

**INSTITUTO TECNOLÓGICO DE BUENOS – ITBA**  
**ESCUELA DE POSTGRADO**

# **PORTAL DE PUBLICIDAD INMOBILIARIA CENTRADO EN EL USUARIO**

**AUTOR:** Mariano Diego Miró – Legajo 105088  
**TUTORA:** Dra. Leticia Gómez

**TRABAJO FINAL PRESENTADO PARA LA OBTENCIÓN DEL TÍTULO DE ESPECIALISTA EN CIENCIA  
DE DATOS**

**Ciudad de Buenos Aires, Noviembre 2022**

## Índice de Contenidos

1. Introducción .....	4
2. Contexto .....	4
3. Presentación del Problema .....	7
4. Justificación del Estudio .....	7
5. Alcances del Trabajo y Limitaciones .....	8
6. Objetivos.....	8
6.1. Objetivo General.....	8
6.2. Objetivos Específicos .....	9
7. Metodología Aplicada .....	9
7.1. Variables .....	9
7.2. Modelado de Datos.....	10
7.3. Técnicas y Herramientas .....	12
8. Desarrollo del Modelo .....	14
8.1. Fuentes de Información .....	14
8.2. Creación y Conexión de la Base de Datos.....	15
8.3. Proceso de ETL Aplicado .....	16
8.4. Armado del Portal.....	17
9. Resultados Experimentales .....	24
10. Conclusiones y Trabajo a Futuro .....	31
11. Referencias.....	34
11. Anexo .....	35
Anexo 12.1. Creación y Conexión de la Base de Datos .....	35
Anexo 12.2. Código del proceso ETL.....	36
Anexo 12.3. Código de la Implementación.....	38



## 1. Introducción

La vida del ser humano está expuesta a múltiples desafíos. Uno de ellos es la búsqueda de un inmueble para alquilar o comprar. Muchas veces las personas pueden estar mucho tiempo realizando esa búsqueda. Varios estudios (Xiaofang Yuan, 2013) han señalado que como mínimo toma tres semanas poder encontrar un inmueble que satisfaga las necesidades que tiene una persona, referido a ubicación, luminosidad, barrio en el que se encuentra, cercanía a lugar de interés (trabajo, colegios, esparcimiento). Y en algunas ocasiones las personas no quedan del todo conformes con su elección generando un arrepentimiento en la compra o alquiler efectuado.

Este trabajo pretende ahondar en las razones de esta dificultad. Aún en la era digital en la que estamos los sitios de publicidad de inmuebles no ofrecen información contextual del inmueble (barrio, escuelas, lugares de esparcimiento, etc.). Se estudiarán los antecedentes encontrados, cómo el marco teórico que rige actualmente en los sitios de búsqueda de inmuebles impide que las personas puedan obtener toda la información que precisan. Se estudiarán y ampliarán los conceptos que deben ser incorporados, y se trabajará en la diagramación y conceptualización de un nuevo sitio web, donde las personas puedan tener la mayor información posible no solamente de los inmuebles sino también del entorno en el cual están los mismos ubicados.

También se analizarán diversas herramientas que puedan contribuir para mostrar información contextual del inmueble, del barrio y de los servicios, de manera que le permitan al usuario tener control sobre qué quiere ver realmente.

## 2. Contexto

El uso de la tecnología y los datos en las actividades humanas ha ido incrementándose, acompañado por el desarrollo de internet y la mayor capacidad y menor costo de los sistemas computacionales. El uso de los datos mediante Big Data posibilita manipular grandes volúmenes de datos y hacerlos asequibles para el público consumidor, y el uso de nuevas tecnologías como la realidad aumentada y el escaneo 3D permiten la visualización de un inmueble de una manera más ampliada. Dicha tecnología ha comenzado a utilizarse en mercados más desarrollados como los Estados

Unidos y Gran Bretaña (Ullah, Sepasgozar, & Wang, 2018). Aquí en la Argentina su uso es muy esporádico y difícil de acceder para las múltiples inmobiliarias existentes.

Diversos autores han estudiado el uso de tecnologías disruptivas en el mercado inmobiliario. Se han identificado dentro de ellas 9 tecnologías que pueden ser especialmente importantes para modificar los estándares actuales del mercado inmobiliario:

- Drones: uso de drones para visualizar mejor las propiedades.
- Internet de las Cosas: la utilización de aparatos cotidianos para el intercambio de información mediante internet.
- Nubes: la guarda de los datos en servidores especializados (AWS, Azzure, Google Cloud, etc.).
- Software como un servicio: software que es accedido mediante una suscripción en lugar de una compra de licencia.
- Big Data: la utilización de datos masivos interconectados para nutrir de información.
- Escaneo 3D: el uso de escáneres para crear modelos 3D de los inmuebles publicados.
- Tecnología usable en el cuerpo: tecnología que es usada por el usuario como una prenda.
- Realidad Virtual y Realidad Aumentada: el uso de realidad virtual para ubicar a un usuario en un lugar creado virtualmente. La realidad aumentada para permitir a un usuario ubicar objetos virtuales en un lugar real.
- Inteligencia Artificial: simulación del razonamiento humano por un sistema informático.

En la Argentina estas tecnologías son utilizadas de manera dispar y no existe una integración completa de todas ellas. Actualmente las publicaciones de inmuebles están mayormente conformadas por imágenes, algunas también cuentan con videos y fotos de 360° que permiten un recorrido diferente al usuario (ver [publicación de ejemplo](#)). No se han incorporado las nuevas tecnologías que permiten ampliar la visión que un consumidor pueda tener de un inmueble.

### **Sitios Web Centrados en el Objeto**

Los actuales sitios de publicidad de internet persiguen el objetivo de ubicar al objeto a vender/alquilar como centro de la información. Por tal razón la búsqueda que realiza un usuario está muy limitada a los parámetros preestablecidos por el sitio y la información brindada por el que publica. En prácticamente todos los sitios web, la consulta inicial que hace el usuario se compone de tres parámetros iniciales: Tipo de operación: compra o alquiler; tipo de inmueble: Casa, Departamento, Local, etc. y ubicación: Barrio, Localidad, provincia. A partir de allí, y de acuerdo a su elección, se le muestran al usuario inmuebles que reúnan dichas características. Si quiere buscar por otras características (precio, metros cuadrados, cantidad de dormitorios, baños, etc.), dicha información se ingresa en una segunda búsqueda y el sistema se va refrescando cada vez que una característica se aplica.

### **Sitios Web Centrados en el Usuario**

En los antecedentes analizados (Ullah, Sepasgozar, & Wang, 2018) (Xiaofang Yuan, 2013.) podemos ver la nueva tendencia de posicionar al usuario como centro de todo el sitio. Conocer primero bien cuáles son sus prioridades más importantes acerca de una propiedad, con preguntas que no solo tiendan a buscar características del inmueble, sino también del entorno donde está ubicado, para recién allí mostrarle opciones que puedan ser mucho más útiles para dicho usuario. Según diferentes estudios muchos usuarios privilegian lugares cercanos a su trabajo, donde el tiempo de viaje desde y hacia el trabajo sea corto. Otros usuarios privilegian acceso a lugares de recreación, zonas comerciales y parques. Mientras que usuarios con hijos/as privilegian lugares cercanos a buenos centros educativos.

Reuniendo esa información, el portal podría mostrarle diferentes opciones al usuario, incluso en barrios que quizás no había considerado inicialmente. De esa forma se trata de satisfacer las necesidades principales del usuario, reduciendo el tiempo de búsqueda y también facilitando la rapidez en las operaciones.

En la literatura se identifican cinco conceptos correspondientes al problema mencionado:

1. Usuario: ubicación del usuario como centro de interés para brindar información clara y suficiente para que pueda tomar una decisión.
2. Información: complementar la información específica del inmueble con información relativa al barrio donde está ubicado.
3. Asimilación de la Tecnología existente: utilización de la tecnología para tener un mayor conocimiento sobre el inmueble y su barrio. (Drones, Realidad Virtual y Aumentada y Big Data principalmente)
4. Sistematización de la información: la información que se muestre al usuario debe guardar un mismo criterio de cantidad y calidad sin importar que provenga de fuentes distintas.
5. Visualización clara: visualización de la información en forma de gráficos que faciliten la comprensión de la misma por el usuario.

### **3. Presentación del Problema**

Los avisos publicitarios de búsqueda de alquiler o venta de inmuebles muestran una información específica del inmueble (descripción, fotos y videos), y en algunos casos, su geolocalización, pero despliegan poca o ninguna información relativa al barrio al cual pertenece ese inmueble, lo cual puede ser importante para una toma de decisión.

Por otro lado, tampoco se hace uso de la enorme y variada información accesible en nuestros días desde múltiples fuentes, ni de las tecnologías disruptivas que permitirían una mejor visualización de los inmuebles.

### **4. Justificación del Estudio**

El presente estudio se justifica por la necesidad de modificar el actual sistema de publicidad del mercado inmobiliario, poniendo al usuario en el centro de interés de la búsqueda, y utilizando aquellas tecnologías basadas en Big Data que realmente provoquen una disrupción en el actual sistema de búsqueda.

Teniendo en cuenta esto, junto al hecho de que no se hace suficiente hincapié en la necesidad de brindarle al usuario información clara y estandarizada como principal punto de partida para que éste pueda tomar una decisión, aparece la necesidad de unificar en un mismo sitio información sistematizada de los inmuebles con la información del barrio donde están ubicados, colocando al usuario como centro del sistema.

## **5. Alcances del Trabajo y Limitaciones**

El presente trabajo abarcará la concepción teórica de un portal de servicio de publicidad de inmuebles que le permita al usuario contar con información suficiente y reducir el tiempo de búsqueda de un inmueble. Se colocará al usuario como centro de la atención del sitio, permitiendo que pueda conocer diferentes inmuebles, tomando en cuenta opciones que actualmente no son brindadas por los actuales sitios, como, por ejemplo, cercanía de escuelas, preferencias por lugares como parques o centros comerciales, ofertas gastronómicas, etc.

Dado que el objetivo principal es presentar la visualización de los inmuebles junto a información de contexto elegida por el usuario y a datos relevantes del barrio que contiene a la propiedad, no se avanzará en el desarrollo de un portal completo de búsqueda, sino que se generarán las componentes necesarias para exponer una prueba de concepto.

También cabe mencionar que el presente trabajo estará limitado únicamente a los inmuebles de la Ciudad de Buenos Aires.

## **6. Objetivos**

### **6.1. Objetivo General**

Presentar una prueba de concepto de un portal inmobiliario, que recopile e integre información contextual de los barrios de la ciudad (lugares de interés, estadísticas de robo y delincuencia, etc.) para combinarla con la información de los inmuebles ofrecidos, presentándole al usuario un tablero de información integrada que lo ayude a tomar una decisión sobre la compra o alquiler de un inmueble.

## 6.2. Objetivos Específicos

- Obtener datos de ubicación de establecimientos educativos, comisarías, zonas comerciales, principales medios de transporte.
- Crear de base de datos donde pueda volcarse la información anterior.
- Elaborar gráficos de estadísticas de inseguridad por barrio.
- Establecer una radio geográfico mínimo de información contextual que se exhibirá en el mapa al usuario, tomando como punto central la ubicación del inmueble elegido.
- Desarrollar una aplicación de visualización de los datos del inmueble publicado, a la cual se le integren los datos complementarios obtenidos.

## 7. Metodología Aplicada

La información contextual sobre transporte, educación, salud entre otras, puede mejorar la calidad de los avisos publicitarios inmobiliarios en la Ciudad de Buenos Aires, reduciendo a los usuarios los tiempos de búsqueda de un nuevo inmueble para residir.

A los fines de mostrar una prueba conceptual sobre la solución propuesta, desarrollaremos una aplicación piloto, que luego podrá ir creciendo en el tiempo. Para ello, se utilizará metodología ágil, ya que permite la elaboración por pasos, nutriendo la base de datos con la información y realizando testeos a medida que se va avanzando en la construcción.

### 7.1. Variables

Las variables a tener en cuenta son:

- Inmuebles publicitados para la compra o alquiler.
- Localización de los inmuebles por barrio.
- Servicios de educación (colegios, universidades)
- Establecimientos de salud
- Comisarías y centrales de bomberos
- Lugares culturales y de gastronomía

- Radio máximo a considerar para mostrar las diferentes opciones de establecimientos educativos, de salud y comerciales.

## 7.2. Modelado de Datos

Si bien este trabajo se limita a la visualización de los inmuebles junto a información contextual relevante del entorno de la propiedad, el modelado de los datos se focalizó sobre todos los escenarios posibles de un portal inmobiliario, habida cuenta de que el desarrollo conceptual puede seguir avanzando hasta lograrse un portal operativo completo.

En ese aspecto, los escenarios propuestos son los siguientes:

- La publicidad de un inmueble que un usuario desea ofrecer.
- Las búsquedas que un usuario puede realizar respecto de inmuebles que deban reunir las características deseadas para su futuro hogar, permitiendo crear, editar, guardar y borrar búsquedas personalizadas para ser utilizadas en el futuro.
- La posibilidad que los usuarios puedan almacenar la información de inmuebles favoritos encontrados.

Entre las colecciones principales, encontramos:

- **Usuarios:** de los usuarios es importante contar con un nombre de usuario, un correo electrónico que debe ser único, no aceptándose dos usuarios con el mismo correo electrónico. Y finalmente, una contraseña para acceder al sitio y poder realizar las acciones previstas en los escenarios.

Salvo acciones propias del usuario que desee suprimir su perfil o del administrador que por violaciones de las condiciones de uso ameriten una expulsión, la permanencia del usuario no tendrá caducidad alguna.

- **Inmuebles:** el primer y tercer escenario están íntimamente vinculados ya que involucran a un inmueble en particular. Su permanencia en la base de datos estará ajustada específicamente al tiempo de duración del aviso abonado por el usuario que realiza la publicación. Puede establecerse un sistema de guardado

de los datos del inmueble, una vez que se ha vencido el plazo del aviso por un tiempo para que el usuario que publicó el aviso no deba cargarlo nuevamente.

El modelo de los datos que debe ingresar el usuario que realiza la publicación debe ser de carácter rígido para ciertos datos. El dato de la ubicación entra en esta rigidez, debiendo validarse la dirección ingresada con una búsqueda que puede realizarse en Open Street Map. A partir de esa validación se puede obtener las coordenadas del inmueble y el barrio donde se encuentra ubicado, no dejando al usuario la posibilidad de elegir un barrio que no corresponde de acuerdo con la ubicación del inmueble, y exigiendo de esa manera que se publique la veracidad completa de la ubicación del inmueble.

Las coordenadas y el barrio son importantes para poder realizar la búsqueda geoespacial de los servicios cercanos al inmueble y el barrio para mostrar diferencialmente las estadísticas de seguridad, resaltando el barrio donde está ubicado el inmueble.

Las medidas de los ambientes que conforman el inmueble y su tipo deben estar incluidas en los datos a publicar.

Otro dato que también se considera importante es que cada foto tenga una leyenda explicativa de que es lo que el usuario está mirando. Así se deberá forzar a que cada foto que se suba incluya una leyenda descriptiva.

La mayor cantidad de datos descriptivos de los inmuebles surge importante para mostrar la mayor información al usuario que está mirando el aviso.

- **Avisos:** La colección de Avisos contendrá la información relevante del aviso, con la fecha de su creación, el precio del inmueble, los detalles de la operación, esto es, tipo de operación (venta – alquiler), moneda de operación (peso – dólar) y el importe de esta. Además, contendrá la clave foránea del inmueble que forma parte del aviso y del usuario-anunciante con los datos de contacto.

Las colecciones de información contextual se construirán a partir de los datos obtenidos en los Datasets del Gobierno de la ciudad de Buenos Aires. Los Datasets analizados y agregados mediante ETL serán de: Bomberos, Comisarias, Escuelas, Universidades, Culturales, Gastronomía y Hospitales. Se incluirán datos resumidos de cada uno, pero el dato más importante son las coordenadas geográficas de ubicación del establecimiento, ya que a partir de ella se hará la consulta para poblar el mapa que se exhibirá al usuario. A los efectos de no sobre saturar el mapa y brindar información clara y concisa, se exhibirán solamente los datos de establecimientos que estén en un radio máximo de 1500 metros desde el inmueble que el usuario está observando.

Finalmente se agregará una colección que contenga la información de estadística de delitos en la ciudad de Buenos Aires. Debido a que el GCBA no tiene un dataset con dicha información se ha creado un dataset a partir de la información que el GCBA publica en un informe anual. La información está actualizada al año 2021. La información relevante a considerar es: homicidios, robos, y robos de automotores.

Con el esquema planteado, se ha armado el modelo lógico de la base de datos, que puede consultarse en la Figura 1.

### **7.3. Técnicas y Herramientas**

El punto inicial es el armado de la base de datos. La misma puede ser del tipo SQL o no SQL. La primera opción ayuda a mantener una base ordenada y estructurada de datos, los cuales han sido previamente cargados. Sin embargo, cuando se trata de obtener datos en tiempo real provenientes de servicios web, las bases de datos NoSQL tienen una mayor dinámica, especialmente para el manejo de grandes volúmenes de datos. Para el armado de la base de datos NoSQL hemos optado por MongoDB, herramienta reúnen los requisitos necesarios para el desarrollo de la base de datos deseada. Además, MongoDB ofrece un tier gratuito de por vida para probar aplicaciones, el cual utiliza los servicios de AWS (Amazon Web Services). La base de datos está alojada en la Ciudad de San Pablo, Brasil. La creación de la base de datos se hizo directamente en la página de MongoDB.

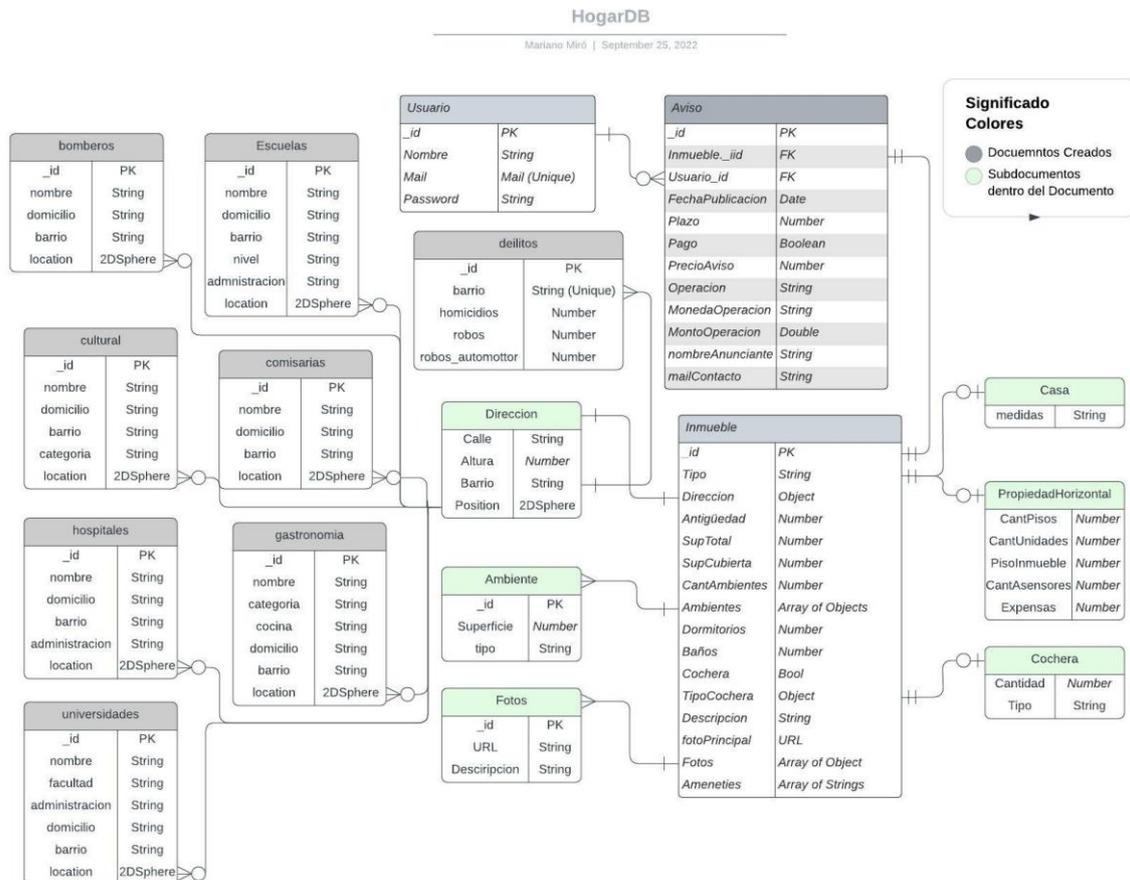


Fig. 1 – Modelo de la base de datos

Para la conexión a la base de datos se utilizó el Driver de node.js, para lo cual se utilizaron dos librerías pertenientes a NPM (Node Package Manager): mongodb y mongoose.

Para efectuar el ETL se utilizó Python con las librerías Numpy y Pandas para la carga, modificación de los datos y modelado de los mismos para después ser cargados en la base de datos con la librería pymongo.

Para la confección de la página web se optó por utilizar el framework de Next.js perteneciente a Vercel, haciendo uso de las librerías node.js, react.js, next.js, bcrypt.js, jsonwebtoken.js, d3.js, react-leaflet.js y react-embla-corousel.js.

Para los gráficos de barras se estudió la posibilidad de usar las librerías Vega y Vega Lite, pero debido a que se precisaba una mayor libertad para realizar el diseño del gráfico se optó por utilizar la librería D3.js para la unión y mapeo de los datos, armando el gráfico directamente en SVG.

## 8. Desarrollo del Modelo

### 8.1. Fuentes de Información

Las fuentes de datos utilizadas son las siguientes:

- Open Street Map (URL: <https://nominatim.openstreetmap.org>): Se usa la página para confirmar la dirección ingresada por el usuario que está cargando un aviso. A partir de la respuesta del servidor se obtiene el barrio donde está el inmueble publicado y las coordenadas geográficas. Es de destacar que aún cuando se ingrese una dirección errónea por altura (por ejemplo, Reconquista 14050 que no existe en CABA con esa altura) Open Street Map envía una respuesta ubicando geográficamente un punto que señala a la calle o avenida buscada. Sin embargo, cuando la respuesta contiene una altura correcta, esta es la primera en aparecer en el cuerpo de la respuesta. Para poder verificar que la altura es correcta se creó un Regex para verificar que la respuesta empiece con un número: '^([0-9])', 'g'. Este regex verifica que haya números al inicio de la respuesta. A partir de la verificación del número se obtiene de Open Street Map el barrio del inmueble y las coordenadas geográficas. Para que las queries geográficas funciones en mongoDB se deben almacenar las coordenadas primero con la longitud y después con la latitud. A diferencia de lo que ocurre con las páginas de mapas más usadas (Google Maps, Open Street Map y Waze por nombrar algunas) El radio mínimo de la vista o zoom del mapa se estableció en 14.
- La información contextual de lugares cercanos al inmueble se obtiene de los Datasets que publica el GCBA en <https://data.buenosaires.gob.ar/dataset>

- Las estadísticas de delitos se ha obtenido de un informe que ha elaborado la Subsecretaría de Seguridad Comunal e Investigación Criminal del Ministerio de Justicia y Seguridad del GCBA, el cual puede ser consultado en [https://cdn.buenosaires.gob.ar/datosabiertos/datasets/ministerio-de-justicia-y-seguridad/delitos/Informe\\_complementario.pdf](https://cdn.buenosaires.gob.ar/datosabiertos/datasets/ministerio-de-justicia-y-seguridad/delitos/Informe_complementario.pdf)

## 8.2. Creación y Conexión de la Base de Datos

La base de datos para este ejemplo se creó en MongoDB. Se eligió esta base de datos NoSQL porque ofrece la dinámica necesaria que precisamos debido a que en el caso de la colección de inmuebles debido a que los tipos de datos y la cantidad de datos varía de inmueble a inmueble exige que se puedan almacenar mas o menos datos según el tipo de inmueble, todo de acuerdo con el modelo ya expuesto en el punto 7.2.

Para la conexión a la base de datos se utilizó el Driver de node.js, para lo cual se utilizaron dos librerías pertenecientes a NPM (Node Package Manager): mongodb y mongoose. La primera se utilizó para realizar las queries geográficas para la ubicación de lugares de interés cercanos al inmueble que se visualiza y la segunda se utilizó para crear y guardar los datos de los usuarios, avisos e inmuebles publicados. La razón de utilizar las dos librerías obedece a que MongoDB es un tipo de base de datos que en principio puede funcionar sin un esquema de los datos a ingresar. La librería de mongodb no permite la creación de un esquema de datos, pero en su lugar mongoose sí lo permite, manteniendo un orden y uniformidad en aquellos datos que se consideran relevantes. Para ello se crearon dos scripts para cada conexión.

En el Anexo 12.1. se amplían los detalles de dichos scripts.

Debido a que el ETL fue hecho en Python, también se efectuó una conexión a la base de datos de MongoDB mediante el driver de Python. Dicha carga es única ya que los datos no cambian de manera dinámica, y por ello la conexión a la base de datos se efectuó por única vez.

### 8.3. Proceso de ETL Aplicado

El ETL fue utilizado para poblar la base de datos con la información contextual que enriquece la visión del usuario del lugar donde está el inmueble.

Los Datasets del GCBA en general se encuentran curados en la mayoría de los casos, aunque igualmente se pudo observar que en algunos casos faltaban algunos datos o la información estaba repetida. Para efectuar el ETL se utilizó Python con las librerías Numpy y Pandas para la carga, modificación de los datos y modelado de los mismos para después ser cargados en la base de datos con la librería pymongo. Los Datasets estaban almacenados en archivos .csv y fueron cargados mediante Pandas utilizando la codificación de UTF-8 para que pueda reconocer las tildes del idioma castellano.

Se debió normalizar algunas direcciones y en algunos casos se debió unir la información de calle y altura para poder ser guardada en la base de datos. En el caso de las universidades muchas tienen diferentes facultades, pero varias de ellas ubicadas en el mismo edificio. Debido a ello existían muchos datos duplicados referentes a la misma universidad, misma ubicación, pero diferente facultad. Agregarlo así a la base de datos hubiera implicado mostrar en el mapa varios tags de ubicación referido al mismo lugar. Debido a ello se efectuó una transformación, indicando que en un lugar funcionaban varias facultades sin especificar cuáles, y se eliminaron los datos duplicados de las universidades tomando como punto de referencia la dirección y altura de esta.

Debido a que para poder hacer queries geográficas en MongoDB se deben guardar los datos en un tipo de específico de datos (2DSphere) e ingresando las coordenadas en un array donde primero esté la longitud y después la latitud. Se debió catalogar correctamente los datos para que sean cargados en esta forma.

Una vez completado todo el ETL se armó un *dict* de Python donde se colocaban cada dato del lugar de interés. Luego, todos los *dicts* se agregaron a un *list* de Python el cual fue insertado en la base de datos usando la opción de `insert_many` que provee la librería pymongo y que permite la inserción de varios documentos en una colección.

Asimismo, desde la página de MongoDB se agregaron a cada colección los índices especiales sobre los datos geográficos, indispensables para poder hacer las queries geográficas.

En el Anexo 12.2 se puede ver el detalle de la implementación del proceso ETL.

#### **8.4. Armado del Portal**

Para la confección de la página web piloto se evaluaron diferentes alternativas. Primero se comenzó con una página clásica en HTML, CSS y Javascript utilizando a node.js como servidor. Sin embargo, la imposibilidad de que la página tuviera el dinamismo necesario se modificó para utilizar al framework de react.js, el cual poseía ventajas que su construcción modular agrega, permitiendo crear diferentes componentes que pueden ser reutilizados en diferentes partes de la página. Nuevamente nos encontramos con un tema a resolver que se necesitaba que la página pudiera armarse con direcciones dinámicas. Es decir, cada vez que un usuario quería ver un inmueble en particular, la página debía mostrar una dirección específica para dicho inmueble. React es un framework optimizado para páginas que solamente tienen una página (Single Page Applications) y modifica lo que el usuario ve mediante el cambio de componentes que conforman la página. Y como se precisa mostrar un sitio con diferentes páginas y siguiendo la propia recomendación que surge de la documentación de React se optó por utilizar el framework de Next.js perteneciente a Vercel. Dicho framework ofrece la posibilidad de creación dinámica de páginas diferentes y optimización de obtención de datos de la base de datos mediante las funciones `getServerSideProps` y `getStaticProps`.

Ya definida la tecnología, el stack utilizado para el armado de la página consistió en:

- Node.js para las funciones de conexión backend entre la base de datos y el frontend efectuado en react.js
- React.js para el armado de los componentes que conforman la página y la interfaz gráfica que utiliza el usuario.
- Next.js para la creación dinámica de la página diferenciando entre las distintas páginas que se crean y además haciendo uso de Vercel para poner en producción la aplicación.

- MongoDB: La base de datos elegida para guardar toda la información con la cual se nutre la página.
- Bcrypt.js: Librería de Javascript que permite el hasheo de la contraseña que el usuario ingresa cuando se da de alta para su posterior almacenamiento en la base de datos y también para después verificar la contraseña cuando el usuario hace un login.
- Jsonwebtoken.js: Librería de Javascript que permite asignar un token al usuario que ha hecho un login y permite controlar que un usuario tiene acceso a las diferentes partes de la página, donde se precisa controlar el acceso.
- D3.js: Librería de Javascript que se utilizó para la confección del gráfico de estadísticas criminales.
- react-leaflet.js: Librería de Javascript utilizada para la confección del mapa que muestra al usuario los lugares de interés cercanos al inmueble que está observando.
- React-embla-carousel.js: Librería de Javascript utilizada para mostrar el Carrusel de fotos del inmueble y de los inmuebles destacados de la página inicial.

Se comenzó dándole forma a la página inicial de la aplicación donde se mostrarían los inmuebles destacados y las principales notas del blog de la aplicación. Se dio forma al header que contendría el logo de aplicación, el botón para iniciar sesión y los links de navegación que lleva al inicio, blog y búsqueda de inmuebles. El botón de iniciar sesión lleva a otra página donde un usuario puede iniciar sesión y da la opción de crear una cuenta si no se posee una.

La página de creación cuenta exige la entrada de tres datos:

- Nombre del usuario: En formato de texto.
- Email del usuario: En formato de email, el cual es verificado mediante la aplicación frontend que reúna los requisitos de un correo electrónico. Para ello se utiliza el siguiente Regex: `/^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$/`
- Password: Ingreso de una contraseña de como mínimo 6 dígitos aceptándose cualquier tipo de texto.

Ingresados los datos, la librería bcrypt procede a transformar la contraseña ingresada en un hash y esta contraseña es almacenada con los demás datos en la base de datos. Debido a que se ha establecido en el esquema de creación del usuario que el email es de tipo único, el sistema arroja un error cuando se quiere almacenar un usuario con un correo electrónico ya registrado en la aplicación.

La página de login precisa para su inserción el email y la contraseña registrados. Una vez enviado los datos por el usuario primero se verifica que exista el correo electrónico en la base de datos, arrojando un error en caso de que el usuario no existe en la base de datos y después la librería bcrypt compara la contraseña con el hash almacenado en la base de datos y si es correcto efectúa el login. Si es incorrecto arroja un error. Ambos errores mencionados son exhibidos al usuario con el mensaje genérico "email o contraseñas erróneos".

Efectuado el login, el componente header se modifica para exhibir el mensaje de bienvenida al usuario el cual haciendo click en el mismo muestra las opciones que el usuario tiene: Ir a su profile o cerrar la sesión.

Después se le dio forma al footer de la página que contiene la información relevante de la aplicación la cual se debe completar a futuro.

Terminado esto, se armó el carrusel que contendría los inmuebles destacados. Dicho componente del carrusel además de servir para mostrar los inmuebles destacados puede ser utilizado para mostrar en el futuro los inmuebles que arroja una búsqueda personalizada que un usuario haga de ellos. Como se dijo anteriormente se utilizó la librería react-embla-carrousel que ofrece una programación simple y sencilla y que permite además ajustarla a las necesidades de la aplicación. Se agregaron links a la foto que exhibe y al título del aviso que llevan al usuario a ver ese inmueble en particular.

Finalmente se procedió a confeccionar un listado de notas importantes a exhibir del blog mostrando el nombre del autor, título y subtítulo de la nota. (El armado de las colecciones que las agrupan y la api de conexión queda para una etapa posterior al presente trabajo).

Para incorporar los datos al carrusel de inmuebles destacados se utilizó la función de next.js “getServerSideProps”. Esta función permite que a nivel de servidor se realice la conexión con la base de datos para obtener los datos necesarios y se confeccione la página a nivel del servidor y no del cliente, de esa manera no se exhibiría al usuario.

Debido a que, para armar el carrusel no se precisan todos los datos del inmueble, sino los más relevantes, se hizo una query a la base de datos usando la función populate de mongoose, la cual tiene un comportamiento análogo de un left join de SQL. Es decir, se eligen aquellos avisos que tienen establecido una bandera de destacados y se les agrega los datos del inmueble utilizando la “\_id” del inmueble con el dato de “inmueble\_id” contenido en la colección Avisos. Dicha query completa es la siguiente:

```
const result = await Aviso.find({destacado: true}).select('nombreAnunciante monedaOperacion montoOperacion operacion').populate({path: 'inmueble_id', select: 'titulo supTotal cantAmbientes dormitorios direccion fotoPrincipal'})
```

Finalmente, el resultado se mapea y se pasa como props a la página de inicio, la cual a su vez se lo trasmite al carrusel que es el componente hijo de dicha página. El detalle de las componentes se puede ver en la Figura 2.



Fig. 2 – Componentes de la Página de Inmuebles

## Páginas Dinámicas Creadas Mediante Apis Dinámicas

Como se explicó anteriormente next.js permite la creación de apis dinámicas cuya denominación dependerá de la “\_id” del inmueble en cuestión. Allí es donde se mostrará la información del inmueble que el usuario está mirando, el mapa con los lugares de interés cercanos y el gráfico con las estadísticas de seguridad.

La confección de dicha página está dividida en seis componentes: header, footer, carrusel, darts del inmueble, mapa y gráfico de barras.

El header y footer son los mismos que para inmuebles destacados, del cual ya se ha hablado anteriormente.

El carrusel de fotos, a diferencia del carrusel de inmuebles destacados tiene las fotos del inmueble en particular y una leyenda indicando al usuario a que corresponde dicha foto. Otra diferencia con es que el carrusel de destacados que mostraba unos botones para ir de un inmueble a otro, mientras que este carrusel muestra fotos en miniatura del inmueble permitiendo al usuario hacer click en ellos para pasar de una foto a otra. Ver Figura 3.

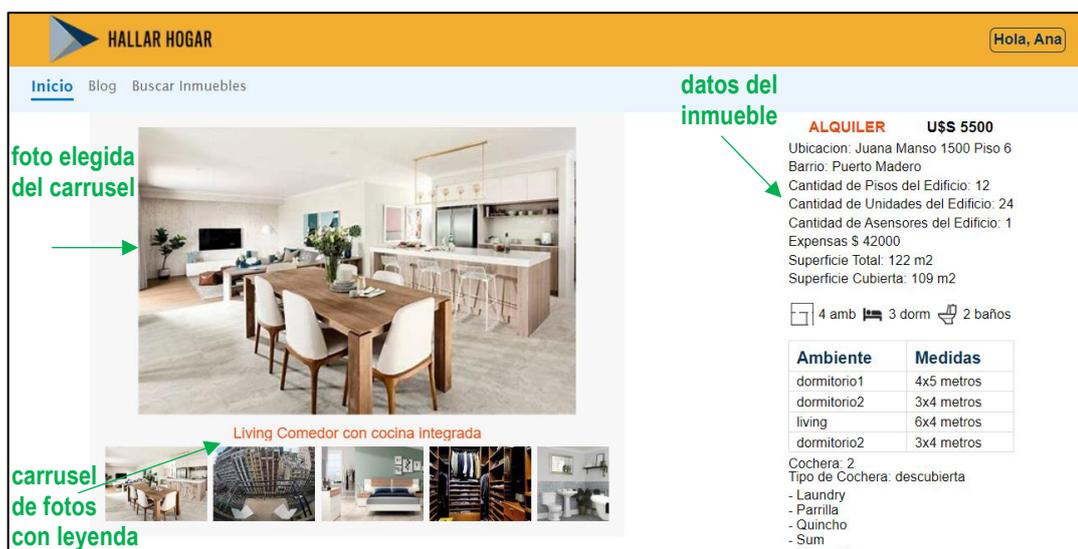
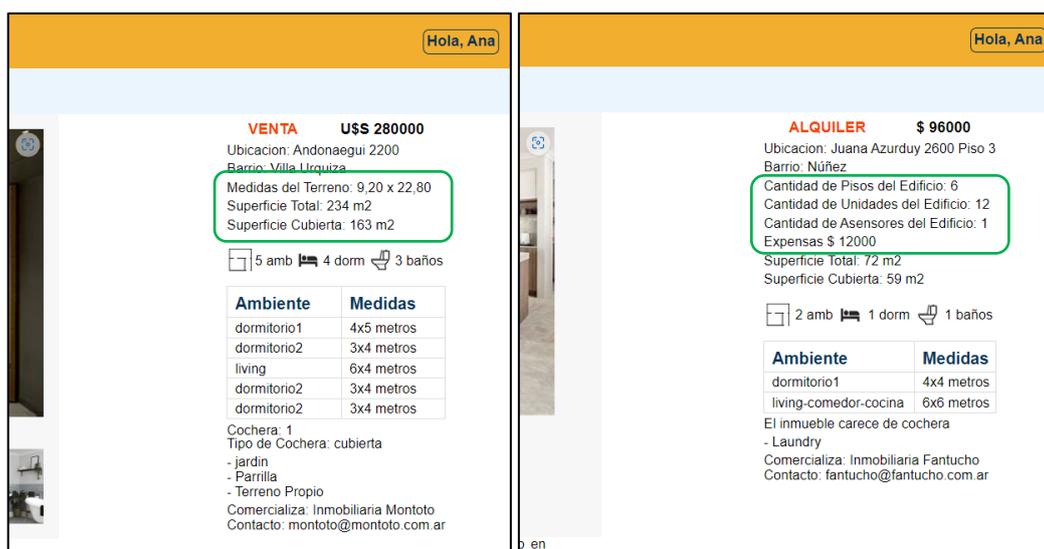


Fig. 3 – Componentes de la Página de un Inmueble Elegido

El componente que muestra los datos del inmueble debió trabajarse con dos componentes diferentes, uno para una casa y otro para un departamento, dado que la información que tiene uno u otro varía por las cualidades propias de cada inmueble. El sistema elige un componente u otro en función del dato “tipo” contenido en la colección “Inmuebles” y renderiza el que corresponde. Además, también elige dinámicamente la información a mostrar en función de que el inmueble tenga o no cochera. Para la elección de un componente u otro se usaron las funciones de React (llamadas hooks), las que permiten manejar el estado de los componentes, modificando el componente a renderizar. Dichos hooks son `useEffect()` y `useState()`. A modo de ejemplo, en la Figura 4 se pueden consultar los distintos datos que se despliegan según el inmueble sea tipo casa o departamento.



(a)

(b)

Fig. 4 – Diferencia en la componente de datos según tipo de inmueble.  
En este caso: (a) casa; (b) departamento

Respecto del componente del mapa se trabajó en tres etapas: El ETL de los datos se trabajó en Python utilizando los Datasets provistos por el GCBA. Desde los mismos scripts de Python se cargaron los datos, previa limpieza y acondicionamiento, a la base de datos en MongoDB. Para la realización del mapa se importó la librería de react-leaflet que permite la renderización de mapas usando Open Street Map. Para el tile se suscribió mediante una API a MapBox para obtener uno que se asemeja al que exhibe Google

Maps, y con el cual la gente está mas familiarizada. Los Markers fueron creados artesanalmente tratando de asemejar por color y diseño al tipo de información que se mostraría en el mapa. Finalmente, la carga de datos se hace dinámicamente y de acuerdo con la ubicación geográfica del inmueble que el usuario está observando, haciendo uso de una query geoespacial de MongoDB que permite seleccionar elementos de las colecciones que se exhiben, en un radio y tomando como punto la ubicación del inmueble. Respecto de la amplitud del radio como ya se dijo se optó por seleccionar aquellos elementos que se encuentran a 1500 metros del inmueble. En la Figura 5 se muestran los detalles de este componente.

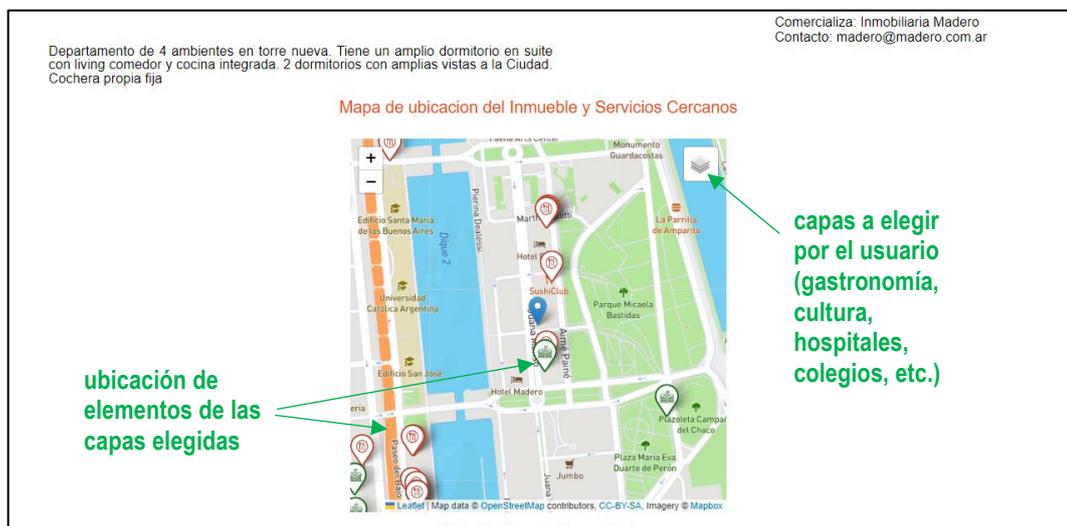


Fig. 5 – Componente de Mapa

El último componente es el gráfico de barras, realizado en SVG, con el uso de la librería D3.js para la unión y mapeo de los datos. Esta elección de tecnología posibilitó darle un diseño que podrá ajustarse a cambios de diseño en la página y por las diferentes tamaños de pantallas que se utilizan. Haciendo uso nuevamente del hook “useState” de React se cambia el componente renderizado en función de la estadística elegida para visualizar. Asimismo, usando también D3.js se mapea los datos dotándole a las barras correspondientes al barrio donde está el inmueble de un color diferente que el resto de las barras para que el usuario pueda identificar de manera rápida las estadísticas que corresponden a dicho barrio. Ver Figura 6.

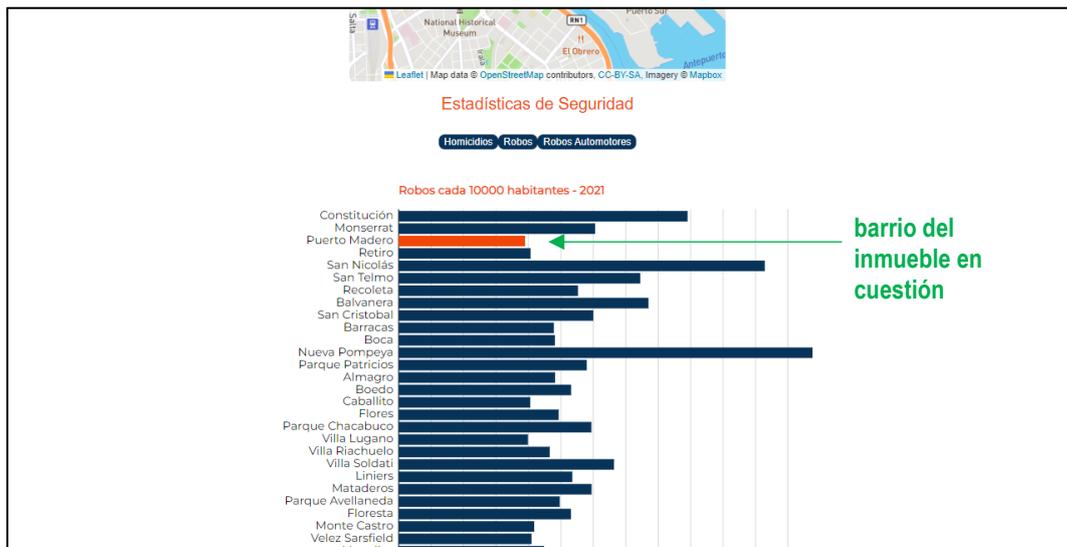


Fig. 6 – Componente Gráfico de Delitos

Todos los detalles de la implementación se muestran en el Anexo 12.3.

## 9. Resultados Experimentales

Finalizado la programación del Backend y el Frontend, se procedió a poner la página online, para lo cual se utilizó el servicio que otorga la empresa Vercel (creadora y mantenedora de Next.js). Realizando algunos ajustes de conexión, la página quedó alojada en la url [tfi-itba.vercel.app](https://tfi-itba.vercel.app). Allí, se puede acceder a la página principal, crear el usuario, hacer el login en la página y ver los inmuebles destacados, junto al enriquecimiento de información contextual dada por el mapa con servicios del entorno y los gráficos de estadísticas criminales de la zona.

Este enriquecimiento ofrece una información complementaria mayor a la contenida en otros portales inmobiliarios como Zonaprop y Argenprop, por nombrar dos de los más populares, que circunscriben la información únicamente al inmueble publicitado.

El mapa diagramado le muestra al usuario una información contextual de los alrededores del inmueble que está observando, lo cual le permite ampliar su conocimiento del barrio y de los servicios cercanos ofrecidos en forma integrada, sin necesidad de acudir a otros portales de información.

Finalmente, el gráfico de estadísticas de inseguridad muestra de manera clara y concisa las principales variables de inseguridad que más preocupan a la ciudadanía. En el mismo, el usuario puede ver los datos correspondientes a todos los barrios, identificándose en color rojo el que corresponde al barrio del inmueble exhibido en la búsqueda.

Presentamos a continuación, una posible navegación realizada por un usuario sobre la búsqueda de un departamento para alquilar en Puerto Madero.

La navegación comienza con un carrusel de propiedades destacadas o que responden a una búsqueda puntual, y se inicia la elección de un departamento de 4 ambientes en alquiler, con la típica información mínima de superficie, cantidad de ambientes y de dormitorios, ofrecida habitualmente por cualquier portal inmobiliario, como se muestra en la Figura 7.



Fig. 7 – Departamento elegido del carrusel inicial

Una vez que se lo elige para profundizar la navegación, aparece más información general y detalles sobre cada ambiente. Una primera diferencia es que las fotos por las que se puede ir pasando presenta texto que indica a qué ambiente corresponde, lo cual supera las típicas fotos sueltas y sin orden de los portales conocidos, en las cuales muchas veces no queda claro que distribución se está presentando, a menos que haya

algún plano. En las Figuras 8 a 11 se pueden ver los distintos ambientes elegidos del carrusel de fotos, cada uno con su descripción. Gracias a ello, en este caso, queda claro cuál es el baño de la suite y cuál es el baño secundario.

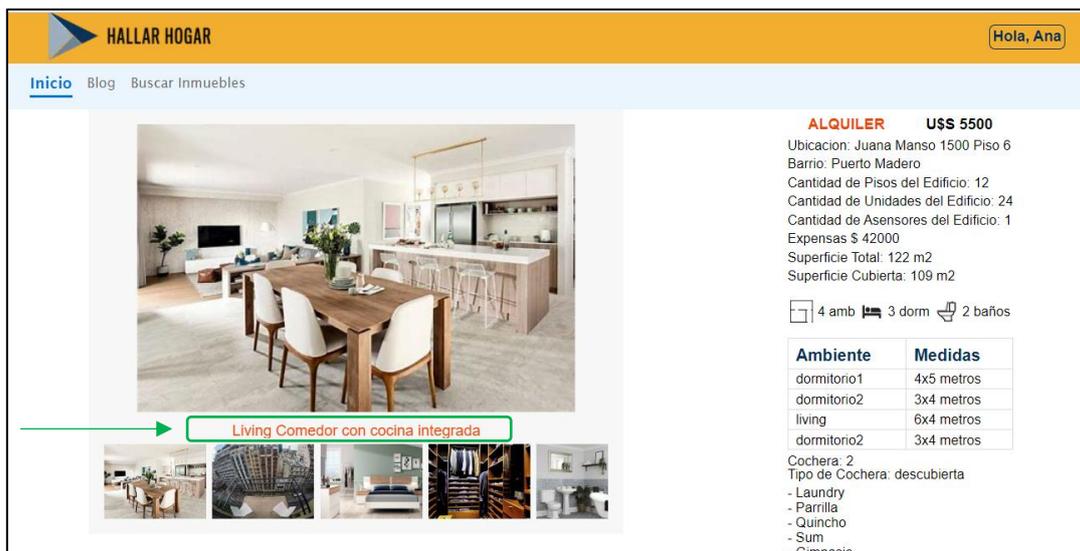


Fig. 8 – Detención en el ambiente de la primera foto del carrusel de ambientes

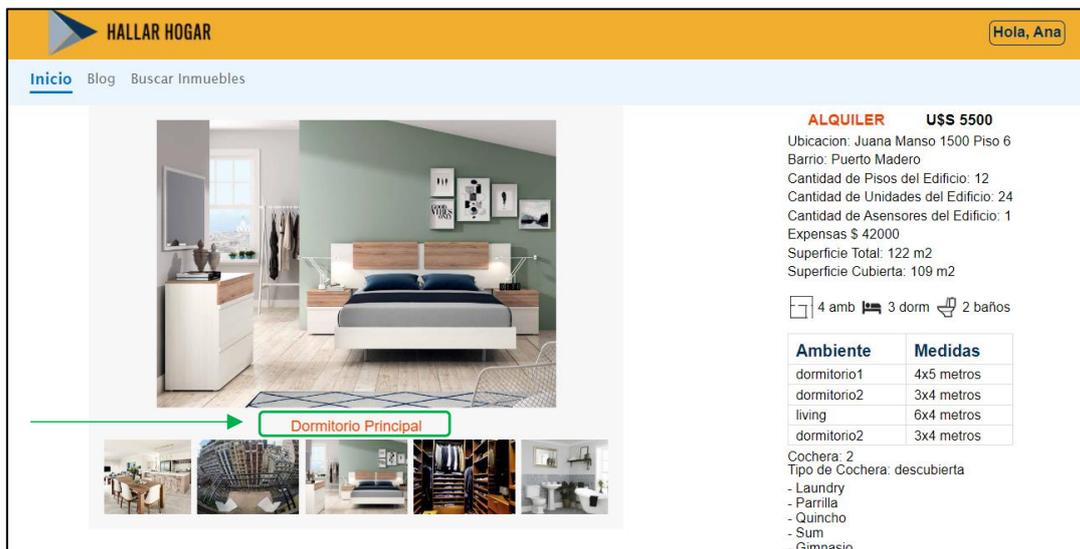


Fig. 9 – Detención en el ambiente de la tercera foto del carrusel de ambientes

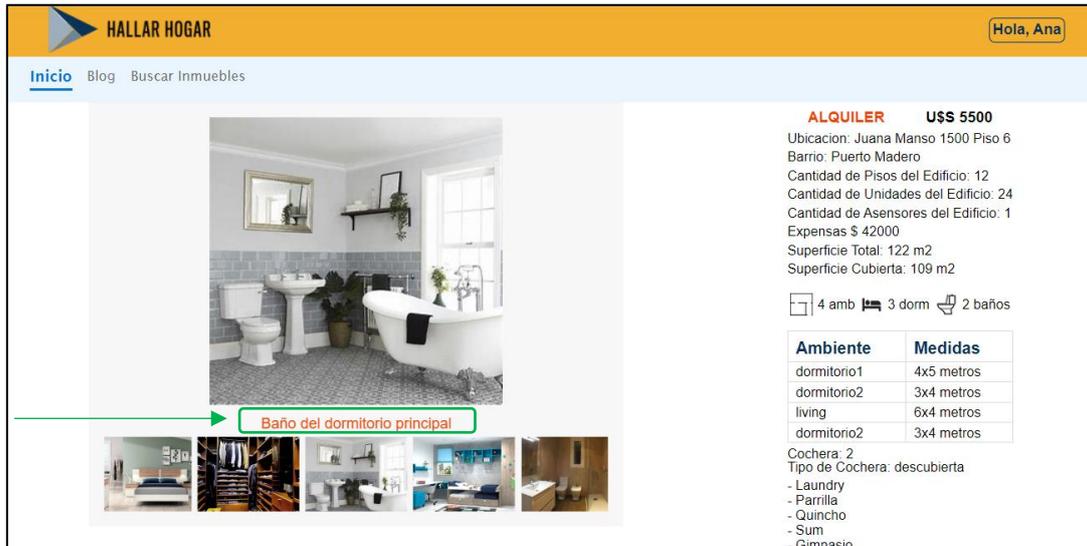


Fig.10 – Detención en el ambiente de la quinta foto del carrusel de ambientes

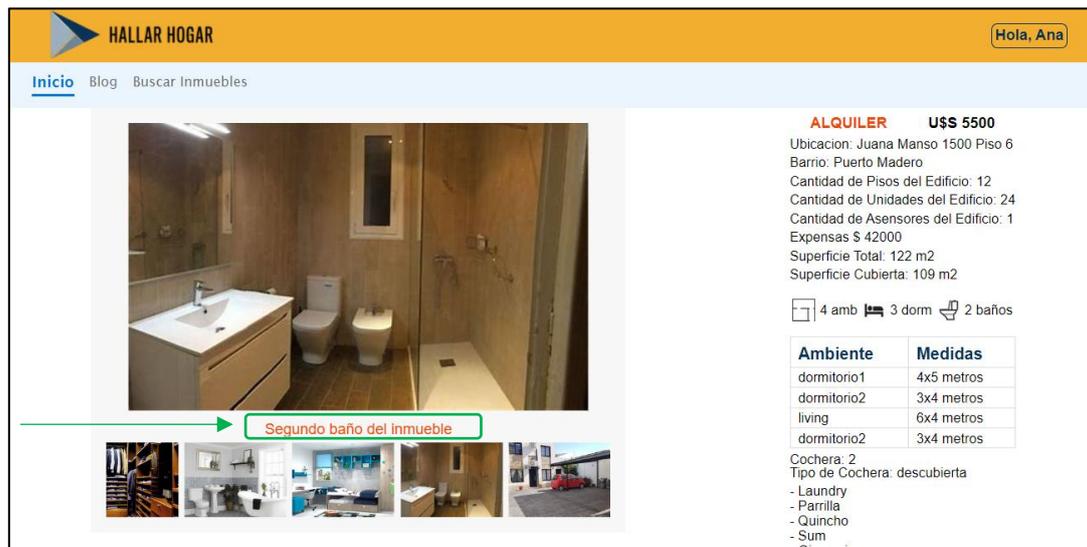


Fig. 11 – Detención en el ambiente de la séptima foto del carrusel de ambientes

Como aporte importante de integración de múltiples fuentes, el usuario puede elegir entre una serie de capas de objetos geolocalizados, los cuales pasan a visualizarse en el mapa que contiene la ubicación del inmueble. Más allá de eso, si se posicionas en cada objeto filtrado, se le muestran los detalles sobre el mismo. En la Figura 12, el usuario elige visualizar en el mapa todos los colegios de la zona, y luego se focaliza en los detalles de dos de ellos. También se puede elegir visualizar varias capas en simultáneo, como se muestra en la Figura 13.

Departamento de 4 ambientes en torre nueva. Tiene un amplio dormitorio en suite con living comedor y cocina integrada. 2 dormitorios con amplias vistas a la Ciudad. Cochera propia fija

- Sum  
- Gimnasio  
Comercializa: Inmobiliaria Madero  
Contacto: madero@madero.com.ar

Mapa de ubicacion del Inmueble y Servicios Cercanos

Departamento de 4 ambientes en torre nueva. Tiene un amplio dormitorio en suite con living comedor y cocina integrada. 2 dormitorios con amplias vistas a la Ciudad. Cochera propia fija

- Sum  
- Gimnasio  
Comercializa: Inmobiliaria Madero  
Contacto: madero@madero.com.ar

Mapa de ubicacion del Inmueble y Servicios Cercanos

Nombre: Jardin de Infantes Integral N° 13/04  
Domicilio: Avda. Calabria y Rosario Vera Peñalosa  
Barrio: Puerto Madero  
Nivel: Nivel Inicial Común  
Tipo de Escuela: Pública

Departamento de 4 ambientes en torre nueva. Tiene un amplio dormitorio en suite con living comedor y cocina integrada. 2 dormitorios con amplias vistas a la Ciudad. Cochera propia fija

- Sum  
- Gimnasio  
Comercializa: Inmobiliaria Madero  
Contacto: madero@madero.com.ar

Mapa de ubicacion del Inmueble y Servicios Cercanos

Nombre: El Elefante Trompita  
Domicilio: Azopardo 1051  
Barrio: San Telmo  
Nivel: Otros Servicios Educativos de la modalidad Común  
Tipo de Escuela: Privada

Fig. 12 – Elección de colegios de la zona del inmueble y focalización en dos de ellos

Departamento de 4 ambientes en torre nueva. Tiene un amplio dormitorio en suite con living comedor y cocina integrada. 2 dormitorios con amplias vistas a la Ciudad. Cochera propia fija

- Sum  
- Gimnasio  
Comercializa: Inmobiliaria Madero  
Contacto: madero@madero.com.ar

Mapa de ubicacion del Inmueble y Servicios Cercanos

Departamento de 4 ambientes en torre nueva. Tiene un amplio dormitorio en suite con living comedor y cocina integrada. 2 dormitorios con amplias vistas a la Ciudad. Cochera propia fija

- Sum  
- Gimnasio  
Comercializa: Inmobiliaria Madero  
Contacto: madero@madero.com.ar

Mapa de ubicacion del Inmueble y Servicios Cercanos

Departamento de 4 ambientes en torre nueva. Tiene un amplio dormitorio en suite con living comedor y cocina integrada. 2 dormitorios con amplias vistas a la Ciudad. Cochera propia fija

- Sum  
- Gimnasio  
Comercializa: Inmobiliaria Madero  
Contacto: madero@madero.com.ar

Mapa de ubicacion del Inmueble y Servicios Cercanos

Fig. 13 – Elección en simultáneo de aspectos culturales y gastronómicos, y focalización en dos de ellos

Finalmente, como otro aporte de integración de información contextual importante para el usuario, se pueden consultar los detalles de estadísticas de homicidios, robos en general y robo de automotores en las zonas en la cual se encuentra el inmueble, y compararlo con otras zonas de la Ciudad de Buenos Aires. Las Figuras 14 a 16 muestran las barras estadísticas de los tres aspectos, remarcando en naranja la correspondiente a la zona del inmueble, para facilitar la observación.

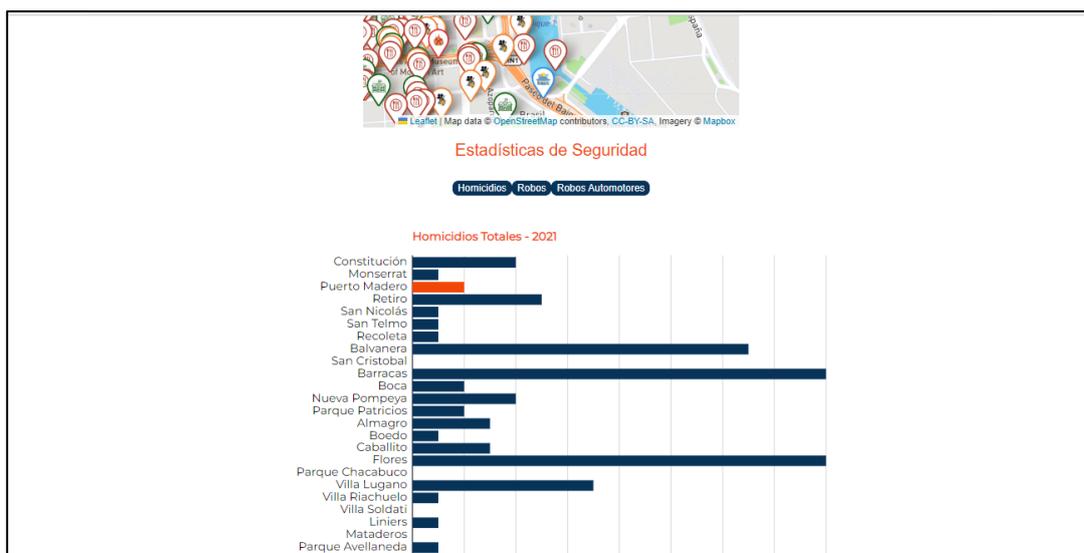


Fig. 14 – Estadísticas de homicidios en la zona del inmueble elegido

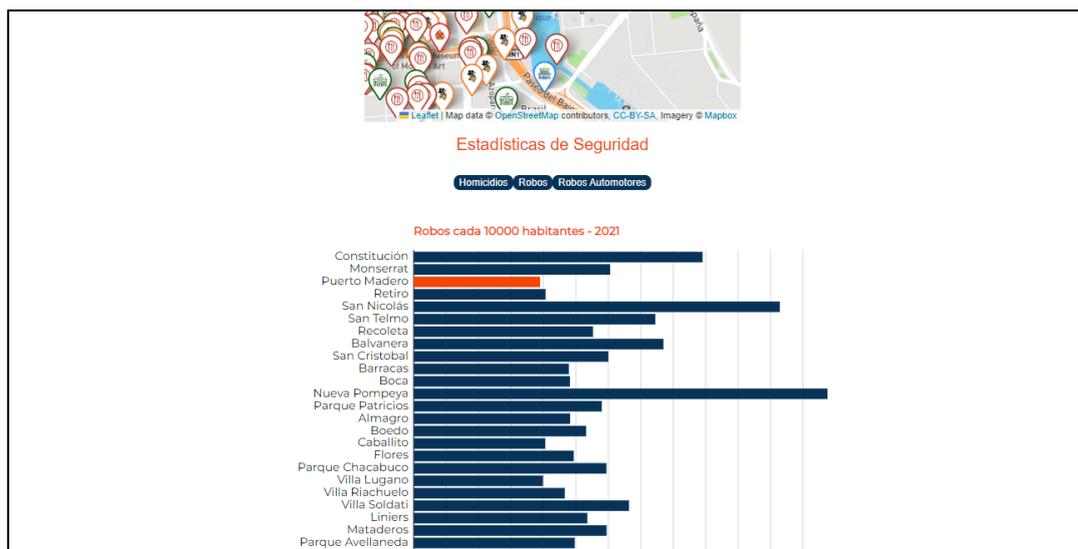


Fig. 15 – Estadísticas de robos en la zona del inmueble elegido

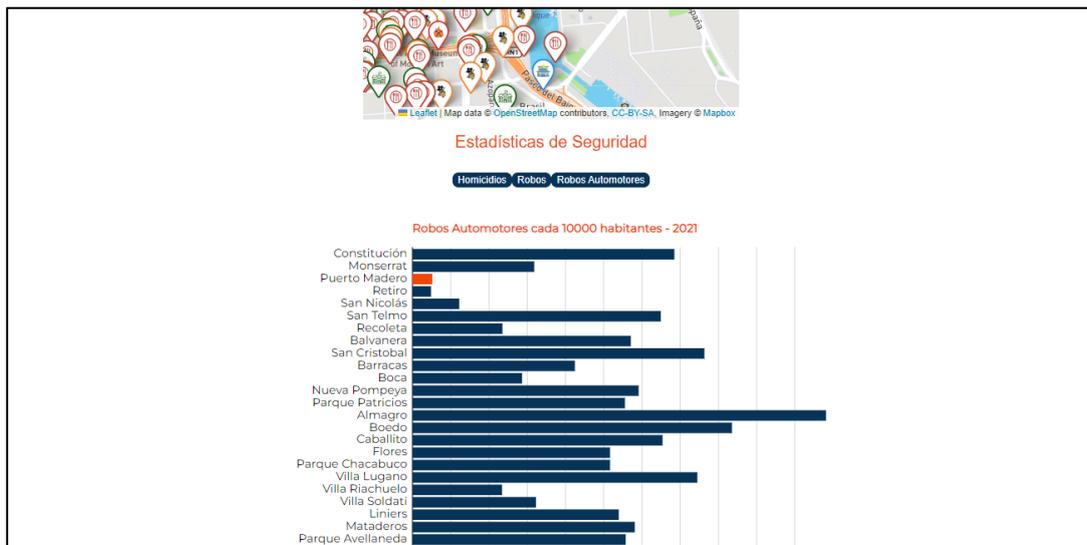


Fig. 16 – Estadísticas de robos de automotores en la zona del inmueble elegido

## 10. Conclusiones y Trabajo a Futuro

Como conclusión, podemos decir que este trabajo ha exhibido una forma diferente de poder mostrar un aviso inmobiliario, logrando incluir información que muestra de manera uniforme todo tipo de aviso, sin importar el tipo de operación y la localización geográfica donde está ubicado el inmueble. Esta información agrega valor al aviso porque permite al usuario tener un mayor conocimiento del entorno donde está ubicado el inmueble que ha despertado su interés, y le permite explorar las diferentes opciones que dicho entorno ofrece para poder tomar una mejor decisión a la hora de alquilar o comprar un inmueble.

Cabe mencionar que, entre los Datasets examinados en la página oficial de GBCA, se ha encontrado la ubicación de ferias y mercados, pero los mismos no incluyen a los supermercados, minimercados, etc., por lo cual no se han incluido estos últimos en la información contextual agregada al entorno del inmueble. La creación de un dataset con dicha información queda como trabajo a futuro, como un trabajo más artesanal que involucra la inspección de las distintas páginas web de supermercados y minimercados.

Como trabajo a futuro, se pueden incluir tiempos de viajes en los cuatro medios de locomoción más utilizados por las personas (moto, auto, bicicleta y transporte público). Para ello la API de Google Routes sería la mejor opción, pero dado que dicha interfaz

permite un limitado número de consultas con su crédito gratuito mensual de USD 200, puede encarecer el mantenimiento y aumentar considerablemente los costos de la aplicación. En su lugar lo que se propone es crear una nueva colección dentro de la base de datos, la que se puede ir poblando de manera gradual y haciendo uso del crédito gratuito de Google. Para ello, se propone seleccionar una N cantidad de puntos geográficos dentro de la Ciudad de Buenos Aires, y conectar cada punto entre sí, calculando de esa manera las distancias entre los diferentes puntos. Debido a que en una ciudad, ir del punto A al punto B no siempre resulta igual en tiempo y distancia que ir del punto B al punto A (manos de las calles, etc.), se deben calcular todos los puntos entre sí, creando así una colección que tendrá una cantidad de documentos equivalentes a  $N^2 - N$ , resultando una cantidad de consultas igual a  $4 * (N^2 - N)$ , debido a que se debe calcular el tiempo en los 4 medios de transporte elegidos. Una vez creada la colección se usarán las coordenadas del inmueble para elegir aquellos documentos que estén mas cerca de dicho punto y se elegirán los tiempos y distancias asociados al punto al cual el usuario quiera ir. De esa manera se puede mostrar otra información que le será útil al usuario para poder elegir un inmueble por sobre otro/s.

El objetivo de este prototipo era mostrar una prueba de concepto de un portal inmobiliario, que recopile información de múltiples fuentes y aplique técnica de ETL para integrarla con la información de los inmuebles ofrecidos, con el fin de ayudar a un usuario a tomar la decisión sobre compra o alquiler de un inmueble a través de información contextual. Por este motivo, en este desarrollo piloto no se hizo foco en la implementación de una pantalla de búsqueda, o en la UI para que un usuario pueda crear un aviso y cargar los datos relativos a un inmueble, por considerarlas comunes a todos los portales existentes.

Sin embargo, una próxima posible extensión tendrá en cuenta estas funcionalidades, habida cuenta de que, una vez completada la colección de tiempos de viajes mencionada anteriormente, se podría incluir como opción de búsqueda que el usuario elija un tiempo máximo de viaje a un destino determinado y posibles medio/s de transporte, para que la aplicación le exhiba todos los inmuebles que se encuentren en ubicaciones con un tiempo de viaje menor al indicado.

Respecto a la información de transporte público, se considera que mostrar en el mapa todas las paradas de colectivos sobresaturaría el mismo, con demasiada

información que se superpondrá entre sí. En una futura versión, lo mejor sería capturar toda esta información, pero exhibir al usuario únicamente la referencia de las líneas de colectivo y estaciones de tren y subte más cercanos al inmueble. Esta información podría ser exhibida en una sección nueva de movilidad junto, a la sección de “tiempo de viaje” ya mencionada anteriormente.

A modo de cierre, otra tecnología que valdrá la pena explorar es la posibilidad de que el usuario pueda visitar el inmueble en el metaverso, para que de esa forma pueda hacerse una mejor idea de las dimensiones y distribución del inmueble. Para ello, se deberá crear un modelo 3D del inmueble e incluir la posibilidad de que el usuario pueda acceder a él mediante un Avatar cuyo tamaño debe guardar relación al tamaño promedio de un humano, siempre en relación al inmueble que el usuario está observando.

## 11. Referencias

- Benli Li. (2021). *Research on Real Estate Information System of the Real Estate Market Based on Big Data Technology*. E3S Web of Conferences 257, 02037.
- Bostock, M., Ogievetsky, V. & Heer. J. (2011). *D3: Data-Driven Documents*. IEEE InfoVis.
- Kaufmann, M. & Meier. A. (2019). *SQL & NoSQL Databases*. Springer Vieweg. ISBN 978-3-658-24548-1.
- Marte, J. (2022). *The Biggest Regrets People have after Buying a Home*. Available online: [https://www.washingtonpost.com/news/get-there/wp/2017/04/20/the-biggest-regrets-people-have-afterbuying-a-home/?noredirect=on&utm\\_term=.79bc863bc842](https://www.washingtonpost.com/news/get-there/wp/2017/04/20/the-biggest-regrets-people-have-afterbuying-a-home/?noredirect=on&utm_term=.79bc863bc842) (accesado el 20/03/2022)
- Ullah, F. & Sepasgozar, S. (2020). *Key Factors Influencing Purchase or Rent Decisions in Smart Real Estate Investments: A System Dynamics Approach Using Online Forum Thread Data*. Sustainability 2020, 12, 4382. <https://doi.org/10.3390/su12114382>.
- Ullah, F., Sepasgozar, S. & Wang, C. (2018). *A Systematic Review of Smart Real Estate Technology: Drivers of, and Barriers to, the Use of Digital Disruptive Technologies and Online Platforms*. Sustainability. 10. 3142. 10.3390/su10093142.
- Yuan, X., Lee, J., Kim, S. & Kim, Y. (2013). *Toward a user-oriented recommendation system for real estate websites*. Information Systems. Volume 38, Issue 2. Pages 231-243. ISSN 0306-4379. <https://doi.org/10.1016/j.is.2012.08.004>.

## 11. Anexo

Todo el Proyecto se encuentra disponible en [GitHub - BJRomPal/TFI-NEXTJS](https://github.com/BJRomPal/TFI-NEXTJS), y la página piloto\_está actualmente hosteada en [Hallar Hogar \(tfi-itba.vercel.app\)](https://tfi-itba.vercel.app).

### Anexo 12.1. Creación y Conexión de la Base de Datos

```
import mongoose from 'mongoose'

const MONGODB_URI = process.env.MONGODB_URI

if (!MONGODB_URI) {
  throw new Error(
    'Please define the MONGODB_URI environment variable inside
    .env.local'
  )
}

let cached = global.mongoose
if (!cached) {
  cached = global.mongoose = { conn: null, promise: null }
}

async function dbConnect() {
  if (cached.conn) {
    return cached.conn
  }

  if (!cached.promise) {
    const opts = {
      bufferCommands: false,
    }

    cached.promise = mongoose.connect(MONGODB_URI,
    opts).then((mongoose) => {
      return mongoose
    })
  }
  cached.conn = await cached.promise
  return cached.conn
}
```

```
export default dbConnect
```

### *Explicación del archivo:*

En MONGODB\_URI es donde se almacena los datos de usuario y contraseña de acceso a la base de datos. Dichos datos están almacenados en un archivo .env el cual no está en el repositorio de GitHub. Los datos de conexión son indicados a Vercel (host de la página web) para que la conexión pueda establecerse.

Si MONGODB\_URI no tiene datos arroja un error.

La función dbConnect() es la que hace la conexión con la base de datos. Dicha función es la que se exporta para ser usada en todo el programa.

## **Anexo 12.2. Código del proceso ETL**

Debido a que el proceso de ETL fue hecho en varios archivos (uno para cada colección) se exhibe el efectuado para Universidades que es el que más trabajo demando.

```
import numpy as np
import pandas as pd
import pymongo

# Lectura del dataset en utf-8 para que lea bien los datos en español
df = pd.read_csv('universidades.csv', delimiter=',', encoding='utf-8')

# Se eliminan los duplicados tomando como punto la dirección_norm
df.drop_duplicates(subset=['direccion_norm'], inplace=True)

# Se individualiza aquellas universidades que solo aparecen una vez y
se obtiene su listado.
df_counts = df.universida.value_counts().to_frame()
df_counts_unicos = df_counts[22:]
listado_unicos = df_counts_unicos.index.to_list()
```

```
# Se cambia la columna de esas universidades donde debe ir la facultad
a Varias Facultades.
for i in range(len(df)):
    if df.iloc[i, 1] in listado_unicos:
        df.iloc[i, 3] = 'Varias Facultades'
    else:
        continue

# Se pasan la longitud y latitud a una lista
longitud = df.long.tolist()
latitud = df.lat.tolist()

# Función que crea dict para los datos geom con el formato requerido
por MongoDB
def create_geom (longitud, latitud):
    lista_coordenadas = []
    for i in range(len(longitud)):
        long_y_lat = [longitud[i], latitud[i]]
        lista_coordenadas.append(long_y_lat)
    listado_geom = []
    for lista in lista_coordenadas:
        dic = {'type': 'Point', 'coordinates': lista}
        listado_geom.append(dic)
    return listado_geom

listado_geom = create_geom(longitud, latitud)

# Se cambia la dirección y Barrio del ITBA para actualizarla.
df.iloc[1, 11] = 'San Martín 202'
df.iloc[1, 15] = 'San Nicolás'

lista_nombre = df.universida.tolist()
lista_facultad = df.unidad_aca.tolist()
lista_regimen = df.regimen.tolist()
lista_barrios = df.barrio.tolist()
lista_direcciones = df.direccion_norm.tolist()
```

```
# Se crea una lista que contendrá dicts, formato requerido para
pymongo para subir a MongoDB
dic_universidades = []
for i in range(len(lista_barrios)):
    universidad = {'nombre': lista_nombre[i], 'facultad':
lista_facultad[i], 'administracion': lista_regimen[i],
'domicilio' : lista_direcciones[i], 'barrio' :
lista_barrios[i], 'location' : listado_geom[i]}
    dic_universidades.append(universidad)

from pymongo import MongoClient
# SE BORRÓ POR SEGURIDAD LA CONEXIÓN A LA BASE DE DATOS

#Subida de datos a MongoDB
db = client['hogarDB']
collection = db['universidades']
collection.insert_many(dic_universidades)
```

### Anexo 12.3. Código de la Implementación

```
//importación de las librerías a utilizar.
import axios from 'axios';
import { useState, useEffect } from 'react';
import { scaleBand, scaleLinear, max } from 'd3';

// función que busca la información de los delitos en la colección. Se
realiza mediante una conexión
// a la api creada
async function fetchData() {
    const configuration = {
        method: "get",
        url: `./api/inmuebles/delito`,
        contentType: 'application/json',
        Accept: 'application/json',
    };
    return await axios(configuration)
}
```

```
// Se establecen las medidas de ancho, alto y margen del gráfico de
barras
const width = 770;
const height = 920;
const margin = {
  top: 80,
  right: 20,
  bottom: 40,
  left: 200
}

// Se establecen las medidas internas donde estarán las barras.
const innerHeight = height - margin.top - margin.bottom;
const innerWidth = width - margin.right - margin.left;

// función de React que crea el gráfico de barras. Recibe como
parámetros el Barrio, el título y el delito.
// Este componente es hijo del componente ButtonBarchart el cual
cambia el gráfico de acuerdo al delito que
// se quiere visualizar.
export default function Barchart({Barrio, titulo, delito}) {
  const [data, setData] = useState(null);

  // función de React que permite crear un efecto cuando se carga el
  componente. En este caso es buscar los datos
  // de la api y asignar colores a las barras. Serán todas azules salvo
  la barra del barrio pasado en el parámetro
  // que será naranja.
  useEffect(() => {
    fetchData().then((result) => {
      let datos = result.data;
      datos.map((d) => {
        if (d.barrio === Barrio) {
          d.color = '#F24607'
        }
        else d.color = "#083359"
      });
      setData(datos)
    })
  }, [])
```

```

// Este pequeño if permite que el componente no arroje un error
mientras está la búsqueda de datos en la MongoDB.
  if(!data) {
    return <pre>Cargando...</pre>
  }

  // Genera la escala vertical donde estarán los barrios, se establece
un padding de 15 para separar las barras entre sí.
  // Debido a se trata de datos categoricos se usa la función de
scaleBand() de D3.
  const yScale = scaleBand()
    .domain(data.map(d => d['barrio']))
    .range([0, innerHeight])
    .padding(.15)

  // Función que genera las barras horizontales de acuerdo a la
cantidad recibida por la base de datos.
  // Debido a que los datos son cuantitativos se usa la función
scaleLinear() de D3.
  const xScale = scaleLinear()
    .domain([0, max(data, d => d[delito])])
    .range([0, innerWidth])

  // Esto es lo que devuelve el componente Barchart. El gráfico de
barras con toda la información recibida por la
// base de datos la cual fue también manipulada por este archivo.
return (
  <svg width={width} height={height}>
    <defs>
      <style type="text/css">@import
url("https://fonts.googleapis.com/css2?family=Montserrat:wght@300;400&
display=swap");</style>
    </defs>
    <text x={margin.left} y={60} style={{fontSize : "0.7em", fill:
"#F24607", fontFamily: "Montserrat", fontWeight:
"bold"}}>{titulo}</text>
    <g transform={`translate(${margin.left}, ${margin.top})`} >
      {xScale.ticks().map(tickValue => (
        <g key={tickValue} transform={`translate(${xScale(tickValue)},
0)`}>
          <line y2={innerHeight} stroke='#d3d3d3' />
          <text style={{textAnchor: "middle", fontFamily:
"Montserrat", fontSize: "0.7em" }} dy=".71em" y={innerHeight +
5}>{tickValue}</text>
        </g>
      )
    </g>
  </svg>
)

```

```

    </g>
  )))
  <line x2={innerWidth} y1={innerHeight} y2={innerHeight}
stroke='black' />
  {yScale.domain().map(tickValue => (
    <text
      key={tickValue}
      style={{textAnchor: "end", fontSize: '.7em', fontFamily:
"Montserrat"}}
      x={-7}
      y={yScale(tickValue) + yScale.bandwidth() / 2 +
4}>{tickValue}</text>
    )))
  <line y2={innerHeight} stroke='black' />
  {data.map(d => <rect
    key={d.barrio}
    x={0}
    y={yScale(d.barrio)}
    width={xScale(d[delito])}
    height={yScale.bandwidth()}
    fill={d.color}
  />)}
  </g>
  <text x={width - 125} y={height} style={{fontSize : "0.8em",
fill: "#F24607", fontFamily: "Montserrat", fontWeight:
"bold"}}>Fuente: GCBA</text>
  </svg>
)
}

```