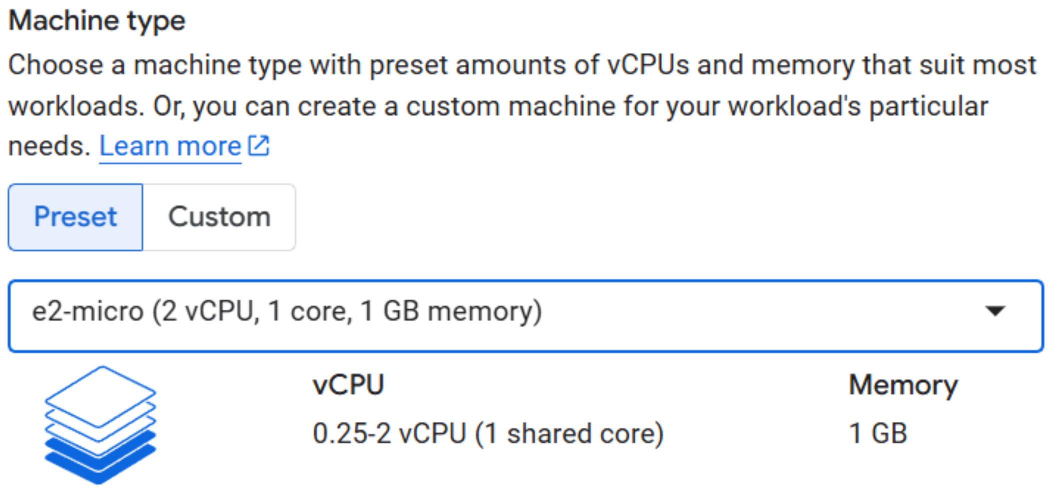
Kubernetes (GKE) Practical Lab Guide

✓ Task 1 – Create a GKE Cluster & Connect via Console

Steps 1 To Create GKE Cluster

1. Go to **Kubernetes Engine > Clusters**.
2. Click “**Create”** > Choose “**Standard”**.
3. Configure the following:
    - **Cluster Name**: my-Cluster1
    - **Location Type**: Zonal (e.g., us-central1-a)
    - **Number of Nodes**: 3



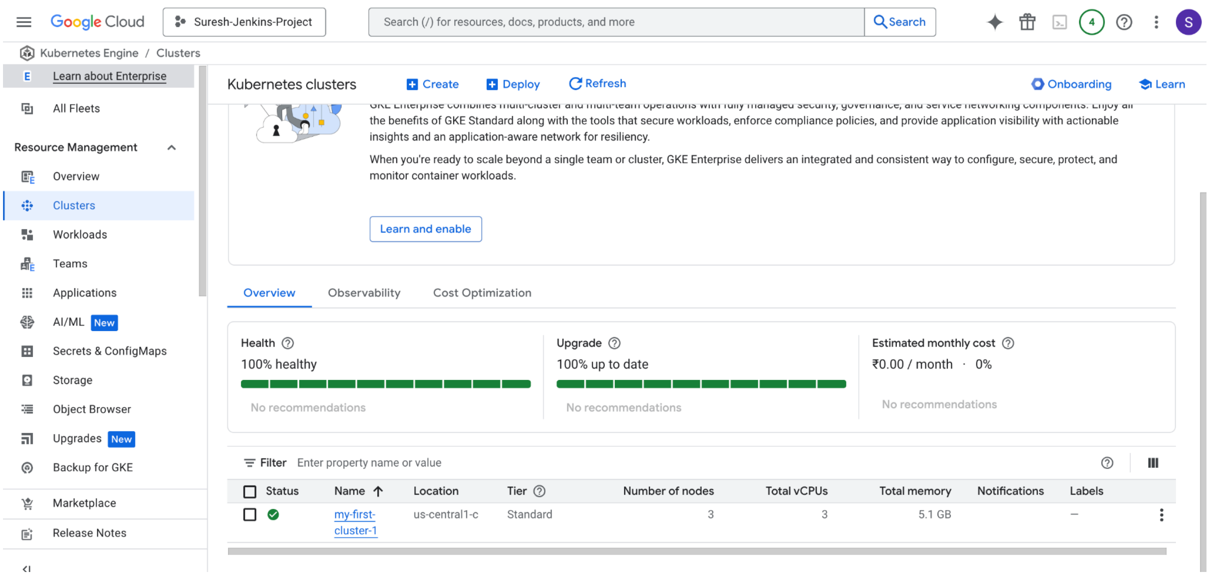
Machine type

Choose a machine type with preset amounts of vCPUs and memory that suit most workloads. Or, you can create a custom machine for your workload's particular needs. Learn more

e2-micro (2 vCPU, 1 core, 1 GB memory)

vCPU	Memory
0.25-2 vCPU (1 shared core)	1 GB

4. Click “**Create”** and wait for the cluster to be provisioned.



GKE Enterprise combines multi-cluster and multi-team operations with fully managed security, governance, and service networking components. Enjoy all the benefits of GKE Standard along with the tools that secure workloads, enforce compliance policies, and provide application visibility with actionable insights and an application-aware network for resiliency.

When you're ready to scale beyond a single team or cluster, GKE Enterprise delivers an integrated and consistent way to configure, secure, protect, and monitor container workloads.

Health: 100% healthy — No recommendations

Upgrade: 100% up to date — No recommendations

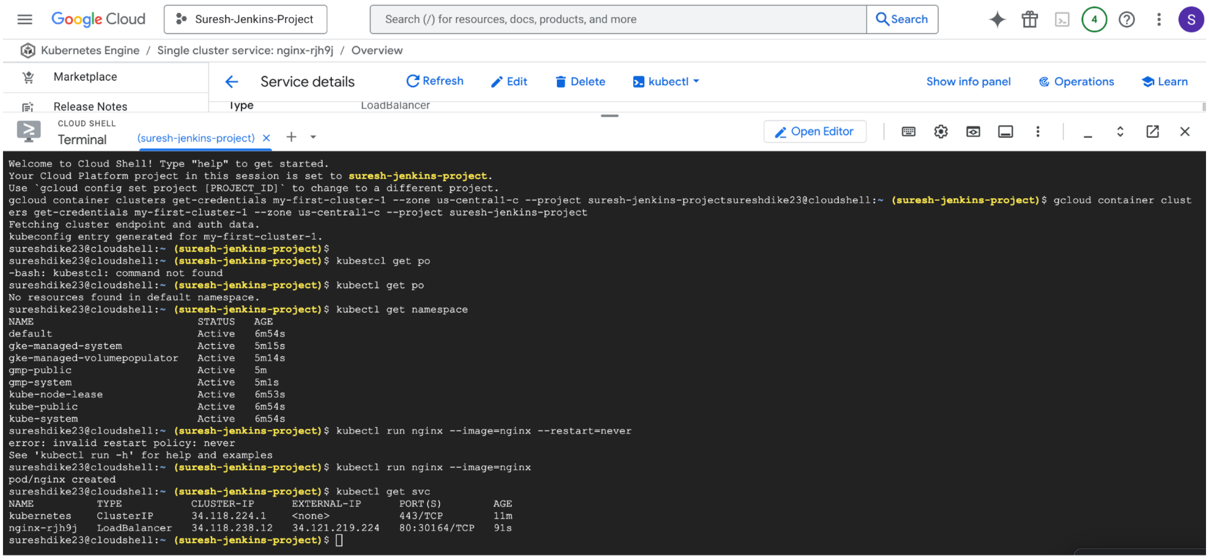
Estimated monthly cost: ₹0.00 / month · 0% — No recommendations

Filter Enter property name or value

□	Status	Name ↑	Location	Tier	Number of nodes	Total vCPUs	Total memory	Notifications	Labels
□	✓	my-first-cluster-1	us-central1-c	Standard	3	3	5.1 GB		–

Step 2: Connect to GKE Cluster

- Once the cluster is ready, click on its name.
- Click “**Connect”** > “**Run in Cloud Shell”**.
- This will automatically run the following command:
  gcloud container clusters get-credentials my-Cluster1 --zone us-central1-a



```
Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to suresh-jenkins-project.
Use `gcloud config set project [PROJECT_ID]` to change to a different project.
gcloud container clusters get-credentials my-first-cluster-1 --zone us-central1-c --project suresh-jenkins-projectsureshdike23@cloudshell:~ (suresh-jenkins-project)$ gcloud container clust
ers get-credentials my-first-cluster-1 --zone us-central1-c --project suresh-jenkins-project
Fetching cluster endpoint and auth data.
kubeconfig entry generated for my-first-cluster-1.
sureshdike23@cloudshell:~ (suresh-jenkins-project)$
sureshdike23@cloudshell:~ (suresh-jenkins-project)$ kubestcl get po
-bash: kubestcl: command not found
sureshdike23@cloudshell:~ (suresh-jenkins-project)$ kubectl get po
No resources found in default namespace.
sureshdike23@cloudshell:~ (suresh-jenkins-project)$ kubectl get namespace
NAME                              STATUS   AGE
default                           Active   6m54s
gke-managed-system                Active   5m15s
gke-managed-volumepopulator       Active   5m14s
gmp-public                        Active   5m
gmp-system                        Active   5m1s
kube-node-lease                   Active   6m53s
kube-public                       Active   6m54s
kube-system                       Active   6m54s
sureshdike23@cloudshell:~ (suresh-jenkins-project)$ kubectl run nginx --image=nginx --restart=never
error: invalid restart policy: never
See 'kubectl run -h' for help and examples
sureshdike23@cloudshell:~ (suresh-jenkins-project)$ kubectl run nginx --image=nginx
pod/nginx created
sureshdike23@cloudshell:~ (suresh-jenkins-project)$ kubectl get svc
NAME          TYPE           CLUSTER-IP      EXTERNAL-IP      PORT(S)        AGE
kubernetes    ClusterIP      34.118.224.1    <none>           443/TCP        11m
nginx-rjh9j   LoadBalancer   34.118.238.12   34.121.219.224   80:30164/TCP   91s
sureshdike23@cloudshell:~ (suresh-jenkins-project)$ 
```

✓ Task 2 – Deploy Nginx Pod & Expose It on Port 80

Step 1: Create Nginx Pod

kubectl run nginx --image=nginx --restart=Never

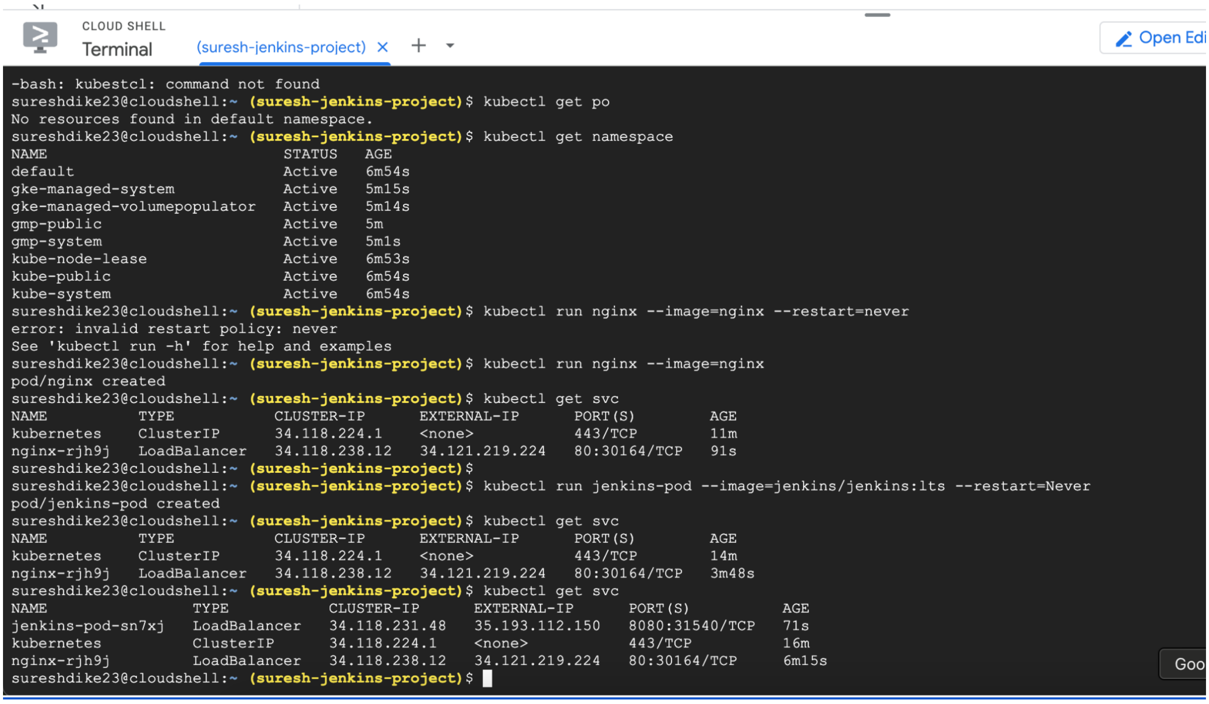
Step 2: Expose the Pod using LoadBalancer

kubectl expose pod nginx-pod --type=LoadBalancer --port=80

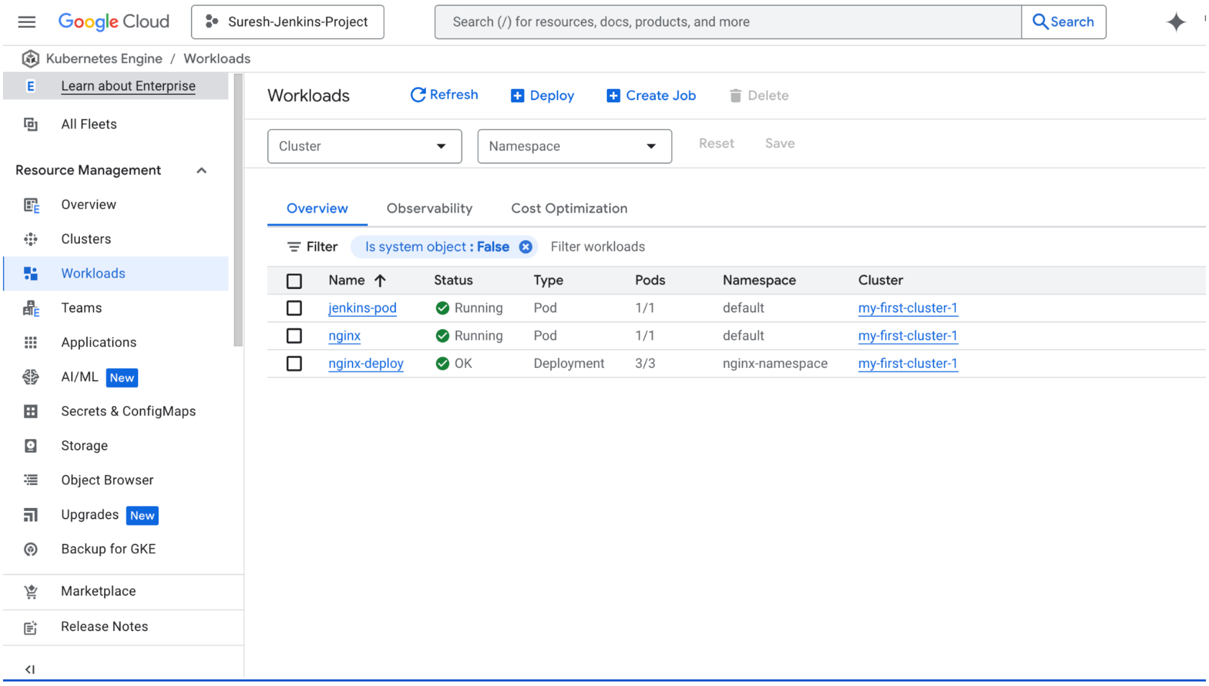
Step 3: Get External IP

kubectl get svc

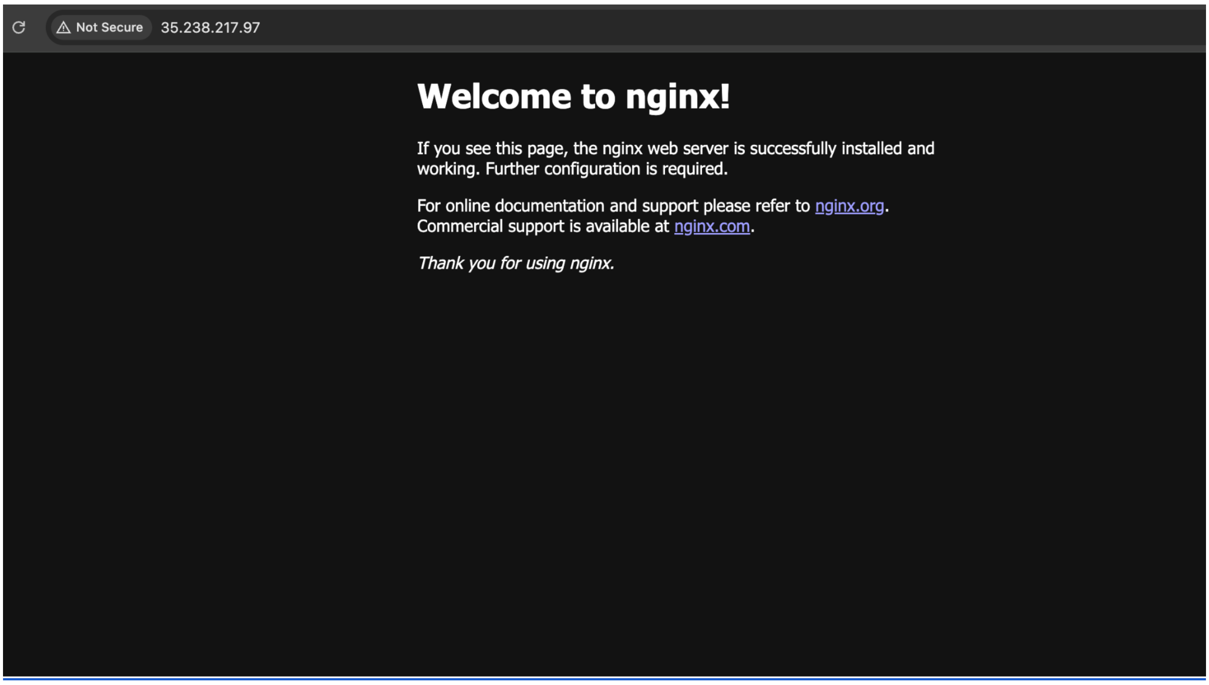
- Wait for EXTERNAL-IP to be assigned (may take 1-2 mins).
- Access your Nginx app using:
  http://<EXTERNAL-IP>



```
-bash: kubectl: command not found
sureshdike23@cloudshell:~ (suresh-jenkins-project)$ kubectl get po
No resources found in default namespace.
sureshdike23@cloudshell:~ (suresh-jenkins-project)$ kubectl get namespace
NAME                              STATUS   AGE
default                           Active   6m54s
gke-managed-system                Active   5m15s
gke-managed-volumepopulator       Active   5m14s
gmp-public                        Active   5m
gmp-system                        Active   5m1s
kube-node-lease                   Active   6m53s
kube-public                       Active   6m54s
kube-system                       Active   6m54s
sureshdike23@cloudshell:~ (suresh-jenkins-project)$ kubectl run nginx --image=nginx --restart=never
error: invalid restart policy: never
See 'kubectl run -h' for help and examples
sureshdike23@cloudshell:~ (suresh-jenkins-project)$ kubectl run nginx --image=nginx
pod/nginx created
sureshdike23@cloudshell:~ (suresh-jenkins-project)$ kubectl get svc
NAME          TYPE           CLUSTER-IP       EXTERNAL-IP      PORT(S)        AGE
kubernetes    ClusterIP      34.118.224.1     <none>           443/TCP        11m
nginx-rjh9j   LoadBalancer   34.118.238.12    34.121.219.224   80:30164/TCP   91s
sureshdike23@cloudshell:~ (suresh-jenkins-project)$
sureshdike23@cloudshell:~ (suresh-jenkins-project)$ kubectl run jenkins-pod --image=jenkins/jenkins:lts --restart=Never
pod/jenkins-pod created
sureshdike23@cloudshell:~ (suresh-jenkins-project)$ kubectl get svc
NAME          TYPE           CLUSTER-IP       EXTERNAL-IP      PORT(S)        AGE
kubernetes    ClusterIP      34.118.224.1     <none>           443/TCP        14m
nginx-rjh9j   LoadBalancer   34.118.238.12    34.121.219.224   80:30164/TCP   3m48s
sureshdike23@cloudshell:~ (suresh-jenkins-project)$ kubectl get svc
NAME                TYPE           CLUSTER-IP       EXTERNAL-IP      PORT(S)          AGE
jenkins-pod-sn7xj   LoadBalancer   34.118.231.48    35.193.112.150   8080:31540/TCP   71s
kubernetes          ClusterIP      34.118.224.1     <none>           443/TCP          16m
nginx-rjh9j         LoadBalancer   34.118.238.12    34.121.219.224   80:30164/TCP     6m15s
sureshdike23@cloudshell:~ (suresh-jenkins-project)$
```



□	Name ↑	Status	Type	Pods	Namespace	Cluster
□	jenkins-pod	✓ Running	Pod	1/1	default	my-first-cluster-1
□	nginx	✓ Running	Pod	1/1	default	my-first-cluster-1
□	nginx-deploy	✓ OK	Deployment	3/3	nginx-namespace	my-first-cluster-1



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

✓ Task 3 – Deploy a Jenkins Pod & Expose It on Port 8080

Step 1: Deploy Jenkins Pod

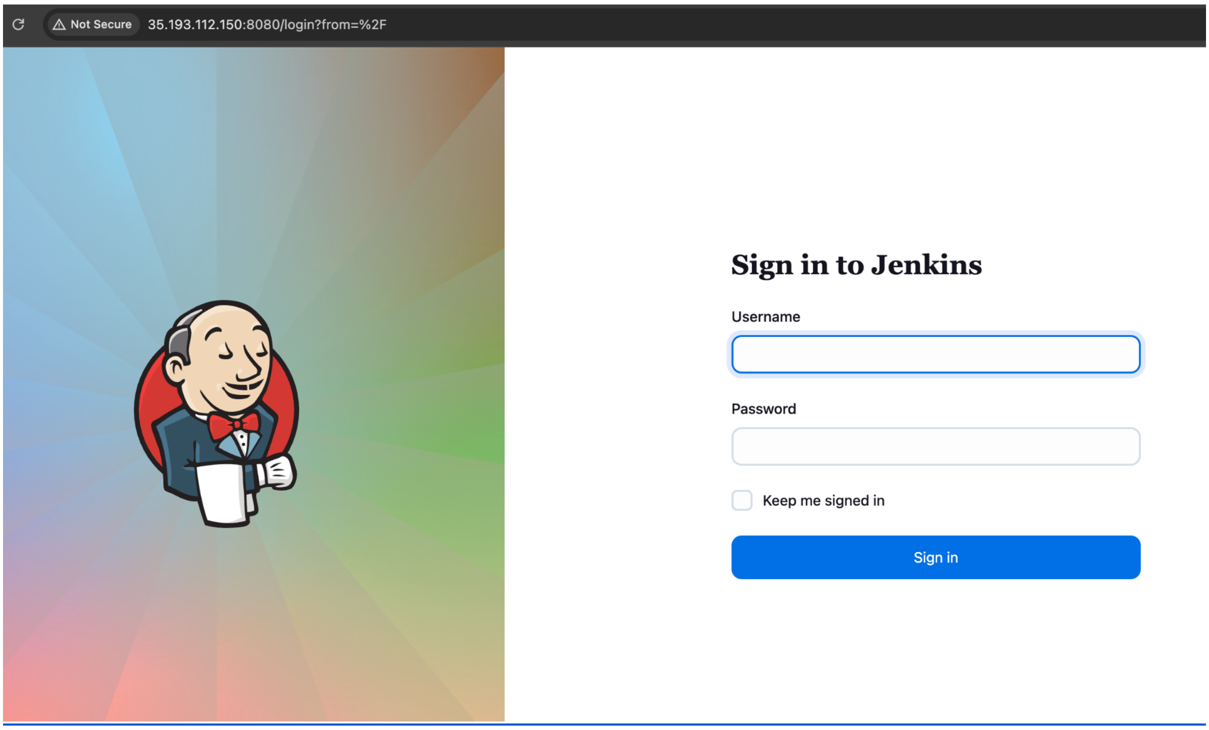
kubectl run jenkins --image=jenkins/jenkins:lts --restart=Never

Step 2: Expose the Pod as LoadBalancer

kubectl expose pod jenkins-pod --type=LoadBalancer --port=8080

Step 3: Access Jenkins UI

1. Run:
   kubectl get svc
2. Get the **NodePort** and **External IP** of your cluster node.
3. Access Jenkins in the browser:
   http://<EXTERNAL-IP>:<NODE-PORT>



✔ Task 4 – Create an Nginx Deployment & Delete One Pod

Step 1: Create Nginx Deployment

kubectl create deployment nginx-deploy --image=nginx

Step 2: Check Running Pods

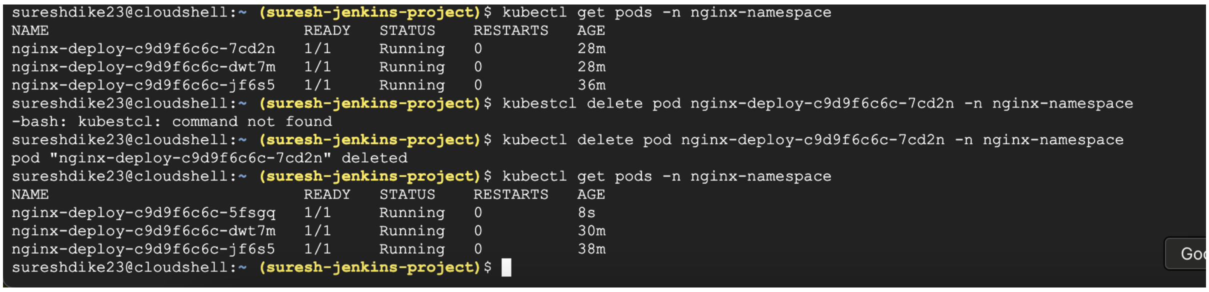
kubectl get pods

Step 3: Delete One Pod from the Deployment

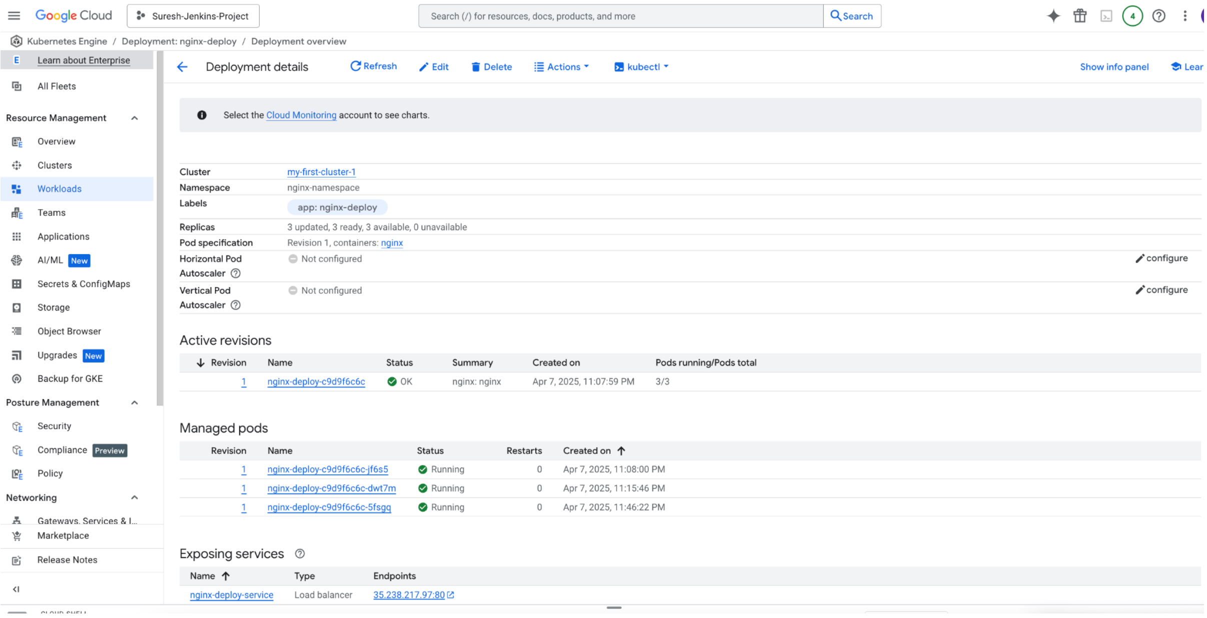
Step 4: Observe Auto-Healing

kubectl get pods

- The deleted pod will be automatically recreated by the deployment controller.



```
sureshdike23@cloudshell:~ (suresh-jenkins-project)$ kubectl get pods -n nginx-namespace
NAME                            READY   STATUS    RESTARTS   AGE
nginx-deploy-c9d9f6c6c-7cd2n    1/1     Running   0          28m
nginx-deploy-c9d9f6c6c-dwt7m    1/1     Running   0          28m
nginx-deploy-c9d9f6c6c-jf6s5    1/1     Running   0          36m
sureshdike23@cloudshell:~ (suresh-jenkins-project)$ kubestcl delete pod nginx-deploy-c9d9f6c6c-7cd2n -n nginx-namespace
-bash: kubestcl: command not found
sureshdike23@cloudshell:~ (suresh-jenkins-project)$ kubectl delete pod nginx-deploy-c9d9f6c6c-7cd2n -n nginx-namespace
pod "nginx-deploy-c9d9f6c6c-7cd2n" deleted
sureshdike23@cloudshell:~ (suresh-jenkins-project)$ kubectl get pods -n nginx-namespace
NAME                            READY   STATUS    RESTARTS   AGE
nginx-deploy-c9d9f6c6c-5fsgq    1/1     Running   0          8s
nginx-deploy-c9d9f6c6c-dwt7m    1/1     Running   0          30m
nginx-deploy-c9d9f6c6c-jf6s5    1/1     Running   0          38m
sureshdike23@cloudshell:~ (suresh-jenkins-project)$ 
```



Select the Cloud Monitoring account to see charts.

Cluster	my-first-cluster-1
Namespace	nginx-namespace
Labels	app: nginx-deploy
Replicas	3 updated, 3 ready, 3 available, 0 unavailable
Pod specification	Revision 1, containers: nginx
Horizontal Pod Autoscaler	Not configured
Vertical Pod Autoscaler	Not configured

Active revisions

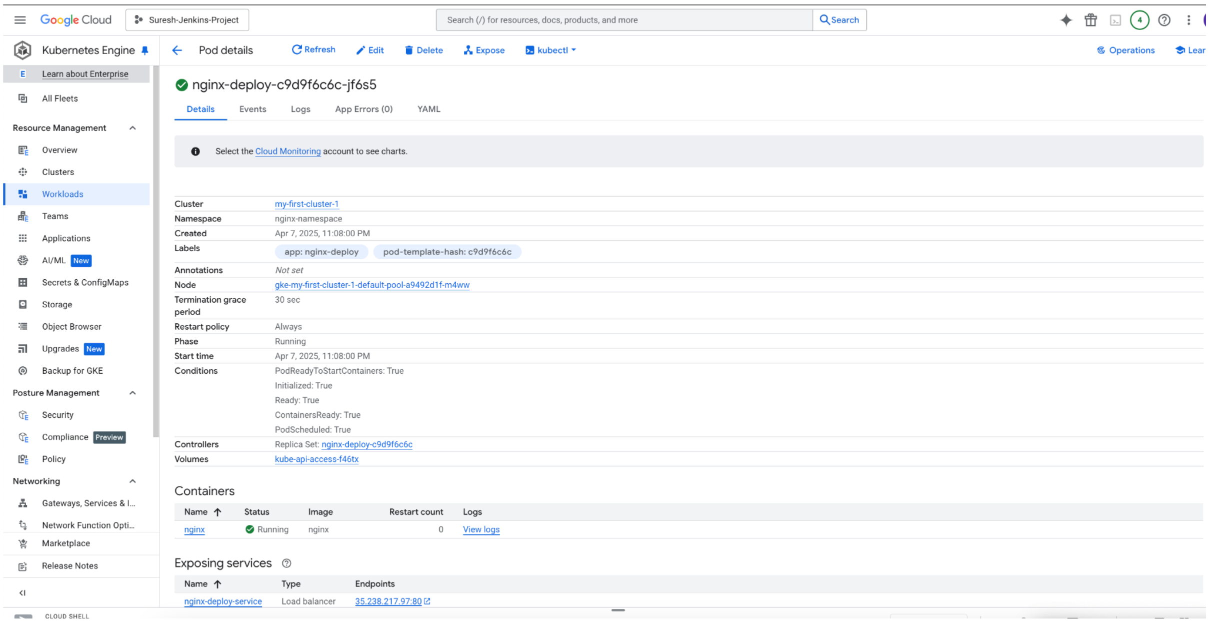
Revision	Name	Status	Summary	Created on	Pods running/Pods total
1	nginx-deploy-c9d9f6c6c	✓ OK	nginx: nginx	Apr 7, 2025, 11:07:59 PM	3/3

Managed pods

Revision	Name	Status	Restarts	Created on
1	nginx-deploy-c9d9f6c6c-jf6s5	✓ Running	0	Apr 7, 2025, 11:08:00 PM
1	nginx-deploy-c9d9f6c6c-dwt7m	✓ Running	0	Apr 7, 2025, 11:15:46 PM
1	nginx-deploy-c9d9f6c6c-5fsgq	✓ Running	0	Apr 7, 2025, 11:46:22 PM

Exposing services

Name	Type	Endpoints
nginx-deploy-service	Load balancer	35.238.217.97:80



✓ nginx-deploy-c9d9f6c6c-jf6s5

Select the Cloud Monitoring account to see charts.

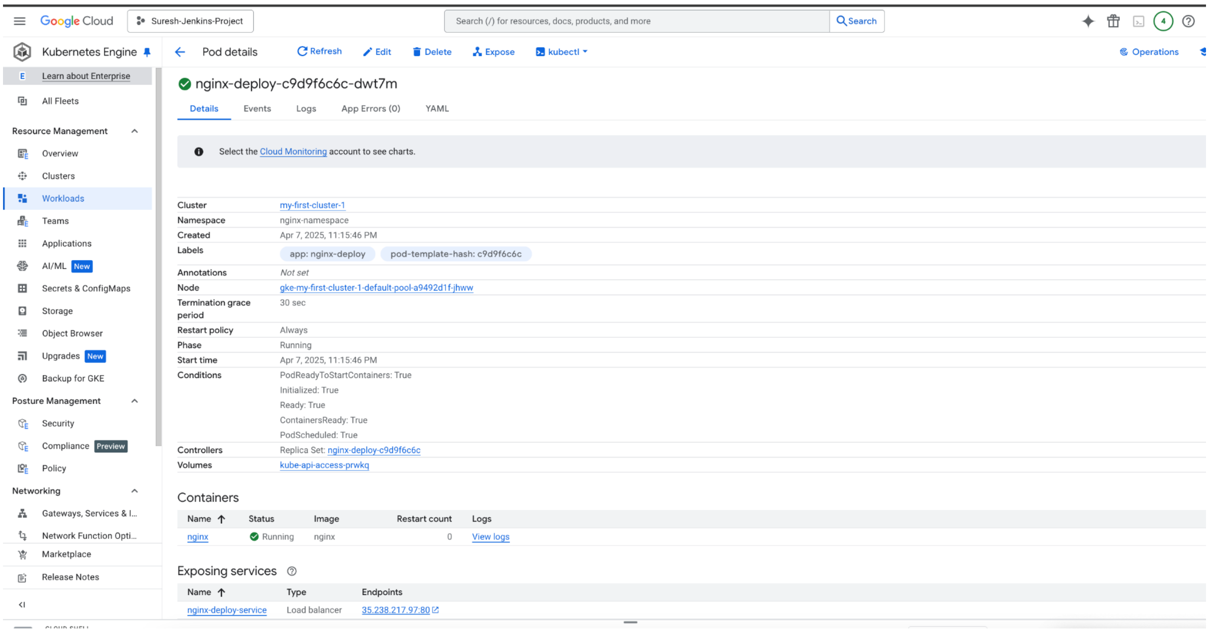
Cluster	my-first-cluster-1
Namespace	nginx-namespace
Created	Apr 7, 2025, 11:08:00 PM
Labels	app: nginx-deploy pod-template-hash: c9d9f6c6c
Annotations	*Not set*
Node	gke-my-first-cluster-1-default-pool-a9492d1f-m4ww
Termination grace period	30 sec
Restart policy	Always
Phase	Running
Start time	Apr 7, 2025, 11:08:00 PM
Conditions	PodReadyToStartContainers: True Initialized: True Ready: True ContainersReady: True PodScheduled: True
Controllers	Replica Set: nginx-deploy-c9d9f6c6c
Volumes	kube-api-access-f46tx

Containers

Name	Status	Image	Restart count	Logs
nginx	✓ Running	nginx	0	View logs

Exposing services

Name	Type	Endpoints
nginx-deploy-service	Load balancer	35.238.217.97:80



nginx-deploy-c9d9f6c6c-dwt7m

Select the Cloud Monitoring account to see charts.

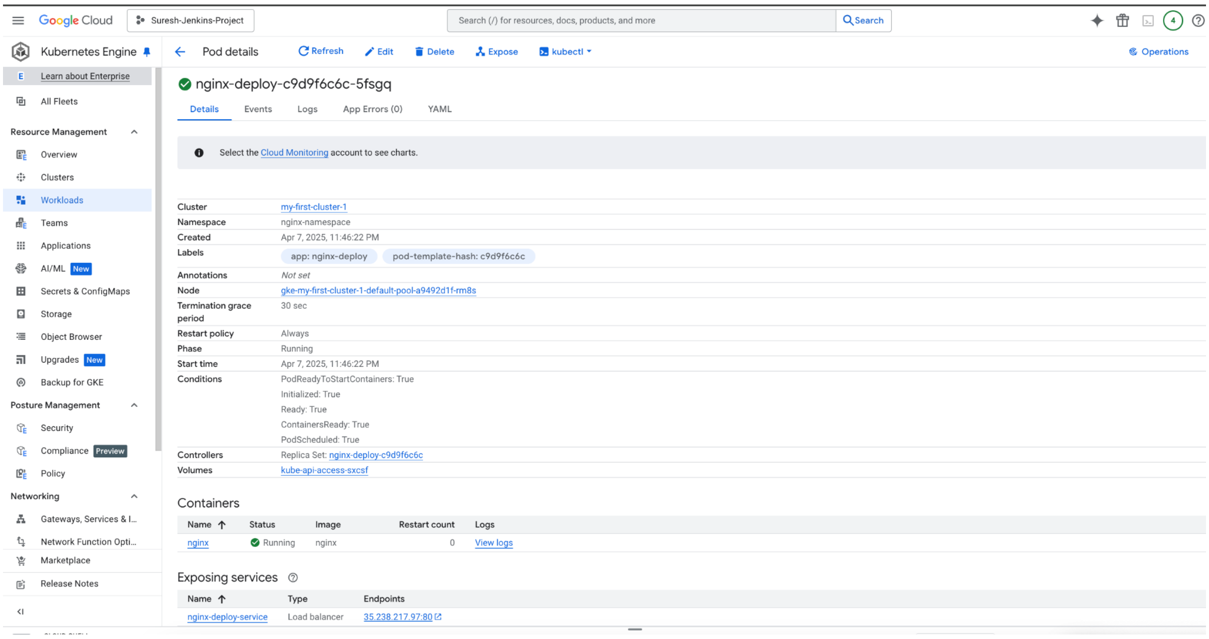
Cluster	my-first-cluster-1
Namespace	nginx-namespace
Created	Apr 7, 2025, 11:15:46 PM
Labels	app: nginx-deploy pod-template-hash: c9d9f6c6c
Annotations	*Not set*
Node	gke-my-first-cluster-1-default-pool-a9492d1f-jhww
Termination grace period	30 sec
Restart policy	Always
Phase	Running
Start time	Apr 7, 2025, 11:15:46 PM
Conditions	PodReadyToStartContainers: True Initialized: True Ready: True ContainersReady: True PodScheduled: True
Controllers	Replica Set: nginx-deploy-c9d9f6c6c
Volumes	kube-api-access-prwkq

Containers

Name ↑	Status	Image	Restart count	Logs
nginx	✓ Running	nginx	0	View logs

Exposing services

Name ↑	Type	Endpoints
nginx-deploy-service	Load balancer	35.238.217.97:80



nginx-deploy-c9d9f6c6c-5fsgq

Select the Cloud Monitoring account to see charts.

Cluster	my-first-cluster-1
Namespace	nginx-namespace
Created	Apr 7, 2025, 11:46:22 PM
Labels	app: nginx-deploy pod-template-hash: c9d9f6c6c
Annotations	*Not set*
Node	gke-my-first-cluster-1-default-pool-a9492d1f-rm8s
Termination grace period	30 sec
Restart policy	Always
Phase	Running
Start time	Apr 7, 2025, 11:46:22 PM
Conditions	PodReadyToStartContainers: True Initialized: True Ready: True ContainersReady: True PodScheduled: True
Controllers	Replica Set: nginx-deploy-c9d9f6c6c
Volumes	kube-api-access-sxcsf

Containers

Name ↑	Status	Image	Restart count	Logs
nginx	✓ Running	nginx	0	View logs

Exposing services

Name ↑	Type	Endpoints
nginx-deploy-service	Load balancer	35.238.217.97:80

✓ Task 5 – Set Up Horizontal Pod Autoscaler (HPA) in Kubernetes

Objective:

To deploy an Nginx application with CPU resource limits and configure a Horizontal Pod Autoscaler (HPA) to scale pods based on CPU usage.

Step-by-Step SOP

Step 1: Create the Nginx Deployment

Create a deployment named nginx-deploy1 with the official Nginx image:

kubectl create deployment nginx-deploy1 --image=nginx

Step 2: Set CPU Resource Requests and Limits

Apply CPU resource limits to the deployment:

kubectl set resources deployment nginx-deploy1 --limits=cpu=200m --requests=cpu=100m

This ensures that the deployment is eligible for autoscaling based on CPU.

Step 3: Apply the Horizontal Pod Autoscaler (HPA)

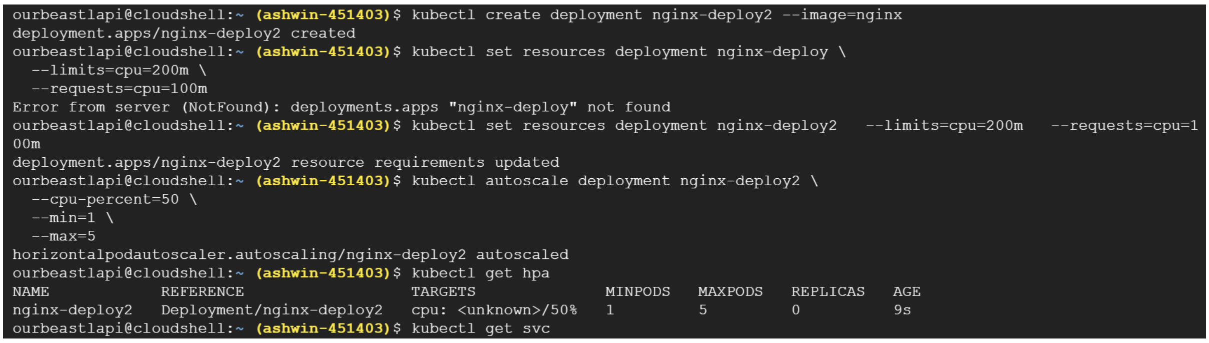
Configure HPA to scale the deployment between 1 and 5 pods if CPU usage exceeds 50%:

kubectl autoscale deployment nginx-deploy1 --cpu-percent=50 --min=1 --max=5

Step 4: Verify the HPA Status

Check if the HPA has been created and is monitoring the deployment:

kubectl get hpa



```
ourbeastlapi@cloudshell:~ (ashwin-451403)$ kubectl create deployment nginx-deploy2 --image=nginx
deployment.apps/nginx-deploy2 created
ourbeastlapi@cloudshell:~ (ashwin-451403)$ kubectl set resources deployment nginx-deploy \
  --limits=cpu=200m \
  --requests=cpu=100m
Error from server (NotFound): deployments.apps "nginx-deploy" not found
ourbeastlapi@cloudshell:~ (ashwin-451403)$ kubectl set resources deployment nginx-deploy2   --limits=cpu=200m   --requests=cpu=1
00m
deployment.apps/nginx-deploy2 resource requirements updated
ourbeastlapi@cloudshell:~ (ashwin-451403)$ kubectl autoscale deployment nginx-deploy2 \
  --cpu-percent=50 \
  --min=1 \
  --max=5
horizontalpodautoscaler.autoscaling/nginx-deploy2 autoscaled
ourbeastlapi@cloudshell:~ (ashwin-451403)$ kubectl get hpa
NAME            REFERENCE                  TARGETS              MINPODS   MAXPODS   REPLICAS   AGE
nginx-deploy2   Deployment/nginx-deploy2   cpu: <unknown>/50%   1         5         0          9s
ourbeastlapi@cloudshell:~ (ashwin-451403)$ kubectl get svc
```

Thank You.

Suresh Dike.