# Documentación del Desafío 12 – Helm Chart

## Objetivo

Este desafío tiene como objetivo aplicar los conocimientos adquiridos sobre Helm para empaquetar, desplegar y gestionar de forma eficiente una aplicación y su base de datos MongoDB en Kubernetes.

## Estructura del Helm Chart

El Helm Chart desarrollado incluye los siguientes archivos clave:

* - deployment.yaml: Define el Deployment de la aplicación.
* - service.yaml: Define el Service para exponer la aplicación.
* - values.yaml: Contiene los valores personalizables del Chart.
* - requirements.yaml: Dependencia con MongoDB (si aplica).

## Uso del Chart

Para desplegar el Chart, se utiliza el siguiente comando:  
`helm install educacionit-app ./educacionit-chart`

## Parámetros configurables (values.yaml)

|  |  |
| --- | --- |
| Parámetro | Descripción |
| image.repository | Ruta del repositorio de la imagen Docker. |
| image.tag | Etiqueta de la imagen. |
| service.port | Puerto de exposición del servicio. |
| mongodb.enabled | Indica si se incluye la base MongoDB. |

## Ventajas de usar Helm

Helm permite reutilizar manifiestos Kubernetes con plantillas y valores dinámicos. Reduce la duplicación de código, facilita el versionado de deployments y permite gestionar entornos de forma más eficiente.

## Mejoras Futuras

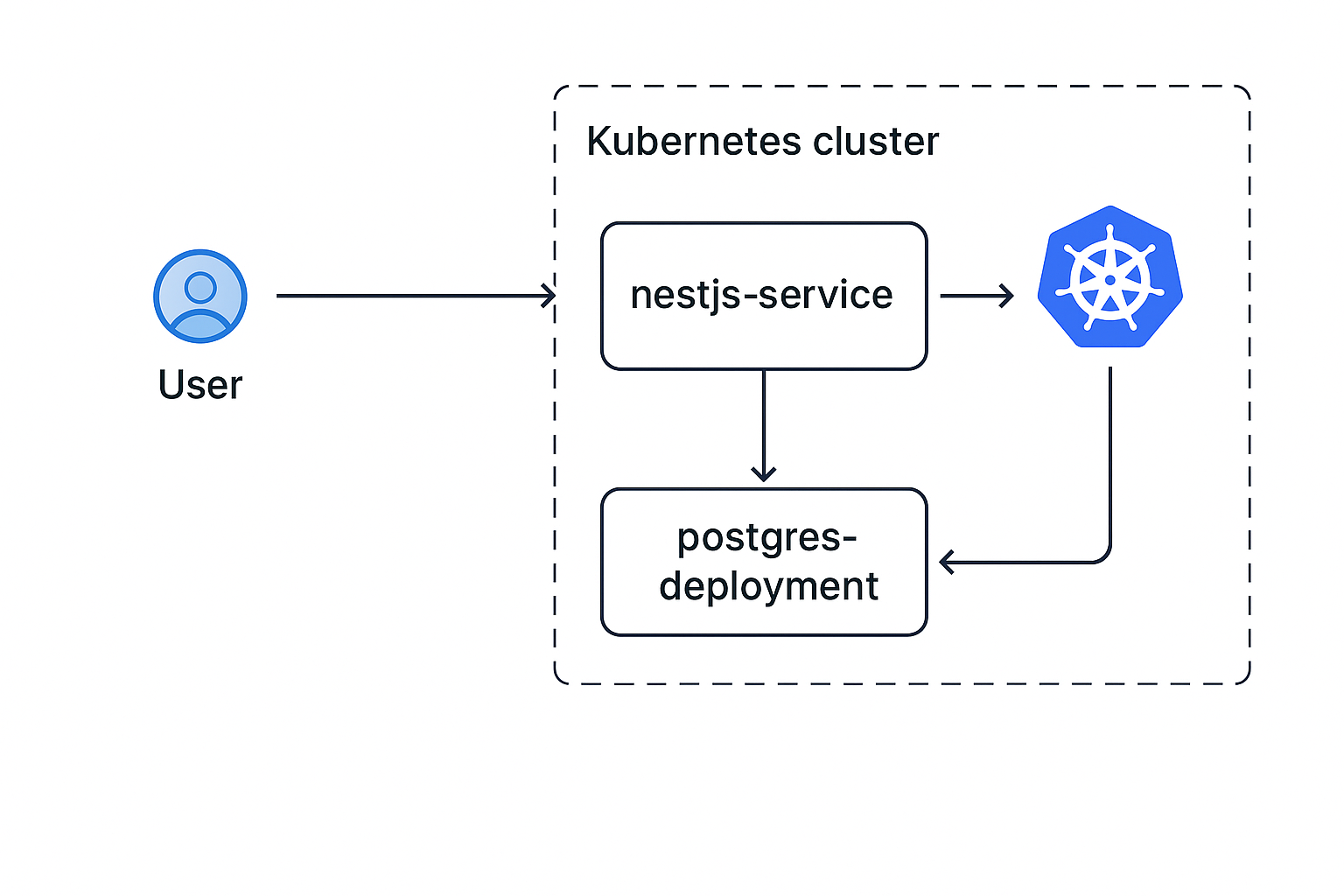
- Integrar el Helm Chart en un pipeline de CI/CD con GitHub Actions.  
- Implementar tests de salud y readiness probes.  
- Incorporar valores secretos usando Sealed Secrets u otros mecanismos.

## Evidencia de despliegue

La evidencia del despliegue exitoso se encuentra en el archivo `helm\_output.txt`, que contiene la salida del comando `helm install`.

## Diagrama

Se incluye el siguiente diagrama que resume la arquitectura del despliegue:



## Conclusión

Se logró encapsular correctamente la lógica de despliegue de la aplicación y su base de datos MongoDB usando Helm, facilitando su gestión futura y cumplimiento con las prácticas recomendadas de Kubernetes.