### **Parte 1: Desarrollo de la Aplicación Web con Spring y PostgreSQL**

#### **Tarea 1: Crear una aplicación web para mostrar información sobre Inmuebles**

1. **Objetivo:** Crear una aplicación web utilizando Spring para mostrar información sobre inmuebles en la CDMX.
2. **Requisitos funcionales:**
   * **Página principal:** Mostrar una lista de inmuebles con enlaces a sus respectivas páginas detalladas.
   * **Página detallada:** Mostrar información adicional sobre cada inmueble, incluyendo nombre, dirección, teléfono, capacidad de aforo (opcional).
3. **Requisitos técnicos:**
   * Utilizar Spring para crear la aplicación.
   * Implementar la persistencia de datos utilizando JPA/Hibernate con una base de datos PostgreSQL.
   * Crear entidades y repositorios para manejar los datos de los inmuebles.
   * Utilizar Thymeleaf para las vistas.
   * Configurar el proyecto para conectarse a una base de datos PostgreSQL.

### **Parte 2: Despliegue de la Aplicación en Kubernetes**

#### **Tarea 2: Desplegar la aplicación en un clúster de Kubernetes**

1. **Objetivo:** Configurar y desplegar la aplicación desarrollada en un clúster de Kubernetes.
2. **Requisitos técnicos:**
   * Crear un Dockerfile para construir la imagen de la aplicación.
   * Crear archivos de configuración de Kubernetes (deployment.yaml, service.yaml).
   * Configurar un ConfigMap para almacenar la URL de la base de datos.
   * Configurar un Secret para almacenar las credenciales de la base de datos.
   * Asegurar que la aplicación sea accesible desde fuera del clúster utilizando un LoadBalancer o Ingress.

### **Parte 3: Desarrollo de API RESTful con Spring Boot**

#### **Tarea 3: Crear una API RESTful para gestionar inmuebles**

1. **Objetivo:** Crear una API RESTful utilizando Spring Boot que maneje la información de los inmuebles.
2. **Requisitos funcionales:**
   * **GET /inmuebles:** Obtener la lista de todos los inmuebles.
   * **GET /inmuebles/{id}:** Obtener los detalles de un inmueble específico por su ID.
   * **POST /inmuebles:** Crear un nuevo inmueble.
   * **PUT /inmuebles/{id} (Opcional):** Actualizar la información de un inmueble existente.
   * **DELETE /inmuebles/{id} (Opcional):** Eliminar un inmueble por su ID.
3. **Requisitos técnicos:**
   * Utilizar Spring Boot para crear la API.
   * Implementar la persistencia de datos utilizando JPA/Hibernate con PostgreSQL.
   * Implementar una operación de API utilizando un procedimiento almacenado en PostgreSQL.
   * Probar la API utilizando una herramienta como Postman.

Realizado completamente

Realizado parcialmente, falta terminarlo