Soporte a la gestión de datos con Programación Visual

TPI Grupo 4 - Scraping

Alumnos:

Elías, Juan Ignacio

Scarano, Lucas

Docentes:

Castagnino, Mario

Torres, Juan Ignacio

2022

ARGENPROP-SCRAPING

Descripción del proyecto

Utilizando la biblioteca de Python **‘BeautifulSoup**’ se realiza scraping de la página de 'Argenprop' (sitio de alquiler y venta de propiedades) para obtener datos de interés.
Los datos de interés de cada propiedad listada en Rosario (con la posibilidad de ampliar a más ciudades) son:

1. dirección,
2. barrio al que pertenece
3. precio
4. moneda (USD o ARS)
5. tipo de operación (venta o alquiler)

Una vez recolectada toda la información, se almacena en una base de datos para su posterior análisis.

Se llevó a cabo un análisis estadístico de los precios de las propiedades en cada barrio de Rosario. Se calcularon promedios, valores mínimo y máximo de precios, y se consideró la conversión de moneda para las propiedades listadas en dólares americanos.

Se implementó una interfaz de usuario interactiva que permite visualizar los resultados del análisis estadístico en un Leaflet Map. Los usuarios pueden visualizar los diferentes barrios de la ciudad, y al hacer clic en cada uno, se muestra la información de precios del barrio correspondiente y la cantidad de propiedades listadas en ese barrio. Además, se incorporó una opción para filtrar entre propiedades en alquiler y en venta, lo que permite a los usuarios refinar su búsqueda.
Existe escalabilidad en la posibilidad de agregar más filtros, como por ejemplo cantidad de habitaciones, aunque esto requeriría el scraping de más datos.

Requerimientos funcionales

1. Extracción de información: La herramienta debe ser capaz de extraer la dirección, el barrio y el precio de cada propiedad de la página web de 'Argenprop'.

2. Almacenamiento de información: La herramienta debe guardar la información recolectada en una base de datos para su posterior análisis.

3. Análisis estadístico: La herramienta debe ser capaz de realizar un análisis estadístico de los precios de las propiedades en cada barrio de la ciudad de Rosario, incluyendo la obtención de valores promedio, mínimo y máximo de precios.

4. Conversión de moneda: La herramienta debe ser capaz de detectar si una propiedad está listada en dólares americanos y realizar la conversión correspondiente a pesos argentinos.

5. Interfaz de usuario interactiva: La herramienta debe presentar los resultados del análisis estadístico en un Leaflet Map y permitir a los usuarios explorar los diferentes barrios de la ciudad. Además, debe incluir un filtro para seleccionar entre propiedades en alquiler y en venta.

6. Facilidad de uso: La herramienta debe ser fácil de usar para los usuarios, con instrucciones claras y una interfaz intuitiva.

Requerimientos no funcionales

1. Fiabilidad: La herramienta debe ser confiable y precisa en la extracción y análisis de datos para garantizar la exactitud de los resultados.

2. Velocidad: La herramienta debe ser capaz de extraer y analizar grandes cantidades de datos en un tiempo razonable.

3. Mantenibilidad: La herramienta debe ser fácil de mantener y actualizar en caso de cambios en la página web de 'Argenprop' o en los requisitos de la herramienta.

4. Usabilidad: La herramienta debe ser fácil de usar y entender para los usuarios, sin requerir conocimientos técnicos especializados.

5. Portabilidad: La herramienta debe ser compatible con diferentes sistemas operativos y navegadores web para asegurar que los usuarios puedan acceder a la herramienta sin limitaciones.

6. Eficiencia: La herramienta debe ser eficiente en términos de uso de recursos y capacidad de respuesta, minimizando el uso de memoria y tiempo de ejecución para mejorar la experiencia del usuario.

7. Escalabilidad: La herramienta debe ser capaz de manejar grandes cantidades de datos y ser escalable para futuras actualizaciones y mejoras.

8. Reusabilidad: La aplicación debe ser diseñada para permitir la integración con otros sistemas y herramientas, para mejorar su funcionalidad y usabilidad.

Stack Tecnológico

Capa de Datos

- SQLite: sistema de gestión de base de datos relacional para almacenar los datos recolectados.

- Django ORM: se definen modelos en Django que representan las propiedades y los barrios. Django ORM utiliza el mapeo objeto-relacional para conectar los objetos de Python con las tablas de la base de datos. Para realizar consultas a la base de datos utilizando SQL. Migraciones de base de datos para actualizar la estructura de la base de datos en el desarrollo.

Capa de Negocio

- Python: lenguaje de programación para realizar la extracción de datos con BeautifulSoup, la manipulación y análisis de datos con pandas y numpy, y la construcción de la interfaz de usuario con Django.

- Numpy: librería de Python que se utiliza para realizar operaciones matemáticas y científicas eficientes en matrices y arrays de datos numéricos.

- BeautifulSoup: librería de Python para extraer información de páginas web.

- PythonAnywhere: plataforma en la nube para implementar y alojar la aplicación web.

Capa de Presentación

- Django: framework de Python para construir la interfaz de usuario y el backend de la aplicación web.

- HTML, CSS, y JavaScript: lenguajes de marcado y programación para construir la interfaz de usuario.

- Leaflet: librería de JavaScript para construir el mapa interactivo.

Reglas de Negocio

1. Recopilación de datos: La recopilación de datos de la página web ‘Argenprop’ debe realizarse respetando las leyes y normativas aplicables, así como los términos y condiciones de uso del sitio web.

2. Precisión de datos: La información recolectada debe ser precisa y actualizada. Por lo tanto, cualquier error en la extracción de datos debe ser corregido lo antes posible.

3. Conversión de moneda: El proyecto debe garantizar que la conversión de moneda sea precisa y actualizada, y que se realice utilizando los tipos de cambio oficiales o los tipos de cambio acordados.

4. Cálculo de promedios: El cálculo de los promedios de precios, el mínimo y el máximo de cada barrio debe ser preciso y actualizado.

5. Filtro de búsqueda: El filtro de búsqueda por tipo de propiedad (alquiler o venta) debe funcionar correctamente y garantizar que se muestren solo los resultados correspondientes.

6. Interfaz de usuario: La interfaz de usuario debe ser clara y fácil de usar, y garantizar que la información sea fácilmente accesible para los usuarios.

Casos de Uso Principales

Nombre del caso de uso: Consulta de precios de propiedades por barrio y tipo de operación.

Actor principal: Usuario.

Objetivo: Permitir al usuario buscar propiedades inmobiliarias según el barrio y el tipo de propiedad (alquiler o venta) en la ciudad de Rosario.

Precondiciones: El usuario ha ingresado a la página del proyecto.

Flujo de eventos:

1. El usuario selecciona entre alquiler y venta para ver los resultados que desea buscar.

2. El sistema actualiza los barrios disponibles según las propiedades disponibles en la base de datos.

3. El usuario pasa el cursor sobre uno de los barrios.

4. El sistema muestra el nombre del barrio en la esquina superior derecha.

5. El usuario selecciona el barrio de su interés en el mapa interactivo.

6. El sistema muestra los datos estadísticos correspondientes al barrio y tipo de operación seleccionados.

7. El usuario puede realizar una nueva búsqueda cambiando el barrio o el tipo de operación.

Postcondiciones: El sistema muestra los resultados de la búsqueda de acuerdo con los criterios seleccionados por el usuario.

Nombre del caso de uso: Reiniciar la base de datos.

Actor principal: Usuario.

Objetivo: Permitir al usuario reiniciar la base de datos del proyecto mediante el botón "Reiniciar Base de Datos".

Precondiciones: El usuario ha ingresado a la página del proyecto.

Flujo de eventos:

1. El usuario hace clic en el botón "Reiniciar Base de Datos".

2. El sistema muestra un mensaje de confirmación para asegurarse de que el usuario desea continuar con la operación.

3. El usuario confirma que desea reiniciar la base de datos.

4. El sistema inicia el proceso de reinicio de la base de datos mediante el scraping de la página "Argenprop".

5. El sistema elimina todos los datos existentes en la base de datos y carga los nuevos datos obtenidos mediante el scraping.

6. El sistema muestra un mensaje de éxito indicando que la operación se ha completado correctamente.

Postcondiciones: La base de datos ha sido reiniciada y contiene los datos más recientes obtenidos mediante el scraping de la página "Argenprop".

Flujo alternativo:

Si el usuario cancela la operación de reinicio de la base de datos en el paso 4, el sistema muestra un mensaje indicando que la operación ha sido cancelada y la base de datos no ha sido reiniciada.

Diagrama Entidad Relación

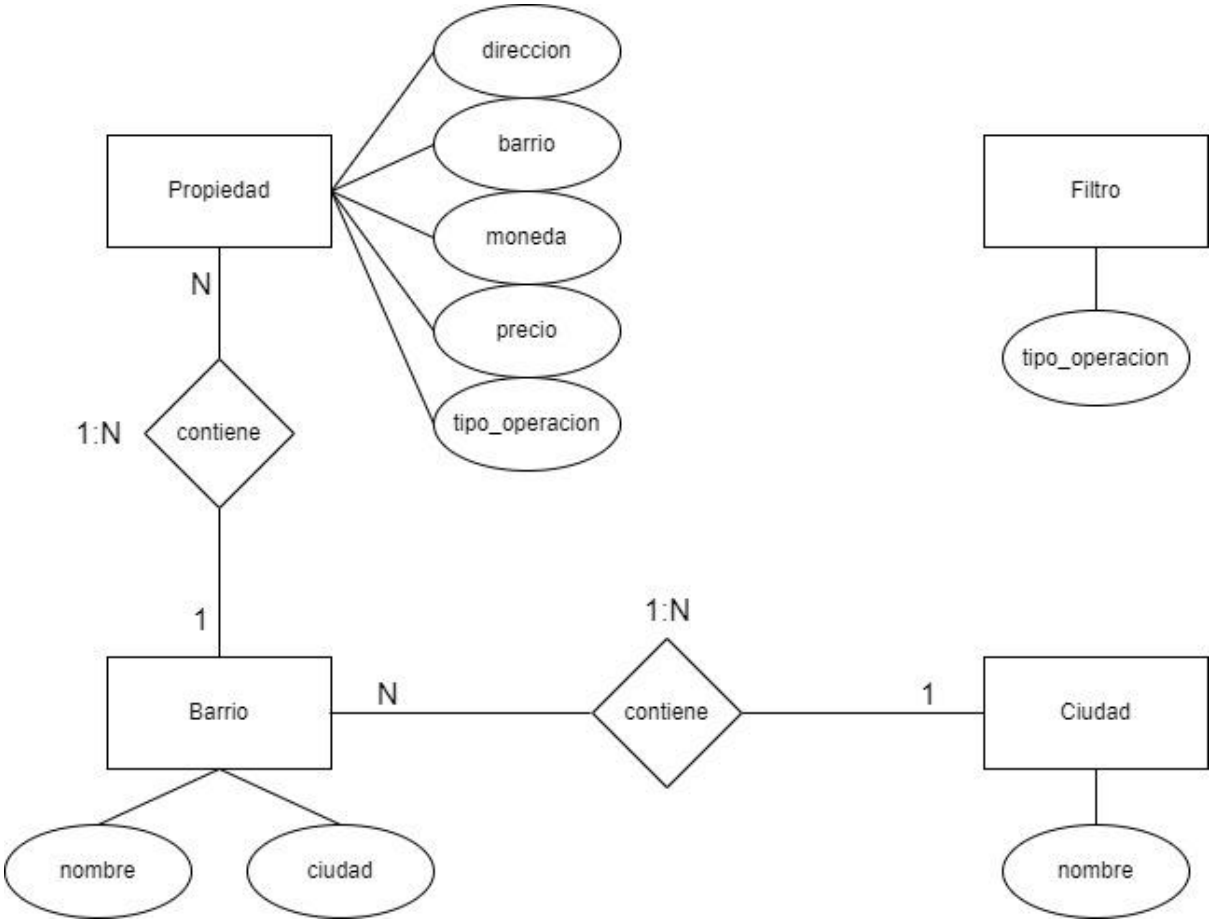
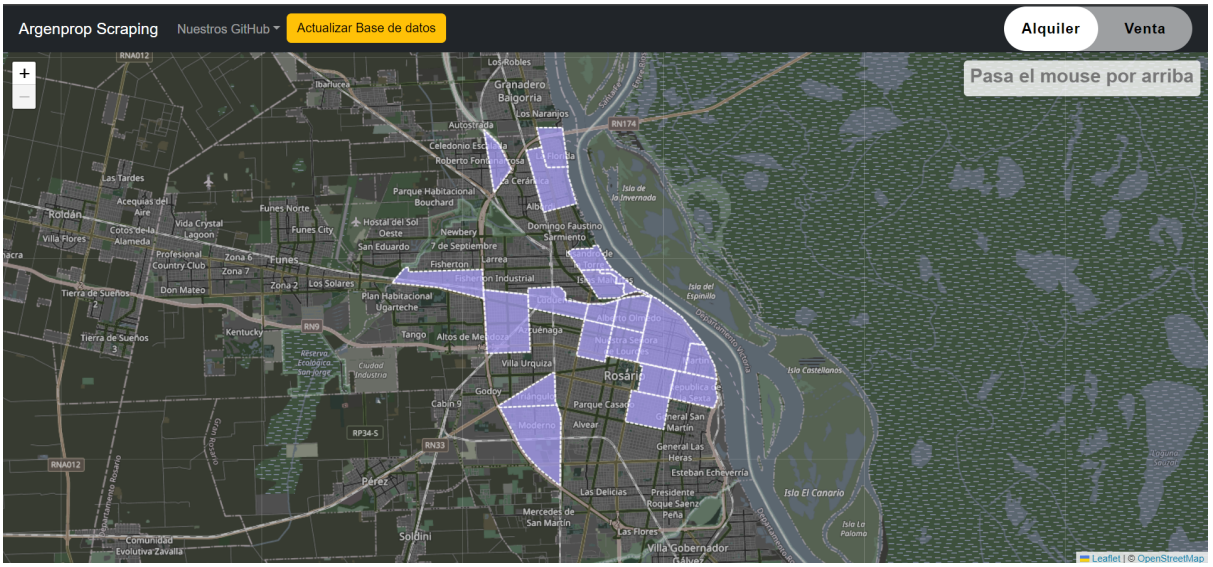


Imagen pantallas



Bibliografía

[Documentación de Django]

[Documentación de la biblioteca Leaflet JS]

[Uso de JSON en JavaScript]

[GeoJSON]

Link proyecto desplegado

http://juani347.pythonanywhere.com

Nota: Al actualizar la base de datos puede que se exceda la quota de procesamiento del servicio de hosting y se ralentice la carga.

Link código fuente

https://github.com/lucasscarano/frro-python-2022-04/tree/TPI-Scraping