your actual values. - This is a basic example, and you may need to adjust the configuration based on your specific service layer and database implementation. ,

NEW QUESTION # 14

You have a Kubernetes deployment named 'my-app' that runs an application with a specific configuration defined in a ConfigMap named 'my-config'. You need to implement a strategy to automatically update the deployment when the ConfigMap is changed.

Answer:

Explanation:

See the solution below with Step by Step Explanation.

Explanation:

Solution (Step by Step) :

1. Create a 'ConfigMap' named 'my-configs' with the following contents:

```

apiVersion: v1
kind: ConfigMap
metadata:
  name: my-config
data:
  config.json: |
    {
      "key1": "value1",
      "key2": "value2"
    }

```

2. Create a 'Deployment' named 'my-app' that mounts the 'my-config' ConfigMap as a volume:

```

apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  replicas: 2
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
        - name: my-app
          image: my-app-image:latest
          volumeMounts:
            - name: my-config-volume
              mountPath: /etc/config
          volumes:
            - name: my-config-volume
              configMap:

```

3. Apply the ConfigMap and Deployment bash `kubectl apply -f configmap.yaml` `kubectl apply -f deployment.yaml` 4. Update the ConfigMap with new values:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: my-config
data:
  config.json: |
    {
      "key1": "new-value1",
      "key2": "new-value2"
    }
```

5. Apply the updated ConfigMap: `bash kubectl apply -f configmap.yaml` - The 'kustomization.yaml' file defines the resources (the 'deployment.yaml' file) and the patches to apply. - The 'deployment.yaml' file contains the base configuration for the deployment. - The patch.yaml file applies a strategic merge patch to the deployment, configuring rolling updates and automatic updates triggered by new images. - The 'maxSurge' and 'maxUnavailable' settings in the 'patch.yaml' define the maximum number of pods that can be added or removed during the update process. - The 'imagePullPolicy: Always' ensures that the new image is pulled from Docker Hub even if it exists in the pod's local cache, triggering the update.

NEW QUESTION # 15

• • • • •

CKAD Latest Test Discount: <https://www.braindumpsqa.com/CKAD> braindumps.html

- New Test CKAD Guide | Efficient Linux Foundation CKAD: Linux Foundation Certified Kubernetes Application Developer Exam 100% Pass □ Open website □ www.prep4pass.com □ and search for ➡ CKAD □ for free download ☂CKAD Dump Check
- Linux Foundation CKAD Exam | Test CKAD Guide - Easy to Pass CKAD: Linux Foundation Certified Kubernetes Application Developer Exam Exam □ Download ☀ CKAD □☀□ for free by simply entering (www.pdfvce.com) website □CKAD Valid Guide Files
- CKAD Study Plan □ CKAD Study Plan □ Exam Sample CKAD Online □ ➡ www.prep4away.com □ is best website to obtain □ CKAD □ for free download □Latest CKAD Test Guide
- Free PDF Quiz 2025 Professional Linux Foundation Test CKAD Guide □ Search for ➡ CKAD □ on “ www.pdfvce.com ” immediately to obtain a free download □CKAD Exam Quiz
- Linux Foundation CKAD Exam | Test CKAD Guide - Easy to Pass CKAD: Linux Foundation Certified Kubernetes Application Developer Exam Exam □ Simply search for [CKAD] for free download on (www.pass4leader.com) □ □Latest CKAD Test Guide
- Hot Test CKAD Guide | Pass-Sure CKAD Latest Test Discount: Linux Foundation Certified Kubernetes Application Developer Exam □ Easily obtain free download of (CKAD) by searching on ☀ www.pdfvce.com □☀□ □Latest CKAD Test Sample
- Test CKAD Guide - Efficient CKAD Latest Test Discount and First-Grade Reliable Linux Foundation Certified Kubernetes Application Developer Exam Exam Sims □ Search for ⇒ CKAD ⇐ on ⇒ www.real4dumps.com ⇐ immediately to obtain a free download □CKAD Exam Quiz
- Actual CKAD Test Material Makes You More Efficient - Pdfvce □ Simply search for □ CKAD □ for free download on ⇒ www.pdfvce.com ⇐ □CKAD Valid Guide Files
- CKAD Well Prep □ Exam Cram CKAD Pdf □ Latest CKAD Exam Papers □ Open ► www.prep4away.com ◄ and search for “ CKAD ” to download exam materials for free □Latest CKAD Exam Papers
- CKAD Study Plan □ CKAD Exam Study Guide □ CKAD Well Prep ► Search for { CKAD } and download exam materials for free through ☀ www.pdfvce.com □☀□ □New Brindumps CKAD Book
- Exam CKAD Certification Cost □ Exam Sample CKAD Online □ CKAD Latest Demo □ Open website □ www.exams4collection.com □ and search for ➡ CKAD □ for free download □CKAD Well Prep
- myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, profstudyhub.com, mikemil988.blogadvize.com, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, hrs.iccollege.uk, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, shortcourses.russellcollege.edu.au, skillrising.in, Disposable vapes

2025 Latest Braindumpsqa CKAD PDF Dumps and CKAD Exam Engine Free Share: <https://drive.google.com/open?id=1ChucDbuAO8LW9BXAGOiOdcdqYIWtyg-OO>