**Descripción del Proyecto**

**Observatorio de Obras Urbanas**

**Integrantes:**

**María Florencia Canale**

**Enzo Páez**

**Eduardo Tolaba**

**Pablo Álvarez Acuña**

**Marcelo González**

**Introducción**

El presente trabajo se desarrolló como parte de un proyecto práctico de Programación Orientada a Objetos (POO). El objetivo fue diseñar y desarrollar un software en Python para gestionar datos de obras urbanas en la Ciudad Autónoma de Buenos Aires (CABA). Utilizamos un dataset público que contiene información detallada sobre estas obras, y empleamos herramientas de manipulación de datos y un ORM para la persistencia en una base de datos.

**Planificación**

Para abordar el proyecto, seguimos una planificación estructurada dividida en las siguientes etapas:

1. **Definición de requerimientos:**
   * El software debía permitir la carga, limpieza, normalización y visualización de datos.
   * La persistencia de los datos se realizaría mediante una base de datos SQLite.
2. **Estructura del proyecto:**
   * El proyecto se dividió en cuatro módulos principales:
     + **gestionar\_obras.py:** Encargado de la extracción, limpieza, normalización y carga de datos.
     + **modelo\_orm.py:** Configuración de la base de datos y las tablas usando el ORM Peewee.
     + **normalizador.py:** Funciones para la normalización de datos.
     + **main.py:** Punto de entrada del programa.
3. **Herramientas utilizadas:**
   * **Python:** Como lenguaje principal.
   * **Pandas y NumPy:** Para la manipulación de datos.
   * **Peewee:** Para la gestión ORM.
   * **SQLite:** Para la base de datos.
4. **Dataset:**  
   Se utilizó un archivo llamado *observatorio de obras urbanas.csv*, que contenía información como nombre, dirección, estado, fechas, monto, empresa constructora, y más.

**Implementación**

El desarrollo siguió los siguientes pasos:

1. **Conexión a la base de datos:**  
   Se estableció una conexión con SQLite y se configuraron las tablas mediante el ORM Peewee.
2. **Extracción de datos:**  
   Los datos se leyeron desde el archivo CSV, manejando posibles errores como el formato incorrecto o valores faltantes.
3. **Limpieza y normalización:**  
   Se implementaron funciones en el módulo *normalizador.py* para:
   * Estandarizar nombres y fechas.
   * Validar CUITs.
   * Convertir montos a valores numéricos.
4. **Carga en la base de datos:**  
   Los datos limpios se almacenaron en la base de datos SQLite.
5. **Consulta y visualización:**  
   Se desarrolló funcionalidad para consultar los datos almacenados y mostrarlos en la consola.

**Resultados**

El sistema logró:

* Leer y procesar el dataset de obras urbanas.
* Normalizar los datos y almacenarlos en una base de datos estructurada.
* Permitir la visualización de registros desde la base de datos.

A pesar de algunos desafíos iniciales (como errores en la lectura del archivo CSV y problemas de mapeo de columnas), se logró un software funcional y adaptable a diferentes necesidades.

**Conclusión**

Este proyecto permitió aplicar conceptos de POO, manejo de datos y persistencia en bases de datos. Como pasos futuros, se podrían agregar:

* Una interfaz gráfica para mejorar la usabilidad.
* Funciones avanzadas de filtrado y generación de reportes.
* Validación adicional para evitar inconsistencias en los datos cargados.