# Logo Description automatically generated

#### **📌 Consideraciones Generales**

* Esta prueba está diseñada para evaluar tus conocimientos en **procesamiento, transformación y almacenamiento de datos** en un entorno real.
* Evalúa tu capacidad para manejar grandes volúmenes de datos, integraciones con APIs externas y optimización de consultas.
* Si no puedes completar la prueba en el tiempo asignado, **envía tu progreso**. También evaluaremos tu proceso y metodología.

### **📂 Descripción**

Tienes acceso a un archivo poscodes\_geo.csv que contiene **coordenadas cercanas a algunos códigos postales**.  
El objetivo de esta prueba es **procesar y enriquecer estos datos** usando la API pública [Postcodes.io](https://postcodes.io/), almacenarlos en una base de datos y generar reportes optimizados.

#### **🛠️ Tareas**

1. **Ingesta de Datos**
   * Leer el archivo poscodes\_geo.csv, asegurando una carga eficiente (manejo de errores, validación de datos, detección de duplicados).
   * Transformar los datos para que sean consistentes en formato y estructura.
   * Almacenar los datos en una base de datos relacional o no relacional de tu elección.
2. **Enriquecimiento con API Externa**
   * Consumir la API de postcodes.io para obtener la información detallada del código postal más cercano a cada coordenada.
   * Manejar **errores de API** (timeouts, respuestas vacías, fallos en conexión).
   * Asegurar que todas las coordenadas tengan un código postal. Si no es posible obtenerlo, almacenar el error en un log estructurado.
3. **Optimización y Modelado de Datos**
   * Diseñar un esquema eficiente en la base de datos para almacenar los datos enriquecidos.
   * Crear **índices** y **optimizaciones** para mejorar el rendimiento en consultas futuras.
4. **Generación de Reportes**
   * Crear una consulta optimizada para **obtener los códigos postales más comunes en el dataset**.
   * Calcular estadísticas de calidad de datos (ejemplo: porcentaje de coordenadas sin código postal).
   * Generar un archivo CSV con los datos enriquecidos y estadísticas generales.
5. **Documentación y Entrega**
   * **README.md** explicando la solución, la arquitectura y los pasos para ejecutar el proyecto.
   * **Diagrama** de arquitectura y flujo de datos.
   * Explicar cualquier decisión clave tomada en el diseño del proceso.

### **📌 Requerimientos Técnicos**

✅ Puedes usar **Python o SQL** (puedes integrar Spark o Pandas si es necesario).  
✅ La solución debe funcionar con **Docker** (Docker Compose recomendado).  
✅ Controla el número de peticiones a la API para evitar bloqueos.  
✅ Debes manejar grandes volúmenes de datos de manera eficiente.  
✅ Si hay errores en los datos o API, deben ser registrados de forma clara.

### **📊 Evaluación**

✅ **Manejo y procesamiento de datos:** Cómo estructuraste la ingesta, validación y transformación.  
✅ **Optimización de consultas:** Estructura de la base de datos, índices, rendimiento de consultas.  
✅ **Gestión de APIs externas:** Cómo manejaste errores, optimización de llamadas.  
✅ **Buenas prácticas en código:** Modularidad, documentación, calidad del código.  
✅ **Escalabilidad y eficiencia:** Cómo manejaste grandes volúmenes de datos y optimizaste el rendimiento.

### **🚀 Entregable**

* Código subido a un repositorio en **GitHub**.
* Un **README.md** con explicación clara de la solución y pasos para ejecutarla.
* Scripts para procesar los datos y generar reportes.
* Opcional: Un Jupyter Notebook con análisis exploratorio de los datos.

### **🔥 ¿Qué cambia respecto a la prueba original?**

✅ Más enfoque en **procesamiento y almacenamiento de datos** en lugar de desarrollo de microservicios.  
✅ Se eliminan requisitos estrictos de **MVC y diseño de software**, que no son tan relevantes para un Data Engineer.  
✅ Se incluyen **reportes y optimización de consultas**, aspectos clave para un ingeniero de datos.  
✅ Se mantiene la integración con API, pero con enfoque en manejo eficiente de datos.