

1. Introducción

Este documento describe paso a paso cómo construir, desplegar, validar y probar un servicio Node.js en Google Kubernetes Engine (GKE) utilizando Docker, Terraform y Kubernetes. La guía resume todos los comandos utilizados en el proyecto y explica cómo ejecutar todo el entorno.

2. Pre-requisitos

- Tener instalado: gcloud, kubectl, Docker y Terraform.
- Haber autenticado en Google Cloud con: `gcloud auth login`
- Haber configurado el proyecto: `gcloud config set project`
- Archivo de credenciales JSON para Terraform.
- Variables definidas en `terraform.tfvars`.

3. Construcción y publicación de la imagen Docker

1. `docker build -t jensanchez/orderservice:v7 .`
 2. `docker push jensanchez/orderservice:v7`
 3. Actualizar `var.image` en Terraform con el nuevo tag.
- La imagen debe contener el archivo `orderservice.js` y el Dockerfile debe ejecutar CMD `["node", "orderservice.js"]`.

4. Infraestructura con Terraform

Terraform crea:

- IP estática
- Cluster GKE
- Namespace
- Deployment
- LoadBalancer

Comandos:

```
terraform init
terraform plan
terraform apply -auto-approve
Si aparece un lock: terraform force-unlock .
```

5. Despliegue en Kubernetes

Validar el cluster:

```
gcloud container clusters get-credentials --zone --project
```

Validar pods y servicios:

```
kubectl get pods -n orders
kubectl get svc -n orders
```

Logs:

```
kubectl logs -n orders
```

Exec:

```
kubectrl exec -it -n orders -- sh
```

6. Debugging

- ImagePullBackOff → verificar nombre/tag de imagen.
- CrashLoopBackOff → revisar logs y variables.
- Probes fallando → revisar /health.
- Para inspeccionar un contenedor:

```
kubectrl exec -it -n orders -- sh
```

7. Pruebas de carga

Curl:

```
curl http:///orders/123/status
```

JMeter:

- Thread Group con ramp-up progresivo.
- 300–500 usuarios virtuales.
- HTTPSampler apuntando al endpoint GET /orders/{id}/status.

8. Flujo completo de ejecución

1. Construir imagen Docker.
2. Subir imagen al registry.
3. Ejecutar terraform apply.
4. Obtener IP del LoadBalancer.
5. Validar salud y logs.
6. Ejecutar pruebas de carga.
7. Limpiar entorno si es necesario.

9. Comandos completos usados

Incluye los comandos de build, push, terraform, kubectrl, curl y debugging descritos en secciones previas.

Resumen final

Para ejecutar todo el proyecto: construir la imagen Docker, subirla, aplicar Terraform para desplegar el cluster y los servicios, validar el despliegue, obtener la IP externa y hacer pruebas funcionales y de carga. Todo el proceso es reproducible y automatizado mediante Docker y Terraform.