Anteproyecto:
Gestor de Tareas en Android con Backend en Kubernetes

Autor: Julián Méndez

1. Definición del Proyecto

El objetivo de este proyecto es desarrollar una aplicación Android que permita a los usuarios gestionar sus tareas personales mediante una interfaz simple. La aplicación permitirá crear, editar, eliminar y marcar las tareas como completadas. Para la gestión de los datos, se utilizará un backend que proporcionará una API REST para interactuar con la aplicación. Este backend estará contenedorizado con Docker y desplegado en un cluster de Kubernetes para garantizar la escalabilidad y eficiencia del servicio.

Este proyecto busca integrar el desarrollo de aplicaciones móviles con prácticas modernas de DevOps, utilizando herramientas como Docker y Kubernetes para gestionar el backend de manera eficiente.

2. Módulos del Ciclo Desarrollados

- **Bases de Datos** → Diseño y gestión de MySQL.
- **Programación** → Desarrollo del backend en Java.
- **Desarrollo de Interfaces y P. Multimedia** → Implementación de la aplicación Android.
- **Acceso a Datos** → Conexión entre la app, backend y MySQL.
- **Sistemas de Gestión Empresarial** → Uso de Kubernetes y Docker para despliegue.

3. Tecnologías y Herramientas

- **Lenguajes de programación**:
  - **Android**: Java para desarrollar la aplicación móvil.
  - **Backend**: Java (Spring Boot) para crear la API REST.
- **Frameworks**:
  - **Android Studio** para el desarrollo de la app Android.
  - **Spring Boot** para el desarrollo del backend.
- **Base de datos**: **MySQL** para la gestión de tareas en el backend.
- **Contenerización y despliegue**:
  - **Docker** para contenerizar el backend.
  - **Kubernetes** para el despliegue del backend en un entorno de producción.
- **Herramientas de DevOps**:
  - **Docker Compose** (opcional) para gestionar contenedores locales.
  - **kubectl** y **Helm** para la gestión del clúster Kubernetes.

4. Apartados a Implementar (Requisitos Funcionales)

1. **Gestión de Tareas en la App Android**:
    - Crear, listar, editar y eliminar tareas.
    - Marcar tareas como completadas o pendientes.
    - Almacenar las tareas en el backend mediante peticiones REST API.
2. **Backend Contenedorizado**:
    - Desarrollar una API RESTful para gestionar las tareas.
    - Implementar operaciones CRUD para interactuar con la base de datos.
    - Asegurar que el backend sea capaz de recibir peticiones de la app Android.
3. **Despliegue en Kubernetes**:
    - Dockerizar el backend.
    - Desplegar el backend en un clúster de Kubernetes, asegurando alta disponibilidad y escalabilidad.
    - Exponer la API para ser consumida por la app Android.