CKAD How the Certified Kubernetes Application Developer exam works, what each domain covers, and a workflow that gets you through two hours of hands-on tasks.

CKAD exam guide

8 sections

PREPARED FOR
You

GENERATED
27 November 2026

VERSION
v0.2.449

IN SHORT

The Certified Kubernetes Application Developer (CKAD) exam gives you a terminal, a few Kubernetes clusters and a list of tasks. A grader then checks whether each cluster ends up in the state the task asked for. There are no multiple-choice questions. This guide covers how the exam runs, what the five domains ask of you, and a way of working that fits the time.

1 The format

- **Online and proctored.** A proctor watches through your webcam and checks your room and desk before you start.
- **About two hours** of hands-on tasks (15 to 20), graded on what the clusters look like at the end. The Linux Foundation does not publish how tasks are weighted or how a partly finished one scores, so finish what you can and check it.
- **Performance-based.** You create, fix and change real objects: Deployments, Services, probes, ConfigMaps, Jobs.
- **What changes over time:** the pass mark, the number of tasks, the Kubernetes version, the price and the retake policy. They are in the official CKAD page and the candidate handbook. Check them when you book and again the week before.

2 The environment

You work in a remote desktop with a terminal and a browser. The browser opens the Kubernetes documentation (`kubernetes.io/docs`) and the few other sites the handbook allows. Your own notes and bookmarks are not available, so practice finding pages in the docs by searching them.

Each task is done on a designated host. The task's infobox gives the `ssh` command: run it first, every time, do the task there, and `exit` back to the base system (hostname `base`) before the next task. Nested ssh is not supported.

```bash
ssh <host-from-the-task>
# do the task here
exit
```

The task hosts have `kubectl` with the `k` alias and completion, `yq` , `curl` and `wget` ; `base` has none of them. Work on the wrong host and the grader will not find it, however good it is. (From the Linux Foundation's important instructions for the CKA and CKAD, checked 20 Nov 2026.) When a task names a namespace, set it on the context so you cannot forget it:

```bash
kubectl config set-context --current --namespace=<namespace>
```

3 The domains and their weights

Domain	Weight	What it asks of you
Application Environment, Configuration and Security	25%	ConfigMaps and Secrets, requests and limits, ResourceQuotas and LimitRanges, ServiceAccounts, security contexts
Application Design and Build	20%	Images, choosing the right workload (Deployment, DaemonSet, Job, CronJob), init containers and sidecars, volumes
Application Deployment	20%	Rolling updates and rollbacks, blue/green and canary with ordinary Deployments, Helm and Kustomize
Services and Networking	20%	Services, Ingress, NetworkPolicies, reaching an application from inside and outside the cluster
Application Observability and Maintenance	15%	Probes, logs, `describe` and events, resource use, deprecated API versions

The weights come from the CNCF curriculum; the official page has the current version.

4 The first minute

Set up the shell before the first task. Check what is already there with `alias`, then add what is missing:

```bash
alias k=kubectl
source <(kubectl completion bash)
complete -o default -F __start_kubectl k
export do="--dry-run=client -o yaml"
export now="--force --grace-period 0"
```

And for vim, so pasted YAML keeps its indentation:

```bash
cat >> ~/.vimrc <<'VIM'
set expandtab tabstop=2 shiftwidth=2
VIM
```

5 A workflow for each task

1. **Read the whole task** before typing. Note the context, the namespace, the names and every number.
2. **Switch context** with the command the task gives.
3. **Generate, do not write, YAML.** `k create deploy web --image=nginx --replicas=3 $do > 4.yaml` gives a valid start; add what the generator cannot (a probe, a volume, a security context) in vim.

4. **Apply and check.** `k apply -f 4.yaml`, then look at the result the way the grader will: `k get`, `k describe`, `k logs`.
5. **Move on.** If a task is not working after five minutes, flag it in the exam's notepad and come back. An easy task later is worth the same as this one.

Name files after the task number so you can find them again.

6 Commands worth knowing by heart

```bash
k run tmp --image=busybox --rm -it --restart=Never -- sh   # a throwaway Pod to test
  from
k create deploy web --image=nginx $do                       # a Deployment to edit
k expose deploy web --port=80 --target-port=8080            # a Service for it
k set image deploy/web nginx=nginx:1.27                     # change the image
k rollout status deploy/web && k rollout undo deploy/web    # watch, then roll back
k create job once --image=busybox $do -- date               # a Job
k create cronjob tick --image=busybox --schedule="*/5 * * * *" $do -- date
k create configmap app --from-literal=MODE=prod $do
k create secret generic db --from-literal=PASSWORD=s3cret $do
k explain pod.spec.containers.livenessProbe                  # the field names,
  offline
```

`k explain` is the fastest way to recall a field's name and where it goes, and it never leaves the terminal.

7 In the last ten minutes

- Go back to the tasks you flagged, easiest first.
- Check that each object is in the namespace the task named: `k get <kind> -A | grep <name>`.
- Do not start rewriting a task that already works.

8 How to practice

Reading helps, but the exam rewards people who have typed the commands many times under a clock. Work through the Gazeek CKAD labs in order: Learn shows every command, Practice gives you the goal and a nudge, Prove gives you only the goal, as the exam does. The Memory Gym's CKAD sets keep the facts fresh between labs.

From https://gazeek.com/resources/ckad-exam-guide. For your own practice: the commands need a Kubernetes cluster to run against.