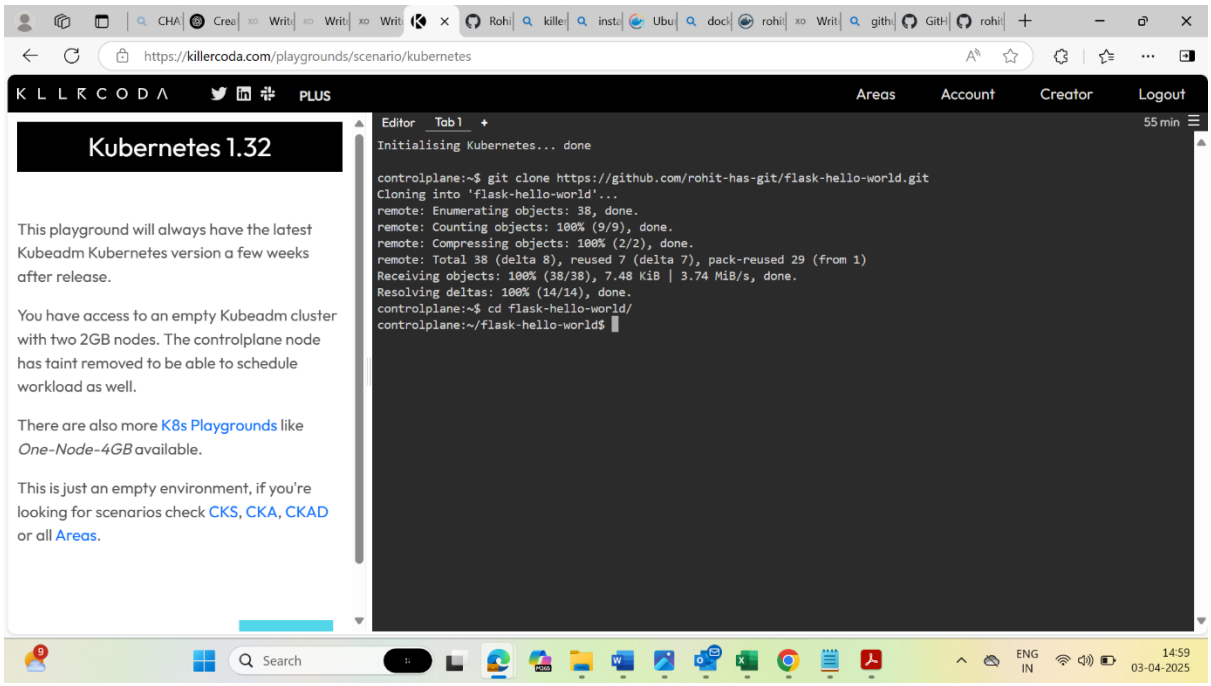
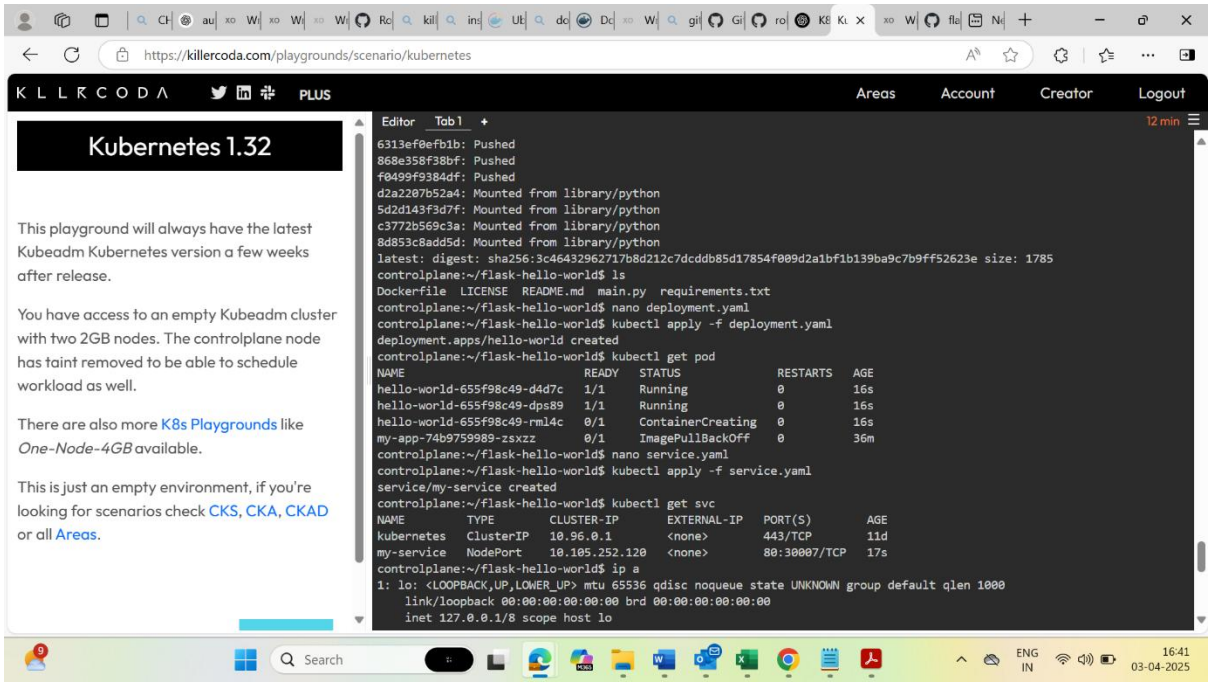
Question 1: Containerize and Push the Application

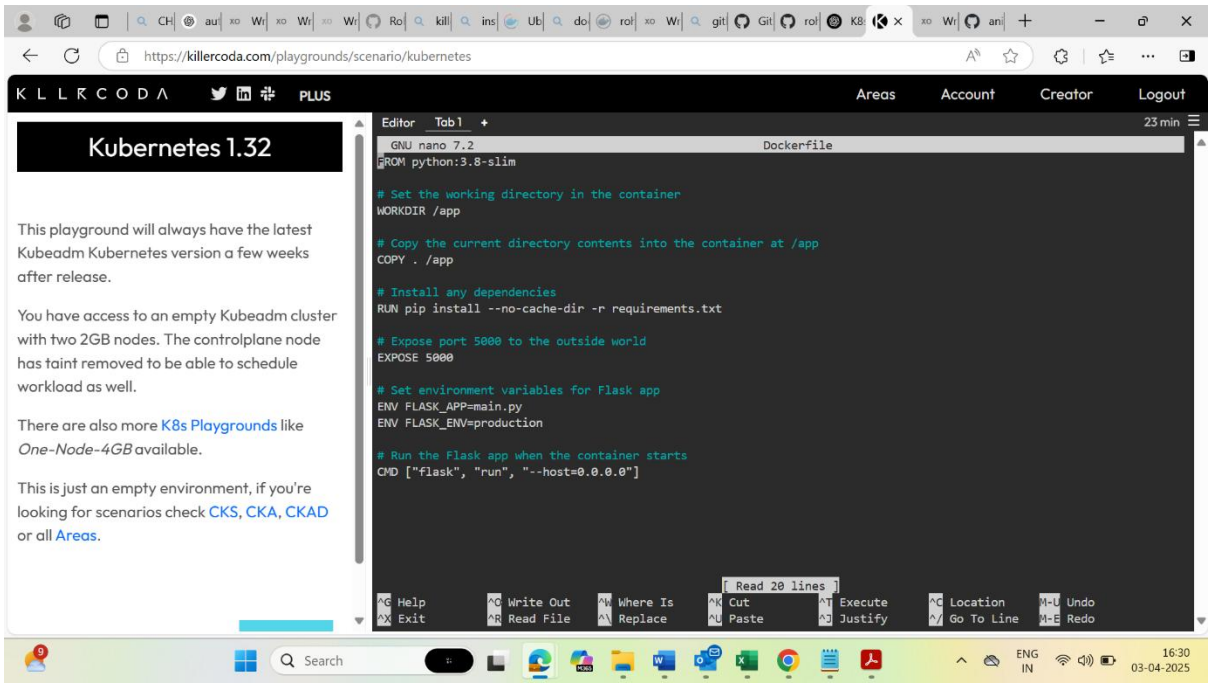
• Fork and clone a sample application repository (Java, Node.js, or Python preferred). • Package the application as required (e.g., JAR, ZIP, etc.). • Write a Dockerfile to containerize the application. • Build the Docker image and push it to DockerHub.

GithubLink

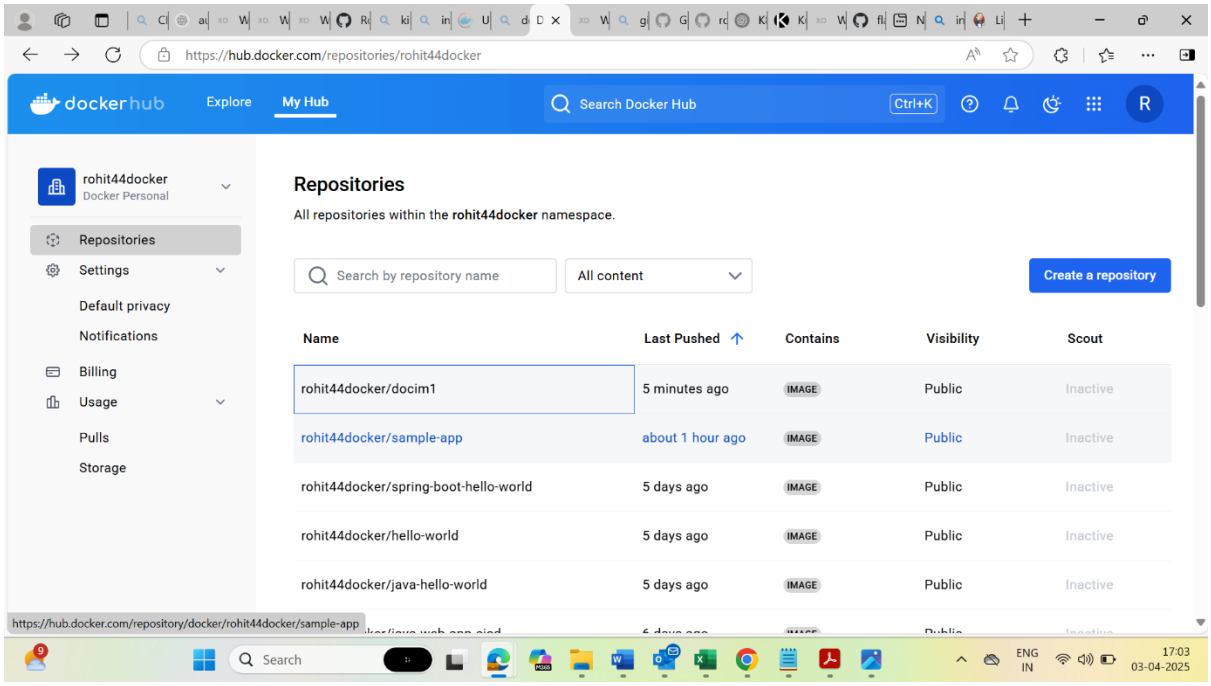




This is Dockerfile



This is Docker Image

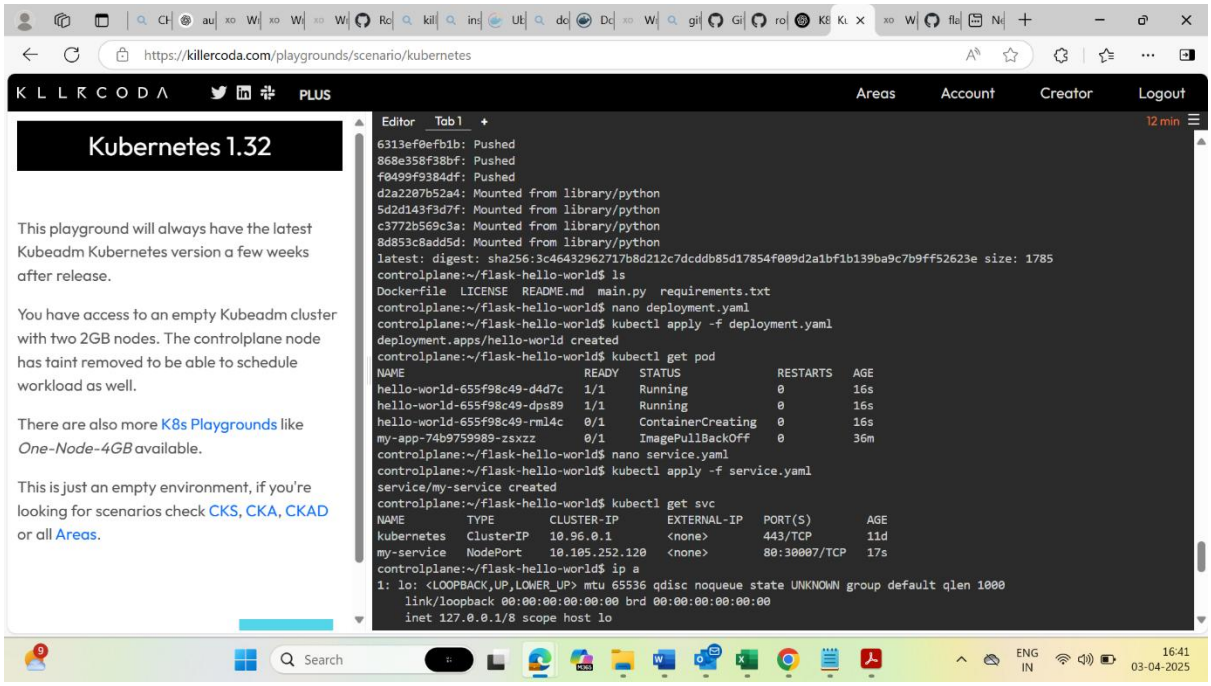


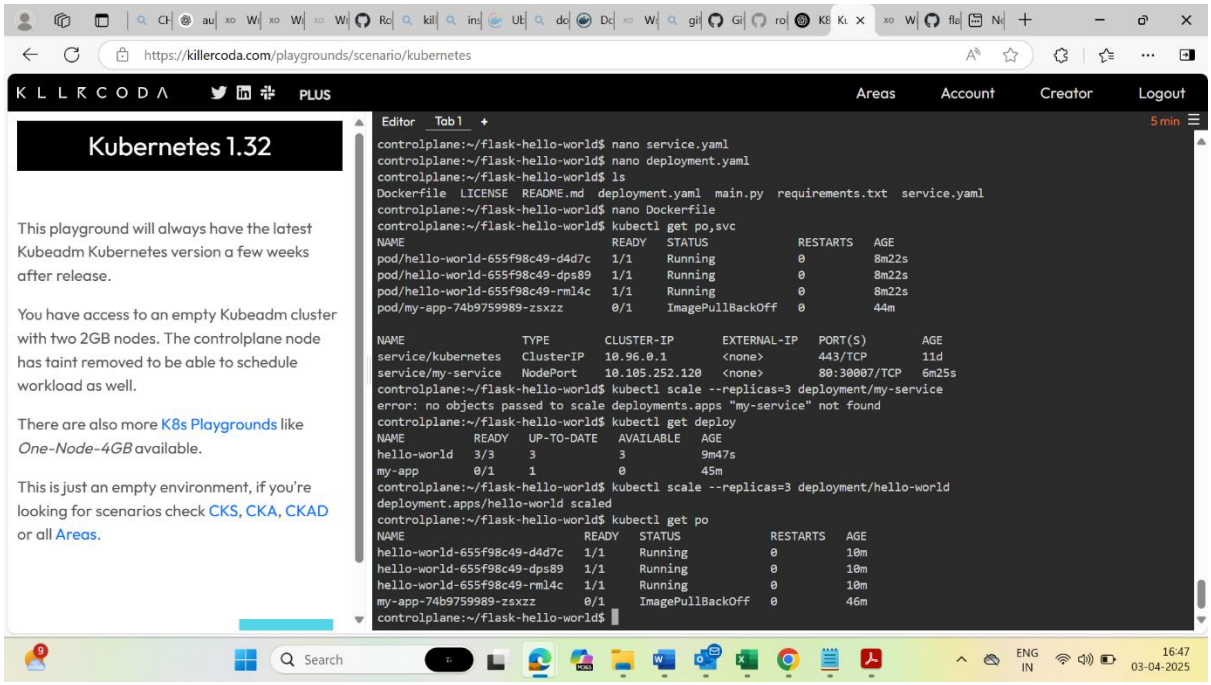
Docker Image link below :

Link

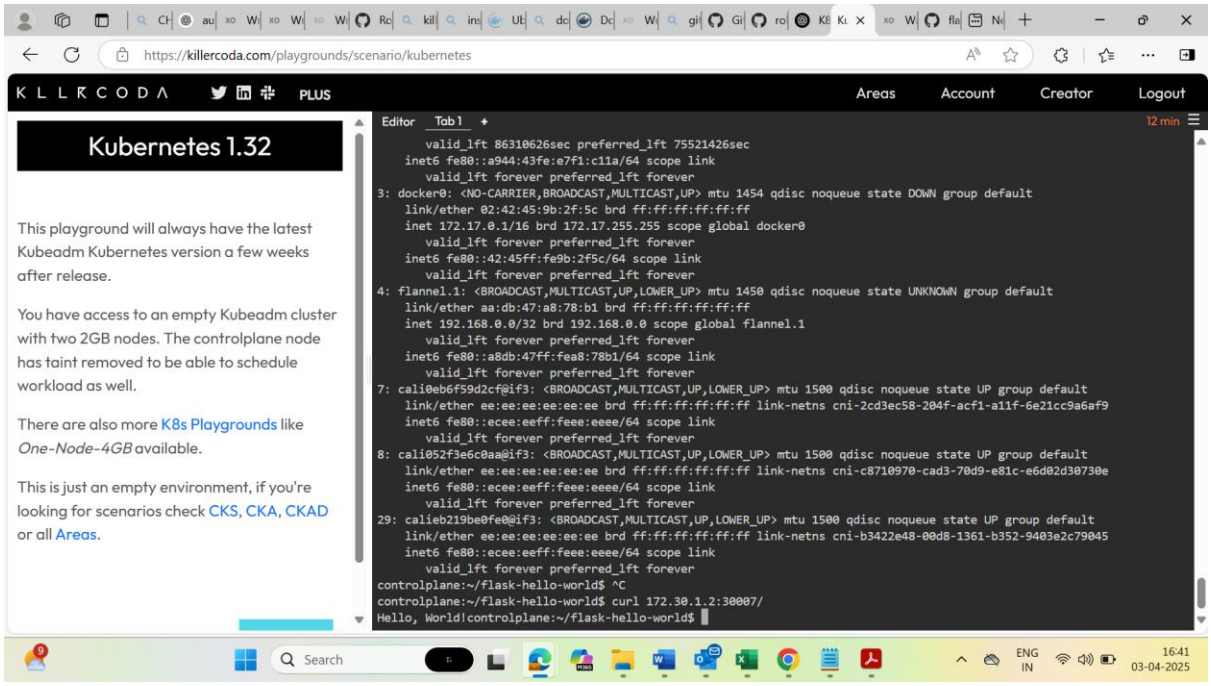
Question 2: Kubernetes Deployment and Service Exposure

• Create Kubernetes `deployment.yaml` to deploy your Docker image. • Create `service.yaml` to expose it using NodePort. • Apply both YAMLs using `kubectl`, and verify deployment. • Use `curl` with public IP and node port to access the service.


https://killercoda.com/playgrounds/scenario/kubernetes
KILLERCODA
PLUS
Areas
Account
Creator
Logout
Kubernetes 1.32
This playground will always have the latest Kubeadm Kubernetes version a few weeks after release.
You have access to an empty Kubeadm cluster with two 2GB nodes. The controlplane node has taint removed to be able to schedule workload as well.
There are also more K8s Playgrounds like One-Node-4GB available.
This is just an empty environment, if you're looking for scenarios check CKS, CKA, CKAD or all Areas.
Editor
Tab 1
12 min
6313ef0efb1b: Pushed
868e358f38bf: Pushed
f0499f9384df: Pushed
d2a2207b52a4: Mounted from library/python
5d2d143f3d7f: Mounted from library/python
c3772b569c3a: Mounted from library/python
8d853c8add5d: Mounted from library/python
latest: digest: sha256:3c46432962717b8d212c7dcddb85d17854f009d2a1bf1b139ba9c7b9ff52623e size: 1785
controlplane:~/flask-hello-world$ ls
Dockerfile LICENSE README.md main.py requirements.txt
controlplane:~/flask-hello-world$ nano deployment.yaml
controlplane:~/flask-hello-world$ kubectl apply -f deployment.yaml
deployment.apps/hello-world created
controlplane:~/flask-hello-world$ kubectl get pod
NAME READY STATUS RESTARTS AGE
hello-world-655f98c49-d4d7c 1/1 Running 0 16s
hello-world-655f98c49-dps89 1/1 Running 0 16s
hello-world-655f98c49-rml4c 0/1 ContainerCreating 0 16s
my-app-74b9759989-zsxzz 0/1 ImagePullBackOff 0 36m
controlplane:~/flask-hello-world$ nano service.yaml
controlplane:~/flask-hello-world$ kubectl apply -f service.yaml
service/my-service created
controlplane:~/flask-hello-world$ kubectl get svc
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 11d
my-service NodePort 10.105.252.120 <none> 80:30007/TCP 17s
controlplane:~/flask-hello-world$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
inet 127.0.0.1/8 scope host lo
Search
ENG
IN
16:41
03-04-2025

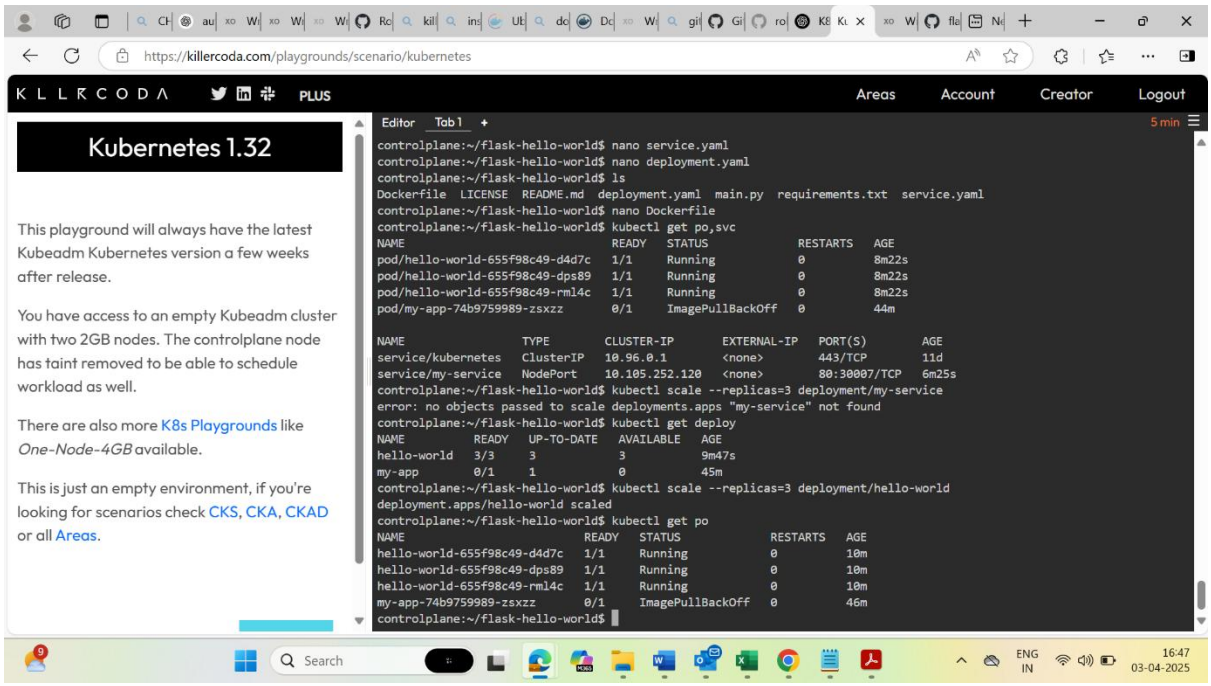

https://killercoda.com/playgrounds/scenario/kubernetes
KILLERCODA
PLUS
Areas
Account
Creator
Logout
Kubernetes 1.32
This playground will always have the latest Kubeadm Kubernetes version a few weeks after release.
You have access to an empty Kubeadm cluster with two 2GB nodes. The controlplane node has taint removed to be able to schedule workload as well.
There are also more K8s Playgrounds like One-Node-4GB available.
This is just an empty environment, if you're looking for scenarios check CKS, CKA, CKAD or all Areas.
Editor
Tab 1
5 min
controlplane:~/flask-hello-world$ nano service.yaml
controlplane:~/flask-hello-world$ nano deployment.yaml
controlplane:~/flask-hello-world$ ls
Dockerfile LICENSE README.md deployment.yaml main.py requirements.txt service.yaml
controlplane:~/flask-hello-world$ nano Dockerfile
controlplane:~/flask-hello-world$ kubectl get po,svc
NAME READY STATUS RESTARTS AGE
pod/hello-world-655f98c49-d4d7c 1/1 Running 0 8m22s
pod/hello-world-655f98c49-dps89 1/1 Running 0 8m22s
pod/hello-world-655f98c49-rml4c 1/1 Running 0 8m22s
pod/my-app-74b9759989-zsxzz 0/1 ImagePullBackOff 0 44m
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
service/kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 11d
service/my-service NodePort 10.105.252.120 <none> 80:30007/TCP 6m25s
controlplane:~/flask-hello-world$ kubectl scale --replicas=3 deployment/my-service
error: no objects passed to scale deployments.apps "my-service" not found
controlplane:~/flask-hello-world$ kubectl get deploy
NAME READY UP-TO-DATE AVAILABLE AGE
hello-world 3/3 3 3 9m47s
my-app 0/1 1 0 45m
controlplane:~/flask-hello-world$ kubectl scale --replicas=3 deployment/hello-world
deployment.apps/hello-world scaled
controlplane:~/flask-hello-world$ kubectl get po
NAME READY STATUS RESTARTS AGE
hello-world-655f98c49-d4d7c 1/1 Running 0 10m
hello-world-655f98c49-dps89 1/1 Running 0 10m
hello-world-655f98c49-rml4c 1/1 Running 0 10m
my-app-74b9759989-zsxzz 0/1 ImagePullBackOff 0 46m
controlplane:~/flask-hello-world$
Search
ENG
IN
16:47
03-04-2025

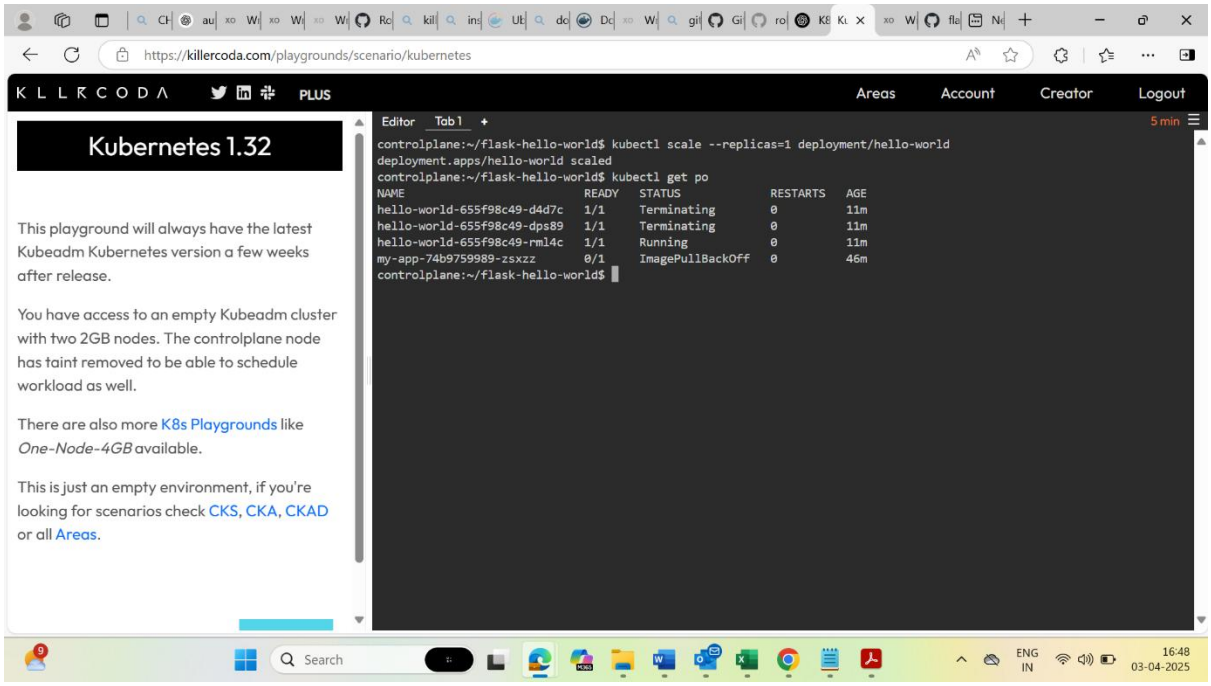
Curl



Question 3: Scale the Deployment

• Scale the application to 3 replicas using `kubectl scale`. • Validate scaling with `kubectl get pods`. • Scale it back down to 1 replica.





Question 4: Automate Deployment with Jenkins

Installation of jenkins