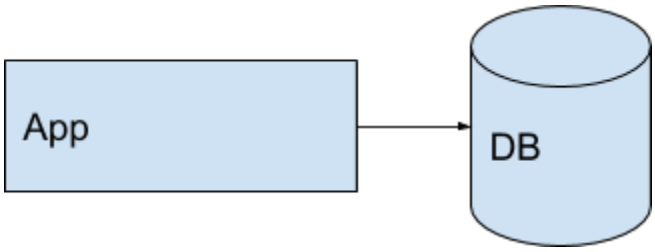
Práctica contenedores

Descripción general

En esta práctica debemos implementar una aplicación consistente en un micro servicio que sea capaz de leer y escribir de una base de datos.

El microservicio y la base de datos son de tu elección.
Puedes basarte en el flask-counter que hemos estudiado durante el curso.
https://github.com/pmoncadaisla/flask-counter
Puedes usar cualquier lenguaje de programación y cualquier framework.



Hitos

1. Crear repositorio GIT con la aplicación.
2. Crear un fichero README.md que explique:
    a. Descripción de la aplicación.
    b. Funcionamiento de la aplicación.
    c. Requisitos para hacerla funcionar.
    d. Instrucciones para ejecutarla en local
    e. Instrucciones para desplegarla en Kubernetes
3. Dockerfile
    a. Que compile / dependencias / pruebe la aplicación
    b. Que la empaquete con los requisitos mínimos (usar Multistage)
4. Docker compose
    a. Que permita ejecutar la aplicación completa en local
5. Logs
    a. Formato JSON con estructura LogEntry de Google a ser posible.
    b. Deben imprimirse por salida estándar de logs y salida de errores.
6. Configuración

    a. La aplicación debe de poder ser configurable, por ejemplo, host de la base de datos, puerto, usuario, contraseña, etc. Mediante fichero de configuración o variables de entorno.
7. Generación de manifests de Kubernetes en directorio **k8s/**, mínimo:
    a. deployment
    b. service
    c. persistentvolumeclaim (para la BBDD)
    d. ingress
    e. configmap
    f. (secret en la documentación, cómo habría que generarlo)
8. Generación de chart de Helm en directorio **charts/**, mínimo:
    a. deployment
    b. service
    c. persistentvolumeclaim (para la BBDD)
    d. ingress
    e. configmap
    f. (secret en la documentación, cómo habría que generarlo)
9. Asegurar que los PODs de la base de datos y la aplicación permanecen lo más juntos posibles al desplegarse en Kubernetes. [OPCIONAL]
10. Asegurar que los PODs de las réplicas de la aplicación permanecen lo más separados posibles. [OPCIONAL]
11. Usar almacenamiento persistente provisionado de forma dinámica para la base de datos.
12. Configuración externa mediante secretos y configMaps.
13. Instalar Ingress Controller: Nginx
14. Exponer la aplicación públicamente mediante Ingress. Puede usarse servicio de nip.io si no se dispone de DNS.
15. Exposición de métricas de Prometheus (alguna métrica custom, o algún exporter del framework utilizado). [OPCIONAL]
16. Instalación de Prometheus y grafana, configurando un Dashboard. [OPCIONAL]
17. Instalación de Prometheus e Ingress controller mediante un Operador. [OPCIONAL]
18. Instalar Istio [OPCIONAL]
19. Configurar un despliegue canary de tipo 80/20 con 2 versiones de aplicación [OPCIONAL]
20. Configurar política de reintentos con Istio [OPCIONAL]

Formato de entrega

- Repositorio GitHub o Gitlab de Keepcoding: público o dándo permisos a “pmoncadaisla”
- Proyecto en **GCP**: permisos de editor a “pmoncadaisla@gmail.com”