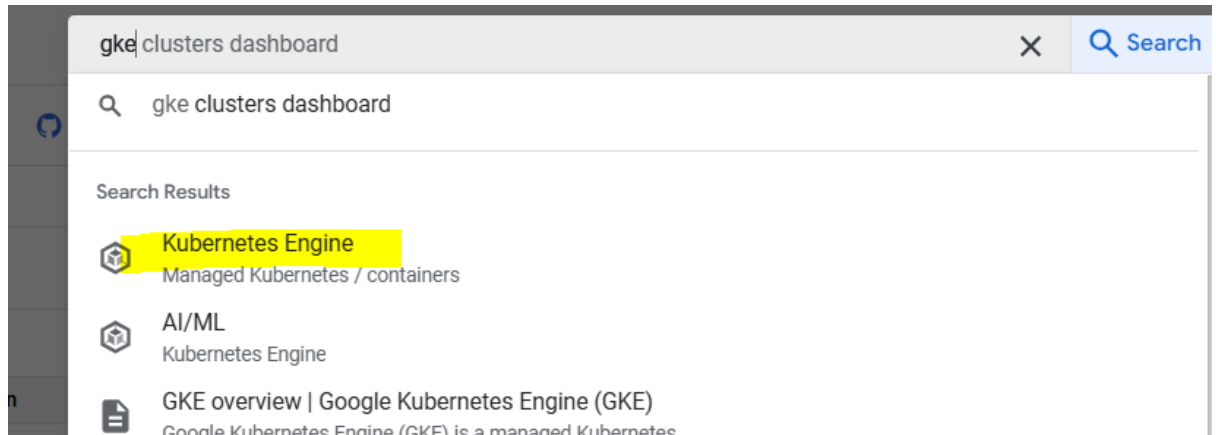


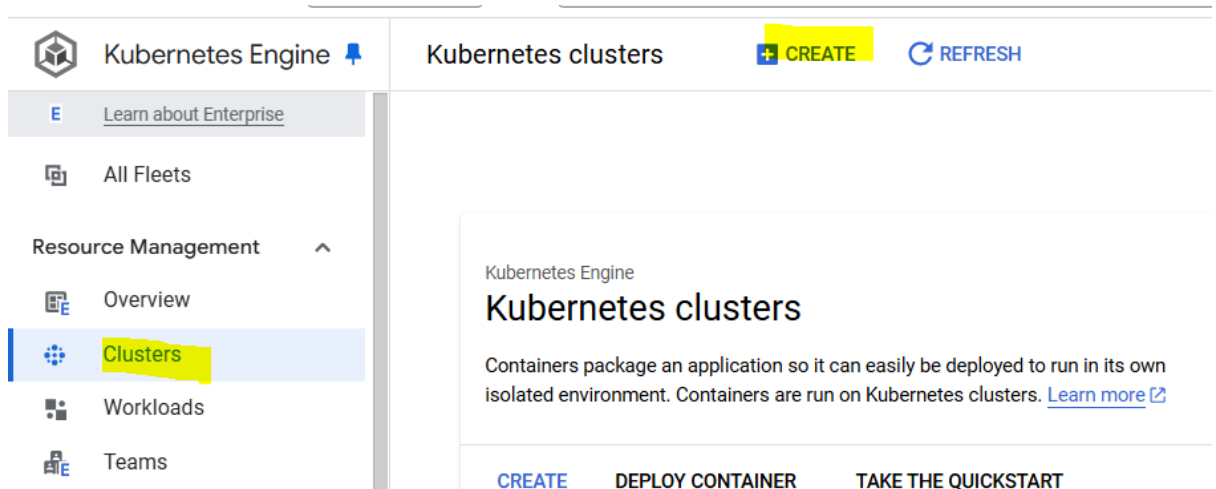
In this lab, we are going to deploy the cart container that we deployed earlier to Cloud Run to GKE in autopilot mode.

- 1) Go to GCP and search for GKE and click Kubernetes Engine



- 2) Enable the API if asked

- 3) Go to clusters and click create



#### 4) Give cluster a name, your region and keep as Standard tier

- Cluster basics**  
Set up basics for your cluster
- Fleet registration**  
Manage multiple clusters together
- Networking**  
Define applications communication in the cluster
- Advanced settings**  
Review additional options
- Review and create**  
Review all settings and create your cluster

cart-cluster

Cluster names must start with a lowercase letter followed by up to 39 lowercase letters, numbers, or hyphens. They can't end with a hyphen. You cannot change the cluster's name once it's created.

Region  
us-central1

The regional location in which your cluster's control plane and nodes are located. You cannot change the cluster's region once it's created.

**Cluster tier**  
Choose a GKE cluster tier based on the complexity of your workloads and your need for advanced features that improve productivity and reduce software deployment times. [View Edition features](#)

Run business critical workloads faster, safer, and easier at enterprise scale  
GKE Enterprise gives you all the power of GKE, plus multi-cluster and multi-team features, and fully managed security, governance, and service networking components. To get started, enable GKE Enterprise in the project.  
[LEARN AND ENABLE](#)

☒ Standard tier ☐ Enterprise tier

#### 5) Go to the next tab, fleet registration, keep all default

##### ← Create an Autopilot cluster

- ☒ **Cluster basics**  
Set up basics for your cluster
- Fleet registration**  
Manage multiple clusters together
- Networking**  
Define applications communication in the cluster
- Advanced settings**  
Review additional options
- Review and create**  
Review all settings and create your cluster

**Fleet registration** New

A fleet lets you logically group and normalize Kubernetes clusters, helping you uplevel management from individual clusters to groups of clusters. To use multi-cluster capabilities and apply consistent policies across your systems, register your cluster to a fleet. [Learn about fleets](#)

☐ Register cluster to the fleet ?

**i** Your cluster will be hosted in project 'still-kit-459403-e2'. To register it to a fleet in another project, skip this step and use the CLI.  
[Learn how to register a GKE cluster](#)

[Previous](#) [Next: Networking](#)

[Reset settings](#)

#### 6) Go to next tab, Networking, check VPC details (keep default)

## ← Create an Autopilot cluster

- ✓ **Cluster basics**  
Set up basics for your cluster
- ✓ **Fleet registration**  
Manage multiple clusters together
- **Networking**  
Define applications communication in the cluster
- **Advanced settings**  
Review additional options
- **Review and create**  
Review all settings and create your cluster

### Networking

#### Control Plane Access

Define from where you can access the control plane.

☐ Access using DNS

**i** DNS-based control plane access is recommended. For a simple, secure, and scalable way to access the control plane, configure IAM-based policies or token-based authentication.

☒ Access using IPv4 addresses

☐ Enable authorized networks [?](#)

**i** To safeguard your operations and your data, add at least one authorized network. This restricts access to your cluster's control plane using an IP-based firewall. [Learn about authorized networks](#)

- 7) Go to the Advanced settings tab, in this list we can see three options, Rapid, Regular, and Stable, so basically what we set here is what is the Kubernetes version that we are going to use and how upgrades are going to be managed.
- 8) This means that when a new version of Kubernetes is released and has passed Google's validation, our cluster will be upgraded to support it.
- 9) Rapid channel means that we are going to be upgraded as fast as possible to the new version, and stable channel is the slowest channel. And only after we are really sure that we want to upgrade our cluster to the new version of Kubernetes, then the upgrade takes place.
- 10) So, we will stick to the regular channel, keep all default and click create

## ← Create an Autopilot cluster

⚠ Ensure that the default GKE node service account has, at a minimum, the Kubernetes Engine Default Node Service Account [role permissions](#) on yo

- ✓ **Cluster basics**  
Set up basics for your cluster
- ✓ **Fleet registration**  
Manage multiple clusters together
- ✓ **Networking**  
Define applications communication in the cluster
- **Advanced settings**  
Review additional options
- **Review and create**  
Review all settings and create your cluster

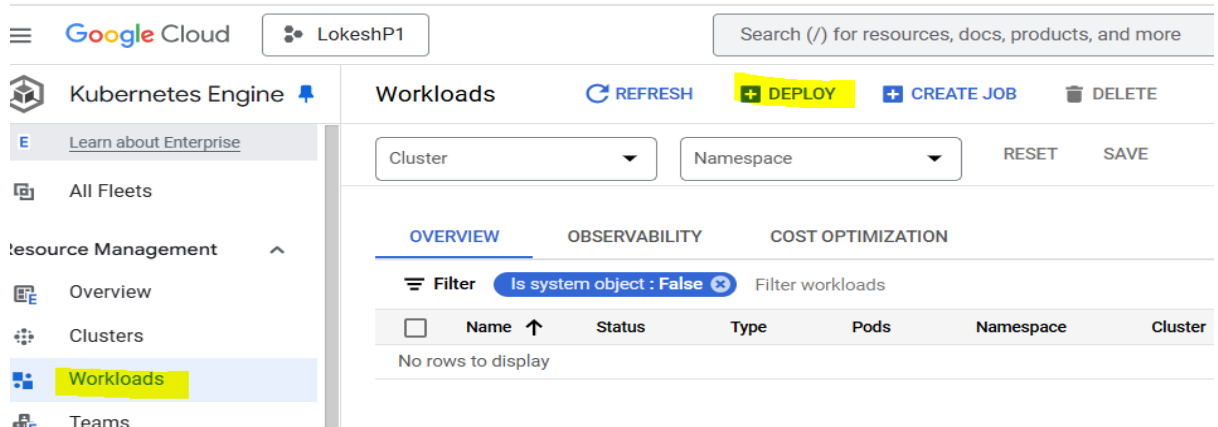
### Advanced settings

**Target release channel**  
Regular (recommended) [?](#)

Versions in the Regular channel have been qualified over a longer period. They offer a balance of feature availability and release stability. We recommend the Regular channel for most users. For known issues and workarounds, review [release notes](#).

- Automation** [?](#)
- Service Mesh** [?](#)
- Backup plan** [?](#)
- Security** [?](#)

- 11) Once the cluster is created, which will take a few minutes, we will deploy our cart image to this cluster. For this, go to Workloads and click Create Deployment



- 12) Give this deployment a name like cart, and make sure the cluster which we just created is selected under Cluster option

## Deployment configuration

A deployment is a configuration which defines how Kubernetes deploys, manages, and scales your container image. Kubernetes will ensure your system matches this configuration. Three replicas will be created by default.

Deployment name \*

cart

Namespace \*

default

## Labels

Use Kubernetes labels to control how workloads are scheduled to your nodes. Labels are applied to all nodes in this node pool.

Key 1 \*

app

Value 1

cart

[+ ADD KUBERNETES LABEL](#)

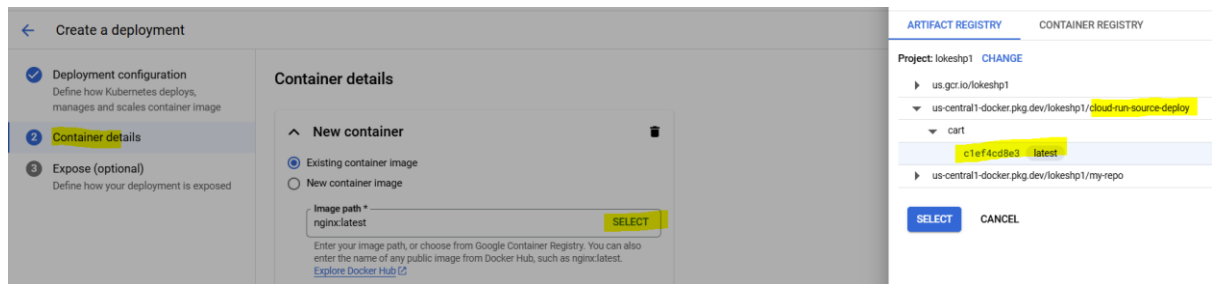
## Cluster

Kubernetes Cluster

cart-cluster (us-central1)

Cluster in which the deployment will be created.

13) Select latest image from repo we created earlier via code

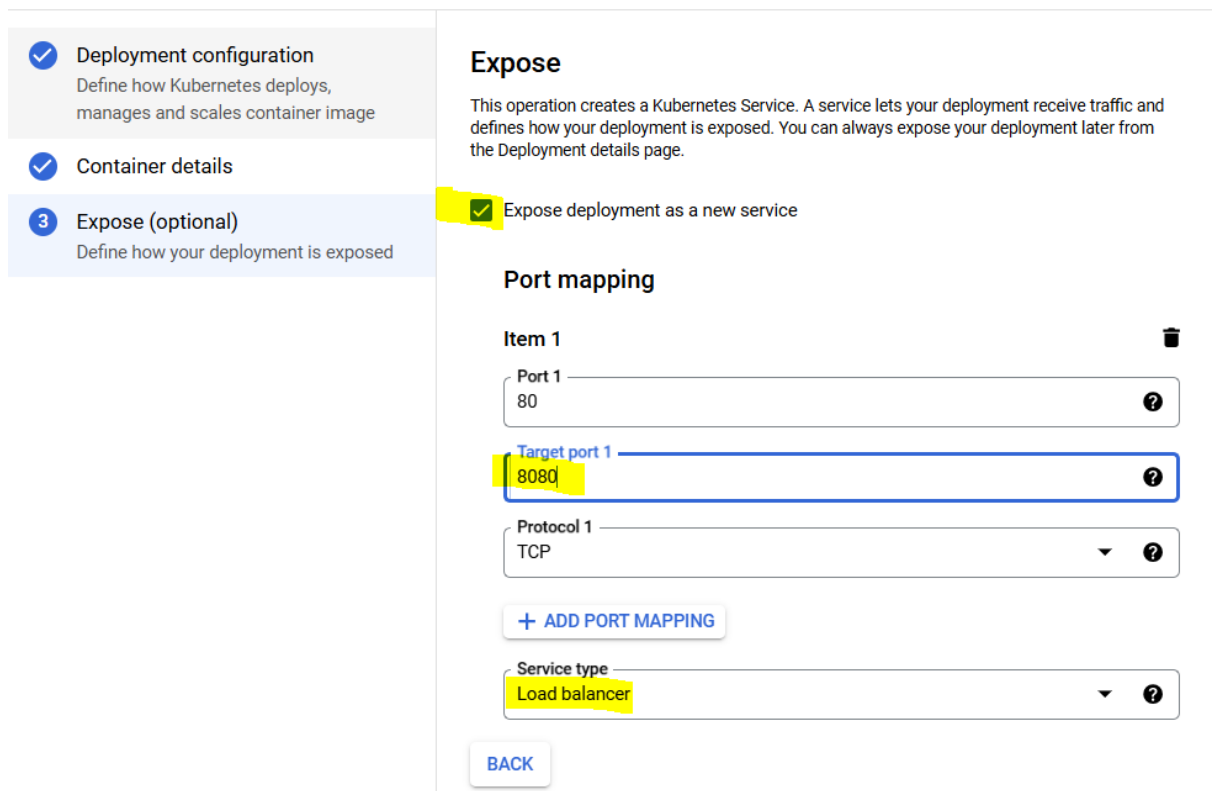


14) In next option, check box to expose the deployment. By this we created a new public endpoint that allows us to access pods in the cluster.

15) Here, service will be exposed to port 80, which is by default and the **target port** is the port at which the pod.

16) In **service type**, we have three options, first is cluster IP, by which only resources in the cloud can access this resource, second is Node port, which creates a service for each node, and third is load balancer, which creates a load balancer with external IP.

17) So here, we will select **load balancer** option (Note, in case you see an error like does not have minimum availability,, ignore that and it will go away in few minutes)



18) Once deployment is completed, scroll down left pane and go to Gateways, services, and Ingress option under networking.

19) Here select your cluster, go to services and here it will show our cart service and type as external load balancer, which we selected while creating

The screenshot shows the Google Cloud Platform console interface. On the left is a navigation menu with categories like Applications, AI/ML, Secrets & ConfigMaps, Storage, Object Browser, Rollout Sequencing, Backup for GKE, Posture Management, Security, Compliance, Policy, Networking, and Features. The 'Networking' category is expanded, and 'Gateways, Services & Ingress' is selected. The main panel is titled 'Gateways, Services & Ingress' and has tabs for 'GATEWAYS', 'ROUTES', 'SERVICES', and 'INGRESS'. The 'SERVICES' tab is active. At the top of the main panel, there are filters for 'Cluster' (set to 'cart-cluster') and 'Namespace'. Below the tabs, there is a description of services and a filter bar. A table lists the services, with one entry highlighted: 'cart-service' with status 'OK' and type 'External load balancer'. The endpoints are listed as '34.42.193.70:80'.

Name	Status	Type	Endpoints	Pods	Name
cart-service	OK	External load balancer	34.42.193.70:80	1/1	default

20) Click on the IP address, and this should open our cart webpage as per selected image

This is a close-up of the 'Services' table from the previous screenshot. It shows the 'cart-service' entry with status 'OK' and type 'External load balancer'. The 'Endpoints' column shows the IP address '34.42.193.70:80' with a link icon. The 'Pods' column shows '1/1', the 'Namespace' is 'default', and the 'Clusters' column shows 'cart-cluster'.

Name	Status	Type	Endpoints	Pods	Namespace	Clusters
<a href="#">cart-service</a>	OK	External load balancer	<a href="#">34.42.193.70:80</a>	1/1	default	<a href="#">cart-cluster</a>

21) Now go back to workloads tab again, open the cart deployment and scroll down, here we can see the pods, revision, exposed service details and all

The screenshot shows the Google Cloud Platform interface for the 'Workloads' tab. On the left is a navigation menu with options like Clusters, Workloads (selected), Teams, Applications, AI/ML, Secrets & ConfigMaps, Storage, Object Browser, Rollout Sequencing, Backup for GKE, Marketplace, and Release Notes. The main content area displays details for the 'cart' deployment. At the top, it shows 'Autoscaler' and 'Vertical Pod Autoscaler' status. Below, the 'Active revisions' table lists one revision with status 'OK'. The 'Managed pods' table shows one pod in 'Running' status. At the bottom, the 'Exposing services' table shows the 'cart-service' as a 'Load balancer' with an endpoint.

Revision	Name	Status	Summary
1	cart-6d8dcd4468	OK	cart-sha256-1: us-central1-docker.pkg.dev/lokeshp1/cloud-run-source-deploy/cart@sha256:c1ef4cd8e3c191210e3d1d4e98c0229c5c99e1

Revision	Name	Status	Restarts	Created on
1	cart-6d8dcd4468-kxlf	Running	0	Jan 26, 2025, 9:01:17 PM

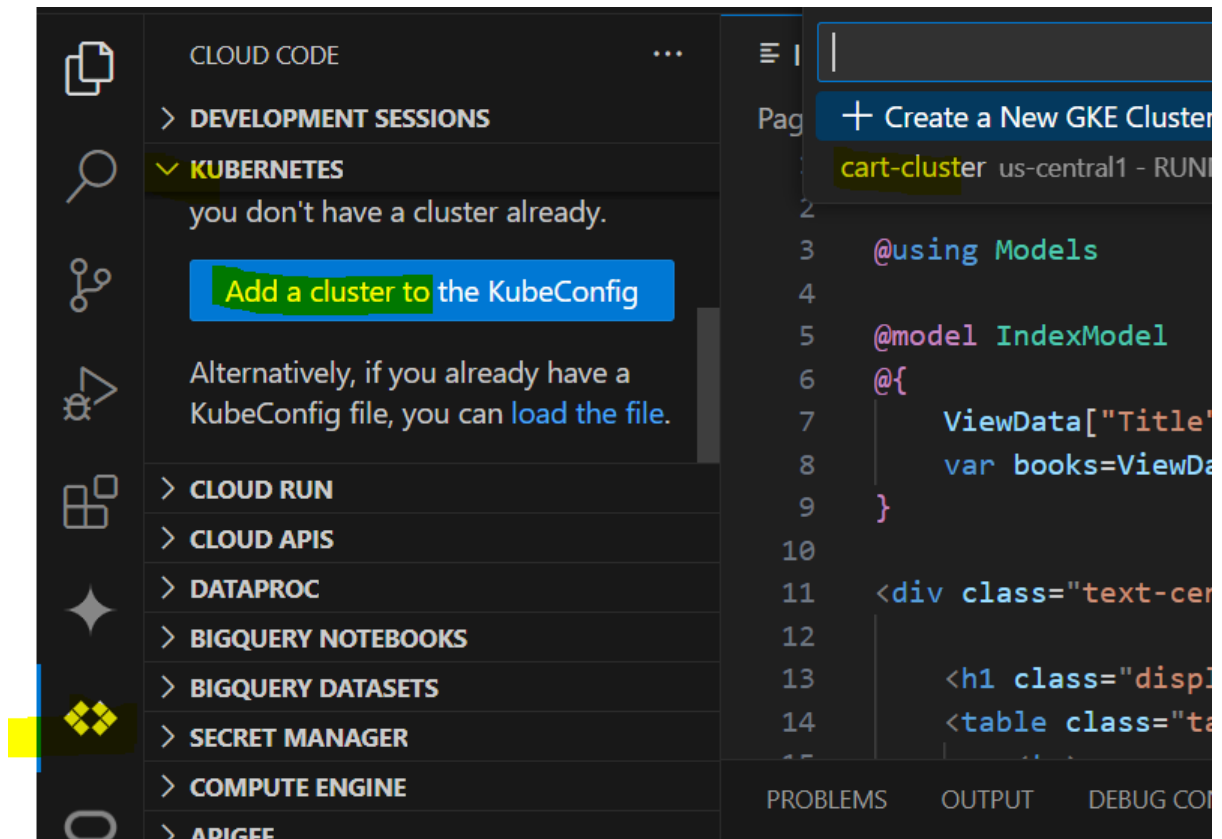
Name	Type	Endpoints
cart-service	Load balancer	34.42.193.70:80

22) Now go to YAML option, and here we can see all the configurations related to this GKE deployment and if you scroll down, there we can see image details as well which is one of most important part

The screenshot shows the 'YAML' tab selected in the Google Cloud Platform interface. The left navigation menu is the same as in the previous screenshot. The main content area displays the YAML configuration for the 'cart' deployment. The configuration includes a 'strategy' section with 'rollingUpdate' settings, a 'template' section with 'metadata' (creationTimestamp, labels), and a 'spec' section with 'containers' (image, imagePullPolicy, name, resources). The 'image' field is highlighted in yellow.

```
110 strategy:
111   rollingUpdate:
112     maxSurge: 25%
113     maxUnavailable: 25%
114   type: RollingUpdate
115 template:
116   metadata:
117     creationTimestamp: null
118   labels:
119     app: cart
120     app.kubernetes.io/managed-by: cloud-console
121 spec:
122   containers:
123     - image: us-central1-docker.pkg.dev/lokeshp1/cloud-run-source-deploy/cart@sha256:c1ef4cd8e3c191210e3d1d4e98c0229c5c99e1f58948
124       imagePullPolicy: IfNotPresent
125       name: cart-sha256-1
126       resources:
127         limits:
128           ephemeral-storage: 1Gi
129       requests:
```

23) Now go to vscode, gcp cloud option, expand Kubernetes and Add a cluster option. Select google cloud and then select cart-cluster



24) This will add our Kubernetes cluster to the vscloud and from here we can check multiple options, like loadbalancer ip under services options and many more configurations  
(If see error related to extension installation, close and launch vscode as



administrator)

