**Guía de Despliegue de una Aplicación Spring Boot en Kubernetes con Docker y Killercoda**

Esta guía describe los pasos para exponer una aplicación Spring Boot en Kubernetes utilizando NodePort, configurando un Ingress y desplegando el servicio en Killercoda.

**1. Clonar el repositorio**

git clone <https://github.com/rmcabrera/productoAPI>

cd productoAPI

**2. Construcción de la Imagen Docker**

gradlew build # Generar el archivo JAR

docker build -t ms-producto-api:v1 .

docker images # Verificar la imagen generada

**3. Publicación en Docker Registry**

docker login

docker tag ms-producto-api:v1 <usuario-docker>/ms-producto-api:v1

docker push <usuario-docker>/ms-producto-api:v1

**4. Desplegar en Killercoda**

Acceder a [Killercoda](https://killercoda.com/playgrounds/scenario/kubernetes) y clonar el repositorio:

git clone <https://github.com/rmcabrera/productoAPI>

cd productoAPI/kubernetes/

ls # Verificar los archivos YAML

Deben existir los siguientes archivos:

* api-namespace.yaml
* h2-deployment.yaml
* h2-service.yaml
* api-deployment.yaml
* api-service.yaml
* api-ingress.yaml

**5. Crear Namespace, Deployment y Service en Kubernetes**

kubectl apply -f api-namespace.yaml

kubectl apply -f h2-deployment.yaml

kubectl apply -f h2-service.yaml

kubectl apply -f api-deployment.yaml

kubectl apply -f api-service.yaml

kubectl get all -n ms-producto-namespace # Verificar recursos creados

**6. Validar Acceso Interno**

kubectl port-forward svc/ms-producto-api-service -n ms-producto-namespace 8484:8484

curl -X GET http://<IP\_SERVICIO>:8484/api/productos # Debería devolver un JSON

**7. Instalar NGINX Ingress Controller**

kubectl apply -f <https://raw.githubusercontent.com/kubernetes/ingress-nginx/main/deploy/static/provider/cloud/deploy.yaml>

kubectl get pods -n ingress-nginx # Verificar que esté en ejecución

**8. Configurar Ingress**

kubectl apply -f api-ingress.yaml

kubectl get ingress -n ms-producto-namespace # Verificar que esté activo

**9. Pruebas con CURL**

**Obtener el puerto del Ingress Controller:**

kubectl get svc -n ingress-nginx #Obtener puerto

kubectl get nodes -o wide #Obtener el IP del Nodo

**GET**

curl -H "Host: ms-producto.server" http://<IP\_NODO>:<PUERTO\_INGRESS>/api/productos

**POST**

curl -X POST -H "Host: ms-producto.server" -H "Content-Type: application/json" \

-d '{"nombre": "Laptop", "descripcion": "Lenovo Yoga", "precio": 1000}' \

http://<IP\_NODO>:<PUERTO\_INGRESS>/api/productos

**PUT**

curl -X PUT -H "Host: ms-producto.server" -H "Content-Type: application/json" \

-d '{"nombre": "Laptop Pro", "descripcion": "Lenovo Yoga 2024", "precio": 1200}' \

http://<IP\_NODO>:<PUERTO\_INGRESS>/api/productos/1

**DELETE**

curl -X DELETE -H "Host: ms-producto.server" http://<IP\_NODO>:<PUERTO\_INGRESS>/api/productos/1

**10. Configurar ngrok para acceso externo**

wget https://bin.equinox.io/c/bNyj1mQVY4c/ngrok-stable-linux-amd64.zip

unzip ngrok-stable-linux-amd64.zip

sudo mv ngrok /usr/local/bin

ngrok config add-authtoken <TU\_AUTHTOKEN> #En caso de tener token

ngrok config add-authtoken 2u5Y731jQWjLgVE7GpTjCOGYSX1\_7GfHZzeHZYcrhFCR3i7ay #usamos este token

**Exponer el servicio con ngrok:**

kubectl get svc -n ingress-nginx # Identificar el puerto del Ingress

ngrok http <puerto> --host-header=ms-producto.server

**Acceder a la API desde Postman o cURL:**

curl -X GET https://<url-ngrok>/api/productos

curl --location 'https:// <url-ngrok>/api/productos' \

--header 'Content-Type: application/json' \

--data '{

"nombre": "Laptop",

"descripcion": "Laptop ACER",

"precio": 3500

}'

curl --location --request PUT 'https://<url-ngrok>/api/productos/2' \

--header 'Content-Type: application/json' \

--data '{

"nombre": "Laptop Sony",

"descripcion": "Laptop con GPU potente",

"precio": 4000

}'

curl --location --request DELETE 'https://<url-ngrok>/api/productos/1'