### Setting up Backstage on Google Kubernetes Engine (GKE)

**Prerequisites**

1. **Google Cloud Account**: Ensure you have a Google Cloud account and a project created.
2. **Billing Enabled**: Make sure billing is enabled for your project.
3. **Google Cloud SDK**: Install the [Google Cloud SDK](https://cloud.google.com/sdk/docs/install) if using the CLI.
4. **kubectl**: Install kubectl (included with the Google Cloud SDK).

**Step 1: Create a GKE Cluster**

**Option 1: Using Google Cloud Console**

1. Go to the [Google Cloud Console](https://console.cloud.google.com/).
2. Navigate to **Kubernetes Engine > Clusters**.
3. Click **Create Cluster**.
4. Choose the **Standard** cluster type.
5. Configure the cluster:
   * **Name**: backstage-cluster
   * **Location Type**: Zonal (choose a zone, e.g., us-central1-a).
   * **Node Pool**: Use the default settings (1 node, e2-medium machine type).
6. Click **Create**.

**Option 2: Using Google Cloud CLI**

Run the following command to create a GKE cluster:

gcloud container clusters create backstage-cluster \

--zone us-central1-a \

--num-nodes 1 \

--machine-type e2-medium

**Step 2: Connect to the GKE Cluster**

**Option 1: Using Google Cloud Console**

1. After the cluster is created, go to **Kubernetes Engine > Clusters**.
2. Click the **Connect** button next to your cluster.
3. Copy the gcloud command provided and run it in your terminal.

**Option 2: Using Google Cloud CLI**

Run the following command to configure kubectl to connect to your cluster:

gcloud container clusters get-credentials backstage-cluster \

--zone us-central1-a

**Step 3: Deploy Backstage**

**Option 1: Using Helm (Simplest Approach)**

1. Install Helm:

curl https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3 | bash

1. Add the Backstage Helm chart repository:

helm repo add backstage https://backstage.github.io/charts

helm repo update

1. Install Backstage:

helm install backstage backstage/backstage

1. Verify the deployment:

kubectl get pods

**Option 2: Using Kubernetes Manifests**

1. Create a backstage.yaml file with the following content:

apiVersion: apps/v1

kind: Deployment

metadata:

name: backstage

spec:

replicas: 1

selector:

matchLabels:

app: backstage

template:

metadata:

labels:

app: backstage

spec:

containers:

- name: backstage

image: backstage/backstage:latest

ports:

- containerPort: 3000

---

apiVersion: v1

kind: Service

metadata:

name: backstage-service

spec:

selector:

app: backstage

ports:

- protocol: TCP

port: 80

targetPort: 3000

type: LoadBalancer

1. Apply the manifest:

kubectl apply -f backstage.yaml

**Step 4: Access Backstage**

1. Get the external IP of the Backstage service:

kubectl get svc backstage-service

1. Open the external IP in your browser (e.g., http://<EXTERNAL\_IP>).

gcloud container clusters delete backstage-cluster \

--zone us-central1-a

# **Deploying Roadie Backstage on Google Kubernetes Engine (GKE)**

This is a step-by-step guide to deploying the **Roadie Backstage Docker image** on **Google Kubernetes Engine (GKE)** with **default settings** (no database, using in-memory or SQLite storage).

## **Step 1: Set Up a GKE Cluster**

### ****1.1 Navigate to Kubernetes Engine****

1. Open the [**Google Cloud Console**](https://console.cloud.google.com).
2. Go to **Kubernetes Engine > Clusters**.
3. Click **Create** to set up a new cluster.

### ****1.2 Configure the Cluster****

1. Select **Standard** mode.
2. Name the cluster:

backstage-cluster

1. Choose a **region** and **zone**:
   * **Region**: us-central1
   * **Zone**: us-central1-a
2. Configure the default node pool:
   * **Machine type**: e2-standard-2
   * **Node count**: 1-3 nodes
3. Click **Create** and wait for provisioning.

### ****1.3 Connect to the Cluster****

1. Click on **backstage-cluster** in the console.
2. Click **Connect** and copy the gcloud command:

gcloud container clusters get-credentials backstage-cluster --region us-central1 --project <your-project-id>

1. Open **Cloud Shell** and run the copied command to configure kubectl.

## **Step 2: Deploy the Roadie Backstage Docker Image**

Roadie provides a **pre-built Backstage image** (docker.io/roadiehq/backstage). Since we're not configuring an external database, Backstage will use **in-memory storage or SQLite**.

### ****2.1 Create the Deployment YAML****

In **Cloud Shell**, create the deployment file:

nano backstage-deployment.yaml

apiVersion: apps/v1

kind: Deployment

metadata:

name: backstage

namespace: default

spec:

replicas: 1

selector:

matchLabels:

app: backstage

template:

metadata:

labels:

app: backstage

spec:

containers:

- name: backstage

image: docker.io/roadiehq/backstage:latest

ports:

- containerPort: 7000

Save and exit (Ctrl+O, Enter, Ctrl+X).

### ****2.2 Deploy to GKE****

Apply the deployment using:

kubectl apply -f backstage-deployment.yaml

## **Step 3: Expose the Backstage Service**

By default, the Backstage UI runs on **port 7000**. To access it externally, create a **LoadBalancer** service.

### ****3.1 Create the Service YAML****

Create the service file:

nano backstage-service.yaml

apiVersion: v1

kind: Service

metadata:

name: backstage

namespace: default

spec:

selector:

app: backstage

ports:

- protocol: TCP

port: 7000

targetPort: 7000

type: LoadBalancer

Save and exit (Ctrl+O, Enter, Ctrl+X).

### ****3.2 Apply the Service****

Deploy the service: kubectl apply -f backstage-service.yaml

### ****3.3 Get the External IP****

Run the following command:

kubectl get services

Look for the **EXTERNAL-IP** under the backstage service. It may take a few minutes for the IP to be assigned.

## **Step 4: Access Backstage**

1. Once the external IP is available, open your browser and go to:

http://<external-ip>:7000

1. You should see the **Backstage UI** running. 🚀

## **Troubleshooting**

### ****4.1 Check Pod Logs****

If the UI doesn’t load, check the logs:

kubectl logs -l app=backstage

### ****4.2 Verify Pod Status****

Check if the pod is running:

kubectl get pods

If the pod is **not running**, check for errors:

kubectl describe pod <pod-name>

## **Step 5: Cleanup (If Needed)**

To delete the deployment and service:

kubectl delete -f backstage-deployment.yaml

kubectl delete -f backstage-service.yaml

To delete the **GKE cluster**:

gcloud container clusters delete backstage-cluster --region us-central1

## **Notes**

✅ This setup **does not persist data** across restarts.  
✅ For production, configure **PostgreSQL** as Backstage’s database.  
✅ The Backstage UI is now accessible via **port 7000**.

### ****That’s it! You’ve successfully deployed Roadie Backstage on GKE!**** 🎉🚀

Let me know if you need any further assistance!