PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR SEDE SANTO DOMINGO

Escuela de Hábitat, Infraestructura y Creatividad

APLICACIÓN WEB CON MACHINE LEARNING PARA LA GESTIÓN INMOBILIARIA DE LA EMPRESA ESCUDERO DEL CANTÓN SANTO DOMINGO

TRABAJO DE INTEGRACIÓN CURRICULAR

Previo a la obtención del título de Ingeniero en Tecnologías de la Información

Línea de investigación: Tecnologías de la información y la comunicación

Autoría:

Alarcón Morillo Jhery Fernando

Flores Vélez Romario Armando

Dirección:

Ocampo Pazos Willian Javier, Mg.

Santo Domingo – Ecuador
Mayo, 2026

PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR SEDE SANTO DOMINGO

Escuela de Hábitat, Creatividad e Ingeniería

HOJA DE APROBACIÓN

APLICACIÓN WEB CON MACHINE LEARNING PARA LA GESTIÓN INMOBILIARIA DE LA EMPRESA ESCUDERO DEL CANTÓN SANTO DOMINGO

Línea de investigación: Tecnologías de la información y la comunicación

Autoría:

Alarcón Morillo Jhery Fernando

Flores Vélez Romario Armando

Revisado por:

Ocampo Pazos Willian Javier, Mg. _____
DIRECTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR

Córdova Galvez Rodolfo Sirilo, Mg. _____
CALIFICADOR

Ulloa Meneses Luis Javier, Mg. _____
CALIFICADOR

Ulloa Meneses Luis Javier, Mg. _____
COORDINADORDE LA CARRERA DE GRADO

Santo Domingo – Ecuador
Mayo, 2026

DECLARACIÓN DE AUTENTICIDAD Y RESPONSABILIDAD

Nosotros, Alarcón Morillo Jhery Fernando, portador de la cédula de ciudadanía 23004851015, y Flores Vélez Romario Armando, portador de la cédula de ciudadanía 0803669688, declaramos que los resultados obtenidos en la investigación que presentamos como informe final, previo a la obtención del Título de Ingeniero en Tecnologías de la Información son absolutamente originales, auténticos y personales.

En tal virtud, declaramos que el contenido, las conclusiones y los efectos legales y académicos que se desprenden del trabajo propuesto de investigación y luego de la redacción de este documento son y serán de nuestra sola y exclusiva responsabilidad legal y académica.

Igualmente, declaramos que todo resultado académico que se desprenda de esta investigación y que se difunda tendrá como filiación la Pontificia Universidad Católica del Ecuador Sede Santo Domingo, reconociendo en las autorías al director del Trabajo de Integración Curricular y demás profesores que amerita.

Además, declaro que el presente trabajo, producto de las actividades académicas y de investigación, forma parte del capital intelectual de la Pontificia Universidad Católica del Ecuador, Sede Santo Domingo, de acuerdo con lo establecido en el artículo 16, literal j), de la Ley Orgánica de Educación Superior.

En tal razón, autorizo a la Pontificia Universidad Católica del Ecuador, Sede Santo Domingo, para que pueda hacer uso, con fines netamente académicos, del Trabajo de Integración Curricular, ya sea de forma impresa, digital y/o electrónica o por cualquier medio conocido o por conocerse, siendo el presente documento la constancia del consentimiento autorizado; y, para que sea ingresado al Sistema Nacional de Información de la Educación Superior del Ecuador para su conocimiento público, en cumplimiento del artículo 103 de la Ley Orgánica de Educación Superior.

_____	_____
Alarcón Morillo Jhery Fernando	Flores Vélez Romario Armando
C.C. 2300485105	C.C. 0803669688

INFORME DE TRABAJO DE INTEGRACIÓN CURRICULAR ESCRITO

Mikel Ugando Peñate, PhD

Responsable de Investigación Formativa

Pontificia Universidad Católica del Ecuador Sede Santo Domingo

De mi consideración,

Por medio del presente informe en calidad de director del Trabajo de Integración Curricular de Ingenieria en Tecnologías de la Información titulado: APLICACIÓN *WEB* CON *MACHINE LEARNING* PARA LA GESTIÓN INMOBILIARIA DE LA EMPRESA ESCUDERO DEL CANTÓN SANTO DOMINGO, realizado por los estudiantes: Alarcón Morillo Jhery Fernando con cédula de ciudadanía 2300485105 y Flores Vélez Romario Armando con cédula de ciudadanía 0803669688, previo a la obtención del título de Ingeniero en Tecnologías de la Información, informo que el presente Trabajo de Integración Curricular escrito se encuentra finalizado conforme a la guía y al formato de la Sede vigente.

Además, certifico haber verificado la originalidad y autenticidad del trabajo de integración curricular por medio del programa anti plagio Turnitin, en respuesta a la normativa institucional vigente.

Santo Domingo, 26/05/2026.

Atentamente,

Mg. Willian Javier Ocampo Pazos

Profesor Titular Auxiliar I

RESUMEN

Esta investigación tiene como finalidad fortalecer la gestión inmobiliaria de la empresa Escudero del cantón Santo Domingo. El presente trabajo se sustenta en la indagación de múltiples fuentes bibliográficas referentes a la gestión inmobiliaria, aplicaciones *web* y *machine learning*. Además, con la información recolectada mediante las visitas *in situ*, se desarrolló la propuesta de intervención que consiste en una aplicación *web* con *React*, *Tailwind CSS*, *Node.js* con *Express*, *PostgreSQL* y *Prisma ORM* con *machine learning* implementado mediante *Python* con *Flask*. Además, se desarrolló con el marco de trabajo ágil Scrum, cuya planificación consistió en 4 *sprints.* Por otro lado, se optó por un enfoque cuantitativo con diseño preexperimental, obteniendo los datos de 65 clientes para validar la aplicación *web* con *machine learning* para la gestión inmobiliaria. Finalmente, para validar la hipótesis se utilizó regresión logística binaria, tomando en cuenta dos escenarios (0 y 1) mediante la herramienta de *software SPSS*.

Palabras clave: Machine Learning, Gestión Inmobiliaria, Inteligencia artificial, Aplicación Web.

ABSTRACT

This research aims to strengthen the real estate management of the Escudero company in the Santo Domingo canton. This work is based on the investigation of multiple bibliographic sources related to real estate management, web applications and machine learning. Furthermore, with the information collected through the on-site visits, the intervention proposal was developed, which consists of a web application with React, Tailwind CSS, Node.js with Express, PostgreSQL and Prisma ORM with machine learning implemented using Python with Flask. Furthermore, it was developed using the Scrum agile framework, whose planning consisted of 4 sprints. On the other hand, a quantitative approach with a pre-experimental design was chosen, obtaining data from 65 clients to validate the web application with machine learning for real estate management. Finally, to validate the hypothesis, binary logistic regression was used, taking into account two scenarios (0 and 1) using the SPSS software tool.

Keywords: Machine learning, Real estate management, Artificial intelligence, Web Aplication.

ÍNDICE DE CONTENIDOS

1. INTRODUCCIÓN

En los últimos años, el avance acelerado de la tecnología ha cambiado de manera radical el paradigma del acceso a la información, la interacción con determinados servicios y la toma de decisiones. Tanto es así que, la Comisión Económica para América Latina y el Caribe (CEPAL, 2021) señala una evolución hacia una economía digital, caracterizada por la integración de tecnologías modernas en los modelos de producción y consumo de los ámbitos sociales y económicos de la sociedad (p. 11). Por lo que, la digitalización ha transformado varios sectores productivos de la sociedad, optimizando procesos, disminuyendo tiempos y mejorando la experiencia de los usuarios.

De forma paralela, el crecimiento poblacional ha dado lugar a un aumento considerable en la construcción de edificaciones de diversa naturaleza en muchas partes del mundo. De hecho, el Programa de las Naciones Unidas para los Asentamientos Humanos (UN-Habitat, 2022) apunta a un aumento del asentamiento humano en zonas urbanas del 56 % en 2021 al 68 % en 2050 durante los próximos treinta años (p. 4). Esta expansión urbanística generó la necesidad de implementar sistemas que gestionen eficientemente información relacionada con el sector y de esta manera poder lograr una administración más ágil y organizada.

1.1. Antecedentes

En este contexto, Obinna y Udo (2022) implementaron en Nigeria un sistema de gestión inmobiliaria bajo una arquitectura *web* compuesta por interfaz de usuario, base de datos *MySQL*, *backend* en *PHP,* y un módulo analítico que integra análisis de datos y algoritmos de *machine learning*, como regresión y *clustering* dinámico. El sistema fue construido a partir de datos de más de 62 empresas del sector con el objetivo de facilitar la búsqueda de propiedades y mejorar la toma de decisiones. Entre los resultados se evidenció una eficiencia significativa en la búsqueda de bienes inmuebles, disminución del

riesgo de fraude mediante la verificación de agentes inmobiliarios y una mejor toma de decisiones por parte de usuarios, agentes y administradores inmobiliarios (pp. 66, 74-75).

De forma similar, en Colombia, Henríquez y Sánchez (2024) desplegaron un sistema *web* inmobiliario que incorporaba tecnologías como *web scrapping*, procesamiento de lenguaje natural y técnicas de *machine learning* para ofrecer sugerencias personalizadas en base a las preferencias e interacciones del usuario en la plataforma. En la fase de evaluación, los usuarios habían reportado un alto grado de satisfacción con las recomendaciones, destacando una mejora en la búsqueda de propiedades, en la reducción del tiempo de navegación y una mejor toma de decisiones. Además, los autores subrayan también el potencial uso del sistema *web* para la optimización de ventas y aumento de rentabilidad en los agentes inmobiliarios (pp. 3, 10-11).

Por otro lado, en la ciudad de Loja - Ecuador, Armijos y Chamba (2022) abordaron en su artículo el problema de búsqueda de inmuebles en dicha ciudad mediante una aplicación *web*. El aplicativo informático integró en su desarrollo un conjunto de tecnologías *web* modernas, tales como *Express*, *Angular*, *Node* y *MongoDB* usando una arquitectura cliente-servidor multicapa para separar la lógica de presentación, la lógica de negocio y la lógica de datos. La implementación de este *software* tuvo excelentes resultados, dado que redujo significativamente el tiempo de búsqueda de oferta de alquiler de inmuebles por parte de los usuarios, quienes empleaban normalmente entre 1 a 3 días a solamente un tiempo de 10 a 30 minutos hasta un máximo de 1 día de búsqueda (pp. 68, 71, 75).

Finalmente, en el estudio realizado por Mendoza y Campoverde (2025) en la ciudad de Cuenca – Ecuador, se desarrolló un prototipo de *software* para la gestión de la Inmobiliaria Inmocañari, la cual enfrentaba problemas como el deficiente manejo de clientes, propiedades, transacciones y citas, dado que se hacía de forma manual. El prototipo incorporó tecnologías como *Python, PyQt5 y PostgreSQL* cuyo desarrollo fue llevado a cabo bajo la metodología de trabajo ágil *Scrum*. Como resultado de la

implementación, se logró reducir errores operativos, agilizar los procesos administrativos de la inmobiliaria y mejorar la atención y satisfacción de sus usuarios finales (p. 2).

1.2. Planteamiento y delimitación del problema

A través de un análisis de los resultados de los cuatro estudios, se observa que dos de los estudios combinan el desarrollo de aplicaciones *web* con componentes de análisis de datos y *machine learning* para la optimización de recomendaciones en el sector inmobiliario. Por otro lado, el tercero y cuarto estudio converge el desarrollo de una aplicación *web* con la mejora de la gestión inmobiliaria de una empresa local.

Acorde a un informe del Observatorio Nacional de Tecnología y Sociedad (ONTSI, 2023) de España, en el año 2022 entre el 56% y 57.8% de *pymes* y grandes empresas del subsector de actividades inmobiliarias, presentan un nivel básico de digitalización. Adicionalmente, el porcentaje total de empresas del sector de actividades inmobiliarias, administrativas y servicios auxiliares con alguna medida de seguridad *TIC*, se redujo en 5.5 puntos porcentuales ese mismo año, alcanzando solo el 51.5%. Asimismo, apenas el 3.3 % de las empresas del sector utiliza inteligencia artificial, lo que las sitúa por debajo del promedio nacional del 4.9 %, incluso entre *pymes* y grandes compañías, su uso sigue siendo limitado, alcanzando sólo un 10.8 % (pp. 12, 37, 74).

Además, según el Laguyás et. al (2022), el sector inmobiliario en América Latina y el Caribe, presenta un nivel bajo de digitalización reflejado en la venta de una propiedad que puede tomar hasta 360 días debido a la burocracia, la asimetría de información y la falta de plataformas digitales eficientes. También, el 70 % de las empresas del sector, se encuentran en una etapa inicial de desarrollo y el 15 % ha logrado expandirse internacionalmente tornando en una lentitud en la transformación digital. Asimismo, la mayoría de los agentes inmobiliarios aún carece de herramientas digitales para agilizar sus procesos de corretaje, pese a tener amplia experiencia en la industria (pp. 8-9, 13, 27).

Por otro lado, según la Asociación de Corredores de Bienes Raíces del Guayas (ACBIR Guayas) y la Asociación de Corredores de Bienes Raíces del Azuay (ACBIR Azuay, 2021) más del 68 % de los profesionales inmobiliarios enfrenta un riesgo de desaparición o transformación debido a la falta de adaptación tecnológica. Esto significa que, dicha cifra revela una grave brecha digital que afecta directamente la sostenibilidad del sector. Asimismo, a pesar de la disponibilidad de herramientas como plataformas digitales, *CRM* o *chatbots*, su uso aún es limitado. Por ejemplo, más del 60 % de la comunicación inmobiliaria en Ecuador sigue dependiendo de medios tradicionales como *WhatsApp* y correo electrónico, lo que muestra una baja implementación tecnológica en procesos clave (pp. 2, 5).

En consecuencia, ACBIR Guayas y ACBIR Azuay (2021), la falta de tecnificación genera desventajas frente a mercados más avanzados, donde la digitalización es una condición mínima de competitividad. Por esta razón, la capacitación constante y el uso estratégico de datos ya no son opcionales, sino una obligación para los actores que desean mantenerse vigentes (pp. 2, 5). Por lo tanto, se llevó a cabo un análisis de la situación actual de la empresa Escudero, esto permitió identificar diversas causas que afectan la adecuada gestión inmobiliaria dentro de la organización. Entre ellas se encuentra la digitalización ineficiente de los procesos, lo cual genera una alta probabilidad de errores en la gestión de la información.

Asimismo, la ausencia de un análisis automatizado de las preferencias de los clientes provoca insatisfacción en los usuarios y una disminución en la concreción de ventas, debido a que no se pueden ofrecer recomendaciones acordes a sus necesidades. De igual manera, la poca visibilidad en línea de las propiedades disponibles ocasiona un alcance reducido en el mercado y una menor captación de nuevos interesados, limitando las oportunidades comerciales de la empresa. Finalmente, la capacidad actual para dar seguimiento a clientes potenciales conduce al desaprovechamiento de contactos valiosos, reduciendo la posibilidad de concretar futuras transacciones inmobiliarias.

1.3. Preguntas de investigación

1.3.1. Pregunta General

¿Cómo fortalecer la gestión inmobiliaria en la empresa Escudero del cantón Santo Domingo?

1.3.2. Preguntas Específicas

- ¿Cuáles son los procesos de gestión inmobiliaria de la empresa Escudero?
- ¿Cuáles son las herramientas tecnológicas y metodologías que pueden emplearse para llevar a cabo el desarrollo de propuestas de intervención?
- ¿Qué solución tecnológica puede aplicarse para el fortalecimiento de gestión inmobiliaria en la empresa Escudero del cantón Santo Domingo?

1.4. Justificación

La Asamblea Nacional del Ecuador en la Constitución de la República del Ecuador (2008) en su artículo 16 numeral 2, garantiza a las personas, tanto a nivel individual como colectivo, el derecho al acceso a las tecnologías de la información y comunicación (p. 14). Asimismo, el artículo 385 numeral 3, señala que la ciencia y la tecnología en el país deben servir para impulsar la producción nacional, aumentar la productividad y eficiencia, mejorar la calidad de vida y contribuir al buen vivir (p. 185). Paralelamente, el artículo 66 numeral 15, garantiza a las personas el libre ejercicio de actividades económicas, de forma individual o grupal, mientras se respeten los principios de solidaridad, responsabilidad social y ambiental (pp. 32, 33). Como resultado, la constitución respalda el desarrollo de la aplicación *web* con *machine learning* al promover el acceso a la tecnología, mejorar la productividad y el desarrollo de actividades económicas dentro del sector inmobiliario.

El presente trabajo de titulación se alinea con dos Objetivos de Desarrollo Sostenible de las Naciones Unidas (2018), uno de ellos es el objetivo 11 en su meta 3, que propone el aumento de la capacidad para la planificación y gestión sostenibles y participativas de los

asentamientos humanos. Asimismo, se relaciona con la meta 5 del objetivo 9 que resalta la innovación e investigación científica de los sectores industriales de todos los países, especialmente los que se encuentran en vías de desarrollo (pp. 44, 52). De manera similar, se relaciona con los objetivos estratégicos 13 y 14 del eje C de la Agenda Digital eLAC2026 impulsada por la Comisión Económica para América Latina y el Caribe (CEPAL, 2022), que promueven el uso de tecnologías digitales emergentes como la inteligencia artificial para fomentar la productividad, la innovación, así como la transformación digital de pequeñas y medianas empresas (p. 5). Como derivación de lo expuesto, los objetivos y estrategias mencionadas, respaldan el uso de una aplicación *web* al fomentar la innovación, la transformación digital y la gestión sostenible dentro del sector inmobiliario.

Adicionalmente, la Asamblea Nacional de Ecuador (2023) en la Ley Orgánica para la Transformación Digital y Audiovisual (p. 8) menciona que todo servicio digital debe estar diseñado bajo principios de interoperabilidad, eficiencia y gobierno de datos. Asimismo, el proyecto presentado por Asamblea Nacional del Ecuador (2024), sobre la Ley Orgánica de Regulación y Promoción de la Inteligencia Artificial en Ecuador (2024, Arts. 2 y 3, p. 2) propone regular el uso del *machine learning* protegiendo los derechos como privacidad y equidad. En consecuencia, se permite integrar estas tecnologías de forma responsable en el ámbito inmobiliario. Por otra parte, en los artículos 1, 9, 17 y 44 de la Ley Orgánica de Ordenamiento Territorial, Uso y Gestión de Suelo del Ecuador (2016), se define el marco jurídico para la gestión y planificación del uso del suelo, incluyendo la clasificación de propiedades, sus usos permitidos y el manejo del inventario urbano y rural (pp. 7, 15, 17, 25). De este modo, las normativas mencionadas respaldan el uso de una aplicación *web* con *machine learning* al promover la eficiencia, la interoperabilidad y el uso responsable de los datos dentro de la gestión inmobiliaria.

Asimismo, la Presidencia Constitucional de la República del Ecuador (2023), en el Reglamento General a la Ley Orgánica para la Transformación Digital y Audiovisual establece reglamentos para plataformas digitales seguras y funcionales. Además, el Art. 11

respalda el uso de *machine learning* para análisis inteligentes de los datos. En este marco, contribuyen a procesos digitales eficientes y automatizados (p. 6-7), en cuanto a la gestión inmobiliaria, esta se sustenta en los artículos 10, 13, 14 y 21 del Reglamento General a la Ley Orgánica de Ordenamiento Territorial, Uso y Gestión del Suelo (Presidencia Constitucional de la República del Ecuador, 2019), que definen los criterios técnicos y normativos de planificación, clasificación y regulación del uso del suelo urbano y rural (pp. 9-11, 16). En consecuencia, los reglamentos mencionados justifican el uso de una aplicación *web* para fortalecer los procesos de gestión inmobiliaria.

1.5. Objetivos de investigación

1.5.1. Objetivo general

Implementar una aplicación *web* con *machine learning* para la gestión inmobiliaria de la empresa Escudero del cantón Santo Domingo durante el período 2025.

1.5.2. Objetivos específicos

- Identificar los procesos de gestión inmobiliaria en la empresa Escudero para la obtención de los requerimientos.
- Determinar las herramientas tecnológicas y metodologías adecuadas para el desarrollo de la aplicación *web*.
- Desarrollar una aplicación *web* con *machine learning* para la gestión inmobiliaria en la empresa Escudero del cantón Santo Domingo.

2. REVISIÓN DE LA LITERATURA

2.1. Fundamentos teóricos

Para los fundamentos teóricos, se elaboraron diagramas que definen la estructura conceptual del proyecto. Según Hernández et al. (2014), este proceso comienza con la elaboración de un índice preliminar de carácter global que se va ajustando de forma progresiva hasta alcanzar un nivel detallado, lo que permite organizar de manera coherente los apartados y vincular las referencias bibliográficas dentro del esquema general (p. 78). La figura 1 describe la variable aplicación *web* como plataforma de interacción para usuarios finales y administradores, la figura 2 se dedica a la representación de los fundamentos del *machine learning* aplicado a la predicción de tendencias inmobiliarias, y la figura 3 detalla los componentes funcionales de la gestión inmobiliaria, incluyendo la administración de propiedades, clientes y agentes.

Figura 1. Variable independiente de aplicación web.

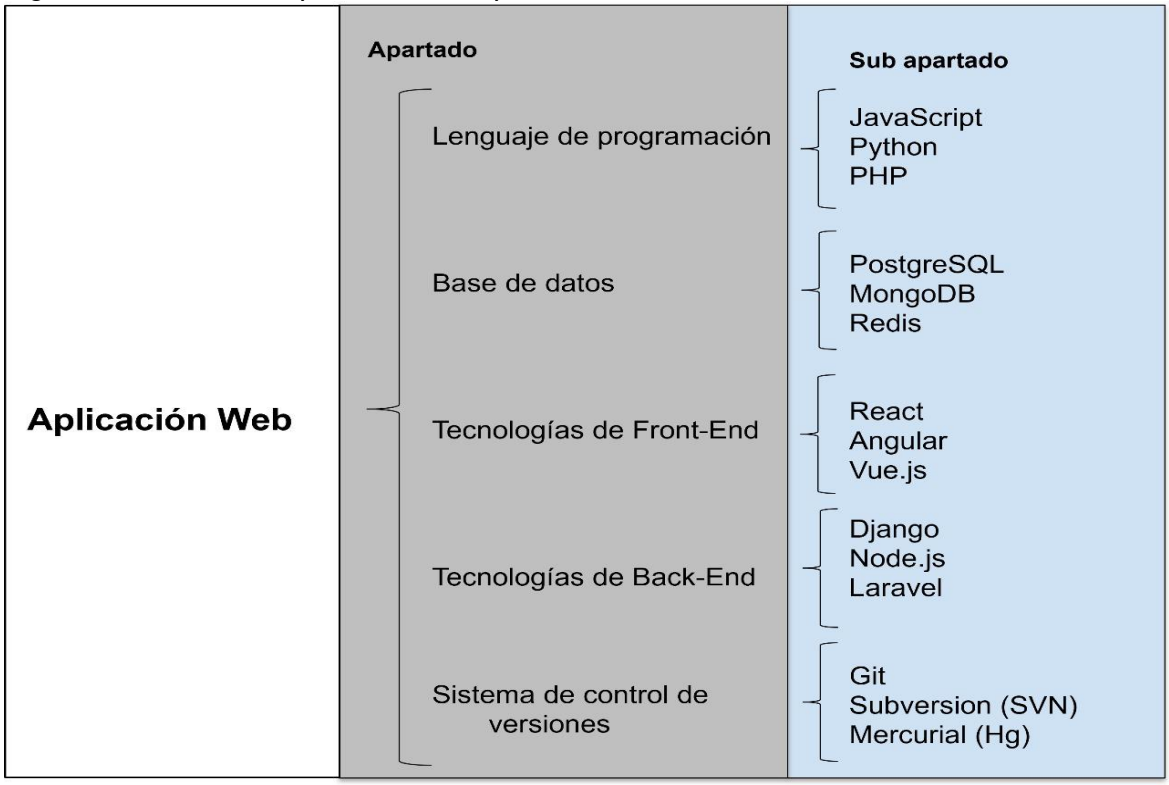


Figura 2. Variable independiente de machine learning

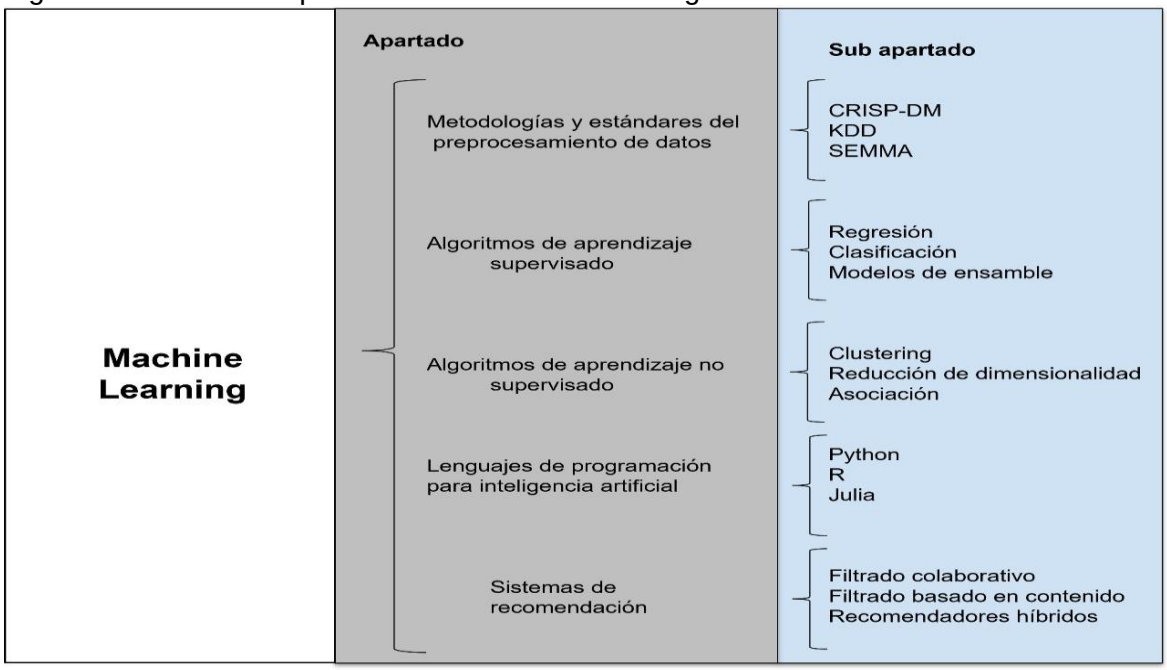
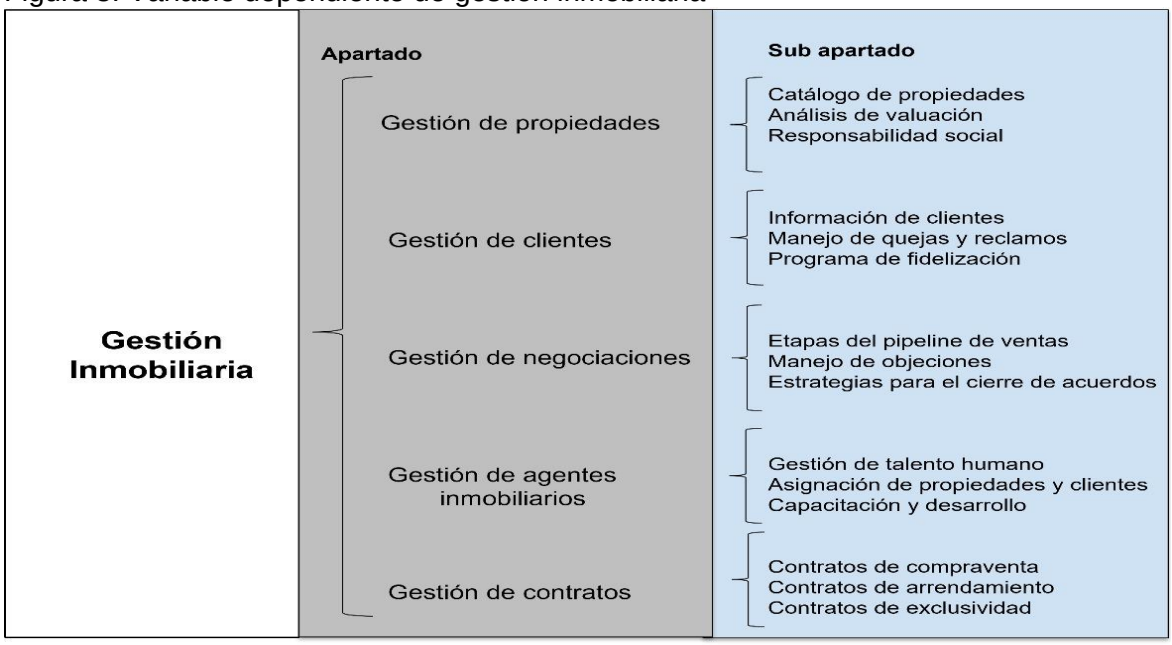


Figura 3. Variable dependiente de gestión inmobiliaria



2.1.1. Aplicación web

Según Ferrer (2012), una aplicación *web* puede entenderse como un sistema basado en clientes livianos (*light clients*), ya que estos no requieren ejecutar procesos complejos para su funcionamiento. Asimismo, desde el enfoque arquitectónico, se distinguen dos componentes principales: el cliente, donde el usuario final accede a la aplicación a través de un navegador (como *Internet Explorer*, *Google Chrome* o *Firefox*), y

el servidor, que es el encargado de almacenar los datos y gestionar la lógica y las reglas de la aplicación (p. 18).

2.1.1.1. Lenguaje de programación

Conforme a lo señalado por Martín et al. (2021), el lenguaje de programación se basa en un modelo computacional que refleja fielmente la arquitectura del ordenador. La ventaja es que los programas se traducen a código máquina eficientemente, ya que se relaciona con el *hardware* directamente, por otro lado, la expresividad del lenguaje se limita, de manera que encaje en la estructura de máquina. donde se dividen en programación funcional que se basa en la evaluación de expresiones construidas por medio de llamadas a funciones y programación lógica que usa un paradigma para formar partículas lógicas simbólicas, denominado cálculo predictivo, donde se empresa en otras ramas como la base de datos (p. 30-31).

2.1.1.1.1. JavaScript

De acuerdo con lo mencionado por Fernández (2023), *JavaScript* es un lenguaje de programación interpretado, basado en estándares *ECMAScript*, Se caracteriza por ser un lenguaje orientado a eventos y basado en prototipos, dinámicos y no demasiado tipado. Esta particularidad facilita la creación de estructuras flexibles y la reutilización de funcionalidades mediante la delegación de comportamientos. Además, es un lenguaje dinámico, ya que permite modificar el contenido de los objetos en tiempo de ejecución, lo cual brinda una gran ventaja a las aplicaciones desarrolladas (pp. 1, 8).

2.1.1.1.2. Java

Según Sznajdleder (2024), describe a Java como un lenguaje que obedece al paradigma orientado a objetos. Es compatible con varios sistemas operativos, lo que lo hace versátil. Esta orientación permite reutilizar código, la modularidad y la mantenibilidad de los proyectos de *software*. *Java* permite ejecutar aplicaciones de manera independiente de la plataforma donde se desarrolla. Asimismo, proporciona un entorno robusto para

implementar los patrones de diseño y otros elementos esenciales que favorecen el desarrollo de aplicaciones (p. 1).

2.1.1.1.3. PHP

Aguirre (2021), menciona que *PHP* es un lenguaje de programación, orientado a objetos, siendo un paradigma que se basa en declarar clases que trabajan como moldes para todos los objetos, el cual se programaran métodos y declaraciones de atributos. Una de las ventajas que ofrece es utilizar clases ya programadas en su núcleo, este mismo se ubica en el momento en el entorno de trabajo, lo cual permite realizar llamadas directas a dichas clases y aprovecharlas en funciones comunes, como la creación de fechas (p. 9).

2.1.1.2. Base de datos

Según Ramos y Ramos (2007), mencionan que una base de datos es un conjunto de información relacionadas entre sí, organizado y estructurado, con información referente a algo. Además, de poder ser utilizada en cosas sencillas como mantener una agenda de registros telefónicos, o tan complicadas como llevar la gestión de una empresa u organización (p. 1).

2.1.1.2.1. PostgreSQL

De acuerdo con Zea et al. (2017), comenta que *PostgreSQL* es un sistema de gestión de bases de datos objeto-relacional, estando bajo licencia *BSD* y con código fuente libre. Útil para modelos cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Una ventaja es que si falla un proceso no afectará a los demás y continuará funcionando con normalidad. También, puede ser definido como un sistema de base de datos relacionales, que permite la manipulación de acuerdo a las reglas de álgebra relacional. Los datos se almacenan en tablas de columnas y renglones, con el uso de llaves primarias, estas tablas podrán relacionarse entre otras (pp. 12-13).

2.1.1.2.2. MongoDB

Conforme a lo señalado por Bradshaw et al. (2020), MongoDB es una base de datos orientada a documentos, lo que significa que almacena datos en documentos en vez de

tablas. Un documento es un conjunto de pares clave-valor, siendo la unidad básica de datos en MongoDB. Los documentos son parecidos a los objetos *JSON*, y se almacenan en una colección. Una colección es un conjunto de documentos que tienen la misma estructura, estos documentos se almacenan en formato *binary JSON* (*BSON*) (p. 3).

2.1.1.2.3. Redis

Sarasa (2019), menciona que *Redis* es una base de datos que almacena toda la información en la memoria, pero con persistencia en el disco duro, lo cual facilita una alta velocidad de escritura y lectura. Sin embargo, esta configuración impone una limitante en cuanto a los conjuntos de datos que puede almacenar, siendo que no puede ser más grande que la memoria. Para gestionar eso se utilizan memorias virtuales que mantiene todas las claves, pero escribe valores menos utilizadas en el disco. Así mismo, la información almacenada en la memoria es que puede ser más sencillo manipular los datos complejos mucho en comparación con los datos en el disco. Utiliza una arquitectura maestro-esclavo para la implementación de clúster, y se realiza una réplica asíncrona de los datos (pp. 11-12).

2.1.1.3. Tecnologías de *Front-End*

Según Paredes (2025), define al *frontend* como la parte de una aplicación o sitio web que los usuarios visualizan y con la que realiza interacciones directas. Además, los lenguajes fundamentales para el desarrollo de *frontend* son *HTML*, *CSS* y *JavaScript,* donde *HTML* es el esqueleto de cualquier página *web*. Su estructura está compuesta por encabezado, párrafos, enlaces, imágenes y otros elementos que componen el contenido. En cambio, *CSS*, es el lenguaje que se encarga de la apariencia visual, puede controlar los colores, tipos de fuentes, tamaño de los elementos y su disposición en la pantalla (pp. 7-8).

2.1.1.3.1. React

De acuerdo con lo mencionado por Aguirre (2021a), *React* es una biblioteca de JavaScript desarrollada y respaldada por Facebook. Está diseñada para crear aplicaciones de una solo página (*SPA*), cuyas siglas significan *Single Page Application*. Las *SPA* se

caracterizan por su interfaz de una sola página, lo que significa que el navegador no recarga la página completa mientras el usuario interactúa con ella, En cambio, varios componentes funcionan juntos o de forma independiente dentro de esa única página, que no se actualiza, excepto en situaciones específicas. Además, *React* simplifica la creación de aplicaciones complejas de forma sencilla en pocos pasos, sin volverse complicado para el programador. *React* permite dividir una aplicación en pequeños módulos, así se podrá trabajar cómodamente, permitiendo modificaciones específicas en secciones concretas y la reutilización de componentes en diferentes partes de la aplicación (pp. 15, 17, 18).

2.1.1.3.2. Angular

Puciarelli (2020), comenta que Angular es un popular *framework* de *JavaScript* de código abierto desarrollado y mantenido por *Google*, está diseñado para crear aplicaciones de una sola página (SPA). Esto significa que la carga inicial de la página va seguida de actualizaciones dinámicas que se producen sin necesidad de recargar la página por completo. Para usar Angular de forma eficaz, los desarrolladores deben tener un conocimiento sólido de *HTML, CSS, JavaScript* y los principios de la programación orientada a objetos. El lenguaje de programación principal que utiliza *Angular* es *TypeScript* (p. 8).

2.1.1.3.3. Vue.js

Según Ayllapan (2025), comenta que *Vue.js* es un *framework* progresivo de *JavaScript* utilizado para construir interfaces de usuario. Se destaca por ser ligero, flexible y fácil de aprender, lo que lo convierte en una excelente opción tanto para principiantes como para proyectos a gran escala. Además, se define como progresivo porque puede empezar integrando poco a poco en proyectos existentes, y después escalar hacia soluciones más complejas (p. 3).

2.1.1.4. Tecnologías de *Back-End*

Según Sebesta (2015), las tecnologías *backend*, o del lado del servidor, son aquellas que corren en un servidor web y son imprescindibles para el procesamiento de

datos, la lógica de negocio, la interacción con bases de datos y la generación dinámica de contenido *web* que luego se refleja en el navegador del cliente. A diferencia del *software* del lado del cliente, el código *backend* no es visible para el usuario final en su navegador (pp. 28-29, 359).

Además, Sebesta (2015) indica que, entre las principales tecnologías de *back-end* se encuentran los servidores *web* que proporcionan documentos a los navegadores de los clientes cuando estos lo solicitan a través del protocolo *HTTP o HTTPS*; los lenguajes de Scripting del lado del servidor siendo algunos de los más populares *PHP, Java, Ruby,* entre otros, *frameworks* y sistemas de bases de datos para la gestión de la información (pp. 7,8).

2.1.1.4.1. Django

De acuerdo con Elman y Lavin (2015), *Django* es un *framework web* de *Python* que fue desarrollado en el año 2005 para que los reporteros pudieran crear contenido *web* rápidamente. *Django* sigue unas filosofías de baterías incluidas lo cual hace alusión a que tiene varias utilidades para desarrollar aplicaciones *web*. Además, cuenta con un diseño desacoplado ya que permite a los desarrolladores elegir y reemplazar componentes para construir las aplicaciones ligeras que desee. De igual forma, se utiliza para construir aplicaciones *web* de una sola página integrando *REST APIs, WebSockets y frameworks MVC* del lado del cliente como Backbone.js (pp. vii, 13).

2.1.1.4.2. Node.js

Según Cantelon et al. (2014), Node.js es un entorno de ejecución para *JavaScript* orientado al desarrollo del lado del servidor, construido sobre el motor V8 de *Google Chrome*. Su principal característica es la ejecución asincrónica y basada en eventos, lo cual permite gestionar múltiples tareas de manera concurrente sin necesidad de múltiples hilos. A diferencia de otros entornos que paralizan procesos mientras esperan a que se ejecuten operaciones de entrada/salida, *Node.js* no lo hace, consiguiendo incluso un menor consumo de recursos (pp. 4-5).

2.1.1.4.3. Laravel

Stauffer (2017) define a Laravel como un *framework* de desarrollo web del lenguaje de programación *PHP*, *open source* y gratuito, que sigue el patrón arquitectónico MVC (Modelo-Vista-Controlador). Fue creado para facilitar tareas comunes en el desarrollo backend, como enrutamiento, autenticación, migraciones de base de datos y gestión de sesiones. Además, integra herramientas modernas como *Eloquent ORM* para la gestión de bases de datos, *Blade* para plantillas y *Artisan* como interfaz de línea de comandos (pp. 8, 29, 31, 250, 339).

2.1.1.5. Sistema de control de versiones

En palabras de Beckman et al. (2020), definen al sistema de control de versiones como una herramienta que permite registrar y gestionar los cambios realizados en un archivo o conjunto de archivos a través del tiempo, permitiendo dar seguimiento a dichas modificaciones y recuperar versiones determinadas de los mismos en caso de revisión o recuperación de errores, convirtiéndose así en un elemento indispensable para el flujo de trabajo individual y colaborativo (pp. 1, 3).

2.1.1.5.1. Git

Ponuthorai y Loeliger (2021) conceptualizan a *Git* como un sistema de control de versiones distribuido, gratuito y open *source*. Su objetivo principal es gestionar y rastrear cambios en archivos y proyectos en el transcurso del tiempo. A diferencia de otros sistemas de control de versiones, *Git* almacena los cambios de los archivos como instantáneas completas, en lugar de registrar únicamente las diferencias que se presentan entre las versiones. Además, *Git* no depende de un servidor central, lo que permite a los desarrolladores trabajar de manera óptima sin conexión para posteriormente sincronizar sus cambios no registrados (pp. 5, 18-19).

2.1.1.5.2. Subversion (SVN)

Pilato et al. (2008) definen a Subversion como un sistema de control de versiones centralizado, *open source* y gratuito que se encarga de gestionar archivos y directorios, así

como el registro de los cambios en los mismos a través del tiempo, lo que permite al desarrollador volver a una versión previa en caso de problemas o errores. Entre sus principales características está que cuenta con un modelo centralizado en el que su historial de versiones se almacena en un único servidor y sus usuarios trabajan con copias locales cuyos cambios se pueden sincronizar luego al repositorio central o actualizar desde el mismo la versión local con los cambios hechos por otros (pp. xi, xiv).

2.1.1.5.3. Mercurial (Hg)

O'Sullivan (2009) describe a Mercurial como un sistema de control de versiones distribuido que permite gestionar el historial de cambios en archivos y proyectos de forma eficaz. En contraste con los sistemas de versiones centralizados, los usuarios pueden hacer una copia completa del repositorio y trabajar de manera local, permitiéndose hacer operaciones como *commit, merge o branching*. Entre sus principales características destacan su rapidez, escalabilidad y baja curva de aprendizaje, lo que lo convierte en una herramienta accesible incluso para desarrolladores principiantes. Aunque su popularidad ha disminuido frente a herramientas como *Git*, sigue siendo una alternativa atractiva para la gestión de versiones en proyectos colaborativos (pp. 4, 6).

2.1.2. Machine Learning

Según Emiro et al. (2023), menciona que *machine learning* es un subconjunto de la inteligencia artificial que consiste en la construcción de modelos matemáticos basados en muestras conocidas como "datos de entrenamiento". Se utiliza para realizar predicciones o toma de decisiones sin que el algoritmo programe de manera explícita una tarea o ejecución de una función. Además de percibirse como un grupo de herramientas que busca el aprendizaje computacional mediante datos sin haber ingresado por codificación. Así mismo, es un diseño y estudio que usa la experiencia para mostrar decisiones futuras (p. 30).

2.1.2.1. Metodología y estándares de preprocesamiento de datos

De acuerdo con el consorcio Chapman et al. (2000), define como una agrupación sistemática de procesos que permite la transformación de datos desordenados e

incompletos en un conjunto de datos final adecuado para el análisis, verificando las etapas de selección, limpieza, construcción, integración y formateo. Lo cual se asegura que la información procesada cumpla con los requisitos de calidad, consistencia y relevancia necesaria para la aplicación de técnicas de minería de datos y *machine learning*. Siendo esto parte esencial del ciclo de la vida analítico para los modelos predictivos o descriptivos se aplique sobre datos confiables y estructurado según el objetivo planteado. Además, esta metodología se apoya no en teorías abstractas, sino en experiencia práctica (pp. 24-26).

2.1.2.1.1. CRISP-DM

Más aún, según Bemthuis et al. (2025), menciona que *CRISP-DM* (*Cross-Indrustry Standard Process for Data Mining*) es un modelo de proceso estándar público y no privado, diseñado para guiar proyectos de minería de datos y ciencia de datos mediante un enfoque de seis fases: comprensión del negocio, comprensión de los datos, preparación de los datos, modelado, evaluación y despliegue. Donde este método permite obtener iteraciones y retroalimentar entre las etapas, lo cual beneficia a la alineación con los objetivos empresariales, resultando en una producción de alta calidad (pp. 1, 6).

2.1.2.1.2. KDD

El descubrimiento de conocimientos de base de datos (KDD, en sus siglas en inglés, *Knowledge Discovery in Databases*), se define, según Fayyad, et al. (1996), como un proceso computacional emergente y de múltiples pasos. Donde su intención principal es apoyar a los usuarios en la extracción de información útil a partir de volúmenes de datos en constante crecimiento. Esto busca abordar el problema fundamental de mapear datos de nivel bajo (aquellos datos difíciles de entender directamente). Este campo se relaciona con una variedad de áreas como el *machine learning*, estadísticas y base de datos (pp. 40-41).

2.1.2.1.3. SEMMA

SEMMA es una metodología desarrollada por SAS Institute Inc. (2018) para asistir en los procesos de minería de datos, siendo un componente clave y estructurante dentro de su plataforma *SAS Enterprise Miner*. Esta metodología se articula en cinco fases

fundamentales de la minería de datos: Sample selecciona un subconjunto representativo de datos. Además, Explore realiza un análisis inicial para entender la estructura de los datos, *Modify* prepara los datos para el análisis, Model construye los modelos predictivos o descriptivos, *Assess* valora el rendimiento y la validez de los modelos (pp. 6-7).

2.1.2.2. Algoritmo de aprendizaje supervisado

Bottini (2025), menciona que el algoritmo de aprendizaje supervisado requiere la intervención humana en el proceso, pero esto no se refiere a un humano en el proceso, sino que los datos fueron tratados anteriormente para etiquetar datos, resultados y brindar una información catalogada y limpia al modelo (p. 24).

2.1.2.2.1. Regresión

Según Velasco (2024), describe al algoritmo como un subalgoritmo del algoritmo de aprendizaje supervisado, donde se divide en regresión lineal que es un modelo clásico que busca establecer una relación lineal entre una o múltiples variables independientes y una variable dependiente, y regresión logística, utilizado para clasificación binaria de problemas, estimando probabilidades de que una instancia pertenezca a una clase particular (p. 18).

2.1.2.2.2. Clasificación

De acuerdo con lo mencionado por Bobadilla (2020), el algoritmo de clasificación es un modelo que intenta predecir con exactitud a qué conjunto pertenece cada muestra de datos obtenida. Este tipo de estructura trabaja con información categórica en muchos escenarios, pero no predice valores continuos (como lo haría el algoritmo de regresión), sino predice un grupo selecto de datos existentes (p. 46).

2.1.2.2.3. Modelos de ensamble

Raschka (2024), menciona que es un método que combina predicciones diferentes modelos para optimizar el rendimiento general de la predicción. Entre las técnicas más conocidas se encuentra el *bagging*, que entrena diferentes modelos independientes sobre varios subconjuntos de datos. Sin embargo, la desventaja de utilizar una gran variedad de modelos es el aumento de la carga informática y el consumo de recursos computacionales.

lo que dificulta su implementación en entornos con restricciones de procesamiento o memoria (pp. 9-10).

2.1.2.3. Algoritmo de aprendizaje no supervisado

Conforme a lo señalado por Ahmad (2024), el aprendizaje no supervisado consiste en detectar y utilizar los patrones en un conjunto de datos para estructurarlo y comprenderlo con mayor claridad. Este tipo de algoritmo identifica relaciones, agrupaciones o tendencias sin necesidad de disponer de etiquetas. Entre los métodos más representativos se encuentran el *clustering*, que agrupa datos similares en categorías o clústeres, y la reducción de dimensionalidad, que busca simplificar la representación de la información preservando la mayor cantidad de variabilidad posible, contribuyendo a una comprensión y representación más profunda de los datos (pp. 38-39).

2.1.2.3.1. Clustering

Según Han et al. (2012) describe al *clustering* como un algoritmo de agrupación que, desempeña un papel fundamental en la identificación de patrones y estructuras dentro de vastos conjuntos de datos. El principal objetivo es agrupar estados o experiencias similares, lo que permite generalizar los conocimientos y políticas de las relaciones. Al aprovechar estas características puede mejorar su eficiencia, escalabilidad y capacidad de toma de decisiones (p. 85).

2.1.2.3.2. Reducción de dimensionalidad

Según lo indicado por Bobadilla (2020), la reducción de dimensionalidad permite disminuir el tamaño de los datos, lo que consiste en crear un nuevo grupo de dimensiones partiendo de las características ya existentes, generando un conjunto reducido que conserve la información más importante de los datos originales. El número de agrupaciones creadas y seleccionadas debe ser necesariamente menor que el número original, lo que ayuda a disminuir la complejidad computacional y mejorar el rendimiento de los algoritmos (p. 141).

2.1.2.3.3. Asociación

De acuerdo con lo mencionado por Emiro et al. (2023), explica que las reglas de asociación relacional (RAR) son conceptos de minería y análisis de datos que se basa en las reglas de asociación tradicionales. Estas normas permiten identificar patrones, relaciones o dependencias que no se limitan al mismo conjunto, sino que pueden añadirse relaciones más complejas entre entidades y atributos de diferentes tablas o estructura de datos. Ofreciendo una representación más rica y flexible de los datos, facilitando la obtención de la información relevante que puede ser utilizada en diferentes ámbitos (p. 30).

2.1.2.4. Lenguaje de programación para inteligencia artificial

Ramírez y Ramírez (2023), menciona que las herramientas dependen principalmente de la elección de los lenguajes de programación, como *C/C++, Java, C#, Python, Matlab, Ruby, R, Julia, Go, etc. Donde C y C* + + son los lenguajes más populares, porque pueden ejecutarse mucho más rápido que las otras herramientas. Además, de albergar muchas librerías orientada a la *IA*, como *OpenCV* para visión artificial, en cambio *java* es otro lenguaje popular, ya que todas las aplicaciones telefónicas están diseñadas en *Java* (p. 18).

2.1.2.4.1. Python

Según Ramírez y Ramírez (2023), describe a *Python* como el lenguaje de programación más utilizado en la actualidad. Se diferencia de otros compiladores por ser un programa interactivo y fácil de usar. El código de *Python* se ejecuta línea por línea, entonces, aunque haya un error en el código se ejecuta las líneas antes del error y se detiene solo hasta llegar a dicho error. Tiene una gran variedad de librerías para el desarrollo de IA como son *Numpy, Pandas, Matplotlib y NLTK*. También puede acoplar *frameworks* como son: *Scikit-Learn, Keras, Tensor Flow de Google, PyTorch de Facebook y Paddle de Baidu* (pp. 18-21).

2.1.2.4.2. R

Según lo expuesto por Heesen (2024), R es un lenguaje de programación distribuido bajo la Licencia Pública General de GNU (GPL), este entorno ha adquirido gran importancia en los últimos años al considerarse como una herramienta útil para el análisis estadístico, minería de datos y aprendizaje automático. Su popularidad proviene de su extensa colección de bibliotecas especializadas para implementar algoritmos avanzados, procesar gran cantidad de información y representar los resultados de forma gráfica. Esto hace a R un referente en el ámbito de la inteligencia artificial y el machine *learning* permitiendo disponer de un ecosistema robusto y flexible (p. 57).

2.1.2.4.3. Julia

Conforme a lo señalado por Dash (2024), Julia es un lenguaje de programación de alto nivel y rendimiento, diseñado específicamente para la computación técnica. Su sintaxis es familiar para los usuarios por otros entornos, lo que facilita su adaptabilidad, así mismo se especializa en la manipulación de datos y al análisis numérico, ya que añade una serie de datos integrados que permiten trabajar de manera más eficiente grandes volúmenes de datos. Entre las estructuras se destaca los arreglos (*array*) que ordena los valores del mismo tipo, pudiendo transformarse en unidimensional (vectores) o multidimensional (matrices) (p. 104).

2.1.2.5. Sistema de recomendación

De acuerdo a lo mencionado por Smarandache y Leyva (2022), los sistemas de recomendación son un conjunto de técnicas y procesos computacionales donde el propósito principal es predecir la preferencia o el interés que un usuario puede manifestar por un ítem determinado. Esto facilita la personalización de datos y mejora la experiencia del usuario, donde clasifica en tres grandes enfoques, filtrado colaborativo, que usa los valores y comportamiento de otros usuarios con preferencia similares, el filtrado basado en contenido y métodos híbridos (p. 80).

2.1.2.5.1. Filtrado colaborativo

Según Huyen (2023), describe al filtrado colaborativo como una red neuronal simple que aprende a identificar patrones y preferencias en los usuarios, este modelo es utilizado para realizar predicciones personalizadas en función del historial del usuario. En este enfoque, el sistema recopila información sobre las valoraciones, clasificación o acción realizada por un conjunto de usuarios y posteriormente reentrenar a la red neuronal para que reconozca similitudes en los patrones (p. 191).

2.1.2.5.2. Filtrado basado en contenido

Según lo expuesto por Ortega (2022), es un espacio para asignar características a los elementos y dependiendo de eso realiza recomendaciones. Esto significa que la información se deriva de los elementos caracterizados y no dependa de ningún dato entrante de los usuarios, solo que se utilizara en las recomendaciones y no en el tiempo computacional. El propósito de este sistema es básicamente monitorizar los hábitos de consumo del usuario y aprenderá de sus preferencias en forma de palabras claves (p. 120).

2.1.2.5.3. Recomendaciones híbridas

Según Ortega (2022), menciona que la recomendación híbrida es la combinación de los tres tipos de filtrados anteriormente vistos. Además, de combinar la inteligencia artificial, lo cual realiza recomendaciones basándose en el hábito de observación y búsqueda de usuarios, así mismo se ofrece películas con características que el usuario a valorado positivamente (p. 214).

2.1.3. Gestión Inmobiliaria

La gestión inmobiliaria para Pfnür (2011) es el conjunto de actividades destinadas a la planificación, dirección y control de los recursos y procesos relacionados con las propiedades inmobiliarias, con el objetivo de optimizar su uso y maximizar su valor. Además, abarca tres enfoques principales: el primero, centrado en el usuario, en el que los inmuebles se consideran herramientas de soporte para la operación y eficiencia de las

empresas. El segundo, desde la perspectiva del propietario o inversor, en el que se consideran a las propiedades como activos financieros y se prioriza su rentabilidad. Y el tercero, orientado a la gestión operativa, que integra el ciclo de vida del inmueble compuesto por la planificación, construcción, explotación y comercialización del mismo (pp. 8-9).

2.1.3.1. Gestión de propiedades

Haupt et al. (2017) aborda la gestión de propiedades como la administración, operación y mantenimiento de un bien inmueble por parte de un profesional del sector que trabaja en representación del propietario. El tipo de inmuebles tratados consiste en un abanico amplio de propiedades como edificios de apartamentos, viviendas unifamiliares de alquiler, espacios de oficinas, propiedades comerciales y propiedades industriales (pp. 1, 14).

2.1.3.1.1. Catálogo de propiedades

Hamilton (2006) señala que, para lograr la comercialización efectiva de un bien inmueble, el agente inmobiliario a cargo de su venta o alquiler debe conocer a detalle cada uno de sus aspectos, como la ubicación exacta, el tipo de propiedad (residencial, comercial, terreno). En el área de terreno y de construcción, la cantidad de habitaciones y baños, estacionamientos, el estado de conservación, entre otras características (pp. 162-164, 347-348).

2.1.3.1.2. Análisis de valuación

Según Appraisal Institute (2013), el análisis de valuación en el ámbito inmobiliario responde a un proceso sistemático que una persona denominada “tasador” utiliza para determinar un valor creíble y fundamentado sobre un bien inmueble. Este proceso suele basarse en la recopilación e interpretación de los datos, la cual, esta última, puede ser mediante el análisis del mercado, análisis del uso más alto y mejor de la propiedad o de análisis estadístico. Finalmente, integra la aplicación de los tres enfoques de valuación, como lo son el enfoque de comparación de ventas, el enfoque de capitalización de ingresos

y el enfoque de costos, para culminar en una opinión de valor bien respaldada (pp. 36, 44, 275).

2.1.3.1.3. Responsabilidad social

Glavinich (2008) argumenta que la responsabilidad social en la construcción inmobiliaria implica una actitud proactiva hacia el medio ambiente y el negocio, lo cual resulta en quien lo practica un buen ciudadano corporativo. Por lo tanto, el sector inmobiliario debe edificar con responsabilidad hacia el medio ambiente, asegurándose de que el proceso de construcción sea eficiente, use recursos renovables y minimice la utilización de recursos y la generación de residuos dentro de unos límites acordados (pp. 2-3).

2.1.3.2. Gestión de clientes

Peppers y Rogers (2017) indican que la gestión de clientes es una estrategia clave para las empresas, ya que pone su concentración más en el cliente que en el propio producto, con el fin de aumentar el valor que la empresa obtiene a largo plazo. Esta táctica busca atraer, mantener y hacer crecer a los clientes más valiosos, entendiendo que no todos los clientes deben ser tratados por igual. Asimismo, para aplicar lo mencionado, las empresas deben seguir el modelo que se compone de la identificación de cada cliente; diferenciación según su valor y necesidades, interacción con ellos para conocerlos mejor, y el ofrecimiento de ofertas o comunicaciones personalizadas (pp. 5, 40, 79-80).

2.1.3.2.1. Información de clientes

De acuerdo con Dintsis (2020), el procesamiento de la información del cliente consiste en su recopilación y análisis detallados con el fin de construir relaciones duraderas, así como adoptar estrategias comerciales informadas entre la marca y el cliente. Los tipos de información de cliente que se recopilan comúnmente son los que incluyen datos personales y demográficos como los nombres, teléfonos y otros datos de contacto, información personal general, género y edad. Se destacan también datos que abarcan la interacción del cliente con la empresa, los cuales son capturados a través de diferentes

canales de comunicación como el chat en vivo, las redes sociales, correos electrónicos, el sitio web de la empresa, así como a través de plataformas tecnológicas (pp. 20, 27-28).

2.1.3.2.2. Manejo de quejas y reclamos

Fitzsimmons y Fitzsimmons (2011) definen el manejo de quejas y reclamos o recuperación del servicio como el proceso que se lleva a cabo para convertir un cliente insatisfecho en un cliente leal con la empresa. Asimismo, se debe considerar la queja de los clientes como un “regalo”, ya que ellos están invirtiendo su tiempo en informar a la empresa sobre algún problema existente, de los cuales un 60% se quedarían si su problema se resuelve, y un 95% si se resuelve rápidamente (pp. 136-137, 140).

Por otra parte, los autores sugieren una política de manejo de quejas efectiva integrada con la capacitación de todo el personal que mantiene contacto con el cliente, la cual consiste en dar la bienvenida a las quejas y animar a los clientes a presentarlas, resolverlas de manera rápida, recompensar a los empleados por su buena gestión, así como tener registros de las quejas para el mejoramiento continuo del negocio (p. 137).

2.1.3.2.3. Programa de fidelización

Kumar y Reinartz (2018) plantean que un programa de fidelización es el proceso de generar recompensas para los clientes que tienen compromiso o fidelidad con la empresa. Asimismo, es una herramienta fundamental que se emplea para la identificación, recompensación y retención de los clientes rentables mediante el cumplimiento de objetivos de la empresa tales como la construcción de una fidelidad genuina, ganancias por eficiencia y ganancias por efectividad. No obstante, se debe procurar una alineación de valor a través del equilibrio entre el costo de servir a un cliente con el valor que aporte a la empresa, asegurando así un servicio óptimo para los clientes más valiosos (pp. 182, 184).

2.1.3.3. Gestión de negociaciones

Fisher et al. (1998) consideran a la gestión de negociaciones como la aplicación efectiva del método de negociación de principios, que consiste en una forma de negociar equilibrada con los valores personales, obteniendo lo que se quiere lograr de manera

amigable y eficiente. De igual forma, esta gestión implica transformar las diferencias entre las partes en una búsqueda colaborativa de soluciones, en lugar de un enfrentamiento de posturas (p. 28).

Por consiguiente, los autores mencionan que para lograr este objetivo el método se basa en cuatro puntos: distinguir entre las personas del problema. Además, enfocarse en los intereses reales de todas las partes, en lugar de sus posiciones iniciales, explorar varias alternativas para el beneficio mutuo antes de tomar decisiones, e insistir en el uso de criterios justos y objetivos, en lugar de ceder ante presiones (pp. 28-30).

2.1.3.3.1. Etapas del pipeline de ventas

Acorde a Kazanjy (2020), describe al pipeline de ventas como la representación de las etapas por las que atraviesa un proceso de negociación desde el primer contacto hasta el cierre del acuerdo. También, cabe mencionar que dicho proceso implica manejar hasta cientos de oportunidades de venta concurrentes, cuyo principal objetivo es la maximización de ventas.

El ciclo de un pipeline de ventas comienza con el contacto y compromiso, cuyo objetivo es conseguir una cita comercial con el prospecto, sigue la presentación del producto en la que se induce al cliente a que la solución propuesta se ajusta a su problema. Finalmente, se encuentra la fase de negociación y cierre, donde el esfuerzo se centra en lograr que el prospecto firme el contrato (pp. 165, 214).

2.1.3.3.2. Gestión de objeciones y toma de decisiones

Blount y Hunter (2018) resaltan que, el manejo de objeciones es un proceso que busca entender y superar las verdaderas preocupaciones de los clientes sobre una negociación con el objetivo de concretar el posible acuerdo. Además, se enfatiza que una objeción no representa necesariamente un rechazo directo, sino que se manifiesta como señal de confusión, aversión al riesgo, sobrecarga cognitiva o miedo al cambio por parte del cliente. De igual forma, dada la naturaleza emocional de las objeciones, se las debe abordar

en primer lugar desde la emoción antes que la lógica, la cual, esta última, puede levantar discusiones y resistencia por parte del cliente (pp. 31, 51).

2.1.3.3.3. Estrategias para cierres de acuerdos

Acorde a Tracy (2007), existen varias técnicas de cierre en negociaciones, como el cierre ascendente, que consiste en formular una serie de preguntas de manera cuidadosa, de tal manera que se obtiene una respuesta afirmativa por cada una de ellas para intensificar la emoción del cliente potencial. De igual manera, menciona el cierre de invitación que atiende a la invitación directa de compra al cliente luego de la presentación del producto o servicio. Asimismo, nombra al cierre Benjamín Franklin, cuyo funcionamiento se asemeja a cómo el ser humano toma decisiones importantes; el vendedor sugiere al cliente listar las razones a favor y en contra de la decisión, lo cual conlleva a una decisión de compra lógica (pp. 163, 173, 225).

2.1.3.4. Gestión de agentes inmobiliarios

Acorde a Johnson (1989), la gestión de agentes inmobiliarios se puede entender como un proceso vital y estratégico para el crecimiento y supervivencia de una empresa inmobiliaria, siendo su enfoque principal el de desarrollar y mantener un personal de alto rendimiento. Por lo que, en esencia, se trata de la capacidad de una empresa de reclutar, capacitar, compensar y retener un equipo de gestión y ventas firme y productivo, ya que la calidad del personal es el factor más importante del éxito a largo plazo de la organización (p. 15).

2.1.3.4.1. Gestión de talento humano

Según Dessler y Varela (2011), la gestión de talento humano se refiere al proceso de identificar, reclutar, contratar y desarrollar empleados con un alto potencial, por lo que es tarea de las organizaciones buscar garantizar una oferta adecuada para los puestos de trabajo actuales, y futuros que surgen a partir de la estrategia de negocios. Adicionalmente, implica planificar y administrar la carrera del personal para satisfacer las necesidades de la empresa, así como las aspiraciones laborales de los colaboradores (p. 91, 96).

2.1.3.4.2. Asignación de propiedades y clientes

Ahumada (2023) menciona que, la generación de leads (clientes potenciales) se puede producir a través de plataformas tecnológicas en donde se configuran preguntas para los clientes y, una vez respondidas, los leads se añaden a un *CRM* para ser gestionados por un agente individual. De manera similar, se señala la importancia de que el *CRM* notifique de manera inmediata a los agentes cuando un cliente muestra interés por una propiedad determinada al marcarla como favorita o cuando la comparte, para que el agente pueda contactarlo. Además, se menciona un proceso para "cortar" a los agentes de los nuevos leads si no están al día con la gestión de estos, lo que implica una asignación activa y una supervisión del rendimiento (pp. 43-44, 52).

2.1.3.4.3. Capacitación y desarrollo

McAdams et al. (2004) consideran que la capacitación en el sector inmobiliario va más allá de la educación para obtener la licencia de bienes raíces, ya que la segunda no siempre lleva el conocimiento a la práctica. Estos programas de capacitación suelen estar dirigidos a agentes inmobiliarios principiantes y experimentados, gerentes y personal administrativo, donde se abarcan temas como estrategias de venta, asuntos de política de la empresa, asuntos legales y reducción de riesgos, temas motivacionales y de desarrollo personal, entre otros. Asimismo, la capacitación puede dictarse a partir de programas elaborados enteramente por la propia empresa o por parte de terceros. De igual forma, se sugiere refuerzo de lo aprendido a través del coaching individual del gerente, sesiones de seguimiento o en reuniones de ventas (pp. 329-331, 333, 340).

2.1.3.5. Gestión de contratos

Según Hinkel (2008), se lo entiende como el proceso integral compuesto por la preparación, revisión y administración continua de acuerdos legales en el ámbito inmobiliario. Por tanto, se recomienda una revisión minuciosa de los contratos para asegurar que la información necesaria de las partes esté incorporada, así como el manejo de un lenguaje claro y preciso, para evitar posibles ambigüedades. De igual forma, se

destaca el papel que desempeñan los asistentes legales, quienes están a cargo de la preparación o revisión de los documentos bajo la supervisión de un abogado (p. 73).

2.1.3.5.1. Contratos de compraventa

Mettling y Cusic (2019) definen al contrato de compraventa como un documento legal en el cual participan el comprador (la persona que acepta adquirir la propiedad) y un vendedor (el que define los términos de venta del bien inmueble). Por lo que, este tipo de acuerdo se caracteriza por ser de carácter bilateral, ya que ambas partes se comprometen a cumplir sus respectivas obligaciones. Además, para que sea ejecutable, el contrato debe ser escrito, se debe identificar a las partes principales, la descripción clara y precisa del inmueble materia de la transacción, el valor de compra y también la firma de los actores principales (pp. 194-195).

2.1.3.5.2. Contrato de arrendamiento

Acorde a lo mencionado por Hinkel (2008), un contrato de arrendamiento es un documento legal mediante el cual una persona (arrendador) transfiere el uso de un bien inmueble de tipo comercial o residencial a otra persona (arrendatario), a cambio de un monto preestablecido conocido como renta. De igual forma, se detallan también aspectos como la identificación clara de las partes, una descripción precisa del bien arrendado, la duración exacta del contrato y el pago de la renta, que puede ser un porcentaje de las ventas o una cantidad fija (pp. 408-409, 411).

2.1.3.5.3. Contrato de exclusividad

Mettling y Cusic (2019) señalan que un contrato de exclusividad es un acuerdo formal entre el propietario de un bien inmueble y un corredor de bienes raíces, en el que el primero otorga el derecho al agente o agencia inmobiliaria de ofrecer la propiedad que está en venta o arriendo en un tiempo previamente determinado. Asimismo, se establece que el dueño de la propiedad se debe prestar a que el inmueble pueda ser mostrado, proporcione información precisa, así como que cumpla con las leyes estatales y federales. Por otra

parte, el agente se compromete a encontrar un comprador por el valor acordado que sea aceptable para el propietario, además de asistir en la respectiva negociación (pp. 155, 157).

2.2. Predicción científica

H0: La aplicación *web* con *Machine Learning* no incide en el fortalecimiento de la gestión inmobiliaria en la empresa Escudero del cantón Santo Domingo.

H1: La aplicación *web* con *Machine Learning* incide en el fortalecimiento de la gestión inmobiliaria en la empresa Escudero del cantón Santo Domingo.

3. METODOLOGÍA

3.1. Enfoque, alcance, diseño y tipo de investigación

Para este proyecto, se ha aplicado un método cuantitativo porque permite recopilar y analizar datos digitales para medir el impacto del sistema. Según Render et al. (2006), "la investigación cuantitativa está relacionada con la definición de un problema, desarrollar un modelo, recopilar datos, desarrollar una solución, probar esa solución, analizar los resultados e implementar los resultados" (p. 3). Esto facilita la evaluación objetiva satisfacción y eficacia de la aplicación *web* inmobiliaria con recomendación automática.

Por otra parte, el alcance determina el nivel de profundidad con el que se analiza el objeto de estudio. En este sentido, Hernández y Mendoza (2018) proponen la clasificación tradicional en exploratorio, descriptivo, correlacional y explicativo (p. 106). No obstante, Domínguez (2015) plantea una clasificación más amplia al incorporar los niveles predictivo y aplicativo; el primero se orienta a la estimación probabilística de eventos futuros mediante técnicas de análisis predictivo, mientras que el segundo se enfoca en la resolución de problemas concretos, interviniendo directamente en el comportamiento de la variable dependiente (p. 53). En este sentido, se optó por un alcance aplicativo, ya que la investigación se orienta a la resolución de problemas concretos mediante el desarrollo de una aplicación *web* para la gestión inmobiliaria.

De acuerdo con Hernández y Mendoza (2018), los diseños preexperimentales, también denominados preexperimentos, se caracterizan por presentar un bajo nivel de control, ya que generalmente se aplican a un solo grupo sin contar con comparaciones rigurosas. Por ello, sus resultados deben interpretarse con cautela y suelen utilizarse principalmente con fines exploratorios. En contraste, los diseños experimentales o experimentos puros ofrecen un mayor grado de control y validez interna, debido a que incorporan grupos de comparación y garantizan la equivalencia entre dos o más grupos. Además, de contemplar la manipulación de variables independientes y la medición de sus

efectos sobre variables dependientes, frecuentemente mediante el uso de prepruebas y pospruebas (p. 163). Asimismo, Hernández et al. (2010), menciona que en los diseños cuasiexperimentales no se realiza una asignación aleatoria de los sujetos ni un emparejamiento entre ellos, sino que se trabaja con grupos previamente conformados, conocidos como grupos intactos. Cuya formación es independiente del desarrollo del experimento, en este caso se utilizó dos o más grupos (p. 148). En este sentido, se escogió un diseño preexperimental, debido a que la investigación se desarrolla con un solo grupo y un nivel limitado de control, permitiendo evaluar de manera inicial la aplicación propuesta.

Finalmente, la investigación se clasifica como aplicada, ya que su propósito es generar soluciones prácticas dentro de un contexto organizacional. Según Baudean (2015), este tipo de investigación permite a las organizaciones obtener información relevante para la toma de decisiones, especialmente ante situaciones que afectan el desarrollo normal de sus procesos, las cuales se conciben como problemas de acción que no siempre se presentan de manera clara o completamente definida (p. 6). Asimismo, se desarrolló una investigación de campo, en donde, Hernández et al. (2010) señala que este enfoque implica el acercamiento directo al entorno donde se lleva a cabo el estudio, permitiendo identificar fuentes de información, comprender el contexto y verificar la viabilidad de la investigación (p. 8). En este marco, los datos fueron recolectados directamente de los clientes tras su interacción con el sistema, mediante la aplicación de encuestas estructuradas con escala Likert en dos momentos: antes y después de la implementación, específicamente en la empresa Escudero.

3.2. Unidades de análisis

De acuerdo con Grove y Gray (2019), “el muestreo no probabilístico por conveniencia son las personas que se incluyen en el estudio, donde estaban en el lugar correcto y en el momento adecuado, adicionalmente estos individuos son los que están disponibles hasta alcanzar el tamaño del muestreo deseado” (p. 479). Para el presente

trabajo de investigación, se consideró aplicar un método de muestreo no probabilístico, denominado muestreo por conveniencia, dado que se seleccionó un subconjunto accesible de la población disponible, en función de la accesibilidad y disponibilidad de los participantes. Este método fue seleccionado considerando los límites temporales establecidos para el desarrollo del proyecto y el acceso a los usuarios activos que interactuaron directamente con la plataforma durante la fase de implementación experimental.

Como resultado, y considerando el total de la población y las limitaciones operativas del estudio, se optó por una muestra de 65 clientes distribuidas proporcionalmente entre los distintos tipos (compradores y arrendatarios) a la cual se aplicó una encuesta estructurada. Como se detalla en la tabla 1, la población de estudio está conformada por los distintos actores que interactúan directamente con la plataforma de gestión inmobiliaria implementada en este proyecto.

Tabla 1. Población involucrada en la gestión inmobiliaria de Escudero Inmobiliaria

	Categorías	**Subgrupo**	**Cantidad Mensual**
1	Clientes	Compradores	250
		Arrendatarios	150
Total			400

Fuente: Datos proporcionados por el Gerente de Escudero Inmobiliaria

3.3. Técnicas e instrumentos de investigación

En el desarrollo del presente estudio, se empleó la encuesta estructurada como técnica de recolección de datos. En este proyecto, la encuesta fue utilizada como el instrumento central para evaluar las percepciones de los usuarios sobre la funcionalidad, facilidad de uso, rapidez, satisfacción y efectividad del sistema de gestión inmobiliaria. La recolección de datos se llevó a cabo mediante la plataforma digital *Google Forms*, lo que permitió administrar los cuestionarios de manera eficiente, almacenar automáticamente las respuestas y reducir posibles errores en el registro de la información.

Para aplicar la encuesta, se diseñó un cuestionario estructurado, el cual sirvió como instrumento principal de recolección de datos. El cuestionario fue elaborado siguiendo la

escala tipo *Likert* de 5 niveles, lo que permitió cuantificar el grado de familiaridad, satisfacción, rapidez, seguridad, facilidad y efectividad percibidas por los distintos grupos de usuarios. Finalmente, es importante destacar que esta técnica permitió obtener información cuantitativa, fortaleciendo el análisis integral de los datos recolectados.

3.4. Técnicas de análisis de datos

En función de los objetivos establecidos, se adoptó un análisis estadístico descriptivo e inferencial para examinar la información recolectada. La estadística descriptiva permitió obtener una visión general del comportamiento de las variables evaluadas, identificando tendencias, frecuencias y niveles de satisfacción de los usuarios. Según Salafranca *et al.* (2005), “la estadística descriptiva sintetiza los datos recopilados para facilitar su interpretación inicial y detectar patrones relevantes” (p. 224). Posteriormente, se aplicaron técnicas de análisis como la distribución de frecuencias relativas (porcentajes) y el análisis descriptivo comparativo a través de gráficos de barras agrupadas, utilizando la herramienta *Microsoft Excel*.

Una vez organizados y validados, los datos fueron analizados mediante estadística inferencial, empleando la técnica de regresión logística binaria. Para este proceso se utilizó la herramienta *IBM SPSS Statistics* (versión 28), ampliamente reconocido en investigaciones aplicadas por su capacidad para realizar análisis multivariados de forma robusta y precisa.

3.5. Operacionalización de las variables

Tabla 2. Operacionalización variable independiente aplicación web

Conceptualización	Dimensión	Indicadores	Preguntas	Herramientas
Según Ferrer (2012), una aplicación *web* puede entenderse como un sistema basado en clientes livianos (*light clients*), ya que estos no requieren ejecutar procesos complejos para su funcionamiento. Asimismo, desde el enfoque arquitectónico, se distinguen dos componentes principales: el cliente, donde el usuario final accede a la aplicación a través de un navegador (como *Internet Explorer*, *Google Chrome* o *Firefox*), y el servidor,	Lenguaje de programación	*JavaScript* *Java* *PHP*	¿Qué tipo de dispositivo electrónico utiliza regularmente? ¿Qué tan familiarizado está con el uso de aplicaciones web? ¿Con qué frecuencia utiliza internet para la búsqueda de propiedades? ¿Qué tan necesario es para usted que una aplicación muestre información organizada y actualizada de las propiedades para tomar mejores decisiones de compra o arrendamiento?	Encuesta a los clientes
			¿Cree que el uso de una aplicación mejoraría la eficiencia del equipo en cuanto al mejoramiento de sus labores diarias en la empresa? ¿Qué mejoras espera ver en la gestión inmobiliaria después de implementar una solución digital?	Reunión de planificación del *sprint* con el gerente
	Base de datos	*PostgreSQL* *MongoDB* *Redis*	¿Qué tan necesario le resulta saber si una propiedad inmobiliaria se encuentra disponible o no? ¿Qué tan importante es para usted que la información de las propiedades esté actualizada?	Encuesta a los clientes
			¿Cómo llevan el control de las propiedades disponibles, vendidas y reservadas? ¿Cómo registra el avance de una negociación?	Reunión de planificación del *sprint* con el gerente

que es el encargado de almacenar los datos y gestionar la lógica y las reglas de la aplicación (p. 18).	Tecnologías de *Front-End*	*React* *Angular* *Vue.js*	¿Qué tan difícil le resulta encontrar una propiedad inmobiliaria? ¿Está de acuerdo que una aplicación reduzca el tiempo que le tomaría buscar propiedades? ¿Con qué frecuencia encuentra propiedades desactualizadas en plataformas inmobiliarias? ¿Qué tan clara le resulta la información mostrada en publicaciones de propiedades? ¿Está de acuerdo en que una interfaz bonita e intuitiva facilita su experiencia como usuario?	Encuesta a los clientes
			¿Qué procesos siguen ustedes en una negociación desde que un cliente muestre interés hasta que se concrete el arriendo o la venta?	Reunión de planificación del *sprint* con el gerente
	Tecnologías de *Back-End*	*Django* *Node.js* *Laravel*	¿Con qué frecuencia nota errores o falta de datos relevantes en publicaciones de propiedades? ¿Qué tan importante es para usted que una aplicación deba responder rápido y sin errores al navegar entre secciones?	Encuesta a los clientes
	Sistema de Control de versiones	*Git* *Subversion* *Mercurial*	¿Considera importante que una aplicación para la inmobiliaria se mantenga actualizada con nuevas funciones? ¿Está de acuerdo con que una buena app se actualice constantemente según necesidades de los clientes? ¿Qué tan necesario sería para usted que la aplicación mejore con frecuencia sin afectar su uso habitual? ¿Qué tan importante es para usted saber cuándo una propiedad de su interés deja de estar disponible?	Encuesta a los clientes

Tabla 3. Operacionalización variable independiente machine learning

Conceptualización	**Dimensión**	**Indicadores**	**Preguntas**	**Herramientas**
Según Emiro et al. (2023), menciona que *machine*	Metodologías y estándares del preprocesamiento de datos	*CRISP-DM* *KDD* *SEMMA*	¿Qué tan seguro se siente al compartir sus preferencias personales en plataformas inmobiliarias para recibir un mejor servicio?	Encuesta a los clientes

learning es un subconjunto de la inteligencia artificial que consiste en la construcción de modelos matemáticos basados en muestras conocidas como “datos de entrenamiento”. Se utiliza para realizar predicciones o toma de decisiones sin que el algoritmo programe de manera explícita una tarea o ejecución de una función. Además de percibirse como un grupo de herramientas que busca el aprendizaje computacional mediante datos sin haber ingresado por codificación. Así mismo, es un diseño y estudio que usa la experiencia para mostrar decisiones futuras (p. 30).			¿Está de acuerdo en que la aplicación utilice sus datos para hacerle sugerencias personalizadas? ¿Qué tan organizado y útil considera el proceso de digitalización de sus datos personales en plataformas inmobiliarias?	
	Algoritmos de aprendizaje supervisado	Regresión Clasificación Modelos de ensamble	¿Qué tan importante le resulta contar con un apartado de sus propiedades favoritas en una aplicación para su revisión posterior? ¿Está de acuerdo con que una aplicación le recomiende propiedades inmobiliarias que le podrían interesar? ¿Qué tan importante es para usted que la plataforma deba aprender de sus preferencias para hacerle sugerencias más útiles? ¿Considera útil que se le muestre propiedades basadas en las que ha consultado o marcado como favoritas previamente? ¿Está de acuerdo con que el sistema debería predecir qué propiedades podrían interesarle en base a lo que ya ha visto?	Encuesta a los clientes
	Algoritmo de aprendizaje no supervisado	*Clustering* Reducción de dimensionalidad Asociación	¿Cree que sería útil recibir sugerencias de propiedades de otros clientes con gustos similares hayan consultado? ¿Considera necesario que la app agrupe las propiedades por criterios de interés sin que usted lo indique? ¿Qué tan importante es para usted que el sistema relacione sus preferencias con otras propiedades que usted no ha explorado? ¿Qué tan importante es para usted que el sistema relacione sus preferencias con otras propiedades que usted no ha explorado?	Encuesta a los clientes
	Lenguajes de programación para inteligencia artificial	*Python* *R* *Julia*	¿Está de acuerdo en que el lenguaje con el que se desarrolla una plataforma influye en su rapidez o funcionalidad? ¿Considera importante que el sistema esté construido con tecnologías modernas para ofrecerle una mejor experiencia? ¿Está de acuerdo con que la plataforma debe usar herramientas de inteligencia artificial para conocer mejor a los clientes?	Reunión de planificación del *sprint* con el gerente
	Sistemas de recomendación	Filtrado colaborativo Filtrado basado en contenido	¿Qué tan importante es para usted que una plataforma le sugiera propiedades similares a las que le ha gustado antes? ¿Estaría de acuerdo con que las recomendaciones personalizadas mejorarían su experiencia en la app?	Encuesta a los clientes

		Recomendaciones híbridas	¿Considera útil la incorporación de inteligencia artificial para recomendar propiedades a los clientes que usan la parte pública de la aplicación?	Reunión de planificación del *sprint*

Tabla 4. Operacionalización variable dependiente gestión inmobiliaria

Conceptualización	Dimensión	Indicadores	Preguntas	Herramientas
La gestión inmobiliaria para Pfnür (2011) es el conjunto de actividades destinadas a la planificación, dirección y control de los recursos y procesos relacionados con las propiedades inmobiliarias, con el objetivo de optimizar su uso y maximizar su valor. Además, abarca tres enfoques principales: el primero, centrado en el usuario, en el que los inmuebles se consideran herramientas de soporte para la operación y eficiencia de las empresas; el segundo, desde la perspectiva del propietario o inversor,	Gestión de propiedades	Catálogo de propiedades Análisis de valuación Responsabilidad social	¿Considera necesario que se muestre toda la información sobre cada propiedad? ¿Es importante para usted que un catálogo de propiedades se presente de forma clara y ordenada?	Encuesta a los clientes
			¿Cómo registran actualmente las propiedades en venta? ¿Tienen propiedades que permanecen publicadas, aunque ya no estén disponibles?	Reunión de planificación del *sprint*
	Gestión de clientes	Administración de base de datos de clientes Manejo de quejas y reclamos Programa de fidelización	¿Está de acuerdo con que el agente inmobiliario gestione adecuadamente sus datos personales e intereses? ¿Con qué frecuencia el agente recuerda detalles que usted ya le había mencionado antes? ¿Está de acuerdo con que su experiencia mejora cuando el agente tiene un historial detallado de su proceso?	Encuesta a los clientes
			¿Cómo se registra la información de los clientes interesados en las propiedades que ofertan? ¿Llevan fichas con datos personales de los clientes? ¿Dónde las almacenan?	Reunión de planificación del *sprint* con el gerente
	Gestión de negociaciones y seguimiento	Manejo de objeciones Estrategias para cierre de acuerdos Etapas del proceso de negociación (pipeline)	¿Con qué frecuencia siente que un agente inmobiliario le brinda un seguimiento personalizado? ¿Qué tan fácil considera que es recibir seguimiento por parte del agente inmobiliario que le atendió? ¿Está de acuerdo con que la app permite gestionar de forma ordenada las negociaciones activas? ¿Con qué frecuencia ha perdido oportunidades de compra o arriendo por falta de seguimiento por parte de los agentes?	Encuesta a los clientes

en el que se consideran a las propiedades como activos financieros y se prioriza su rentabilidad; y el tercero, orientado a la gestión operativa, que integra el ciclo de vida del inmueble compuesto por la planificación, construcción, explotación y comercialización del mismo (pp. 8-9).			¿Qué tan importante le parece que el sistema registre la etapa de cada negociación en curso?	
			¿Cómo se lleva el seguimiento a los clientes que están interesados por una determinada propiedad? ¿Se puede saber fácilmente en qué etapa está un cliente respecto a una negociación?	Reunión de planificación del *sprint*
	Gestión de agentes inmobiliarios	Gestión de talento humano Asignación de propiedades y clientes Capacitación y desarrollo	¿Qué tan importante es para usted que el agente inmobiliario conozca sus preferencias? ¿Con qué frecuencia ha tenido que repetir su información en cada interacción con el agente inmobiliario? ¿Qué tan útil considera que el agente pueda registrar sus inquietudes y observaciones durante una negociación? ¿Está de acuerdo con que cada cliente sea asignado a un agente para que lo guíe durante todo el proceso?	Encuesta a los clientes
			¿Quiénes pueden intervenir en el proceso de publicación de una propiedad? ¿Cómo gestiona la información de sus agentes?	Reunión de planificación del *sprint*
	Gestión de contratos	Contratos de compraventa Contratos de arrendamiento Contratos de exclusividad	¿Qué tan importante le parece que los agentes puedan subir documentos durante la negociación para fines de organización y agilidad? ¿Está satisfecho con la gestión de documentos que ha recibido por parte de la inmobiliaria?	Encuesta a los clientes
			¿Cómo gestionan los documentos como, por ejemplo, copias de cédula, promesas de compraventa, escrituras, etc., de una determinada negociación?	Reunión de planificación del *sprint*

4. RESULTADOS

4.1. Resultado del primer objetivo específico

La validación de los instrumentos de investigación del presente trabajo de titulación como son la encuesta a los clientes de la empresa fue analizados y validados por docentes expertos, según se evidencia en la tabla 5.

Tabla 5. Experto en validación de instrumentos para recopilación de datos

Nombres	Título académico	Área
Luis Javier Ulloa Meneses	Mg. Big Data y Data Science	Tecnologías de la Información y la Comunicación (TIC)
Moreno Rivera Mario Jonathan	Máster Universitario en Ingeniería de software y Sistemas Informáticos	Desarrollo web

4.1.1. Aplicación de las encuestas a los clientes

En la primera etapa, se aplicó la encuesta antes de la implementación de la propuesta de intervención, la cual fue realizada a los clientes de la empresa Escudero.

4.1.1.1. Presentación de los resultados de la encuesta a los clientes

De acuerdo con la recopilación de datos realizada a los 65 clientes de la empresa Inmobiliaria Escudero, mediante la aplicación de encuestas en la fase de *pretest*, se evidencia que el 29.20% de los usuarios utiliza principalmente el teléfono móvil, seguido de un 24.60% que emplea *laptop* y un 20% computadora de escritorio, lo que refleja la necesidad de contar con soluciones accesibles desde distintos dispositivos. En cuanto al uso de aplicaciones *web*, el 30.80% se considera totalmente familiarizado y el 32.30% familiarizado, aunque aún existe un 33.50% que presenta niveles moderados o bajos de familiaridad, evidenciando la importancia de interfaces intuitivas.

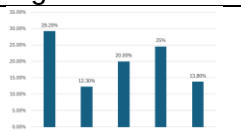
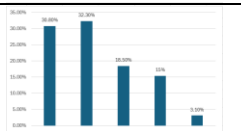
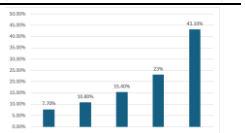
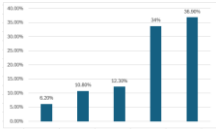
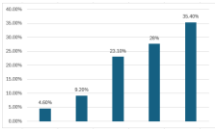
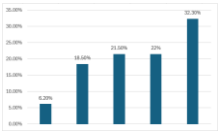
Por otro lado, el proceso de búsqueda presenta dificultades, ya que el 36.90% lo percibe como “muy difícil” y el 21.50% como “difícil”, mientras que solo el 17% lo considera fácil o muy fácil. Además, el 58.40% indica que encuentra propiedades disponibles con

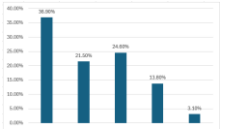
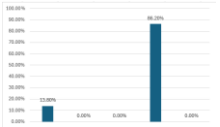
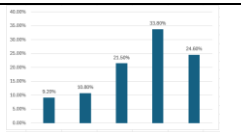
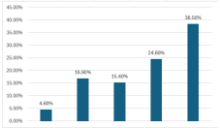
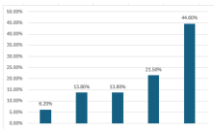
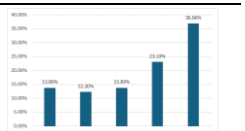
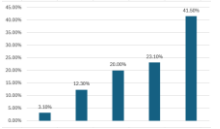
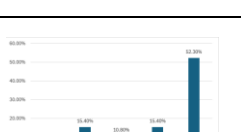
poca o ninguna frecuencia, lo que evidencia limitaciones en el acceso a información actualizada. Además, el 63.10% de los usuarios consideran que la información mostrada es poco o nada clara, y el 66.10% percibe que las interfaces no son intuitivas, lo que afecta negativamente la interacción con las plataformas.

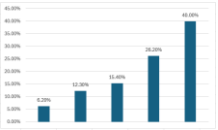
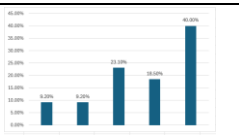
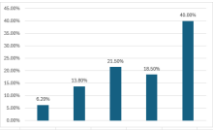
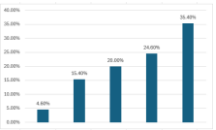
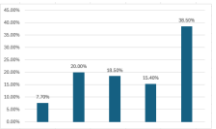
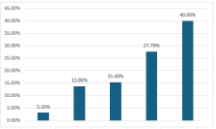
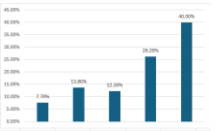
Asimismo, se identifican problemas en el seguimiento y personalización, ya que el 72.30% de los usuarios indica que rara vez o nunca recibe seguimiento personalizado por parte de los agentes, el 75.40% considera difícil recibir dicho seguimiento, y el 41.60% ha perdido oportunidades de compra o arriendo con frecuencia o de forma medianamente frecuente. En conjunto, estos resultados evidencian deficiencias en la organización de la información, el acceso a propiedades y la gestión del seguimiento, lo que justifica la necesidad de implementar una solución tecnológica que optimice la experiencia del usuario y mejore la toma de decisiones.

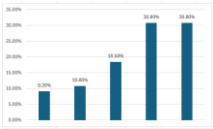
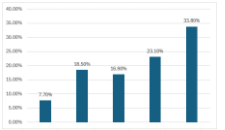
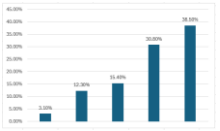
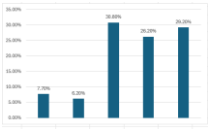
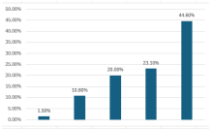
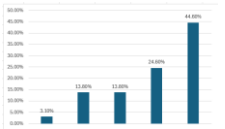
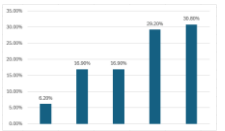
4.1.2. Resultados de la encuesta a los clientes de la empresa Escudero en el Cantón Santo Domingo

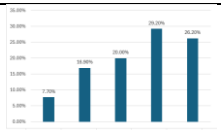
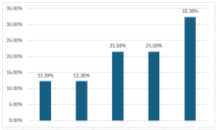
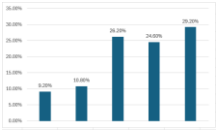
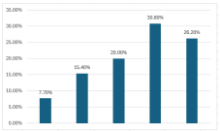
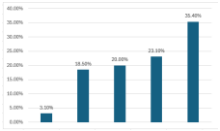
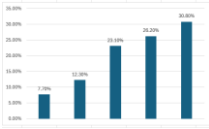
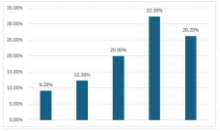
Tabla 6. Datos obtenidos de las encuestas

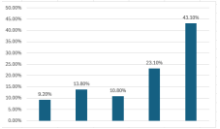
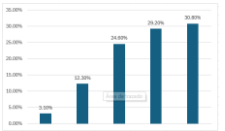
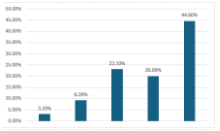
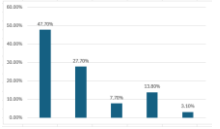
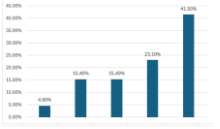
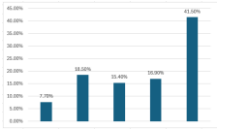
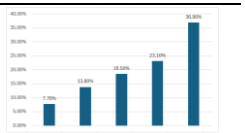
N.º	Preguntas	Escala y %					Figuras
1	¿Qué tipo de dispositivo electrónico utiliza regularmente?	Teléfono móvil	Computadora de escritorio	Laptop	Tablet	Otro	
		29.20%	12.30%	20%	24.60%	13,80%	
2	¿Qué tan familiarizado está con el uso de aplicaciones web?	Totalmente familiarizado	Familiarizado	Moderadamente familiarizado	Poco familiarizado	Nada familiarizado	
		30.80%	32.30%	18.50%	15%	3.10%	
3	¿Con qué frecuencia utiliza Internet para la búsqueda de propiedades inmobiliarias?	Muy frecuente	Frecuente	Medianamente frecuente	Poco frecuente	Nada frecuente	
		7.70%	10.80%	15.40%	23.10%	43.10%	
4	¿Qué tan necesario es para usted que una aplicación muestre información organizada y actualizada de las propiedades para tomar mejores decisiones de compra o arrendamiento?	Totalmente necesario	Necesario	Moderadamente necesario	Poco necesario	Innecesario	
		6.20%	10.80%	12.30%	33.80%	36.90%	
5	¿Qué tan necesario le resulta saber si una propiedad inmobiliaria se encuentra disponible o no?	Totalmente necesario	Necesario	Moderadamente necesario	Poco necesario	Innecesario	
		4.60%	9.20%	23.10%	27.70%	35.40 %	
6	¿Qué tan importante es para usted que la información de las propiedades esté actualizada?	Muy importante	Importante	Medianamente importante	Poco importante	Nada importante	
		6.20 %	18.50 %	21.50 %	21.50 %	32.30 %	
7		Muy difícil	Difícil	Normal	Fácil	Muy fácil	

	¿Qué tan difícil le resulta encontrar una propiedad inmobiliaria de su interés?	36.90%	21.50%	24.60%	13.80%	3.10%	
8	¿Está de acuerdo que una aplicación reduzca el tiempo que le tomaría buscar propiedades?	Totalmente de acuerdo	Bastante de acuerdo	Neutral	Poco de acuerdo	Totalmente en desacuerdo	
		13.80%	0.00%	0.00%	86.20%	0,00%	
9	¿Con qué frecuencia encuentra propiedades disponibles?	Muy frecuente	Frecuente	Medianamente frecuente	Poco frecuente	Nada frecuente	
		9,20%	10.80 %	21.50 %	33.80 %	24.60 %	
10	¿Qué tan clara le resulta la información mostrada en publicaciones de propiedades?	Muy clara	Clara	Medianamente clara	Poco clara	Nada clara	
		4.60 %	16.90 %	15.40 %	24.60%	38.50%	
11	¿Está de acuerdo en que una interfaz bonita e intuitiva facilita su experiencia como usuario?	Totalmente de acuerdo	Bastante de acuerdo	Neutral	Poco de acuerdo	Totalmente en desacuerdo	
		6.20 %	13.80 %	13.80 %	21.50 %	44.60 %	
12	¿Con qué frecuencia nota errores o falta de datos relevantes en publicaciones de propiedades?	Muy frecuente	Frecuente	Medianamente frecuente	Poco frecuente	Nada frecuente	
		13.80 %	12.30%	13.80%	23.10%	36.90%	
13	¿Qué tan importante es para usted que una aplicación deba responder rápido y sin errores al navegar entre secciones?	Muy importante	Importante	Medianamente importante	Poco importante	Nada importante	
		3.10%	12.30 %	20 %	23.10 %	41.50%	
14	¿Considera importante que una aplicación para la inmobiliaria se mantenga actualizada con nuevas funciones?	Muy importante	Importante	Medianamente importante	Poco importante	Nada importante	
		6.20%	15.40 %	10.80 %	15.40 %	52.30 %	

15	¿Está de acuerdo con que una buena app se actualice constantemente según necesidades de los clientes?	Totalmente de acuerdo	Bastante de acuerdo	Neutral	Poco de acuerdo	Totalmente en desacuerdo	
		6.20%	12.30%	15.40 %	26.20 %	40%	
16	¿Qué tan necesario sería para usted que la aplicación mejore con frecuencia sin afectar su uso habitual?	Totalmente necesario	Necesario	Moderadamente necesario	Poco necesario	Innecesario	
		9.20%	9.20 %	23.10 %	18.50 %	40 %	
17	¿Qué tan importante es para usted saber cuándo una propiedad de su interés deja de estar disponible?	Muy importante	Importante	Medianamente importante	Poco importante	Nada importante	
		6.20%	13.80%	21.50%	18.50%	40%	
18	¿Qué tan seguro se siente al compartir sus preferencias personales en plataformas inmobiliarias para recibir un mejor servicio?	Totalmente seguro	Seguro	Moderadamente seguro	Poco seguro	Totalmente inseguro	
		4.60%	15.40 %	20%	24.60 %	35.40 %	
19	¿Está de acuerdo en que la aplicación utilice sus datos para hacerle sugerencias personalizadas?	Totalmente de acuerdo	Bastante de acuerdo	Neutral	Poco de acuerdo	Totalmente en desacuerdo	
		7.70%	20%	18.50%	15.40 %	38.50%	
20	¿Qué tan organizado y útil considera el proceso de digitalización de sus datos personales en plataformas inmobiliarias?	Totalmente útil	Útil	Moderadamente útil	Poco útil	Inútil	
		3.10 %	13.80 %	15.40%	27.70 %	40 %	
21	¿Qué tan importante le resulta contar con un apartado de sus propiedades favoritas en una aplicación para su revisión posterior?	Muy importante	Importante	Medianamente importante	Poco importante	Nada importante	
		7.70%	13.80 %	12.30 %	26.20 %	40 %	

22	¿Está de acuerdo con que una aplicación le recomiende propiedades inmobiliarias que le podrían interesar?	Totalmente de acuerdo	Bastante de acuerdo	Neutral	Poco de acuerdo	Totalmente en desacuerdo	
		9.20%	10.80 %	18.50 %	30.80 %	30.80 %	
23	¿Qué tan importante es para usted que la plataforma deba aprender de sus preferencias para hacerle sugerencias más útiles?	Muy importante	Importante	Medianamente importante	Poco importante	Nada importante	
		7.70%	18.50%	16.90 %	23.10 %	33.80 %	
24	¿Considera útil que se le muestre propiedades basadas en las que ha consultado o marcado como favoritas previamente?	Totalmente útil	Útil	Moderadamente útil	Poco útil	Inútil	
		3.10%	12.30%	15.40 %	30.80 %	38.50 %	
25	¿Está de acuerdo con que el sistema debería predecir qué propiedades podrían interesarle en base a lo que ya ha visto?	Totalmente de acuerdo	Bastante de acuerdo	Neutral	Poco de acuerdo	Totalmente en desacuerdo	
		7.70%	6.20%	30.80 %	26.20 %	29.20 %	
26	¿Cree que sería útil recibir sugerencias de propiedades de otros clientes con gustos similares hayan consultado?	Totalmente útil	Útil	Moderadamente útil	Poco útil	Inútil	
		1.50%	10.80 %	20%	23.10 %	44.60 %	
27	¿Considera necesario que la app agrupe las propiedades por criterios de interés sin que usted lo indique?	Totalmente necesario	Necesario	Moderadamente necesario	Poco necesario	Innecesario	
		3.10%	13.80%	13.80%	24.60%	44.60%	
28	¿Qué tan importante es para usted que el sistema relacione sus preferencias con otras propiedades que usted no ha explorado?	Muy importante	Importante	Medianamente importante	Poco importante	Nada importante	
		6.20%	16.90%	16.90%	29.20%	30.80%	

29	¿Está de acuerdo en que se desarrolle con tecnologías modernas?	Totalmente de acuerdo	Bastante de acuerdo	Neutral	Poco de acuerdo	Totalmente en desacuerdo	
		7.70%	16.90%	20%	29.20%	26.20%	
30	¿Considera importante que el sistema esté construido con tecnologías modernas para ofrecerle una mejor experiencia?	Muy importante	Importante	Medianamente importante	Poco importante	Nada importante	
		12.30%	12.30%	21.50%	21.50 %	32.30 %	
31	¿Está de acuerdo con que la plataforma debe usar herramientas de inteligencia artificial para conocer mejor a los clientes?	Totalmente de acuerdo	Bastante de acuerdo	Neutral	Poco de acuerdo	Totalmente en desacuerdo	
		9.20%	10.80%	26.20 %	24.60 %	29.20%	
32	¿Qué tan importante es para usted que una plataforma le sugiera propiedades similares a las que le ha gustado antes?	Muy importante	Importante	Medianamente importante	Poco importante	Nada importante	
		7.70%	15.40%	20%	30.80 %	26.20%	
33	¿Estaría de acuerdo con que las recomendaciones personalizadas mejorarían su experiencia en la app?	Totalmente de acuerdo	Bastante de acuerdo	Neutral	Poco de acuerdo	Totalmente en desacuerdo	
		3.10%	18.50 %	20 %	23.10 %	35,40%	
34	¿Considera necesario que se muestre toda la información sobre cada propiedad?	Totalmente necesario	Necesario	Moderadamente necesario	Poco necesario	Innecesario	
		7.70%	12.30%	23.10 %	26.20%	30.80%	
35	¿Es importante para usted que un catálogo de propiedades se presente de forma clara y ordenada?	Muy importante	Importante	Medianamente importante	Poco importante	Nada importante	
		9.20%	12.30%	20%	32.30%	26.20 %	
36	¿Está de acuerdo con que el agente inmobiliario gestione adecuadamente	Totalmente de acuerdo	Bastante de acuerdo	Neutral	Poco de acuerdo	Totalmente en desacuerdo	
		9.20 %	15.40 %	21.50 %	20 %	33.80 %	

	sus datos personales e intereses?						
37	¿Con qué frecuencia el agente recuerda detalles que usted ya le había mencionado antes?	Muy frecuente	Frecuente	Medianamente frecuente	Poco frecuente	Nada frecuente	
		9.20%	13.80 %	10.80 %	23.10 %	43.10 %	
38	¿Está de acuerdo con que su experiencia mejora cuando el agente tiene un historial detallado de su proceso?	Totalmente de acuerdo	Bastante de acuerdo	Neutral	Poco de acuerdo	Totalmente en desacuerdo	
		3.10%	12.30%	24.60%	29.20 %	30.80 %	
39	¿Con qué frecuencia siente que un agente inmobiliario le brinda un seguimiento personalizado?	Muy frecuente	Frecuente	Medianamente frecuente	Poco frecuente	Nada frecuente	
		3.10%	9.20%	23.10 %	20 %	44.60 %	
40	¿Qué tan fácil considera que es recibir seguimiento por parte del agente inmobiliario que le atendió?	Muy difícil	Difícil	Normal	Fácil	Muy fácil	
		47.70%	27.70 %	7.70 %	13.80 %	3.10 %	
41	¿Está de acuerdo con que la app permite gestionar de forma ordenada las negociaciones activas?	Totalmente de acuerdo	Bastante de acuerdo	Neutral	Poco de acuerdo	Totalmente en desacuerdo	
		4.60%	15.40%	15.40 %	23.10 %	41.50 %	
42	¿Con qué frecuencia ha perdido oportunidades de compra o arriendo por falta de seguimiento por parte de los agentes inmobiliarios?	Muy frecuente	Frecuente	Medianamente frecuente	Poco frecuente	Nada frecuente	
		7.70 %	18.50%	15.40 %	16.90%	41.50 %	
43	¿Qué tan importante le parece que el sistema registre la etapa de cada negociación en curso?	Muy importante	Importante	Medianamente importante	Poco importante	Nada importante	
		7.70%	13.80%	18.50%	23.10 %	36.90%	
44	¿Qué tan importante es para usted que el agente	Muy importante	Importante	Medianamente importante	Poco importante	Nada importante	

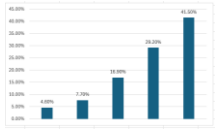
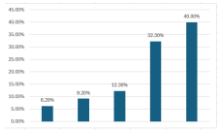
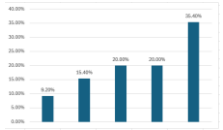
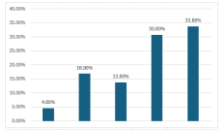
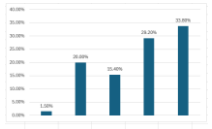
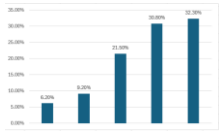
	inmobiliario conozca sus preferencias?	4.60%	7.70%	16.90%	29.20 %	41.50%	
45	¿Con qué frecuencia ha tenido que repetir su información en cada interacción con el agente inmobiliario?	Muy frecuente	Frecuente	Medianamente frecuente	Poco frecuente	Nada frecuente	
		6.20%	9.20%	12.30%	32.30%	40%	
46	¿Qué tan útil considera que el agente pueda registrar sus inquietudes y observaciones durante una negociación?	Totalmente útil	Útil	Moderadamente útil	Poco útil	Inútil	
		9.20%	15.40%	20%	20 %	35.40 %	
47	¿Está de acuerdo con que cada cliente sea asignado a un agente para que lo guíe durante todo el proceso?	Totalmente de acuerdo	Bastante de acuerdo	Neutral	Poco de acuerdo	Totalmente en desacuerdo	
		4.60%	16.90%	13.80%	30.80%	33.80 %	
48	¿Qué tan importante le parece que los agentes puedan subir documentos durante la negociación para fines de organización y agilidad?	Muy importante	Importante	Medianamente importante	Poco importante	Nada importante	
		1.50%	20%	15.40%	29.20%	33.80%	
49	¿Está satisfecho con la gestión de documentos que ha recibido por parte de la inmobiliaria?	Totalmente satisfecho	Satisfecho	Moderadamente satisfecho	Poco satisfecho	Nada satisfecho	
		6.20%	9.20%	21.50%	30.80%	32.30%	
		46.20%	53.80%				

Figura 4. Diagrama BPD del proceso de gestión inmobiliaria

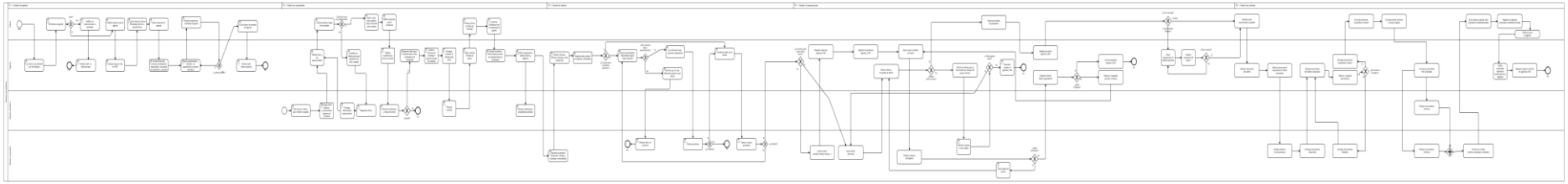


Figura 5. Diagrama BPD del proceso de gestión de propiedades

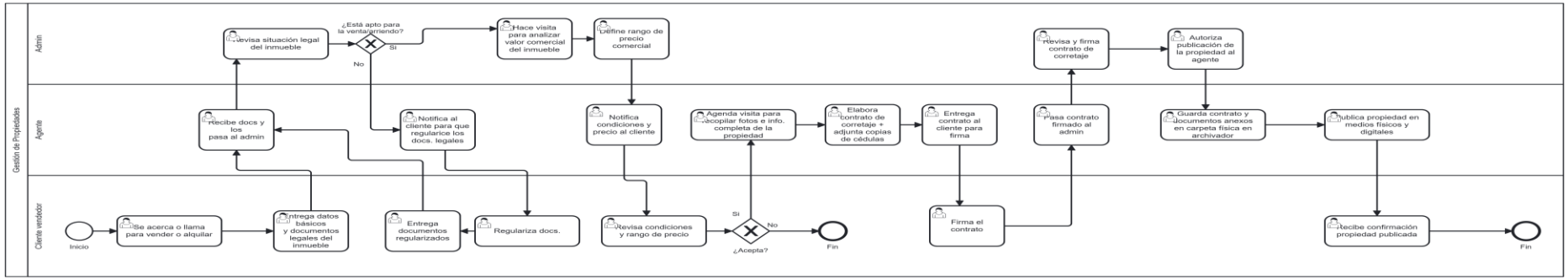


Figura 6. Diagrama BPD del proceso de gestión de agentes inmobiliarios

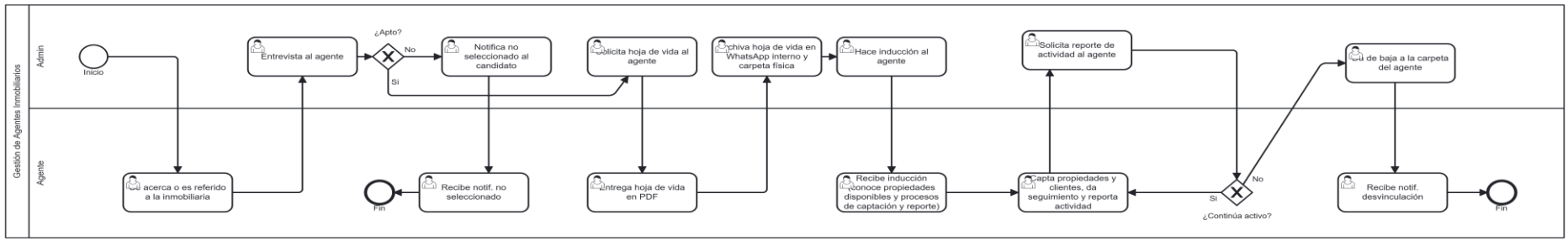


Figura 7. Diagrama BPD del proceso de gestión de clientes

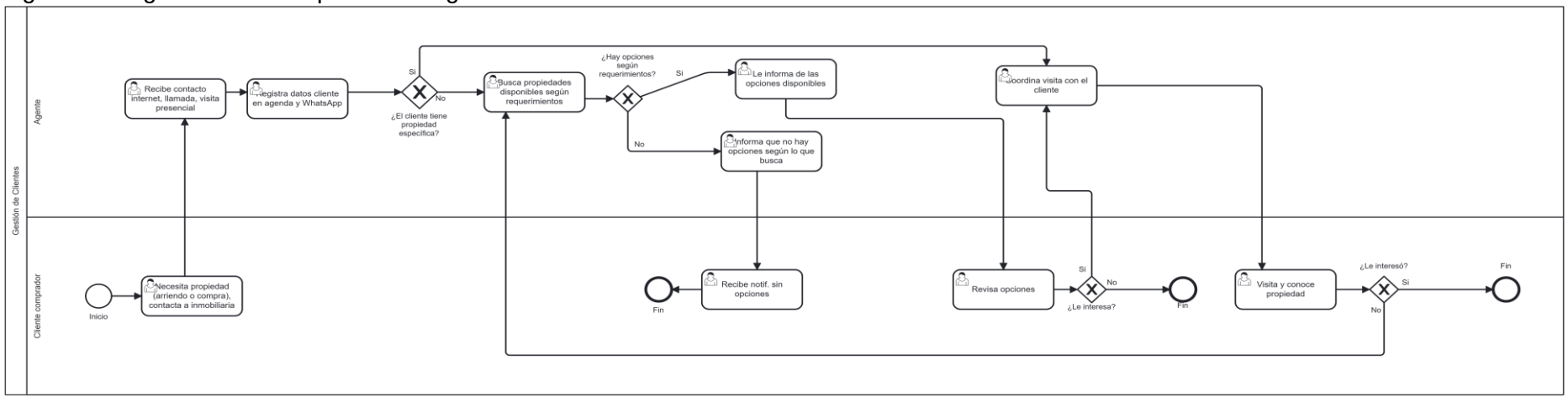
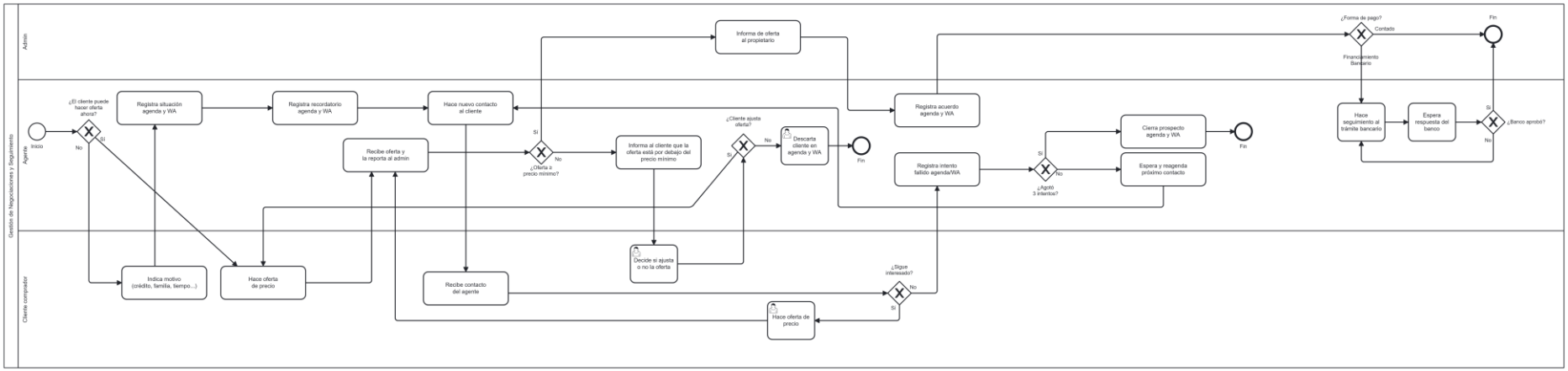


Figura 8. Diagrama BPD del proceso de gestión de negociaciones y seguimiento



4.1.2.1. Identificación de los procesos en la gestión inmobiliaria

Mediante la reunión de planificación del *sprint* llevada a cabo con el gerente de la inmobiliaria Escudero, (véase anexo VI y XI), se identificaron los procesos que forman parte de la gestión inmobiliaria de la empresa: gestión de propiedades, gestión de clientes, gestión de negociaciones y seguimiento, gestión de contratos, y, por último, gestión de agentes inmobiliarios.

4.1.2.2. Diagrama BPD del proceso de gestión inmobiliaria

A través de los instrumentos empleados en el presente trabajo de titulación, a partir de los procesos identificados de la gestión inmobiliaria, se elaboró para cada uno de ellos un *Business Process Diagram* (*BPD*) utilizando la notación *Business Process Model* and *Notation* (*BPMN* 2.0), que se presentan en las figuras 4, 5, 6, 7 y 8.

Por otra parte, cabe recordar que un diagrama *BPD* modela la secuencia de actividades, eventos y decisiones de un proceso, lo cual se convirtió en una clara referencia del funcionamiento del negocio para guiar la definición de los requerimientos del sistema. El diseño de la base de datos, la identificación de los roles de usuario y las funcionalidades que debe contemplar la aplicación expresadas a través de historias de usuario. No obstante, se debe precisar que el diagrama *BPD* corresponde exclusivamente al ámbito del desarrollo de la propuesta de intervención (aplicación *web* con *machine learning* para la gestión inmobiliaria), por lo que no fue considerado como un instrumento que forma parte del enfoque de investigación.

4.2. Resultado del segundo objetivo específico.

En el desarrollo de la aplicación *web* con *machine learning* para la gestión inmobiliaria, se aplicó el principio de separación de responsabilidades y se siguió el marco ágil de trabajo *Scrum* para la organización del proyecto. Asimismo, se hizo un proceso de comparación y selección de herramientas para conformar el *stack* tecnológico de la aplicación compuesto por: tecnologías de *frontend* y *backend*, sistemas de gestión de bases

de datos, control de versiones, estilos arquitectónicos y algoritmos de *machine learning*, seleccionando en cada área la opción más adecuada.

4.2.1.1. Tecnologías de Frontend

En la actualidad, existe una diversidad de *frameworks* y librerías del lado del *frontend* para la construcción de interfaces de usuario interactivas. No obstante, se elaboró un análisis comparativo de características entre los tres *frameworks frontend* más populares y usados en la actualidad que se refleja en la tabla 7.

Tabla 7. Comparativa de tecnologías de Frontend

Características	Tecnologías de Frontend		
	React [a, b]	Angular [b, c]	Vue [d]
Tipo	Librería	Framework (completo)	Framework progresivo
Desarrollador	Meta (Facebook)	Google	Evan You
Lenguaje	JavaScript, TypeScript (opcional)	TypeScript (obligatorio)	JavaScript, TypeScript (opcional)
Arquitectura	Basado en componentes con flujo de datos unidireccional	Basado en componentes con inyección de dependencias	Inspirado en MVVM
DOM	Virtual DOM	Incremental DOM	Virtual DOM

Nota: Obtenido de: [a]Meta Platforms Inc. (2026), [b]MDN Web Docs (2025), [c]Angular Team (2026), [d]Vue.js Core Team (2026)

Posterior al análisis de estas tecnologías, se inclinó por usar *React*, dado que es una librería muy flexible que permite elegir herramientas complementarias según las necesidades que pueda tener el proyecto. Permite construir interfaces *web* con componentes reutilizables, usa un *DOM* virtual que mejora el rendimiento en la renderización de aplicaciones con muchos componentes, asegurando una experiencia rápida para el usuario. Es también, una librería relativamente fácil de aprender y usar para programadores que cuentan con conocimientos en *JavaScript* y *HTML*.

Por otro lado, se usó *Tailwind CSS* como *framework CSS* por su excelente integración con *React,* ya que adopta un enfoque "*utility-first*" al proporcionar clases predefinidas que, permite aplicar estilos directamente al *HTML* o *JSX*, en lugar de hacerlo

con mucho *CSS* personalizado, permitiendo así estilizar las interfaces *web* de forma flexible y rápida.

4.2.2. Tecnologías de Backend

De forma similar, se realizó una tabla comparativa entre tres *frameworks* muy reconocidos del lado *backend*, como se visualiza en la tabla 8.

Tabla 8. Comparativa de tecnologías de Backend

Características	Tecnologías de Backend		
	Express [a, b, c, d]	Django [e]	Laravel [f, g]
Lenguaje base	Node.js (JavaScript)	Python	PHP
Arquitectura	Basada en middleware	Model-View-Template (MVT)	Model-View-Controller (MVC)
Gestor de paquetes	npm, yarn, pnpm	pip	Composer
Soporte ORM	Ninguno integrado; usa librerías externas como Sequelize, Prisma, TypeORM	ORM integrado	Eloquent ORM

Nota: Obtenido de: [a]OpenJS Foundation (2026), [b]Sequelize Team (2026), [c]Prisma Data Inc. (2026), [d]TypeORM Team (2026), [e]Django Software Foundation (2026), [f]Laravel Team (2026), [g]Composer Team (2026)

Se eligió *Express* como *framework backend* debido a que permite utilizar *JavaScript* tanto del lado del cliente como del servidor, facilitando en gran medida el desarrollo al emplear un mismo lenguaje en todo el proyecto.

Asimismo, *Express* al basarse en *Node.js* opera con un modelo no bloqueante y asincrónico que, posibilita manejar múltiples conexiones simultáneas de forma eficiente, lo que lo hace ideal para *APIs REST* de alto rendimiento. Además, posee un ecosistema muy rico en paquetes, gracias a *NPM* (*Node Package Manager*) que son librerías y herramientas con utilidades muy comunes que al usarlas acelera la creación de aplicaciones.

4.2.3. Sistemas de gestión de bases de datos

En la tabla 9, se muestra una comparación entre diferentes sistemas de gestión de base de datos para determinar cuál de ellos cubre las necesidades del proyecto.

Tabla 9. Comparativa de Sistemas de Gestión de Bases de Datos

Características	Sistemas de Gestión de Bases de Datos		
	PostgreSQL [a]	Redis [b]	MongoDB [c]

Tipo	Base de datos relacional/objeto-relacional (SQL)	Base de datos NoSQL en memoria (key-value store)	Base de datos NoSQL orientada a documentos
Modelo de datos	Relacional	Key-value	Documentos flexibles en formato BSON
Escalabilidad	Escalabilidad para grandes volúmenes de datos y cargas de trabajo de alta concurrencia	Escalado horizontal vía Redis Cluster con sharding por hash	Escalado horizontal nativo con sharding y distribución geográfica
Lenguaje de Consultas	SQL estándar con soporte para JSON y funciones avanzadas.	Comandos key-value	MQL (MongoDB Query language)

Nota. Obtenido de: [a] PostgreSQL Global Development Group (2026), [b]Redis Ltd. (2026), [c]MongoDB Inc. (2026)

En base al análisis de la información de la tabla 9, se optó por utilizar *PostgreSQL* como sistema de gestión de bases de datos, debido a la naturaleza relacional que posee el proyecto al facilitar la modelación de relaciones complejas entre datos estructurados. Adicionalmente, se utilizó *Prisma ORM* para facilitar aún más la interacción con la base de datos usando código *JavaScript*/*TypeScript* en lugar del lenguaje *SQL* que maneja *PostgreSQL*.

4.2.4. Sistemas de control de versiones

Un sistema de control de versiones se ha convertido en una herramienta imprescindible en el desarrollo de software, pues permite gestionar de forma eficiente la evolución del código fuente.

Tabla 10. Comparativa de Sistemas de Control de Versiones

Características	Sistemas de Control de Versiones		
	Git [a, b]	Subversion [b, c, d]	Mercurial [b, e, f, g]
Modelo	Distribuido (DVCS)	Centralizado (CVCS)	Distribuido (DVCS)
Branching y Merging	Flexible y eficiente; branching económico y rápido.	Con soporte de branching y merging con seguimiento semi-automatizado	Basado en named branches, bookmarks y anonymous branches; merges generan changesets con múltiples padres
Almacenamiento	Basado en snapshots	Basado en deltas	Basado en changesets

Integridad	Backups distribuidos y checksums (SHA-1)	Repositorio centralizado	Hashing de changesets
Herramientas de hosting	GitHub, GitLab, Azure Repos	Assembla	Heptapod, RhodeCode
Popularidad	Muy alta	Baja	Baja

Nota. Obtenido de: [a]Software Freedom Conservancy (2025), [b]Stack Exchange Inc. (2026), [c]Apache Software Foundation (2026), [d]Assembla (2026), [e]Mercurial Development Team (2026), [f]Octobus y Clever Cloud (2026), [g]RhodeCode GmbH (2026)

Como se observa en la tabla 10, se ha elaborado una comparación entre algunos sistemas de control de versiones, de los cuales se ha escogido a *Git* por su alto rendimiento, arquitectura distribuida, una popularidad y uso sólido por desarrolladores del mundo. Asimismo, se ha optado por usar a *GitHub* como repositorio para alojar el proyecto al permitir centralizar el código fuente y facilitar el trabajo colaborativo entre los desarrolladores.

4.2.5. Estilos arquitectónicos

Un estilo arquitectónico, a veces llamado patrón arquitectónico, se entiende como la estructura general y superior de cómo se organizan la interfaz de usuario y el código *backend*. La tabla 11 proporciona información comparativa entre los estilos como: arquitectura por capas y arquitectura de microservicios, los cuales son patrones que guardan ventajas y desventajas a considerar al decidir cómo se afrontan el desarrollo de un proyecto de *software*.

Tabla 11. Comparativa de Estilos Arquitectónicos

Características	Estilos Arquitectónicos	
	Arquitectura por Capas[a]	Arquitectura de Microservicios [a]
Tipo de Arquitectura	Monolítica (una única unidad de despliegue)	Distribuida (varias unidades de despliegue independientes)
Particionamiento	Partición técnica; el dominio de negocio se extiende a lo largo de las capas.	Centrada en el dominio; cada servicio delimita un contexto específico del negocio.
Escalabilidad	Baja; debido a que el despliegue es monolítico.	Alta; cada servicio se escala de forma individual ajustándose con precisión a la demanda de ese servicio en concreto (fine-tuned scaling).
Rendimiento	Bajo; al requerir que las solicitudes pasen por múltiples capas, lo que	Bajo; afectado por la latencia de la red provocada por la llamada entre servicios y las

	impacta el tiempo de respuesta.	verificaciones de seguridad en cada punto final
Complejidad y Costo	Baja; fácil de entender para los desarrolladores y bajo costo inicial.	Alta; alto costo y complejidad debido a su naturaleza distribuida.

Nota. Obtenido de: [a]Richards y Ford (2020)

Según la información recopilada de estilos arquitectónicos, se eligió la arquitectura por capas debido a su simplicidad al momento de implementar y mantener una solución de software a un bajo costo. Así como también, su estructura de capas que presenta este patrón (Presentación, Negocio, Base de Datos), se alinea muy bien con las tareas correspondientes a los de equipos de desarrollo, haciendo de este estilo una opción viable para muchas aplicaciones de negocio.

4.2.6. Algoritmos de Aprendizaje Supervisado

En el campo de *machine learning*, la selección de algoritmo es ideal para garantizar un modelo predictivo más eficiente, dado que es la base para los procesos de entrenamiento y aprendizaje de los sistemas inteligentes.

Tabla 12. Comparativa de Algoritmos No supervisados

	Algoritmos No supervisados		
Características	Clustering[a]	Reducción de Dimensionalidad[a]	Asociación[a]
Tipo de aprendizaje	No supervisado	No supervisado	No supervisado
Objetivo principal	Agrupar puntos de datos similares en clústeres basados en una métrica de similitud	Reducir el número de características (o dimensiones) en los datos	descubrir relaciones o asociaciones interesantes entre variables en grandes conjuntos de datos.
Modelos comunes	K-means Agrupamiento jerárquico	PCA t-SNE	Algoritmo Apriori

Nota. Obtenido de: [a]Vanitha y Kasthuri (2024)

Por medio de esta comparativa, se optó por el algoritmo no supervisado de reducción de dimensionalidad, ya que permite disminuir el número de características (o dimensiones) de los datos manteniendo la información esencial. Además, resulta adecuado para el análisis de grandes volúmenes de información, facilitando la identificación de

patrones y la simplificación de datos complejos, lo que contribuye a mejorar el rendimiento de los modelos en tareas de análisis y predicción.

4.2.7. Lenguajes de programación para inteligencia artificial

Dentro del ámbito del *Machine Learning*, la elección del lenguaje de programación influye directamente en la eficiencia del desarrollo, precisión del modelado y escalabilidad del sistema, para ello se realizará una tabla comparativa:

Tabla 13. Comparativa de Lenguajes de Programación para Inteligencia Artificial

Características	Lenguajes de Programación para Inteligencia Artificial		
	Python[a]	R[b]	Julia[c]
Licencia	Software libre	Licencia Pública General de GNU (GPL)	Open Source (licencia MIT).
Propósito general	Desarrollo general, IA, análisis de datos.	Análisis estadísticos y minería de datos.	Manejo de grandes datos y análisis numéricos
Ventajas	Fácil de usar, extensa comunidad y librerías como Numpy, Pandas, Matplotlib, Scikit-Learn, TensorFlow.	Potente para estadísticas y gráficos, cantidad de paquetes especializados en análisis de datos.	Excelente rendimiento numérico
Popularidad actual	Más utilizado en IA y ciencia de datos a nivel global	Altamente valorado en entornos académicos y de investigación estadística.	En crecimiento en el campo científico y técnico

Nota. Obtenido de: [a] Ramírez y Ramírez (2023), [b] Heesen (2024), [c] Dash (2024)

De los distintos lenguajes de programación para inteligencia artificial, se decidió utilizar *Python* por ser versátil y accesible para desarrollo de inteligencia artificial y aprendizaje automático. Además, ofreciendo características ideales como su fácil uso, potencia de librerías especializadas, y de su sintaxis sencilla y contabilidad con *frameworks* avanzados.

4.2.8. Sistemas de recomendación

Los sistemas de recomendaciones son una de las aplicaciones más representativas del *Machine Learning*, ya que permite predecir las preferencias de los usuarios y ofrecer contenidos más personalizables.

Tabla 14. Comparativa de los sistemas de recomendación

Características	Sistemas de Recomendación		
	Filtrado colaborativo[a]	Filtrado basado en contenido[b]	Recomendaciones híbridas[c]
Definición	Modelo que identifica patrones de comportamiento entre usuarios	Sistema que asigna características a los elementos según similitudes de contenido	Combina distintos enfoques para mejorar la precisión
Tipo de datos utilizados	Valoraciones, puntuaciones o comportamientos de usuarios.	Características de los ítems	Datos mixtos: interacciones, características y contexto.
Ventajas	Ofrece recomendaciones personalizadas según la experiencia colectiva	No necesita datos de otros usuarios; útil para nuevos usuarios	Aumenta la precisión
Nivel de personalización	Alto	Medio	Muy alto

Nota. Obtenido de: [a] Huyen (2023a), [b] Ortega (2022a), [c] Ortega (2022b)

En base a la tabla 14, se eligió el enfoque de filtrado basado en contenido, porque permite realizar recomendaciones personalizadas analizando las características de los elementos y las preferencias de los usuarios sin depender de información o valoraciones de otros.

4.3. Resultados del tercer objetivo específico

4.3.1. Nomenclatura y Logotipo

Para el nombre de la aplicación se optó por utilizar la nomenclatura *PropTech Hub*, que deriva de la combinación de los términos en inglés *Property Technology* (tecnología inmobiliaria) y *Hub* (centro) que hace alusión a la centralización digital de operaciones inmobiliarias que permite el proyecto, y su logotipo se visualiza en la figura 9.

Figura 9. Logotipo de la aplicación web

Nota. Imagen generada con Google Gemini (2026). Esta imagen se generó en respuesta a la siguiente solicitud: «Emblema circular con borde naranja. Al centro, un ícono naranja de una casa formada por circuitos. Debajo, el texto 'PropTech Hub' y el eslogan 'Soluciones Inmobiliarias Inteligentes' centrados en color negro».

4.3.2. Marco de trabajo Scrum

Para el desarrollo de la aplicación *web* con *machine learning* de la empresa Escudero del cantón Santo Domingo, se utilizó el marco de trabajo ágil *Scrum*, dado su enfoque iterativo e incremental, ya que permite organizar el trabajo en períodos de tiempo corto denominados *sprints*. Los cuales generan un incremento funcional que hace posible la aprobación o la retroalimentación temprana y adaptación a posibles cambios de requerimientos de las funcionalidades requeridas del producto. Además, *Scrum* permitió la priorización de las tareas que aportan mayor valor a la organización, así como también mantener una comunicación constante entre los miembros del equipo para detectar y afrontar inconvenientes de manera eficiente y rápida.

4.3.3. Sprint 1

4.3.3.1. Planificación del Sprint I – Sprint Planning

Como primer paso importante en el desarrollo, es la planificación en la que se determinó las funcionalidades más importantes del *product backlog* y se llevó a cabo en el *sprint*, entendiéndose a este último como aquel período de tiempo de 4 semanas o incluso menos. Dado los requerimientos del cliente y limitaciones de tiempo, el presente proyecto se lo planificó en 4 *sprints*.

4.3.3.1.1. Roles

Acorde al marco de trabajo ágil *Scrum*, se establecieron los roles que desempeñaron cada persona implicada en la consecución del desarrollo de la aplicación como se aprecia en la tabla 15. En primer lugar, se tiene el *Product Owner* quien es el representante del negocio que define y prioriza el *Product Backlog*. Adicionalmente, se encuentra el *Scrum Master*, quien se encarga de llevar de mejor manera los lineamientos de *Scrum*. Finalmente, está el *Development Team* que, se encargó de crear y entregar los incrementos funcionales al finalizar cada *sprint.*

Tabla 15. Distribución de Roles

Persona	Roles	Departamento
Jhery Alarcon	Development Team	Desarrollador Frontend y Backend
Romario Flores	Development Team	Desarrollador Frontend y Backend
Cristian Escudero	Propietario (Product Owner)	Gerente de la Empresa Escudero
Mg. Willian Ocampo	Scrum Master	Docente de la PUCESD

4.3.3.1.2. Arquitectura por capas

Para el desarrollo de la aplicación, se empleó el estilo arquitectónico denominado arquitectura por capas o también conocido como arquitectura en *N* capas, el cual se aplica en la gran mayoría de aplicaciones empresariales, al permitir separar las responsabilidades de trabajo acorde a los dominios técnicos que requiere la solución.

Asimismo, en la arquitectura por capas, los componentes están organizados en capas horizontales, donde cada una realiza un rol específico dentro de la aplicación, sus cuatro capas típicas son:

- **Presentación:** maneja toda la lógica de la interfaz de usuario y comunicación con el navegador.
- **Negocio**: ejecuta las reglas de negocio asociadas a la solicitud.
- **Persistencia**: gestiona el acceso a datos (consultas *SQL*, *DAOs*).
- **Base de datos**: almacena los datos del sistema.

Además, dependiendo de la complejidad de la aplicación, se pueden fusionar las capas de negocio y persistencia en una sola, quedando como total tres capas (Richards, 2022, pp. 15-16).

4.3.3.1.3. Parametrización

Con el fin de garantizar legibilidad, orden y colaboración en el código fuente de la aplicación, se establecieron estándares de nomenclatura acordados por el *Development Team*, los cuales se detallan en la tabla 16.

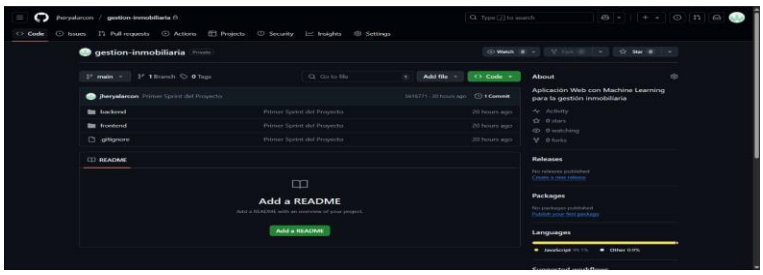
Tabla 16. Parametrización

Capa de datos	Capa de lógica de negocio	Capa de presentación
Modelos (Prisma): Primera letra de cada palabra en mayúscula, por ejemplo: *Usuario* *Propiedad* *ArchivoNegociacion* Columnas de tablas: Nombre en minúscula con guion bajo, por ejemplo: *tipo_propiedad* *estado_publicacion* Enumeraciones (*enums*): Primera letra de cada palabra en mayúscula para el nombre del tipo; valores en minúscula o mayúscula: *enum Rol {cliente, agente, admin}* *enum TipoInteraccion {VISTA, FAVORITO, CONTACTO}* Idioma: español.	Controladores: Primera palabra en minúscula, las siguientes con primera letra en mayúscula, seguido del sufijo *controller.js*, por ejemplo: *propiedad.controller.js* *negociacion.controller.js* Rutas: Primera palabra en minúscula, seguido del sufijo *routes.js*, por ejemplo: *propiedad.routes.js* *negociacion.routes.js* *Middlewares*: Primera palabra en minúscula, las siguientes con primera letra en mayúscula, seguido del sufijo *.js*, por ejemplo: *esAdmin.js* *esAgenteOAdmin.js* Funciones: Primera palabra en minúscula, las siguientes con primera letra en mayúscula, por ejemplo: *crearPropiedad()* *obtenerPropiedadPorId()* Idioma: español.	Páginas: Primera letra de cada palabra en mayúscula, seguido del sufijo *jsx*, por ejemplo: *PanelNegociaciones.jsx* *RegistrarPropiedad.jsx* Componentes: Primera letra de cada palabra en mayúscula, seguido del sufijo*.jsx*, por ejemplo: *RutaPrivada.jsx* *CardPropiedad.jsx* Funciones: Primera palabra en minúscula, las siguientes con primera letra en mayúscula, por ejemplo: *verificarToken()* *obtenerUsuario()* Idioma: español.

4.3.3.1.4. Control de versiones

Para manejar de forma eficiente las diferentes versiones del proyecto, se decidió utilizar *GitHub*, una plataforma en línea que permite alojar, gestionar y colaborar en proyectos de software, usando el sistema de control de versiones *Git*, como se visualiza en la figura 10.

Figura 10. Repositorio de GitHub del proyecto



4.3.3.1.5. Product Backlog

El *Product Backlog* es una lista de todos los requerimientos del sistema, expresados comúnmente mediante historias de usuario, las cuales fueron definidas y priorizadas por medio de reuniones con el *Product Owner*. En las tablas 17 y 18, se pueden observar las funcionalidades que conlleva el proyecto, así como la estimación en complejidad por puntos de historia, la prioridad que tienen en el negocio y la incertidumbre técnica que suponen para el *Development Team*.

Tabla 17. Product Backlog

Product Backlog				
N.º	Historia de Usuario	Estimación	Prioridad de Negocio	Riesgo en Desarrollo
1	Inicio de sesión con roles	5	100	Alto
2	Cierre de sesión	1	98	Bajo
3	Registro de cuenta y verificación (cliente externo)	3	96	Bajo
4	Registro de propiedades	5	94	Medio
5	Visualización de propiedades	3	92	Bajo
6	Edición de propiedades	5	91	Medio
7	Actualización del estado de publicación de propiedades	3	90	Medio
8	Desactivación de propiedades	3	89	Medio
9	Búsqueda y filtrado de propiedades en el portal público	3	88	Bajo
10	Visualización del detalle de la propiedad	3	87	Bajo

11	Guardar propiedad como favorita	3	86	Medio
12	Registro de clientes internos	2	85	Medio
13	Visualización de clientes registrados	1	84	Bajo
14	Edición de clientes registrados	2	83	Bajo
15	Desactivación de clientes registrados	2	82	Bajo
16	Registro de negociación	3	81	Medio
17	Actualización de etapa de una negociación	3	80	Medio
18	Historial de seguimiento por negociación	3	79	Medio
19	Notas internas por negociación	3	78	Medio
20	Adjuntar archivos por negociación	3	77	Medio
21	Registro de agentes	3	75	Medio
22	Visualización de agentes	2	74	Bajo
23	Edición de agentes	2	73	Bajo
24	Desactivación de agentes	2	72	Bajo
25	Recomendación de propiedades personalizada	8	65	Alto
26	Recuperación de contraseña de la cuenta	5	64	Medio
27	Detalle de propiedad (panel administrativo)	3	71	Medio

Tabla 18. Product Backlog Técnico

Product Backlog Técnico				
N.º	Historia de Usuario	Estimación	Prioridad de Negocio	Riesgo en Desarrollo
1	Autenticación y Seguridad	8	100	Alto

4.3.3.1.6. Estimación

Para la estimación de la complejidad de las historias de usuario, se empleó la métrica de puntos de historia, con los números presentes en la secuencia de Fibonacci y consensuados con el *Devolopment Team* a través de la técnica de *Planning Poker*. Por otro lado, se muestra el calendario de trabajo del equipo de desarrollo reflejado en la tabla 19.

Tabla 19. Calendario de trabajo

Calendario de Trabajo			
Mes	Semanas	Días	Horas
1	4	5	4

4.3.3.1.7. Velocidad de desarrollo

La velocidad de desarrollo en *Scrum,* hace alusión a la cantidad de trabajo que puede hacer el equipo de desarrollo por *sprint*. Entonces en el primer *sprint*, la primera

historia tiene un valor de 5 puntos, la segunda historia se estima en 1 punto, la tercera en 3 puntos, la cuarta en 5 puntos, la quinta en 3 puntos y finalmente la sexta en 5 puntos, lo que da un total de 22 puntos de historias en el sprint 1, que equivale a la velocidad de desarrollo.

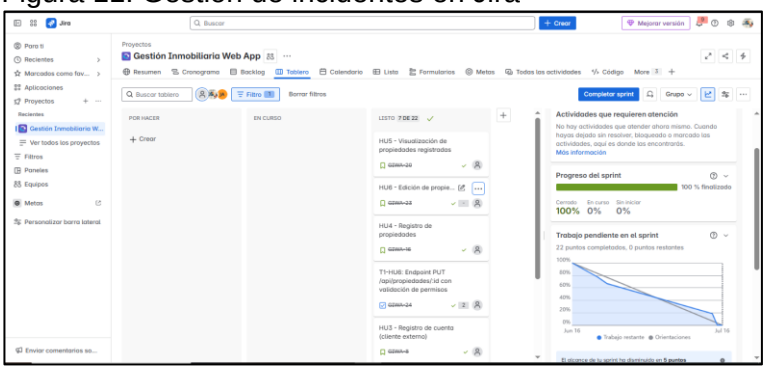
4.3.3.1.8. Escenario de Prueba

Una vez culminadas las historias de usuario (véase anexo IV), se verifica si el sistema cumple con todas las funcionalidades de la aplicación a través de los escenarios de prueba. Por lo tanto, en el anexo V están adjuntas las pruebas de aceptación, que con los escenarios de prueba manejan un lenguaje comprensible para todos los miembros del equipo, incluyendo los no técnicos (Dueño del producto).

4.3.3.1.9. Gestión de Incidentes

En este apartado, se evidencia el avance de las historias de usuarios y sus respectivas tareas a través de la plataforma *Jira*, herramienta que permitió organizar y asignar al equipo de desarrollo que historias tuvieron que terminar (figura 11). Adicional a esto, *Jira* está estructurada en las siguientes secciones: “Por hacer”, donde se visualizan las historias a realizar o pendientes en ese momento; “En curso”, donde se encuentran las historias que se están realizando, y finalmente, el apartado de “Listo”, donde se incorporan las tareas que se han concluido.

Figura 11. Gestión de incidentes en Jira



4.3.3.1.10. Sprint Backlog

Acorde a la tabla 20, para efecto del primer *sprint,* se eligieron las 6 primeras historias de usuario del *Product Backlog* sumando la cantidad de 22 puntos de historia.

También, se definieron las tareas correspondientes junto con su estimación, las cuales se asignaron de manera proporcional a la complejidad de la historia de usuario de la cual se derivan. Además, se identificó al desarrollador responsable de llevar a cabo cada una de estas tareas.

Tabla 20. Sprint Backlog 1

Historia Usuario	Est. His.	Categ	Tarea	Est. Tar.	Responsable	Estado
HU1 – Inicio de sesión con roles	5 pts	Base de Datos	Creación del modelo Usuario. (schema.prisma)	1 pt	Jhery Alarcon	Completado
		Backend	Implementación de endpoint /api/auth/login con validación de credenciales, verificación de usuario activo y generación de JWT. (auth.controller.js, auth.routes.js)	1.5 pts	Romario Flores	Completado
		Backend	Creación de middlewares de autenticación y autorización. (verificarToken.js, esAdmin.js, esAgenteOAdmin.js)	1 pt	Romario Flores	Completado
		Frontend	Elaboración de formulario de login con validaciones, autenticación, redirección según rol del usuario (Login.jsx)	1 pt	Jhery Alarcon	Completado
		Frontend	Creación de componente para proteger rutas según rol. (RutaPrivada.jsx)	0.5 pts	Romario Flores	Completado
HU2 – Cierre de sesión	1 pt	Frontend	Elaboración de componente, eliminación de token del almacenamiento y redirección a la página de inicio. (BotonLogout.jsx)	1 pt	Jhery Alarcon	Completado
HU3 – Registro de cuenta y verificación (cliente externo)	3 pts	Backend	Implementación de endpoint POST /api/auth/register con validación de campos, encriptación de contraseña, creación de usuario con rol cliente, generación de token de verificación y envío de correo; el JWT de sesión se obtiene luego vía login. (auth.controller.js, auth.routes.js)	1.5 pts	Jhery Alarcon	Completado
		Frontend	Creación de formulario para registro de clientes, validación de datos, envío de petición de registro y manejo del mensaje de verificación con redirección al inicio de sesión. (Registro.jsx)	1.5 pts	Romario Flores	Completado

	5 pts	Base de Datos	Creación del modelo Propiedad y definición del modelo Imagen. (schema.prisma)	1 pt	Jhery Alarcon	Completado
		Backend	Configuración de multer para manejo de archivos y gestión de imágenes. (multer.js, images.js)	1 pt	Jhery Alarcon	Completado
HU4 – Registro de propiedades		Backend	Implementación de endpoint POST /api/propiedades con validaciones, guardado de imágenes y registro en la base de datos. (propiedad.controller.js, propiedad.routes.js)	1 pt	Jhery Alarcon	Completado
		Frontend	Creación de formulario para registro de propiedades con vista previa de imágenes. (RegistrarPropiedad.jsx, DocumentManager.jsx)	2 pts	Jhery Alarcon	Completado
HU5 – Visualización de propiedades registradas	3 pts	Backend	Implementación de endpoint GET /api/propiedades para mostrar propiedades. (propiedad.controller.js, propiedad.routes.js)	1 pt	Romario Flores	Completado
		Frontend	Elaboración de la vista y filtros de búsqueda de propiedades para los agentes y administradores. (PanelPropiedades.jsx, CardPropiedad.jsx, FiltroPropiedadesAdmin.jsx)	2 pts	Romario Flores	Completado
HU6 – Edición de propiedades	5 pts	Backend	Implementación de endpoint PUT /api/propiedades/:id con validación de permisos. (propiedad.controller.js, propiedad.routes.js)	2 pts	Romario Flores	Completado
		Frontend	Elaboración de formulario de edición pre-cargado con datos existentes de la propiedad a modificar. (EditarPropiedad.jsx)	2 pts	Jhery Alarcon	Completado
		Backend	Creación de middleware para evitar accesos no autorizados. (esPropietarioOAdmin.js)	1 pt	Jhery Alarcon	Completado

4.3.3.2. *Sprint* 1 – Reuniones diarias

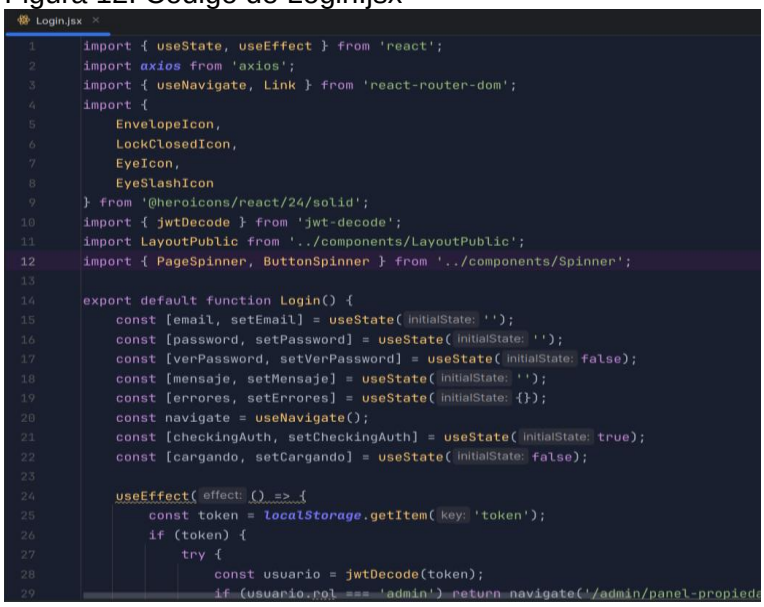
Para la consecución del *sprint* 1, se realizaron reuniones diarias breves, con un tiempo máximo de 15 minutos, donde se expusieron los avances de las tareas, las dificultades encontradas, mientras se culminaban las mismas y la planeación de las

siguientes. Adicional a esto, se utilizó la herramienta *Jira* para llevar un seguimiento de las actividades pendientes y realizadas.

4.3.3.2.1. Historia de Usuario 1: Inicio de Sesión con roles

Se creó la página de inicio de sesión *Login.jsx* (figura 12) usando *React* y *Tailwind CSS* para el diseño de la interfaz. El *login* permite a los usuarios autenticarse al sistema mediante su correo electrónico y contraseña, incluyendo validación de los campos en tiempo real y un botón para mostrar u ocultar la contraseña.

Figura 12. Código de Login.jsx

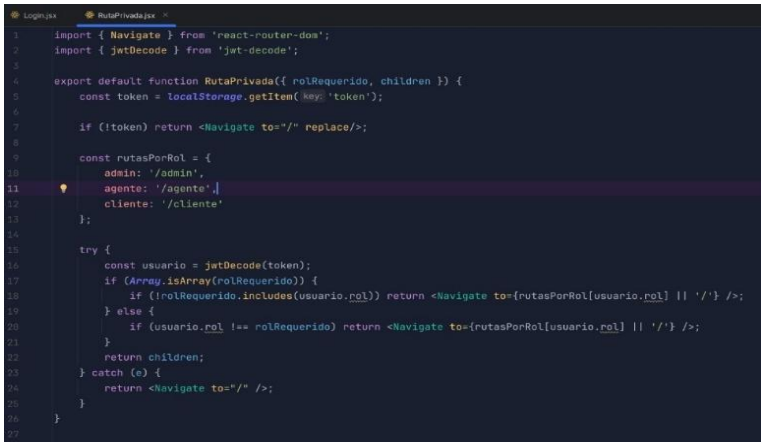


El componente *Login.jsx* incluye *useEffect* que verifica si existe un *token* válido en el *localStorage*, el cual es un espacio de almacenamiento que permite guardar datos en el navegador. De existir, se decodifica dicho *token* usando *jwt-decode* para obtener el rol del usuario autenticado y redirigirlo a su página de inicio correspondiente: administradores a */admin*, agentes inmobiliarios a */agente*, y clientes a /. Al enviar el formulario presente del *login*, la función *handleSubmit* valida los campos si cumplen las condiciones establecidas, realiza una petición a la ruta *POST /api/auth/login* al *backend* mediante *Axios*. Y por último, si resulta ser exitosa guarda el *token JWT* y la información del usuario en el *localStorage* del navegador, y redirige según el rol respectivo.

Asimismo, para proteger las rutas del *frontend*, se diseñó *RutaPrivada.jsx* (figura 13) que valida el *token JWT* y el rol del usuario a través de la función *verificarToken* ubicada en

el archivo *tokenUtils.js*. Si el usuario no está autenticado o no tiene el rol requerido, lo redirige a la página de inicio de sesión o la ruta por defecto acorde al rol.

Figura 13. Código de RutaPrivada.jsx

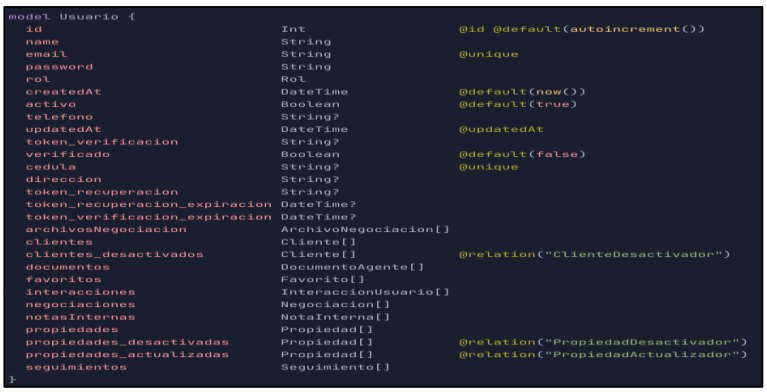


```jsx
import { Navigate } from 'react-router-dom';
import { jwtDecode } from 'jwt-decode';

export default function RutaPrivada({ rolRequerido, children }) {
    const token = localStorage.getItem( key: 'token');

    if (!token) return <Navigate to="/" replace/>;

    const rutasPorRol = {
        admin: '/admin',
        agente: '/agente',
        cliente: '/cliente'
    };

    try {
        const usuario = jwtDecode(token);
        if (Array.isArray(rolRequerido)) {
            if (!rolRequerido.includes(usuario.rol)) return <Navigate to={rutasPorRol[usuario.rol] || '/'} />;
        } else {
            if (usuario.rol !== rolRequerido) return <Navigate to={rutasPorRol[usuario.rol] || '/'} />;
        }
        return children;
    } catch (e) {
        return <Navigate to="/" />;
    }
}
```

Del lado del *backend,* se empleó *Node.js* junto con *Express.js* y *Prisma* como *ORM* para interactuar con el sistema de gestión de base de datos *PostgreSQL*. En cuanto a *Prisma ORM*, se definió el modelo Usuario en *schema.prisma*, tal como se observa en la figura 14, que es el archivo principal de configuración de aquel *ORM* donde se establece la conexión a la base de datos, la configuración del cliente *Prisma*, los modelos que son el símil a las tablas y sus respectivas relaciones.

En dicho modelo, se establecieron campos que suele tener comúnmente los usuarios registrados en sistemas informáticos como nombres, correo electrónico, teléfono, contraseña, así como el rol de tipo *enum,* que es una clase de dato que define un conjunto de valores posibles, en este caso, los roles: cliente, agente y *admin*.

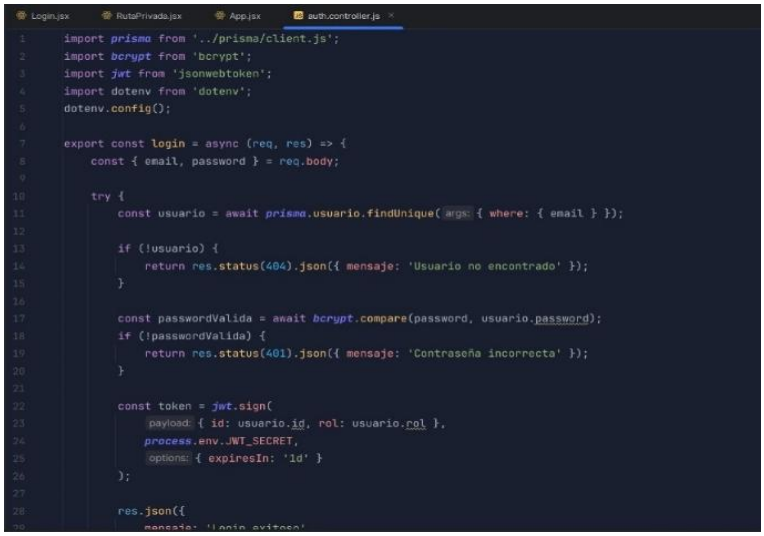
Figura 14. Modelo Usuario en Prisma ORM



```
model Usuario {
  id                          Int                 @id @default(autoincrement())
  name                        String
  email                       String              @unique
  password                    String
  rol                         Rol
  createdAt                   DateTime            @default(now())
  activo                      Boolean             @default(true)
  telefono                    String?
  updatedAt                   DateTime            @updatedAt
  token_verificacion          String?
  verificado                  Boolean             @default(false)
  cedula                      String?             @unique
  direccion                   String?
  token_recuperacion          String?
  token_recuperacion_expiracion DateTime?
  token_verificacion_expiracion DateTime?
  archivosNegociacion         ArchivoNegociacion[]
  clientes                    Cliente[]
  clientes_desactivados       Cliente[]           @relation("ClienteDesactivador")
  documentos                  DocumentoAgente[]
  favoritos                   Favorito[]
  interacciones               InteraccionUsuario[]
  negociaciones               Negociacion[]
  notasInternas               NotaInterna[]
  propiedades                 Propiedad[]
  propiedades_desactivadas    Propiedad[]         @relation("PropiedadDesactivador")
  propiedades_actualizadas    Propiedad[]         @relation("PropiedadActualizador")
  seguimientos                Seguimiento[]
}
```

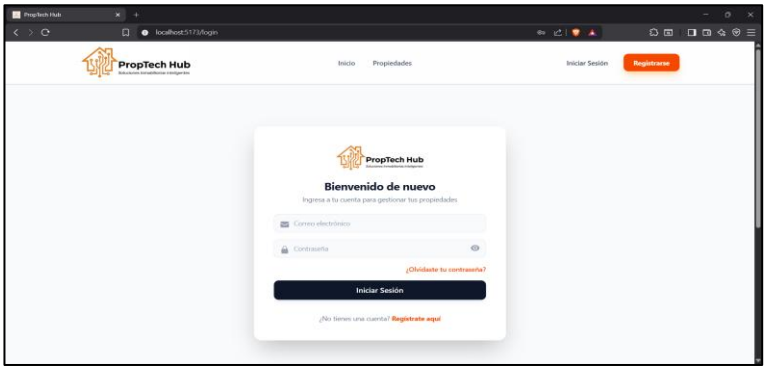
Por otro lado, se creó el archivo *auth.controller.js* con la función *login* (figura 15) que recibe desde el cuerpo de la petición el email y la contraseña. La función *login* busca el usuario por correo electrónico utilizando *prisma.usuario.findUnique*, y verifica que el usuario exista, la contraseña esté correcta, el usuario se encuentre activo. Y por último, cuando pasan todas las validaciones genera un token *JWT* con datos del usuario sin incluir la contraseña.

Figura 15. Función de inicio de sesión



Se definió la ruta *POST /api/auth/login* en *auth.routes.js*, que llama a la función *login* del controlador *auth.controller.js* sin *middlewares* de autenticación, debido a que es un *endpoint* público. Además, para proteger las rutas de lado del servidor se integraron *middlewares* de autenticación y autorización como *verificarToken.js*, *esAdmin.js* y *esAgenteOAdmin.js*. En la figura 16 se puede apreciar la interfaz de inicio de sesión en la plataforma.

Figura 16. Interfaz de Inicio de sesión



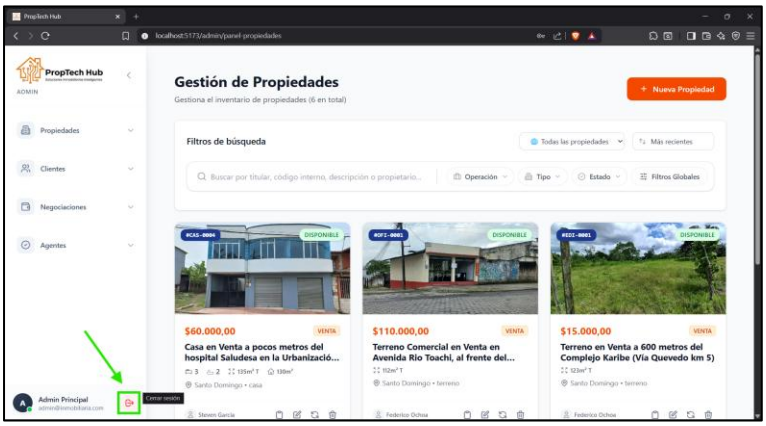
4.3.3.2.2. Historia de Usuario 2: Cierre de Sesión

Para el desarrollo de esta funcionalidad, no se acudió a implementar la lógica de *backend*, ya que el sistema utiliza *JWT stateless* (sin estado) por lo que no requiere comunicación con el servidor. Mediante *React,* se diseñó el componente reutilizable *BotonLogout.jsx* (figura 17) el cual se integra en las páginas *PanelAdmin.jsx* y *PanelAgente.jsx* para permitir cerrar sesión desde los diferentes paneles administrativos.

Figura 17. Código de BotonLogout.jsx

El componente *BotonLogout.jsx* utiliza la función *cerrarSesion()* que hace uso de *useNavigate* de *React Router*. La lógica central de esta función reside en eliminar el *token JWT*, los datos del usuario del *localStorage* usando *localStorage.removeItem* que es una característica propia del navegador y redirigir a la página principal del sistema. En la figura 18, se evidencia el botón para cerrar sesión desde una cuenta administrativa.

Figura 18. Botón para cerrar sesión



4.3.3.2.3. Historia de Usuario 3: Registro de cuenta y verificación

Para el cliente externo, a través de *React,* se creó la página *Registro.jsx* (figura 19), la cual permite a usuarios no autenticados crear una cuenta con el rol de cliente. Dicha página incluye campos para ingresar información del cliente como nombres, correo electrónico, teléfono y contraseña, incluyendo validación en tiempo real, tal como se muestra en la interfaz del formulario de registro de una cuenta, reflejado en la figura 20.

Figura 19. Código de Registro.jsx

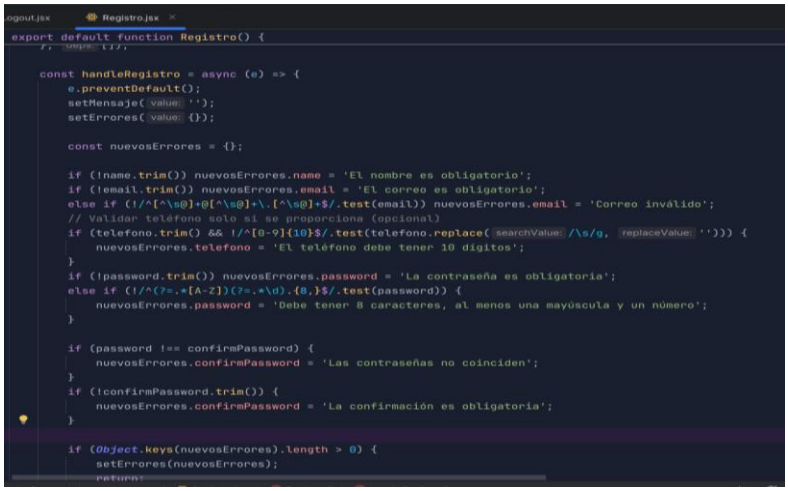
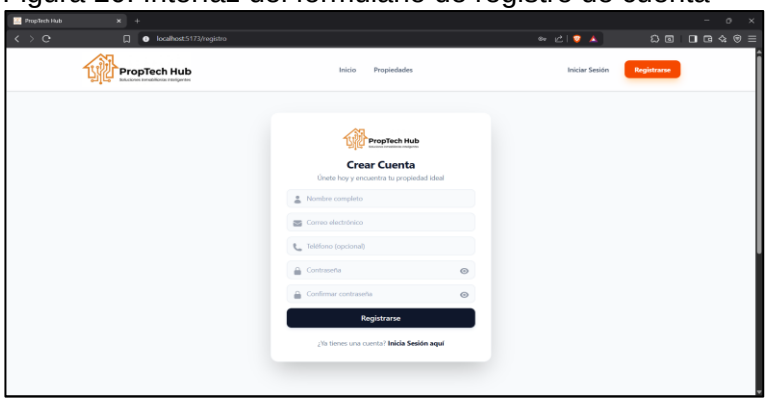


Figura 20. Interfaz del formulario de registro de cuenta

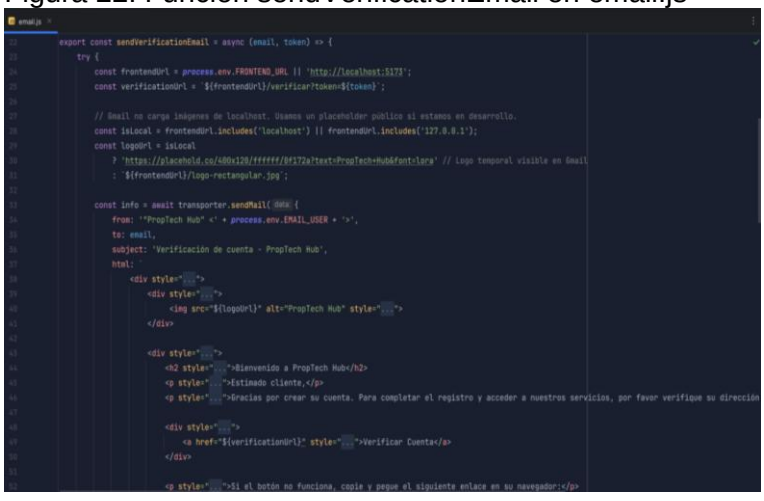


Al enviar los datos del formulario de registro, se ejecuta la función *handleRegistro* que valida los campos y envía la petición *POST /api/auth/register* mediante *Axios*. De ser exitosa la solicitud, se muestra un mensaje indicando que se envió un enlace de verificación al correo y se redirige a la interfaz de inicio de sesión.

Del lado del *backend*, la ruta *POST /api/auth/register* está definida en *auth.routes.js* como *endpoint* público. La función *register* de *auth.controller.js* emplea *bcrypt* para encriptar la contraseña, genera un token de verificación único mediante la librería *crypto* con una expiración de 24 horas, y almacena ambos campos (*token_verificacion* y *token_verificacion_expiracion*) en el registro del usuario creado. Al finalizar, encarga el envío del correo de verificación a la función *sendVerificationEmail* de *email.js*, que utiliza la biblioteca de *nodemailer* con *Gmail* para entregar el enlace de activación, tal como se muestra en las figuras 21 y 22.

Figura 21. Función register en auth.controller.js

Figura 22. Función sendVerificationEmail en email.js



Una vez que el usuario pulsa el enlace recibido, se carga la página *VerificarCuenta.jsx* (figura 23) que extrae el *token* desde la *URL* y realiza una petición *POST /api/auth/verify*. La función *verifyEmail* de *auth.controller.js* valida la existencia y

vigencia del *token*; si es válido, marca la cuenta como verificada (campo verificado = *true*) y limpia el *token*; si ha expirado o es inválido, retorna el mensaje de error correspondiente. La aplicación muestra el resultado visualmente con un ícono de éxito o error, en caso exitoso (figura 24), redirige automáticamente al inicio de sesión tras pocos segundos.

Figura 23. Código de VerificarCuenta.jsx

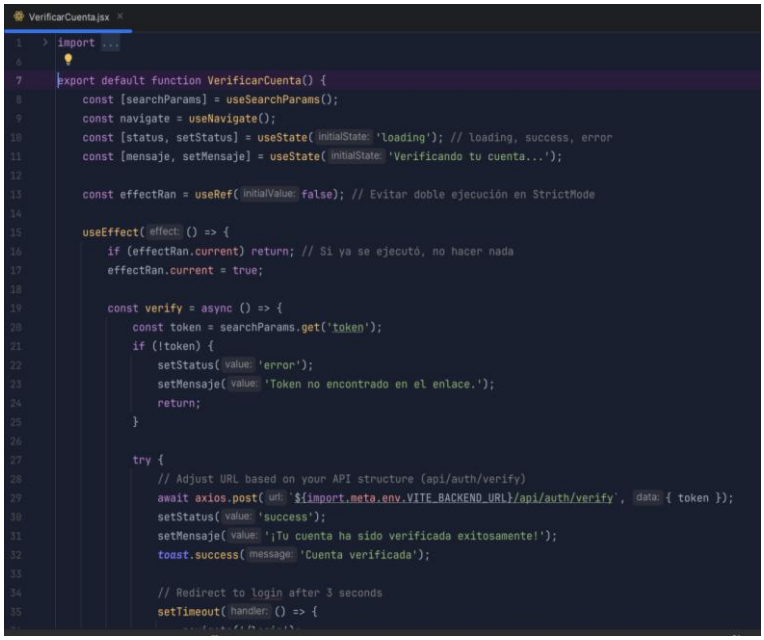
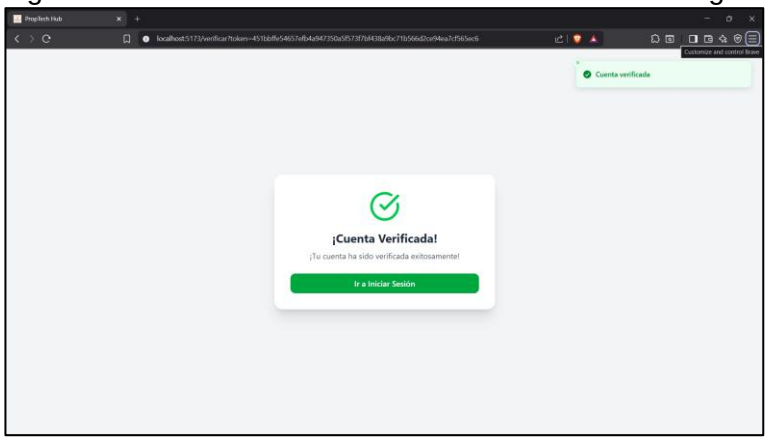


Figura 24. Interfaz de verificación exitosa de cuenta registrada

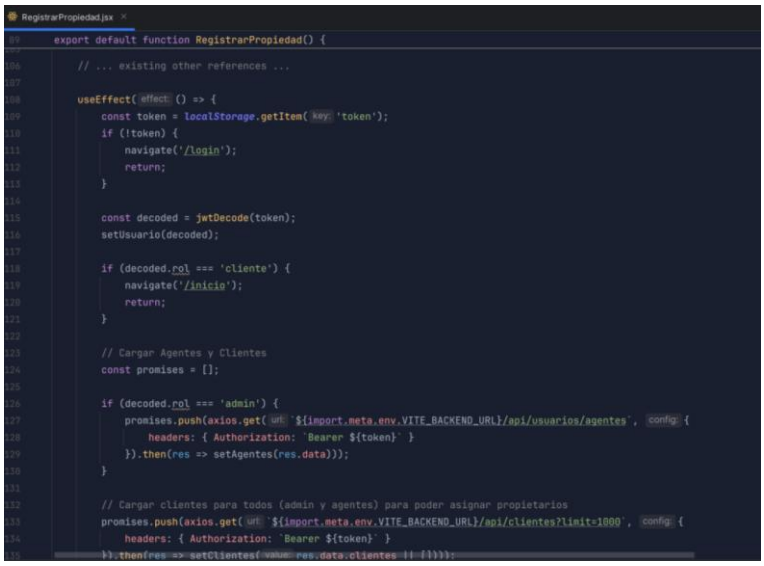


4.3.3.2.4. Historia de usuario 4: Registro de propiedades

Mediante *React,* se creó la página *RegistrarPropiedad.jsx* (figura 25) la cual permite a administradores y agentes inmobiliarios registrar nuevas propiedades mediante un formulario dividido en cinco pasos: identificación, ubicación, características, multimedia y negocio, tal como se muestra en la figura 26. Cada paso incluye su propia validación, si

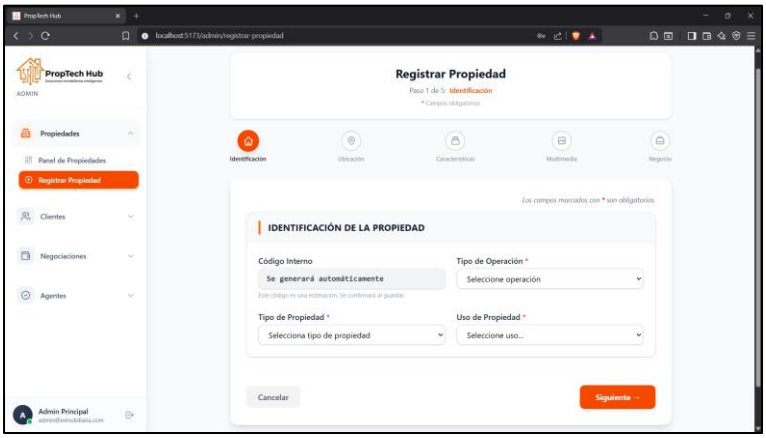
existen campos obligatorios incompletos o inválidos, se muestran mensajes de error y no se permite avanzar al siguiente paso. El botón "Anterior" permite retroceder sin perder los datos ingresados.

Figura 25. Código de RegistrarPropiedad.jsx



En el Paso 1 (Identificación), se capturan el tipo, uso y transacción de la propiedad. El Paso 2 (Ubicación) recoge la provincia, ciudad, sector, dirección y coordenadas geográficas mediante un mapa interactivo. El Paso 3 (Características) registra el estado físico, área y amenidades. El Paso 4 (Multimedia) requiere la subida de imágenes (mínimo 1, máximo 15) en formato JPG, PNG o WEBP.

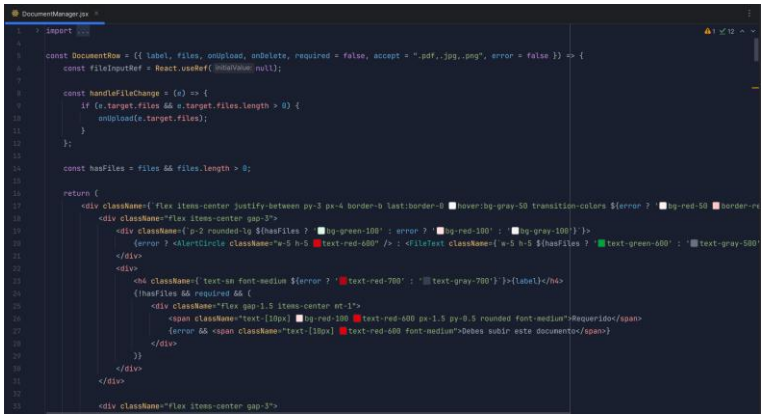
Figura 26. Formulario de registro de propiedad



El Paso 5 (Negocio), destinado a información comercial, captura precio, comisión, tipo y fecha de contrato, propietarios con sus cuotas de participación y la documentación

legal y técnica mediante el componente *DocumentManager.jsx* (figura 27). Si el tipo de contrato es exclusividad, el documento Contrato de Exclusividad es obligatorio; en caso contrario, si se elige Contrato de Autorización se exige la Autorización de Venta. El administrador debe además seleccionar un agente responsable en este paso.

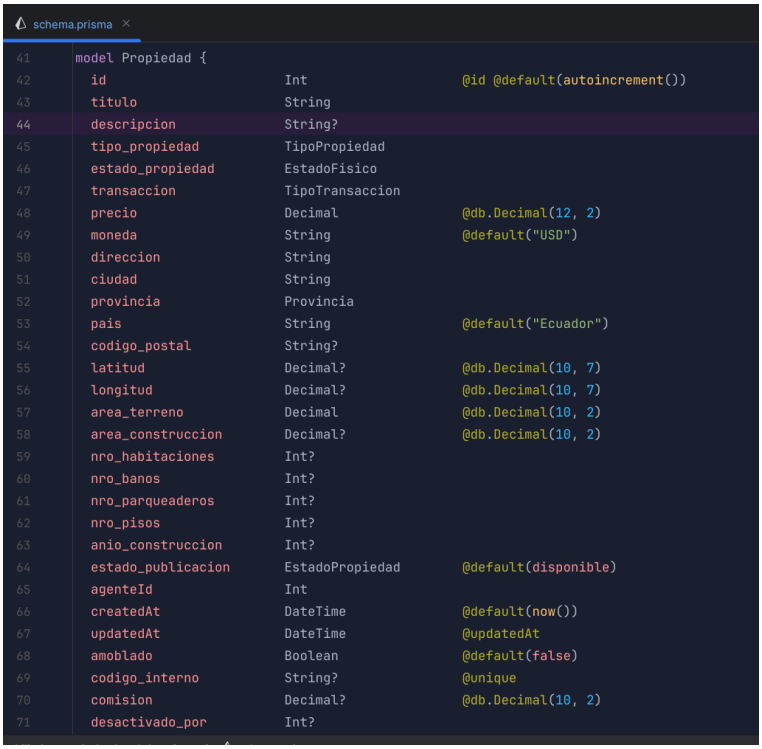
Figura 27. Código de DocumentManager.jsx



Al pulsar el botón "Registrar Propiedad", la función *handleSubmit* construye un objeto *FormData* con los datos e imágenes y ejecuta el *endpoint POST /api/propiedades* mediante *Axios*. Una vez creada la propiedad, los documentos adjuntos se suben en paralelo mediante *POST /api/documentos/propiedad/:propiedadId*, clasificados por tipo y categoría (legal, comercial, técnico, etc.). De ser exitoso, se muestra un *toast* de éxito y redirige al panel de propiedades según el rol del usuario.

Se declararon los modelos Propiedad e Imagen en *schema.prisma,* con sus respectivos campos y relaciones (figuras 28 y 29). La lógica del *backend,* se centralizó en la función *crearPropiedad* del controlador *propiedad.controller.js* (figura 30). El *endpoint POST /api/propiedades* está definido en *propiedad.routes.js* con el *middleware verificarToken* y el middleware *upload.array* de *multer.js* para procesar las imágenes, con un límite de 10 MB por archivo. Adicionalmente, se configuró el *helper* de rutas en *images.js* para la conversión automática de *URLs* absolutas de las imágenes alojadas.

Figura 288. Modelo Propiedad en schema.prisma.

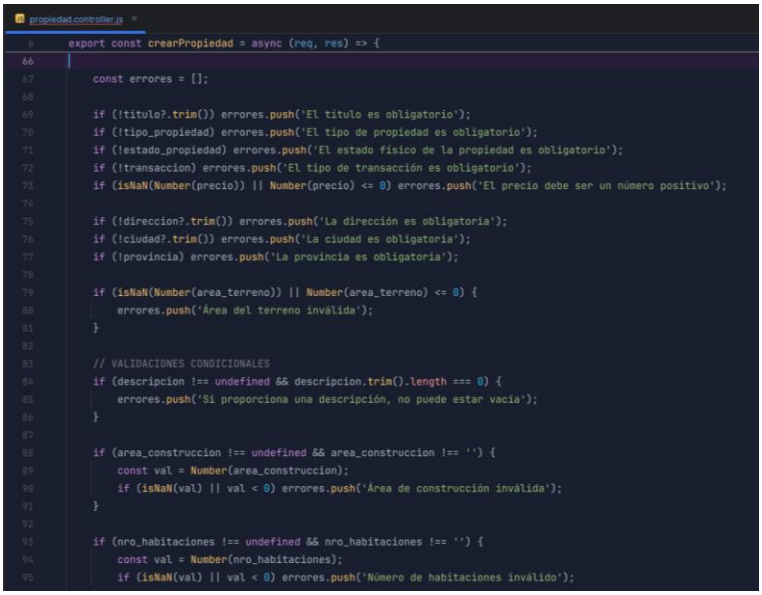


```
schema.prisma
model Propiedad {
  id                    Int                 @id @default(autoincrement())
  titulo                String
  descripcion           String?
  tipo_propiedad        TipoPropiedad
  estado_propiedad      EstadoFisico
  transaccion           TipoTransaccion
  precio                Decimal             @db.Decimal(12, 2)
  moneda                String              @default("USD")
  direccion             String
  ciudad                String
  provincia             Provincia
  pais                  String              @default("Ecuador")
  codigo_postal         String?
  latitud               Decimal?            @db.Decimal(10, 7)
  longitud              Decimal?            @db.Decimal(10, 7)
  area_terreno          Decimal             @db.Decimal(10, 2)
  area_construccion     Decimal?            @db.Decimal(10, 2)
  nro_habitaciones      Int?
  nro_banos             Int?
  nro_parqueaderos      Int?
  nro_pisos             Int?
  anio_construccion     Int?
  estado_publicacion    EstadoPropiedad     @default(disponible)
  agenteId              Int
  createdAt             DateTime            @default(now())
  updatedAt             DateTime            @updatedAt
  amoblado              Boolean             @default(false)
  codigo_interno        String?             @unique
  comision              Decimal?            @db.Decimal(10, 2)
  desactivado_por       Int?
```

Figura 29. Modelo Imagen en schema.prisma.

```
model Imagen {
  id           Int       @id @default(autoincrement())
  url          String
  propiedadId  Int
  propiedad    Propiedad @relation(fields: [propiedadId], references: [id])
}
```

Figura 30. Función crearPropiedad en propiedad.controller.js

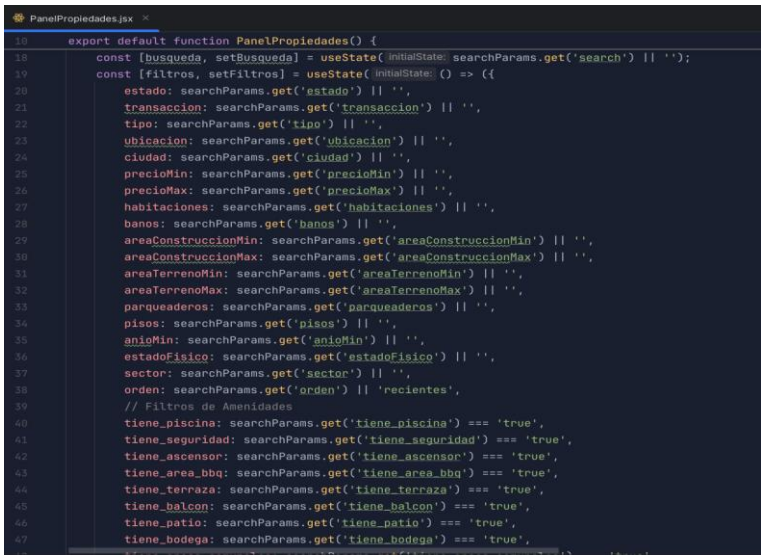


```
propiedad.controller.js
export const crearPropiedad = async (req, res) => {

    const errores = [];

    if (!titulo?.trim()) errores.push('El titulo es obligatorio');
    if (!tipo_propiedad) errores.push('El tipo de propiedad es obligatorio');
    if (!estado_propiedad) errores.push('El estado fisico de la propiedad es obligatorio');
    if (!transaccion) errores.push('El tipo de transacción es obligatorio');
    if (isNaN(Number(precio)) || Number(precio) <= 0) errores.push('El precio debe ser un número positivo');

    if (!direccion?.trim()) errores.push('La dirección es obligatoria');
    if (!ciudad?.trim()) errores.push('La ciudad es obligatoria');
    if (!provincia) errores.push('La provincia es obligatoria');

    if (isNaN(Number(area_terreno)) || Number(area_terreno) <= 0) {
        errores.push('Área del terreno inválida');
    }

    // VALIDACIONES CONDICIONALES
    if (descripcion !== undefined && descripcion.trim().length === 0) {
        errores.push('Si proporciona una descripción, no puede estar vacía');
    }

    if (area_construccion !== undefined && area_construccion !== '') {
        const val = Number(area_construccion);
        if (isNaN(val) || val < 0) errores.push('Área de construcción inválida');
    }

    if (nro_habitaciones !== undefined && nro_habitaciones !== '') {
        const val = Number(nro_habitaciones);
        if (isNaN(val) || val < 0) errores.push('Número de habitaciones inválido');
```

4.3.3.2.5. Historia de usuario 5: Visualización de propiedades

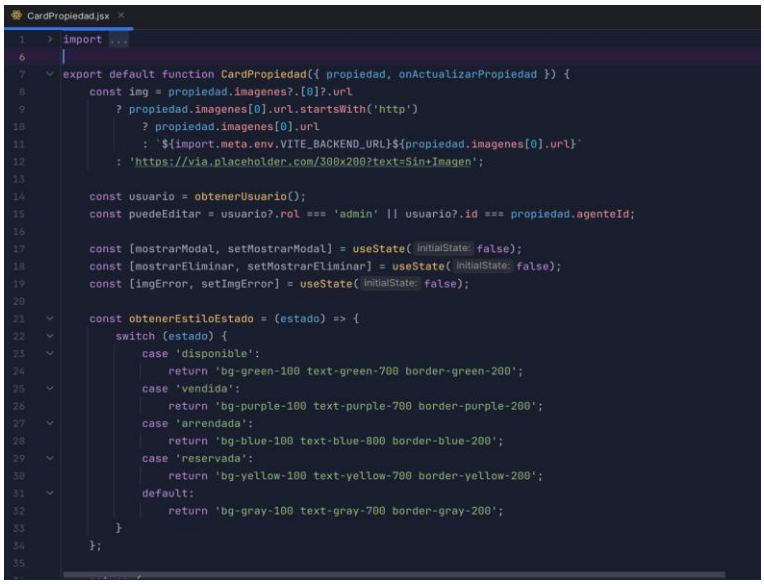
Se desarrolló la página *PanelPropiedades.jsx* (figura 31) con *React* para visualizar las propiedades registradas que gestionan los administradores y agentes inmobiliarios. La interfaz muestra las propiedades en un *grid* de tarjetas mediante el componente reutilizable *CardPropiedad.jsx* (figura 32). Esta incluye imagen principal, código interno, etiqueta de estado con color diferenciado (disponible en verde, reservada en amarillo, vendida en morado, arrendada en azul e inactiva en gris), precio, tipo de transacción, características físicas, ciudad, nombre del agente asignado y botones de acción.

Figura 31. Código de PanelPropiedades.jsx



```
PanelPropiedades.jsx
export default function PanelPropiedades() {
    const [busqueda, setBusqueda] = useState( initialState: searchParams.get('search') || '');
    const [filtros, setFiltros] = useState( initialState: () => ({
        estado: searchParams.get('estado') || '',
        transaccion: searchParams.get('transaccion') || '',
        tipo: searchParams.get('tipo') || '',
        ubicacion: searchParams.get('ubicacion') || '',
        ciudad: searchParams.get('ciudad') || '',
        precioMin: searchParams.get('precioMin') || '',
        precioMax: searchParams.get('precioMax') || '',
        habitaciones: searchParams.get('habitaciones') || '',
        banos: searchParams.get('banos') || '',
        areaConstruccionMin: searchParams.get('areaConstruccionMin') || '',
        areaConstruccionMax: searchParams.get('areaConstruccionMax') || '',
        areaTerrenoMin: searchParams.get('areaTerrenoMin') || '',
        areaTerrenoMax: searchParams.get('areaTerrenoMax') || '',
        parqueaderos: searchParams.get('parqueaderos') || '',
        pisos: searchParams.get('pisos') || '',
        anioMin: searchParams.get('anioMin') || '',
        estadoFisico: searchParams.get('estadoFisico') || '',
        sector: searchParams.get('sector') || '',
        orden: searchParams.get('orden') || 'recientes',
        // Filtros de Amenidades
        tiene_piscina: searchParams.get('tiene_piscina') === 'true',
        tiene_seguridad: searchParams.get('tiene_seguridad') === 'true',
        tiene_ascensor: searchParams.get('tiene_ascensor') === 'true',
        tiene_area_bbq: searchParams.get('tiene_area_bbq') === 'true',
        tiene_terraza: searchParams.get('tiene_terraza') === 'true',
        tiene_balcon: searchParams.get('tiene_balcon') === 'true',
        tiene_patio: searchParams.get('tiene_patio') === 'true',
        tiene_bodega: searchParams.get('tiene_bodega') === 'true',
```

Figura 32. Código de CardPropiedad.jsx



```
CardPropiedad.jsx
import ...

export default function CardPropiedad({ propiedad, onActualizarPropiedad }) {
    const img = propiedad.imagenes?.[0]?.url
        ? propiedad.imagenes[0].url.startsWith('http')
            ? propiedad.imagenes[0].url
            : `${import.meta.env.VITE_BACKEND_URL}${propiedad.imagenes[0].url}`
        : 'https://via.placeholder.com/300x200?text=Sin+Imagen';

    const usuario = obtenerUsuario();
    const puedeEditar = usuario?.rol === 'admin' || usuario?.id === propiedad.agenteId;

    const [mostrarModal, setMostrarModal] = useState( initialState: false);
    const [mostrarEliminar, setMostrarEliminar] = useState( initialState: false);
    const [imgError, setImgError] = useState( initialState: false);

    const obtenerEstiloEstado = (estado) => {
        switch (estado) {
            case 'disponible':
                return 'bg-green-100 text-green-700 border-green-200';
            case 'vendida':
                return 'bg-purple-100 text-purple-700 border-purple-200';
            case 'arrendada':
                return 'bg-blue-100 text-blue-800 border-blue-200';
            case 'reservada':
                return 'bg-yellow-100 text-yellow-700 border-yellow-200';
            default:
                return 'bg-gray-100 text-gray-700 border-gray-200';
        }
    };
```

El componente *FiltroPropiedadesAdmin.jsx* (figura 33) proporciona una barra de búsqueda por texto (título, código interno, descripción o nombre de propietario) y filtros rápidos de operación, tipo y estado. Además, de un modal de filtros globales con opciones adicionales como rango de precio, ubicación, número mínimo de habitaciones, baños y garajes, año de construcción, rangos de área de terreno y construcción, y selección de amenidades. Todos los filtros seleccionados se sincronizan con la *URL* de la página y se envían al *backend* como *query params* mediante el *endpoint GET /api/propiedades*.

Figura 33. Código de FiltroPropiedadesAdmin.jsx

```
ema.prisma    FiltroPropiedadesAdmin.jsx

export default function FiltroPropiedadesAdmin({ filtros, setFiltros, busqueda, setBusqueda }) {
    const [tipos, setTipos] = useState( initialState: []);
    const [transacciones, setTransacciones] = useState( initialState: []);
    const [ciudades, setCiudades] = useState( initialState: []);

    const [activeDropdown, setActiveDropdown] = useState( initialState: null);
    const [showModal, setShowModal] = useState( initialState: false);
    const filterRef = useRef( initialValue: null);

    // Lista de amenidades
    const AMENIDADES = [
        { key: 'tiene_piscina', label: 'Piscina' },
        { key: 'tiene_seguridad', label: 'Seguridad' },
        { key: 'tiene_ascensor', label: 'Ascensor' },
        { key: 'tiene_area_bbq', label: 'Area BBQ' },
        { key: 'tiene_terraza', label: 'Terraza' },
        { key: 'tiene_balcon', label: 'Balcón' },
        { key: 'tiene_patio', label: 'Patio' },
        { key: 'tiene_bodega', label: 'Bodega' },
        { key: 'tiene_areas_comunales', label: 'Áreas Com.' },
        { key: 'tiene_gas_centralizado', label: 'Gas Cent.' },
        { key: 'tiene_cisterna', label: 'Cisterna' },
        { key: 'tiene_lavanderia', label: 'Lavandería' },
        { key: 'amoblado', label: 'Amoblado' }
    ];

    useEffect( effect: () => {
        const cargarDatos = async () => {
            try {
                // CAMBIO: Usar endpoint específico de metadatos administrativos
```

La función *obtenerPropiedades* (figura 34) del controlador *propiedad.controller.js* aplica los filtros recibidos. El *endpoint GET /api/propiedades* está declarado en *propiedad.routes.js* y protegido con el middleware *verificarToken*. Si no hay resultados, se muestra el mensaje un mensaje indicando de que no se encontraron propiedades. En la figura 35 se refleja la interfaz del panel de propiedades.

Figura 34. Función obtenerPropiedades en propiedad.controller.js

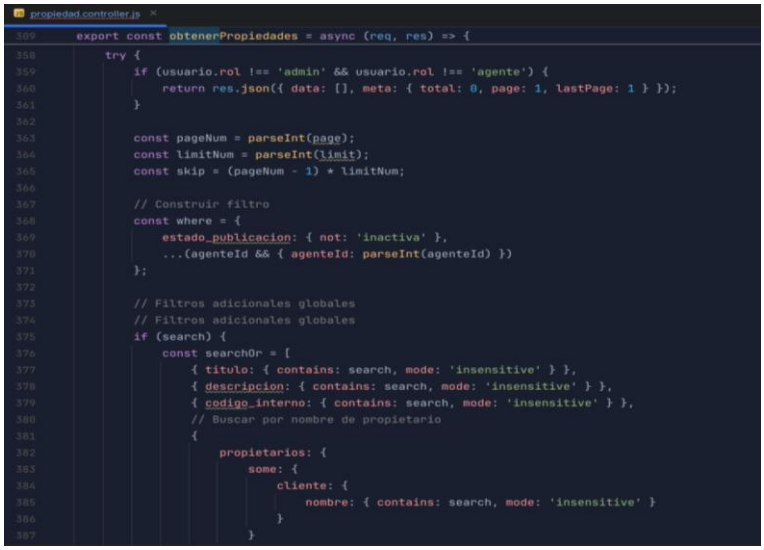
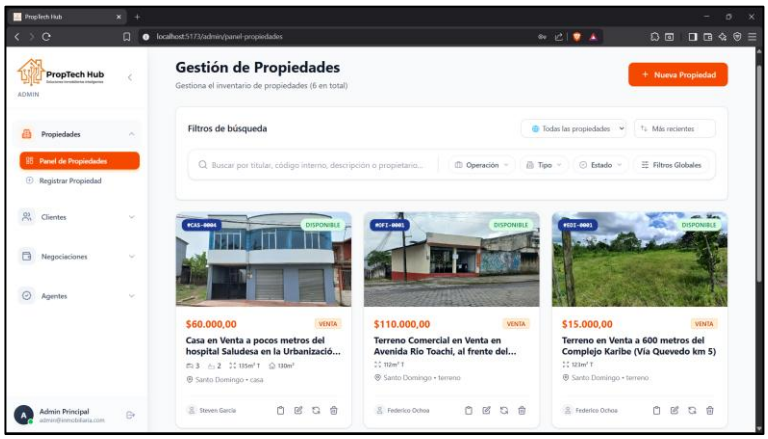


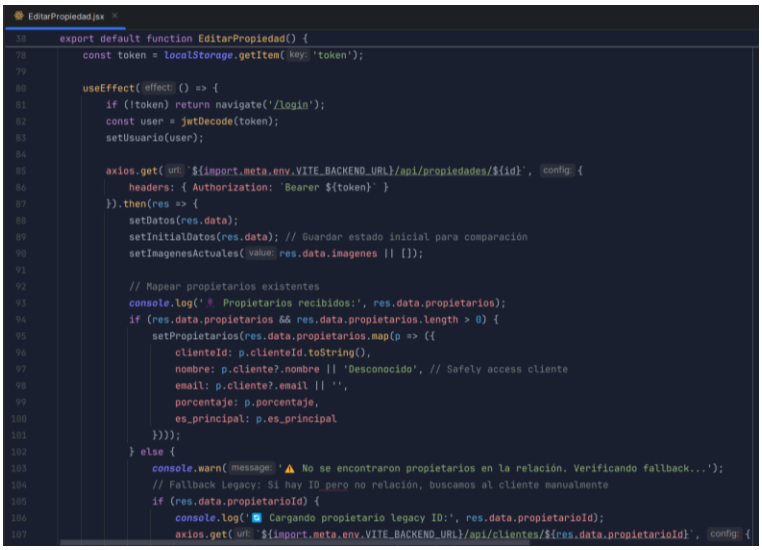
Figura 35. Interfaz del panel de propiedades



4.3.3.2.6. Historia de usuario 6: Edición de propiedades

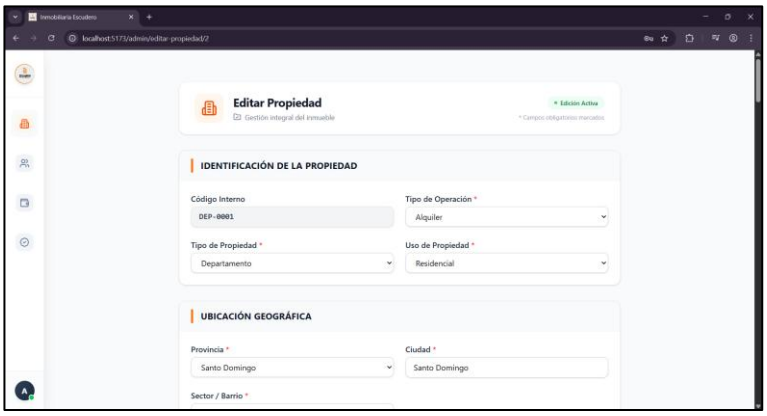
Se creó la página *EditarPropiedad.jsx* (figura 36) con *React* para permitir a administradores y agentes inmobiliarios modificar la información de una propiedad registrada. Al montar el componente, se realiza una petición *GET /api/propiedades/:id* con el *token JWT* para obtener los datos de la propiedad, los cuales se pre-rellenan automáticamente en el formulario, como se refleja en su interfaz de la figura 37. Si un agente intenta acceder a la edición de una propiedad que no le pertenece, se muestra el mensaje de acceso denegado y lo redirige a su panel correspondiente.

Figura 36. Código de EditarPropiedad.jsx



```
export default function EditarPropiedad() {
    const token = localStorage.getItem('token');

    useEffect(() => {
        if (!token) return navigate('/login');
        const user = jwtDecode(token);
        setUsuario(user);

        axios.get(`${import.meta.env.VITE_BACKEND_URL}/api/propiedades/${id}`, {
            headers: { Authorization: `Bearer ${token}` }
        }).then(res => {
            setDatos(res.data);
            setInitialDatos(res.data); // Guardar estado inicial para comparación
            setImagenesActuales(res.data.imagenes || []);

            // Mapear propietarios existentes
            console.log('📍 Propietarios recibidos:', res.data.propietarios);
            if (res.data.propietarios && res.data.propietarios.length > 0) {
                setPropietarios(res.data.propietarios.map(p => ({
                    clienteId: p.clienteId.toString(),
                    nombre: p.cliente?.nombre || 'Desconocido', // Safely access cliente
                    email: p.cliente?.email || '',
                    porcentaje: p.porcentaje,
                    es_principal: p.es_principal
                })));
            } else {
                console.warn('⚠️ No se encontraron propietarios en la relación. Verificando fallback...');
                // Fallback Legacy: Si hay ID pero no relación, buscamos al cliente manualmente
                if (res.data.propietarioId) {
                    console.log('🔄 Cargando propietario legacy ID:', res.data.propietarioId);
                    axios.get(`${import.meta.env.VITE_BACKEND_URL}/api/clientes/${res.data.propietarioId}`, {
```

Figura 37. Formulario de edición de una propiedad



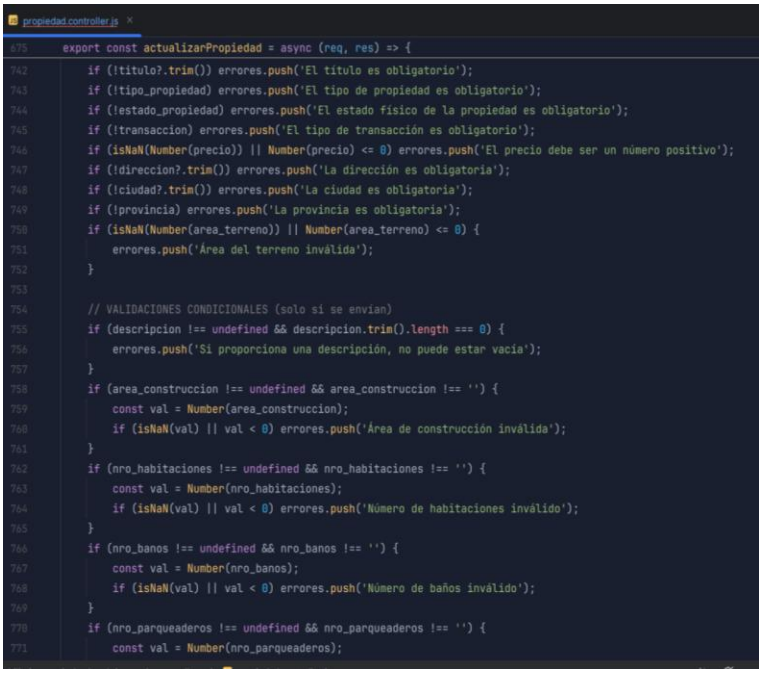
Para la gestión de imágenes, el componente administra tres estados independientes: *imagenesActuales* (imágenes existentes en la base de datos), *imagenesAEliminar* (*IDs* de imágenes marcadas para eliminar) e *imagenesNuevas* (imágenes nuevas a agregar), y la propiedad debe conservar al menos una. Además, la sección de *Documentación* dentro de *FormNegocio* permite agregar nuevos documentos o marcar los existentes para eliminación.

Si la validación es exitosa al enviar el formulario, construye un objeto *FormData* con los datos del formulario, las imágenes nuevas y el arreglo *imagenesAEliminar[]*, y realiza la petición *PUT /api/propiedades/:id* mediante *Axios*. A continuación, de forma paralela, elimina los documentos marcados mediante *DELETE /api/documentos/propiedad/:docId* y

sube los nuevos mediante *POST /api/documentos/propiedad/:id*. De resultar en una petición exitosa, muestra una alerta de éxito y se redirige al panel de propiedades.

En el *backend*, la función *actualizarPropiedad* (figura 38) en *propiedad.controller.js* realiza las validaciones de los campos, actualiza la lista de propietarios, elimina las imágenes marcadas y luego de la base de datos, y finalmente sube las nuevas imágenes a *Cloudinary* y las registra. El *endpoint PUT /api/propiedades/:id* está declarado en *propiedad.routes.js* con los *middlewares verificarToken*, *esPropietarioOAdmin* (que verifica que sea administrador o el agente captador de esa propiedad), u*ploadPropiedadImg.array('imagenes',15)* y *optimizarImagen*.

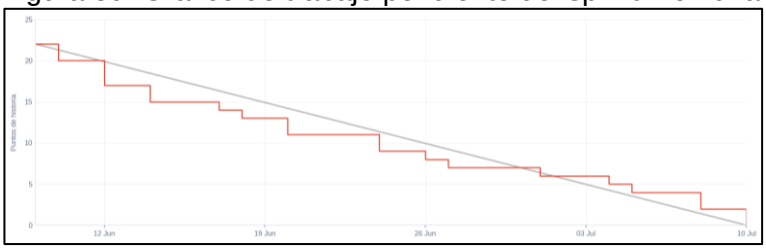
Figura 38. Función actualizarPropiedad en propiedad.controller.js



4.3.3.3. Sprint 1 – Gráfico de trabajo pendiente

Durante el *sprint* 1, se utilizó el gráfico de trabajo pendiente generado por *Jira* (figura 39) para monitorizar las historias de usuarios: inicio de sesión con roles, cierre de sesión, registro de cuentas y verificación de las mismas, registro, visualización y edición de propiedades registradas. Se visualiza en la gráfica una reducción progresiva de los puntos de historia a lo largo de *sprint*, identificando que el desarrollo iba según lo planificado.

Figura 39. Gráfico de trabajo pendiente del sprint 1 en Jira



4.3.3.4. Sprint 1 – Revisión

Al finalizar el *sprint* 1, el *Product Owner* hizo la revisión del incremento del producto que comprendía la autenticación, registro de cuentas y gestión de propiedades a través de los escenarios de prueba de los casos de prueba adjuntos en el anexo V. El *Product Owner* validó que las funcionalidades cumplían con los requisitos y aprobó el avance del *sprint*.

4.3.3.5. Sprint 1 – Retrospectiva

Para obtener la retrospectiva del *sprint* 1, se formularon tres preguntas con el fin de obtener una perspectiva más clara y completa, como se puede apreciar en la tabla 21.

Tabla 21. Sprint 1 – Retrospectiva

Aspectos exitosos en el Sprint	Mejoras que se podrían aplicar en el siguiente Sprint (recomendaciones para la mejora continua)	Errores en el Sprint
Se completaron todas las tareas de ingeniería del sprint 1 dentro del tiempo estipulado. Además, se logró establecer de manera satisfactoria el control de acceso por roles a la aplicación, resultado un inventario de inmuebles en este incremento.	Para la planificación del próximo sprint, se recomienda tener en cuenta la documentación de React, Node.js, Express y Prisma como recursos de apoyo para abordar posibles problemas.	Durante el proceso de desarrollo del sprint 1, se encontraron dificultades en la funcionalidad de edición de propiedades en cuanto a la gestión de imágenes en la base de datos, generando inconsistencias en la visualización de imágenes.

4.3.4. Sprint 2

4.3.4.1. Sprint 2 – Planificación

En la planificación del *sprint* 2, se seleccionaron las historias de usuario número 7 al 15 del *Product Backlog* para continuar el desarrollo, teniendo como resultado el *Sprint Backlog* 2 (tabla 22) con una sumatoria de 22 puntos de historia.

Tabla 22. Sprint Backlog 2

Historia Usuario	Est. His.	Categ	Tarea	Est. Tar.	Responsable	Estado
HU7 - Actualización del Estado de Publicación de Propiedades	3 pts	Backend	Implementación de endpoint PATCH /api/propiedades/:id/estado con verificación de estados administrativos permitidos, reglas de negocio según negociaciones activas y permisos para agente captador o administrador. (propiedad.controller.js, propiedad.routes.js)	1.5 pts	Jhery Alarcon	Completado
		Frontend	Implementación de cambio del estado de una propiedad a través de un modal con selector de estado. (CardPropiedad.jsx, ModalActualizarEstado.jsx)	1.5 pts	Romario Flores	Completado
HU8 - Desactivación de propiedades	2 pts	Backend	Implementación de endpoint DELETE /api/propiedades/:id para cambiar el estado de una propiedad a inactiva, validando que no existan negociaciones críticas activas. (propiedad.controller.js, propiedad.routes.js)	1 pt	Jhery Alarcon	Completado
		Frontend	Elaboración de modal de confirmación de desactivación de propiedad. (ModalConfirmarEliminar.jsx, CardPropiedad.jsx)	1 pt	Romario Flores	Completado
HU9 - Búsqueda y Filtrado de Propiedades en el Portal Público	3 pts	Backend	Implementación de endpoint GET /api/propiedades/publicas con filtros y opciones de ordenamiento. (propiedad.controller.js, propiedad.routes.js)	1.5 pts	Jhery Alarcon	Completado
		Frontend	Creación de componentes de búsqueda y filtros varios. (FiltrosPropiedades.jsx, Propiedades.jsx)	1.5 pts	Romario Flores	Completado
HU10 - Visualización del detalle de la propiedad	3 pts	Backend	Implementación de endpoint GET /api/propiedades/publica/:id para obtener la información necesaria de la propiedad únicamente de propiedades con estado disponible. (propiedad.controller.js, propiedad.routes.js)	1 pt	Jhery Alarcon	Completado
		Frontend	Elaboración de la vista de detalle de propiedad con galería de imágenes, información completa del inmueble, formulario de contacto y secciones de recomendaciones personalizadas. (DetallePropiedad.jsx)	2 pts	Romario Flores	Completado

HU11 - Guardar propiedad como favorita	3 pts	Base de Datos	Creación del modelo Favorito con relación de los modelos Usuario y Propiedad (schema.prisma)	0.5 pts	Jhery Alarcon	Completado
		Backend	Implementación de endpoint POST/DELETE/GET /api/favoritos con validación de permisos (favorite.controller.js, favorite.routes.js)	1 pt	Jhery Alarcon	Completado
		Frontend	Elaboración de toggle de favoritos y vista con listado de propiedades favoritas. (FavoritoIcon.jsx, MisFavoritos.jsx, CardPropiedadPublica.jsx)	1.5 pts	Romario Flores	Completado
HU12 - Registro de clientes internos	5 pts	Base de Datos	Creación del modelo Cliente con relación al modelo Usuario (agente). (schema.prisma)	1 pt	Jhery Alarcon	Completado
		Backend	Implementación de endpoint POST /api/clientes con validaciones, y asignación de agente según rol. (cliente.controller.js, cliente.routes.js)	1.5 pts	Jhery Alarcon	Completado
		Frontend	Elaboración de formulario para registro de clientes internos con selectores, validaciones y carga de documentos. (RegistrarCliente.jsx, SelectTipoCliente.jsx, DocumentManager.jsx)	2.5 pts	Romario Flores	Completado
HU13 - Visualización de clientes registrados	3 pts	Backend	Implementación de endpoint GET /api/clientes con restricciones de visibilidad según rol. (cliente.controller.js, cliente.routes.js)	1.5 pts	Jhery Alarcon	Completado
		Frontend	Elaboración de tabla de clientes con filtros. (PanelClientes.jsx)	1.5 pts	Romario Flores	Completado
HU14 – Edición de clientes registrados	3 pts	Backend	Implementación de endpoint GET/PUT /api/clientes/:id con verificación de permisos, y validación de campos. (cliente.controller.js, cliente.routes.js)	1.5 pts	Jhery Alarcon	Completado
		Frontend	Elaboración de formulario de edición pre-cargado con datos existentes del cliente a modificar y gestión de documentación. (EditarCliente.jsx, SelectTipoCliente.jsx, DocumentManager.jsx)	1.5 pts	Romario Flores	Completado
HU15 - Desactivación de clientes registrados	3 pts	Backend	Implementación de endpoint PATCH /api/clientes/:id/desactivar, /api/clientes/:id/reactivar para desactivar/reactivar clientes con verificación de permisos y validación de negociaciones y propiedades	1.5 pts	Jhery Alarcon	Completado

	activas. (cliente.controller.js, cliente.routes.js)			
Frontend	Elaboración de botón para desactivar/reactivar clientes con modal de confirmación (PanelClientes.jsx)	1.5 pts	Romario Flores	Completado

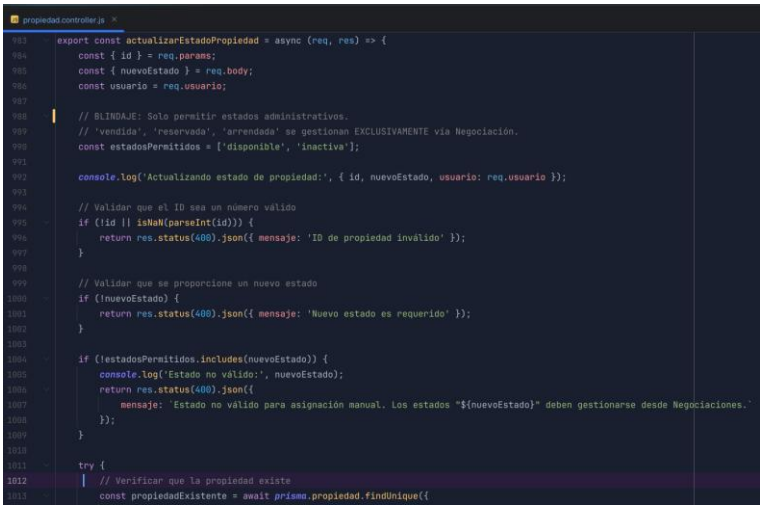
4.3.4.2. Sprint 2 – Reuniones diarias

Con la planificación del *sprint* 2, se establecieron los pasos necesarios para alcanzar el resultado esperado. Se utilizó la herramienta de gestión de incidentes *Jira* para las reuniones diarias, permitiendo supervisar el avance del proyecto, llevando un control continuo de las tareas de ingeniería definidas en las historias de usuario, las cuales se fueron avanzando de forma diaria.

4.3.4.2.1. Historia de Usuario 7: Actualización del estado de publicación

Del lado del *backend*, se implementó el *endpoint PATCH /api/propiedades/:id/estado* en el archivo *propiedad.routes.js* con *middlewares* de autenticación y autorización a usuarios con el rol de administrador o del agente que registró la propiedad. Si los permisos son válidos se ejecuta el controlador correspondiente en la función *actualizarEstadoPropiedad* (figura 40) ubicado en *propiedad.controller.js* para actualizar el estado de la propiedad tratada en la base de datos.

Figura 40. Función actualizarEstadoPropiedad en propiedad.controller.js



En la interfaz administrativa, del lado del *frontend*, se creó el modal *ModalActualizarEstado.jsx* (figura 41) que se abre desde un ícono de flechas circulares

ubicado en la parte inferior de la tarjeta de la propiedad a los usuarios autorizados. Dicho modal cuenta con un selector que muestra el listado de estados que puede tener una propiedad (figura 42), del cual se escoge un valor y se envía la actualización al *endpoint* correspondiente. De concretarse la operación se denota la actualización del estado de la propiedad a través de un *badge* que se implementó en la tarjeta de la propiedad *CardPropiedad.jsx*, cuyo color denota el estado actual de la misma.

Figura 41. Código de ModalActualizarEstado.jsx

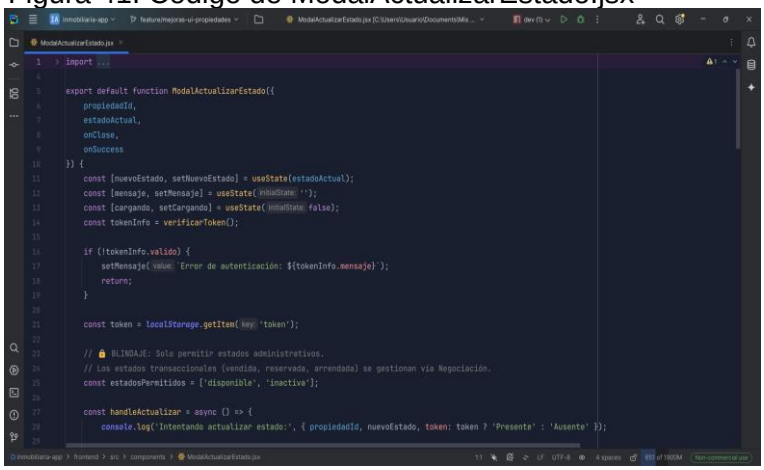
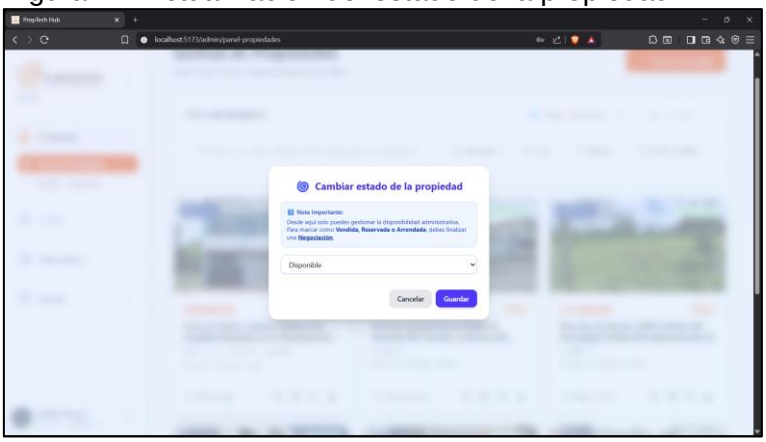


Figura 42. Actualización del estado de la propiedad

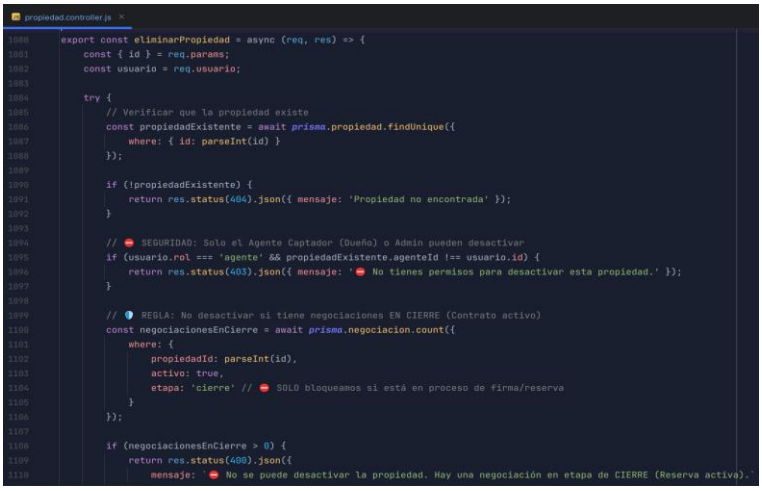


4.3.4.2.2. Historia de Usuario 8: Desactivación de Propiedades

En el *backend*, se añadió el *endpoint DELETE /api/propiedades/:id* al archivo de rutas *propiedad.routes.js* con los *middlewares* para verificar que el usuario autenticado tenga rol de administrador o agente propietario, el cual puede desactivar propiedades. De igual forma, se creó el controlador respectivo en la función *eliminarPropiedad* (figura 43) del

archivo *propiedad.controller.js*, la cual actualiza el campo *estado_publicacion* a “inactiva” en la base de datos haciendo *soft* delete.

Figura 43. Función eliminarPropiedad en propiedad.controller.js



Posteriormente, se procedió a crear el componente *ModalConfirmarEliminar.jsx* (figura 44) el cual es un modal que se muestra a partir de un botón con ícono de papelera, ubicado en la tarjeta de la propiedad de la interfaz administrativa *CardPropiedad.jsx* únicamente a administradores o agentes propietarios. El modal muestra un mensaje de confirmación indicando que la propiedad se muestra como inactiva (figura 45). El usuario puede confirmar o cancelar la desactivación, si el usuario lo confirma se envía una petición *DELETE* al *endpoint* correspondiente, y si la desactivación es exitosa se recarga la página no visualizando más la propiedad desactivada en la vista por defecto en la que se visualizan las propiedades activas.

Figura 44. Código de ModalConfirmarEliminar.jsx

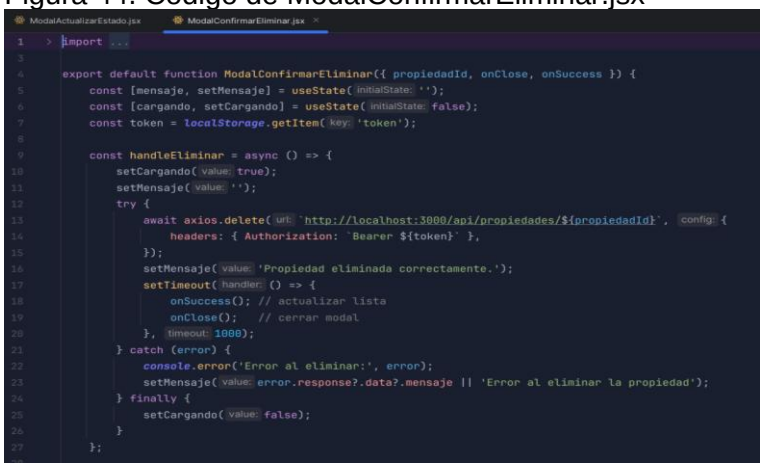
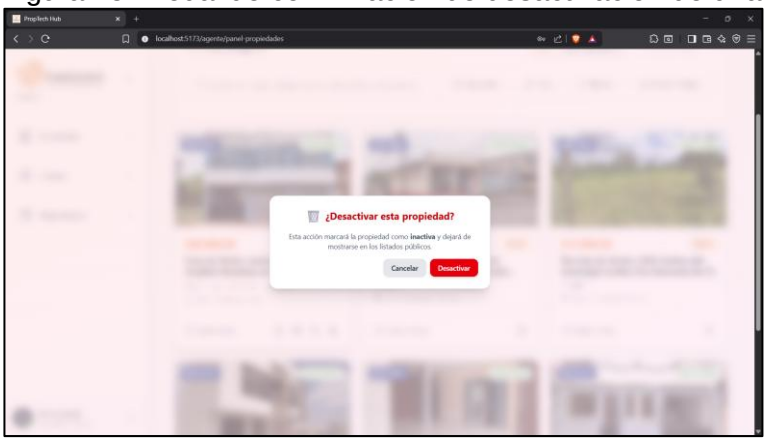


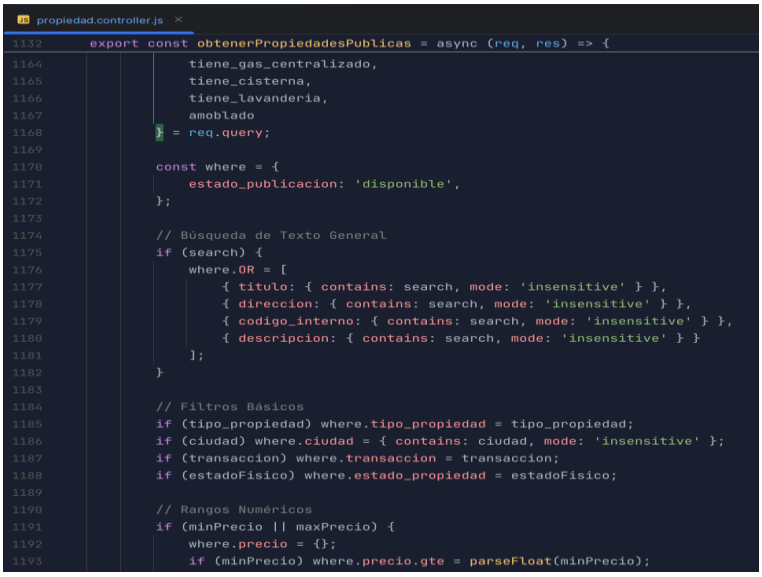
Figura 45. Modal de confirmación de desactivación de una propiedad



4.3.4.2.3. Historia de Usuario 9: Búsqueda y Filtrado de Propiedades

Se creó un controlador que corresponde a la función *obtenerPropiedadesPublicas* (figura 46) en *propiedad.controller.js* que recibe parámetros de *query* opcionales. Además, concernientes a diferentes métricas que puede tener una propiedad como el tipo de propiedad, ciudad donde está ubicada, precio máximo y mínimo, área de construcción, tipo de transacción, entre otros, filtrando siempre a las propiedades cuyo estado de publicación esté disponible. Asimismo, se agregó la ruta *GET /api/propiedades/publicas* en *propiedad.routes.js* sin *middlewares* de autenticación permitiendo el acceso público a las propiedades que estén disponibles.

Figura 46. Función obtenerPropiedadesPublicas en propiedad.controller.js



```
propiedad.controller.js
1132  export const obtenerPropiedadesPublicas = async (req, res) => {
1164              tiene_gas_centralizado,
1165              tiene_cisterna,
1166              tiene_lavanderia,
1167              amoblado
1168          } = req.query;
1169
1170          const where = {
1171              estado_publicacion: 'disponible',
1172          };
1173
1174          // Búsqueda de Texto General
1175          if (search) {
1176              where.OR = [
1177                  { titulo: { contains: search, mode: 'insensitive' } },
1178                  { direccion: { contains: search, mode: 'insensitive' } },
1179                  { codigo_interno: { contains: search, mode: 'insensitive' } },
1180                  { descripcion: { contains: search, mode: 'insensitive' } }
1181              ];
1182          }
1183
1184          // Filtros Básicos
1185          if (tipo_propiedad) where.tipo_propiedad = tipo_propiedad;
1186          if (ciudad) where.ciudad = { contains: ciudad, mode: 'insensitive' };
1187          if (transaccion) where.transaccion = transaccion;
1188          if (estadoFisico) where.estado_propiedad = estadoFisico;
1189
1190          // Rangos Numéricos
1191          if (minPrecio || maxPrecio) {
1192              where.precio = {};
1193              if (minPrecio) where.precio.gte = parseFloat(minPrecio);
```

Por otro lado, en el *frontend*, se hicieron dos implementaciones, la primera, en el portal público, en el componente *Propiedades.jsx* donde se muestran las propiedades de la ruta previamente descrita. Además, una barra de búsqueda por título, así como una variedad de filtros diseñados en *FiltrosPropiedades.jsx* (figura 47), los cuales cargan de forma dinámica según las características seleccionadas de los inmuebles disponibles, tal como se muestra en la interfaz respectiva a través de la figura 48.

Figura 47. Código de FiltrosPropiedades.jsx

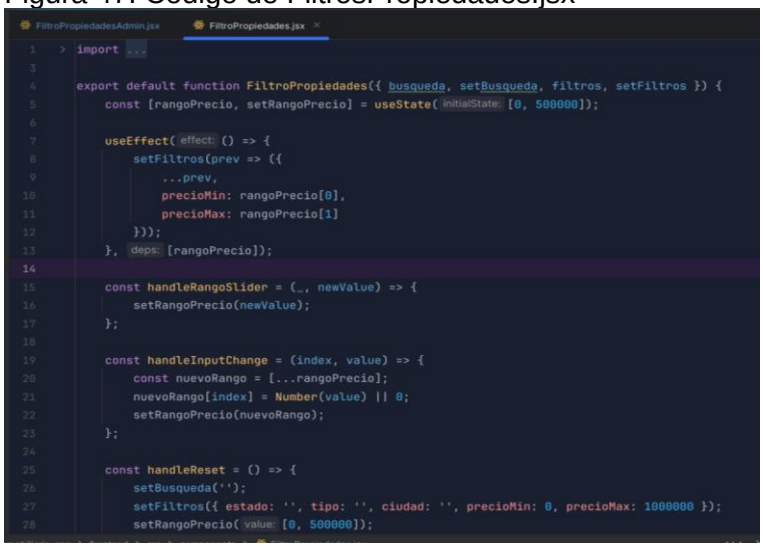
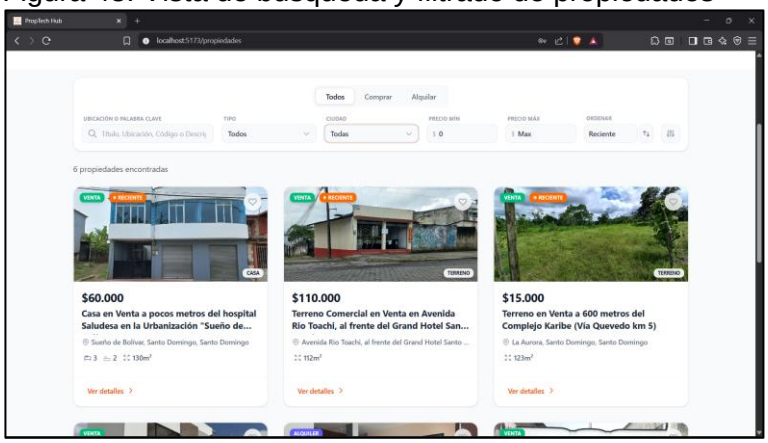


Figura 48. Vista de búsqueda y filtrado de propiedades

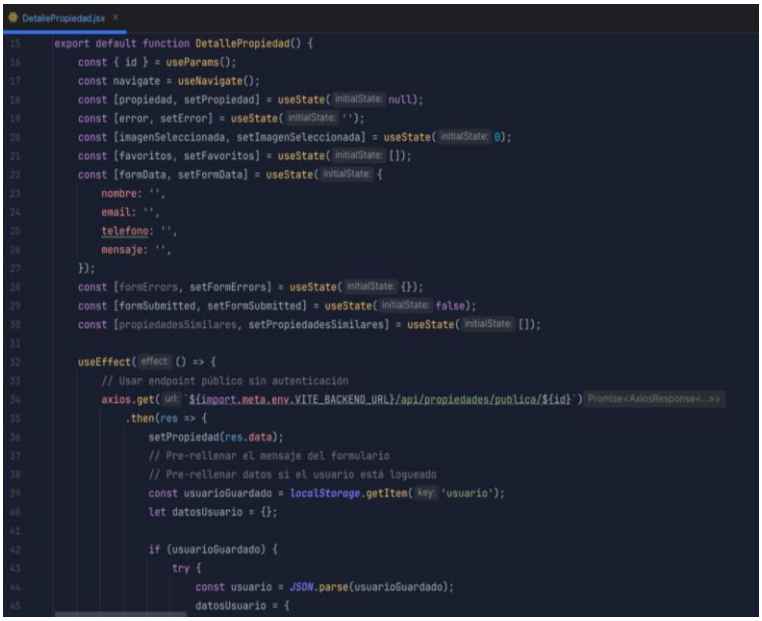


4.3.4.2.4. Historia de Usuario 10: Visualización de la propiedad

Se desarrolló la página *DetallePropiedad.jsx* (figura 49), accesible desde el portal público. Al cargar la página, se realiza una petición de tipo *GET* a la ruta */api/propiedades/publica/:id* mediante *Axios*, cuya lógica está en la función *obtenerPropiedadPublicaPorId* (figura 50) de *propiedad.controller.js,* Este *endpoint* retorna

propiedades con estado disponible, excluyendo datos sensibles como precio mínimo o comisión, y la ruta está declarada en *propiedad.routes.js* sin *middleware* de autenticación.

Figura 49. Código de DetallePropiedad.jsx



```
export default function DetallePropiedad() {
    const { id } = useParams();
    const navigate = useNavigate();
    const [propiedad, setPropiedad] = useState( initialState: null);
    const [error, setError] = useState( initialState: '');
    const [imagenSeleccionada, setImagenSeleccionada] = useState( initialState: 0);
    const [favoritos, setFavoritos] = useState( initialState: []);
    const [formData, setFormData] = useState( initialState: {
        nombre: '',
        email: '',
        telefono: '',
        mensaje: '',
    });
    const [formErrors, setFormErrors] = useState( initialState: {});
    const [formSubmitted, setFormSubmitted] = useState( initialState: false);
    const [propiedadesSimilares, setPropiedadesSimilares] = useState( initialState: []);

    useEffect( effect: () => {
        // Usar endpoint público sin autenticación
        axios.get( url: `${import.meta.env.VITE_BACKEND_URL}/api/propiedades/publica/${id}`) Promise<AxiosResponse<...>>
            .then(res => {
                setPropiedad(res.data);
                // Pre-rellenar el mensaje del formulario
                // Pre-rellenar datos si el usuario está logueado
                const usuarioGuardado = localStorage.getItem( key: 'usuario');
                let datosUsuario = {};

                if (usuarioGuardado) {
                    try {
                        const usuario = JSON.parse(usuarioGuardado);
                        datosUsuario = {
```

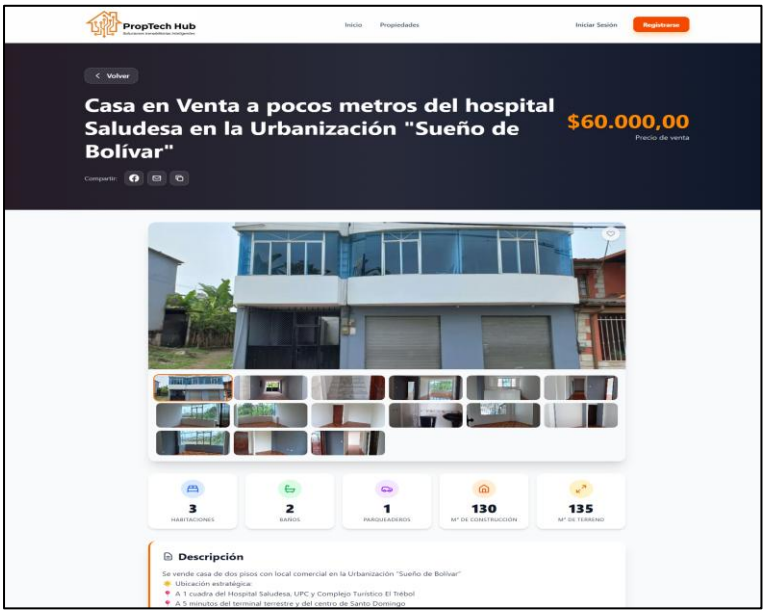
Figura 50. Función obtenerPropiedadPublicaPorId en propiedad.controller.js

```
export const obtenerPropiedadPublicaPorId = async (req, res) => {
    const { id } = req.params;

    try {
        const propiedad = await prisma.propiedad.findFirst({
            where: {
                id: parseInt(id),
                estado_publicacion: 'disponible'
            },
            include: {
                imagenes: true,
                agente: {
                    select: {
                        id: true,
                        name: true,
                        email: true
                    }
                }
            },
        });

        if (!propiedad) {
            return res.status(404).json({ mensaje: 'Propiedad no encontrada o no disponible' });
        }

        res.json(propiedad);
    } catch (error) {
        console.error(error);
        res.status(500).json({ mensaje: 'Error al obtener la propiedad' });
    }
};
```

La interfaz de la ficha de detalles de una propiedad en el portal público, tal como se visualiza en la figura 51, presenta una galería de imágenes principal donde las miniaturas permiten cambiar la fotografía activa, al hacer clic en una miniatura, la librería *react-photo-view* abre un visor a pantalla completa. Seguido, se muestran tarjetas de características con íconos (habitaciones, baños, parqueaderos, áreas), la descripción, detalles técnicos, ubicación y amenidades disponibles. En la cabecera, se incluyen botones para compartir en *Facebook*, correo y copiar enlace. La sección "¿Interesado en esta propiedad?" contiene un

formulario de contacto con los campos nombre, email, teléfono y mensaje. Si el usuario es cliente autenticado, los campos se pre-rellenan con sus datos.

Figura 51. Interfaz del detalle de una propiedad del portal público



4.3.4.2.5. Historia de Usuario 11: Guardar propiedad como favorita

En el archivo *schema.prisma* se declaró un nuevo modelo de datos llamado Favorito (figura 52) que, relaciona los usuarios con rol de cliente con las propiedades que han marcado como favoritas, es decir, establece la relaciones entre los modelos existentes *Usuario* y *Propiedad*. Se implementó el controlador *favorite.controller.js* (figura 53) con funciones para agregar una propiedad favorita al perfil del cliente, removerla de aquella lista y retornar todos los inmuebles preferidos con la información completa de los mismos. En *favorite.routes.js* se configuraron *middleware* de autenticación a las rutas *POST /api/favoritos* (agregar), *DELETE /api/favoritos* (quitar), y *GET /api/favoritos* (listar).

Figura 52. Modelo Favorito en schema.prisma

```
model Favorito {
  id          Int       @id @default(autoincrement())
  usuarioId   Int
  propiedadId Int
  createdAt   DateTime  @default(now())
  propiedad   Propiedad @relation(fields: [propiedadId], references: [id])
  usuario     Usuario   @relation(fields: [usuarioId], references: [id])

  @@unique([usuarioId, propiedadId])
}
```

Figura 53. Código del controlador favorite.controller.js

Se procedió con la elaboración de un componente reutilizable en *FavoritoIcon.jsx* (figura 54) con el ícono de corazón cuyo color cambia según si la propiedad se encuentra como favorita o no en el perfil de cliente (rojo si está, gris si no está). El componente verifica si el usuario está autenticado con el rol de cliente antes de permitir la acción. Si no está autenticado, muestra el icono de corazón con coloración gris y lo redirige a la página de inicio de sesión al hacer clic, y si el usuario no es cliente, muestra un mensaje de error.

De igual manera, se incorporó la página *MisFavoritos.jsx* la cual verifica que el usuario esté autenticado y sea cliente antes de renderizar las propiedades favoritas del mismo. Además, son traídas mediante la ruta *GET /api/favoritos* y se ajustan a un *grid* responsivo utilizando *CardPropiedadPublica.jsx*, incluye ordenamiento por precio, título o fecha ya sea en forma ascendente o descendente, así como el número de propiedades favoritas, tal como se observa en la figura 55.

Figura 54. Código de FavoritoIcon.jsx

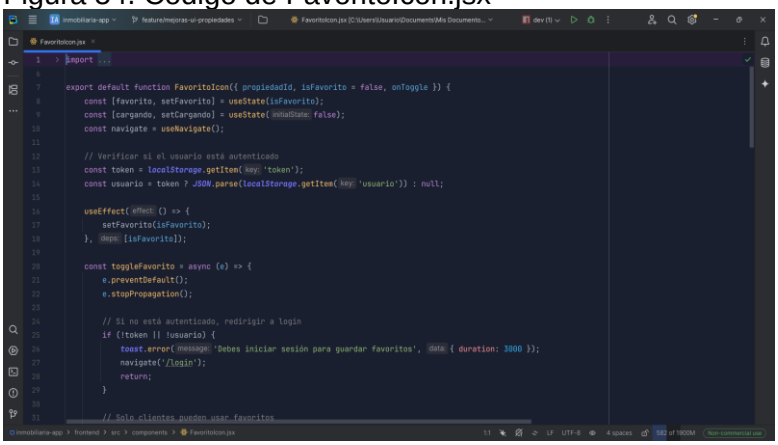
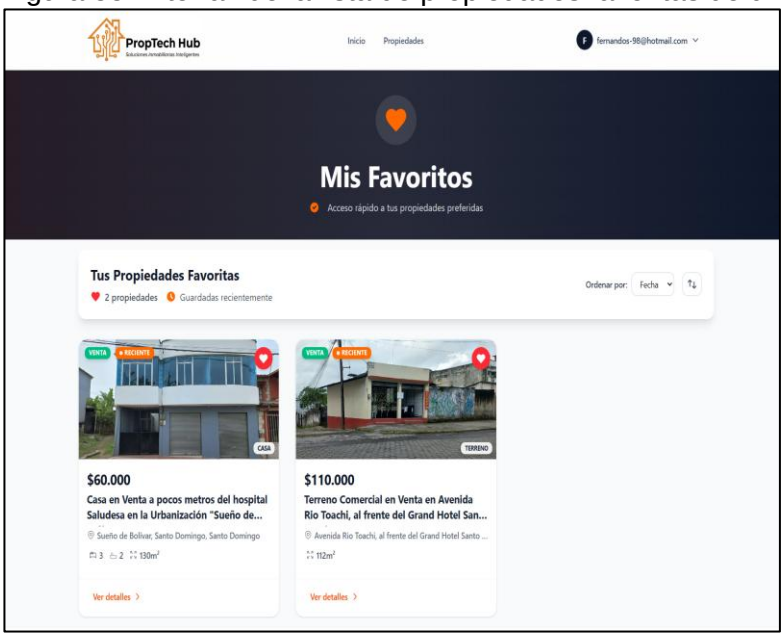


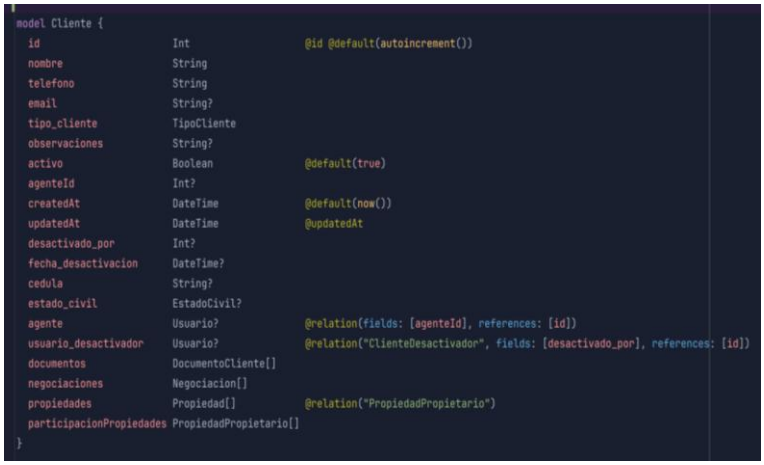
Figura 55. Interfaz de la lista de propiedades favoritas de un cliente autenticado



4.3.4.2.6. Historia de Usuario 12: Registro de clientes internos

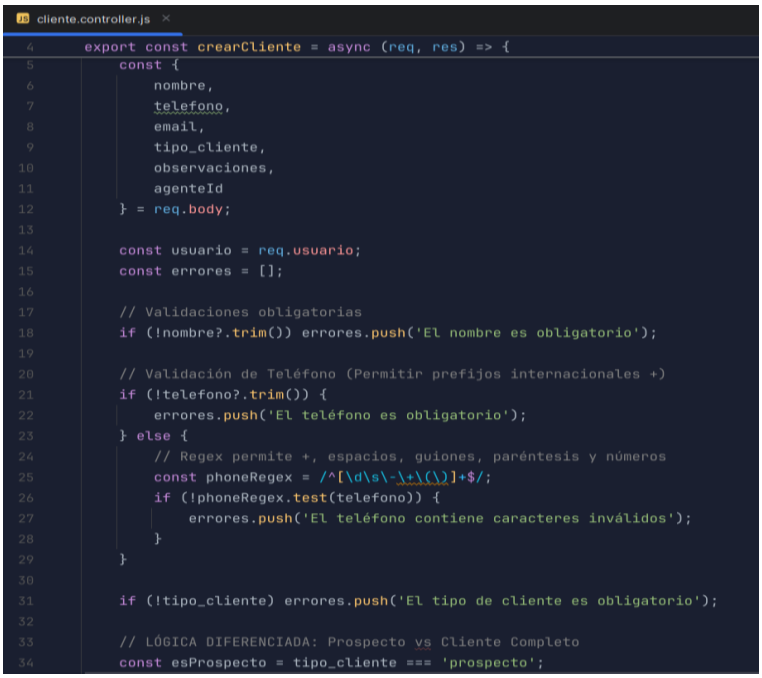
Se definió el modelo *Cliente* en *schema.prisma* con campos obligatorios como nombre, teléfono, correo electrónico, cédula/RUC y tipo de cliente, y, campos opcionales como observaciones, tal como se muestra en la figura 56. Además, en *cliente.controller.js* se configuró la función *crearCliente* (figura 57) que recibe los datos del cliente en el cuerpo de la petición y el usuario autenticado desde *req.usuario*; si el rol es administrador, debe enviar el *agenteId* del responsable, mientras que si es agente, se asigna automáticamente a sí mismo. En *cliente.routes.js,* se declaró la ruta *POST /api/clientes* con los *middlewares verificarToken* y *verificarRol* para restringir el acceso a agentes y administradores.

Figura 56. Modelo Cliente en schema.prisma



```
model Cliente {
  id                    Int                 @id @default(autoincrement())
  nombre                String
  telefono              String
  email                 String?
  tipo_cliente          TipoCliente
  observaciones         String?
  activo                Boolean             @default(true)
  agenteId              Int?
  createdAt             DateTime            @default(now())
  updatedAt             DateTime            @updatedAt
  desactivado_por       Int?
  fecha_desactivacion   DateTime?
  cedula                String?
  estado_civil          EstadoCivil?
  agente                Usuario?            @relation(fields: [agenteId], references: [id])
  usuario_desactivador  Usuario?            @relation("ClienteDesactivador", fields: [desactivado_por], references: [id])
  documentos            DocumentoCliente[]
  negociaciones         Negociacion[]
  propiedades           Propiedad[]         @relation("PropiedadPropietario")
  participacionPropiedades PropiedadPropietario[]
}
```

Figura 57. Función crearCliente en cliente.controller.js



```
cliente.controller.js
export const crearCliente = async (req, res) => {
    const {
        nombre,
        telefono,
        email,
        tipo_cliente,
        observaciones,
        agenteId
    } = req.body;

    const usuario = req.usuario;
    const errores = [];

    // Validaciones obligatorias
    if (!nombre?.trim()) errores.push('El nombre es obligatorio');

    // Validación de Teléfono (Permitir prefijos internacionales +)
    if (!telefono?.trim()) {
        errores.push('El teléfono es obligatorio');
    } else {
        // Regex permite +, espacios, guiones, paréntesis y números
        const phoneRegex = /^[\d\s\-\+\(\)]+$/;
        if (!phoneRegex.test(telefono)) {
            errores.push('El teléfono contiene caracteres inválidos');
        }
    }

    if (!tipo_cliente) errores.push('El tipo de cliente es obligatorio');

    // LÓGICA DIFERENCIADA: Prospecto vs Cliente Completo
    const esProspecto = tipo_cliente === 'prospecto';
```

La página *RegistrarCliente.jsx* (figura 58), presenta el formulario dividido en dos secciones: "Información Personal" e "Información Adicional". Los campos obligatorios son nombre, teléfono y tipo de cliente, si el tipo seleccionado es diferente de prospecto, el correo electrónico y la cédula/RUC también pasan a ser obligatorios. El selector de tipo de cliente utiliza el componente *SelectTipoCliente.jsx*, y la sección adicional incluye un campo de observaciones para notas o preferencias del cliente. El formulario una vez renderizado corresponde al que se visualiza en la figura 59.

Figura 58. Código de RegistrarCliente.jsx

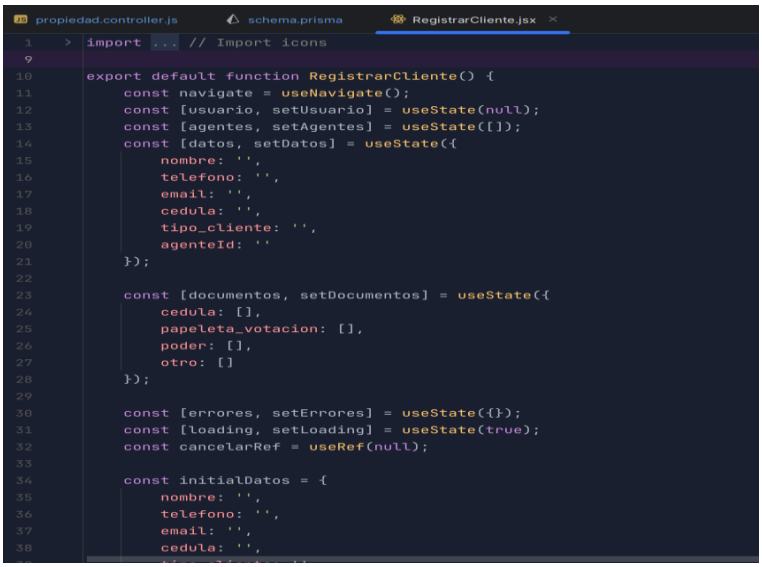
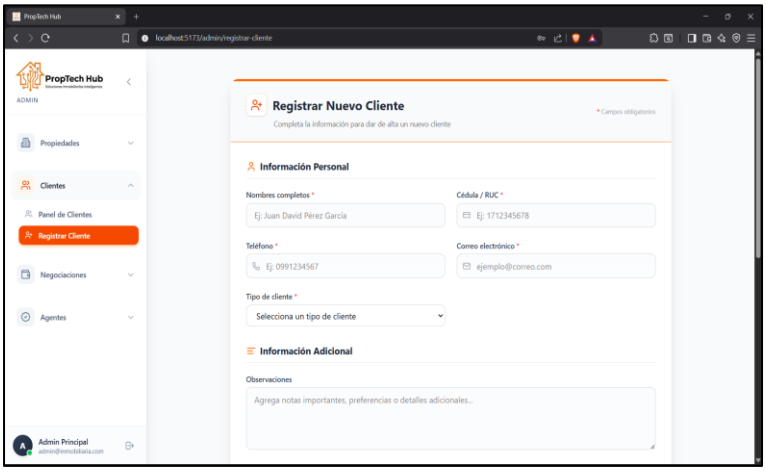


Figura 59. Formulario de registro de cliente



Si el usuario autenticado es administrador, aparece un buscador con autocompletado para seleccionar al agente responsable, cuya lista se obtiene mediante el *endpoint GET /api/usuarios/agentes*. Al pulsar el botón para guardar el cliente, la función *handleSubmit* ejecuta la validación final. Si la validación es exitosa, realiza una petición *POST /api/clientes* con el *token JWT* y, una vez creado el cliente, sube los documentos adjuntos tales como cédula, papeleta de votación, poder y otros mediante la ruta *POST /api/documentos/cliente/:clienteId* usando *DocumentManager.jsx*. De ser exitoso el proceso completo, muestra una alerta de confirmación y redirige al panel de clientes acorde al rol del usuario autenticado.

4.3.4.2.7. Historia de usuario 13: Visualización de clientes registrados

El componente *PanelClientes.jsx* (figura 60) implementa el listado de clientes registrados en el sistema traído mediante el *endpoint GET /api/clientes,* el cual se encuentra definido en *cliente.routes.js* y procesado en la función *obtenerClientes* del archivo *cliente.controller.js*, tal como se muestra en la figura 61. Si el usuario autenticado tiene rol de administrador visualiza todos los clientes del sistema junto con la columna "Agente" (nombre y correo del responsable), mientras que, un agente ve únicamente los clientes que ha registrado. La tabla incluye filtros por tipo, estado (activo/inactivo), agente (solo para administrador) y búsqueda en tiempo real por nombre, cédula, correo o teléfono y botones de acción por fila, como se aprecia en la figura 62.

Figura 60. Código de PanelClientes.jsx

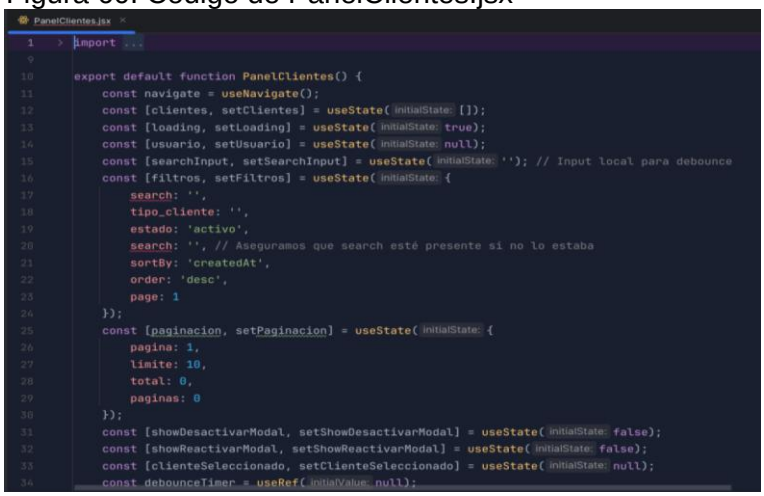
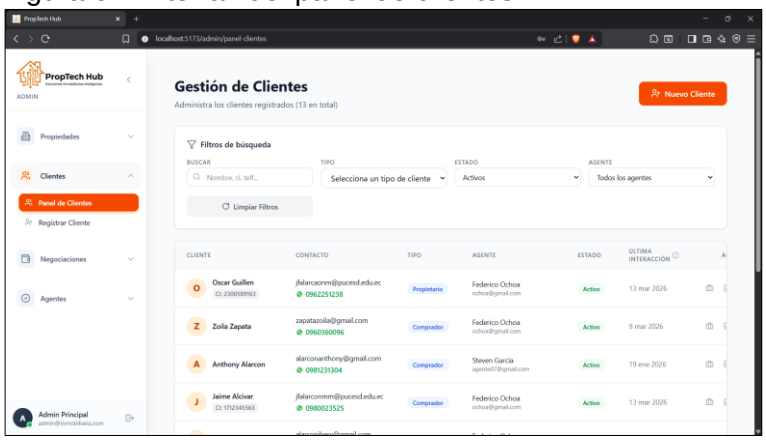


Figura 61. Función obtenerClientes en cliente.controller.js

Figura 62. Interfaz del panel de clientes



4.3.4.2.8. Historia de usuario 14: Edición de clientes

Los componentes *EditarCliente.jsx* (figura 63), junto con *SelectTipoCliente.jsx* y *DocumentManager.jsx*, gestionan el formulario de edición de los datos actuales de cliente. Los datos se obtienen mediante el *endpoint GET /api/clientes/:id*, y al confirmar los cambios se ejecuta la petición *PUT /api/clientes/:id*. Además, ambos *endpoints* están definidos en *cliente.routes.js* y procesados en la función *actualizarCliente* (figura 64) del *cliente.controller.js*, el cual verifica permisos y valida que el correo electrónico y la cédula/RUC sean únicos. Los clientes inactivos no pueden editarse, el agente puede reasignar únicamente si el cliente es de tipo prospecto, mientras que el administrador puede reasignar, en cualquier caso. La interfaz del formulario de edición de la información de un cliente se muestra en la figura 65.

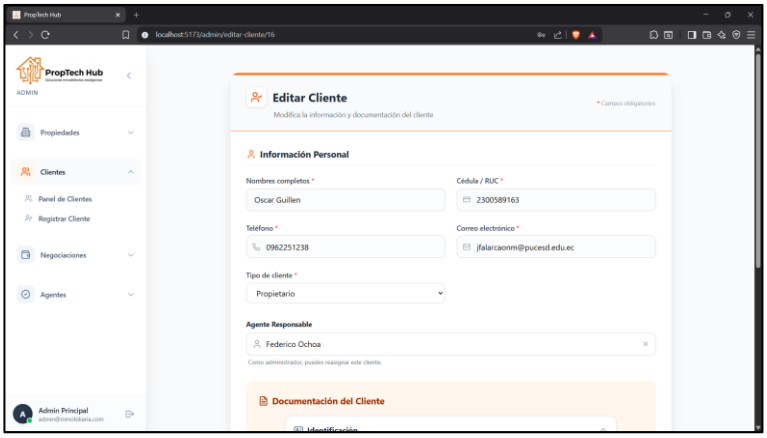
Figura 63. Código de EditarCliente.jsx

```
EditarCliente.jsx
import ...

export default function EditarCliente() {
    const { id } = useParams();
    const navigate = useNavigate();
    const [usuario, setUsuario] = useState( initialState: null);
    const [cliente, setCliente] = useState( initialState: null);
    const [agentes, setAgentes] = useState( initialState: []);

    // Estado para documentos
    const [documentos, setDocumentos] = useState( initialState: {
        cedula: [],
        papeleta_votacion: [],
        poder: [],
        otro: []
    });

    const [datos, setDatos] = useState( initialState: {
        nombre: '',
        telefono: '',
        email: '',
        cedula: '',
        tipo_cliente: '',
        agenteId: ''
    });

    const [errores, setErrores] = useState( initialState: {});
```

Figura 64. Función actualizarCliente en cliente.controller.js

```
cliente.controller.js
323  export const actualizarCliente = async (req, res) => {
324      const { id } = req.params;
325      const {
326          nombre,
327          telefono,
328          email,
329          cedula,
330          tipo_cliente,
331          observaciones,
332          agenteId // Solo permitido para admin
333      } = req.body;
334
335      const usuario = req.usuario;
336      const errores = [];
337
338      try {
339          // Verificar que el cliente existe
340          const clienteExistente = await prisma.cliente.findUnique({
341              where: { id: parseInt(id) }
342          });
343
344          if (!clienteExistente) {
345              return res.status(404).json({
346                  mensaje: 'Cliente no encontrado'
347              });
348          }
349
350          // Verificar permisos
351          if (usuario.rol === 'agente' && clienteExistente.agenteId !== usuario.id) {
352              return res.status(403).json({
353                  mensaje: 'No tienes permisos para editar este cliente'
```

Figura 65. Formulario de edición de un cliente



4.3.4.2.9. Historia de usuario 15: Desactivación de clientes registrados

Desde la página *PanelClientes.jsx* (figura 66), se presentan los botones de desactivar y reactivar en la columna de acciones de la tabla de clientes. Al desactivar, se ejecuta el *endpoint PATCH /api/clientes/:id/desactivar*, al reactivar, trabaja de forma similar con el *endpoint PATCH /api/clientes/:id/reactivar*. Por lo tanto, ambos *endpoints* están declarados en *cliente.routes.js* y gestionados por las funciones *desactivarCliente* y *reactivarCliente* en el archivo *cliente.controller.js*, en donde la primera función verifica que el

cliente a desactivar no tenga negociaciones en curso ni propiedades activas vinculadas, como se observa en la figura 67. Po otro lado, el administrador puede reactivar clientes, si un agente intenta hacerlo, el modal de confirmación muestra la advertencia correspondiente. Adicionalmente, se muestra en la figura 68 el modal de confirmación para desactivar un cliente.

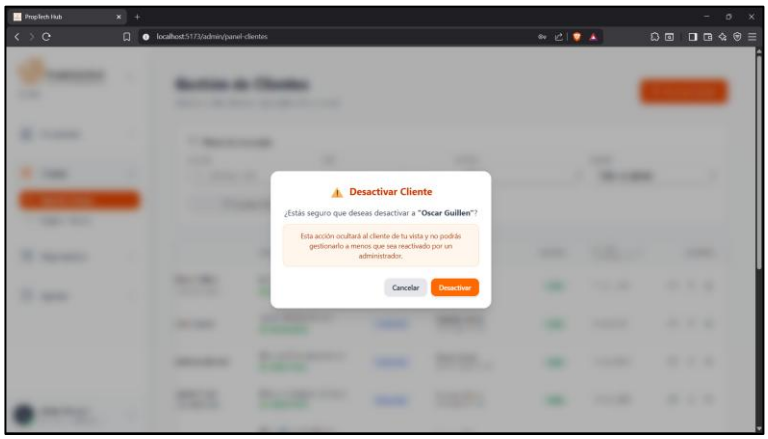
Figura 66. Código de PanelClientes.jsx

```
export default function PanelClientes() {
    <div className="max-w-7xl mx-auto p-4 sm:p-6 lg:p-8">
        <div className="bg-white rounded-lg shadow-sm border border-gray-200 p-6 mb-6">
            <div className="grid grid-cols-1 md:grid-cols-4 gap-4">
                <div>
                    Tipo
                </label>
                <SelectTipoCliente
                    value={filtros.tipo_cliente}
                    onChange={(e) => handleFiltroChange( { target: { name: 'tipo_cliente', value: e.target.value } })}
                />
            </div>

            {/* Estado del cliente */}
            <div>
                <label className="block text-xs font-medium text-gray-500 mb-1 uppercase tracking-wider">
                    Estado
                </label>
                <select
                    name="estado"
                    value={filtros.estado}
                    onChange={handleFiltroChange}
                    className="w-full border border-gray-300 rounded-lg px-3 py-2 text-sm focus:outline-none focus:ring-2 focus:ring-orange-500
                >
                    <option value="activo">Activos</option>
                    <option value="inactivo">Inactivos</option>
                    {usuario?.rol === 'admin' && <option value="todos">Todos</option>}
```

Figura 67. Función desactivarCliente en cliente.controller.js

```
// Desactivar un cliente
export const desactivarCliente = async (req, res) => {
    const { id } = req.params;
    const usuario = req.usuario;

    try {
        // Verificar que el cliente existe
        const cliente = await prisma.cliente.findUnique({
            where: { id: parseInt(id) }
        });

        if (!cliente) {
            return res.status(404).json({
                mensaje: 'Cliente no encontrado'
            });
        }

        // Verificar permisos
        if (usuario.rol === 'agente' && cliente.agenteId !== usuario.id) {
            return res.status(403).json({
                mensaje: 'No tienes permisos para desactivar este cliente'
            });
        }

        // Verificar si ya está inactivo
        if (!cliente.activo) {
            return res.status(400).json({
                mensaje: 'El cliente ya está inactivo'
            });
        }
```

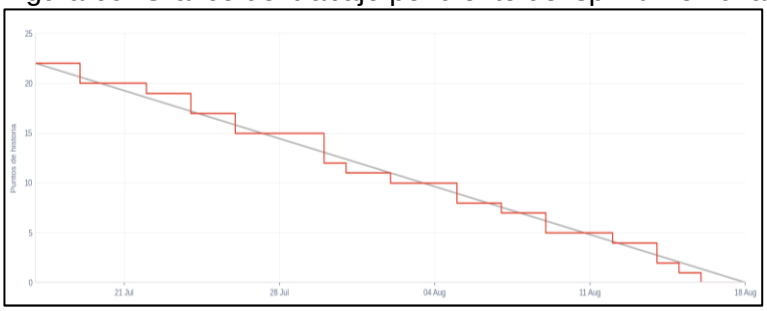
Figura 68. Modal de confirmación para desactivar cliente



4.3.4.2.10. Sprint 2 - Gráfico de trabajo pendiente

Para el *sprint* 2, el gráfico de trabajo pendiente de *Jira* (figura 69), permitió visualizar el progreso de las historias de usuario: actualización del estado de publicación de propiedades, desactivación de propiedades, búsqueda y filtrado de propiedades, visualización del detalle de una propiedad, guardar propiedad como favorita, registro, visualización, edición y desactivación de clientes registrados. Reflejando la evolución de las tareas pendientes, mostrando la velocidad de trabajo del equipo y permitiendo detectar desviaciones respecto a la planificación.

Figura 69. Gráfico del trabajo pendiente del sprint 2 en Jira



4.3.4.3. Sprint 2 – Revisión

Al concluir el *sprint* 2, se llevó a cabo la revisión del incremento del producto con el *Product Owner*, se demostraron las funcionalidades implementadas y se evaluó el cumplimiento de los objetivos. De igual forma, hubo la validación de los escenarios de pruebas de las pruebas de aceptación (véase anexo V), en donde, el *Product Owner* confirmó que las funcionalidades cumplían con las expectativas y aprobó el avance.

4.3.4.4. Sprint 2 – Retrospectiva

Se realizaron tres preguntas fundamentales con el fin de obtener una perspectiva clara, completa y concisa, al culminar el s*print* 2, como se puede apreciar en la tabla 23.

Tabla 23. Sprint 2 - Retrospectiva

Aspectos exitosos en el Sprint	Mejoras que se podrían aplicar en el siguiente Sprint (recomendaciones para la mejora continua)	Errores en el Sprint
Durante el desarrollo del sprint 2, se completaron	Se recomienda realizar pruebas rápidas de cada	Durante el proceso de desarrollo del sprint 2, se

exitosamente todas las tareas. Además, se logró implementar satisfactoriamente el sistema de soft delete para propiedades y clientes, manteniendo la integridad de los datos.	historia antes de cerrarla para detectar errores a tiempo.	encontraron dificultades con la funcionalidad de filtro y búsqueda de propiedades en la página pública. Especialmente, se detectó errores al momento de aplicar múltiples filtros simultáneamente ya que la aplicación no filtraba correctamente ocasionando como resultado resultados incorrectos.

4.3.5. Sprint 3

4.3.5.1. Sprint 3 – Planificación

Para la planificación del *sprint* 3, se seleccionaron las historias de usuario 16 al 23 del *Product Backlog*, dando como resultado 22 puntos de historia, necesarios para culminar este *sprint*, asimismo se elaboró su correspondiente *sprint backlog* que se visualiza en la tabla 24.

Tabla 24. Sprint Backlog 3

Historia Usuario	Est. His.	Categoría	Tarea	Est. Tar.	Responsable	Estado
HU16 - Registro de negociación	5 pts	Base de Datos	Creación de modelo Negociación con relación de los modelos Cliente y Propiedad. (*schema.prisma*)	1 pt	Jhery Alarcon	Completado
		Backend	Implementación de *endpoint POST /api/negociaciones* con las respectivas validaciones. (*negociacion.controller.js, negociacion.routes.js*)	2 pts	Jhery Alarcon	Completado
		Frontend	Elaboración de componente para registrar una negociación. (*CrearNegociacion.jsx*)	2 pts	Romario Flores	Completado
HU17- Actualización de etapa de una negociación	3 pts	*Backend*	Implementación de *endpoint PUT /api/negociaciones/:id*, con validaciones de etapa y actualización automática del estado de la propiedad. (*negociacion.controller.js, negociacion.routes.js*)	1.5 pts	Jhery Alarcon	Completado
		Frontend	Creación de componente para actualizar etapa de negociación. (*ActualizarEtapaForm.jsx, ModalActualizarEtapaNegociacion.jsx*)	1.5 pts	Romario Flores	Completado

HU18-Historial de seguimiento por negociación	5 pts	Base de Datos	Creación de modelo Seguimiento con relación de los modelos Negociacion y Usuario (agente). (*schema.prisma*)	1 pt	Jhery Alarcon	Completado
		Backend	Implementación de *endpoints GET y POST /api/seguimientos/:negociacionId* con validaciones (*seguimiento.controller.js, seguimiento.routes.js*)	2 pts	Jhery Alarcon	Completado
		Frontend	Creación de componente para visualizar y registrar seguimientos a negociaciones. (*HistorialSeguimientos.jsx*)	2 pts	Romario Flores	Completado
HU19-Notas internas por negociación	3 pts	Base de Datos	Creación de modelo NotaInterna con relación de los modelos Negociacion y Usuario (agente). (*schema.prisma*)	0.5 pts	Jhery Alarcon	Completado
		Backend	Implementación de *endpoints GET y POST /api/notas-internas/:negociacionId* con validaciones; cifrado *AES-256-GCM* del contenido de la nota antes de persistir y descifrado al consultar. (*notaInterna.controller.js, notaInterna.routes.js*, *cifrado.js*)	1.5 pts	Jhery Alarcon	Completado
		Frontend	Creación de formulario para registro de notas internas. (*NotasInternas.jsx*)	1 pt	Romario Flores	Completado
HU20-Adjuntar archivos por negociación	5 pts	Base de Datos	Creación de modelo *ArchivoNegociacion* con relación a los modelos Negociacion y Usuario (agente). (*schema.prisma*)	1 pt	Jhery Alarcon	Completado
		Backend	Configuración de *multer* e implementación de *endpoints POST /api/archivos-negociacion/subir, GET /api/archivos-negociacion/negociacion/:negociacionId y GET /api/archivos-negociacion/descargar/:archivoId* con validación de permisos. (*multer.js, archivoNegociacion.controller.js, archivoNegociacion.routes.js*)	2 pts	Jhery Alarcon	Completado
		Frontend	Creación de componentes para subir, listar, descargar archivos. (*ArchivosAdjuntos.jsx, ModalArchivosAdjuntos.jsx*)	2 pts	Romario Flores	Completado
HU21-Registro de	5 pts	*Backend*	Implementación de *endpoint POST /api/agentes/crear* con validaciones.	2.5 pts	Jhery Alarcon	Completado

agentes			(*agentes.controller.js, agentes.routes.js*)			
		Frontend	Elaboración de formulario con validaciones para registrar agentes. (*RegistrarAgente.jsx*)	2.5 pts	Romario Flores	Completado
HU22-Visualización de agentes	3 pts	*Backend*	Implementación de endpoint *GET /api/agentes* con validaciones. (*agentes.controller.js, agentes.routes.js*)	1.5 pts	Jhery Alarcon	Completado
		Frontend	Creación de vista de agentes. (*PanelAgentes.jsx*)	1.5 pts	Romario Flores	Completado
HU23-Edición de agentes	3 pts	*Backend*	Implementación de endpoints *GET* y *PUT /api/agentes/:id* con validaciones. (*agentes.controller.js, agentes.routes.js*)	1.5 pts	Jhery Alarcon	Completado
		Frontend	Elaboración de formulario de edición pre-cargado con datos existentes del agente a modificar. (*EditarAgente.jsx*)	1.5 pts	Romario Flores	Completado

Tabla 25. Sprint Backlog Técnico

Historia de Usuario	Est. His.	Categoría	Tarea	Est. Tarea	Responsable	Estado
HT1-Autenticación y seguridad	6 pts	Backend	Implementación de autenticación con JWT (JSON Web Tokens)	1 pt	Jhery Alarcon	Completado
		Backend	Encriptación de contraseñas con bcrypt	1 pt	Jhery Alarcon	Completado
		Backend	Middlewares de autorización por roles y permisos	2 pts	Romario Flores	Completado
		Frontend	Protección de rutas por medio de React Router	2 pts	Romario Flores	Completado

4.3.5.2. Sprint 3 - Reuniones diarias

Durante el desarrollo del *sprint* 3, se realizaron reuniones diarias utilizando como herramienta de gestión de incidencias *Jira* para verificar los avances de cada integrante del equipo. Asimismo, se llevó un control de las tareas finalizadas de acuerdo con la planificación.

4.3.5.2.1. Historia de usuario 16: Registro de negociación

Se desarrolló el componente *CrearNegociacion.jsx* (figura 70), el cual es un modal desplegado desde *PanelNegociaciones.jsx*, y permite registrar una negociación asociando

un cliente activo con una propiedad disponible mediante campos de búsqueda. Los datos se cargan a través de los *endpoint GET /api/clientes?estado=activo* y *GET /api/propiedades/negociaciones/disponibles?includeAgente=true*.

Al confirmar, se ejecuta la petición *POST /api/negociaciones*, definido en *negociacion.routes.js* y procesado en la función *crearNegociacion* del archivo *negociacion.controller.js* (figura 71), el cual valida permisos, previene la creación de una negociación duplicada y crea la negociación con estado "Interés". Por otro lado, se creó el modelo Negociación con sus relaciones a Cliente y Propiedad, el cual está definido en *schema.prisma*. El modal para registrar una negociación se muestra en la figura 72.

Figura 70. Código de CrearNegociaciones.jsx

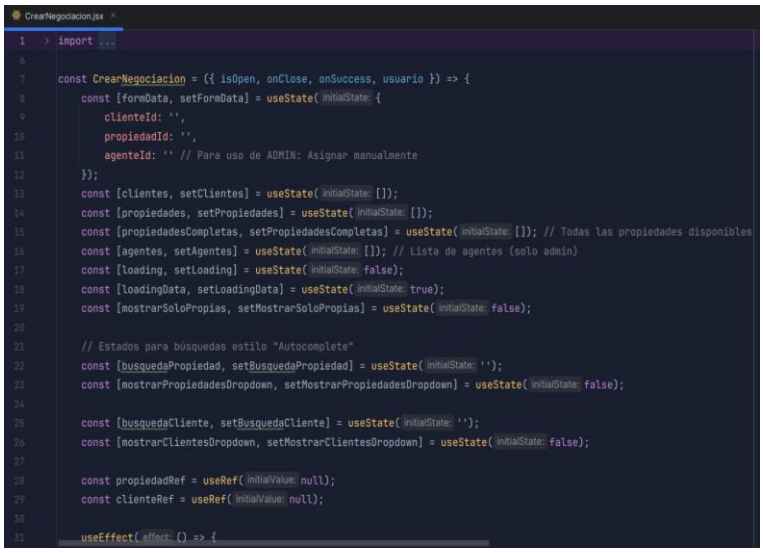


Figura 71. Función crearNegociacion en negociacion.controller.js

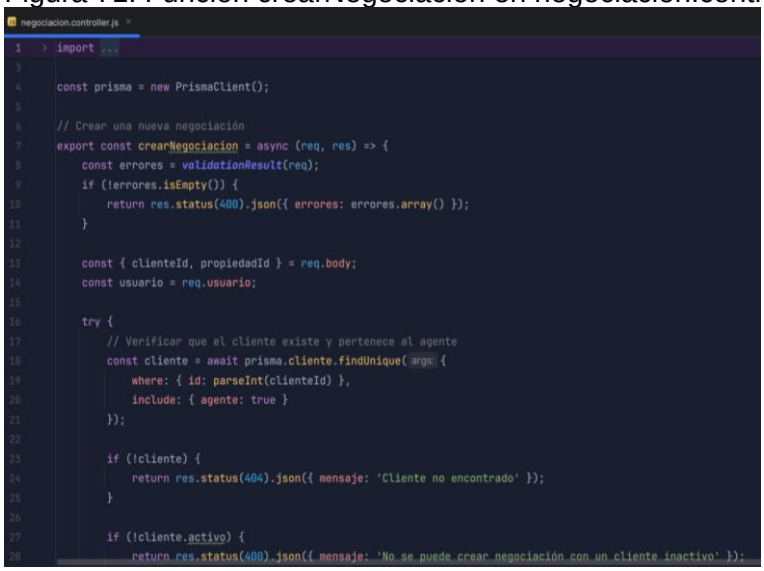
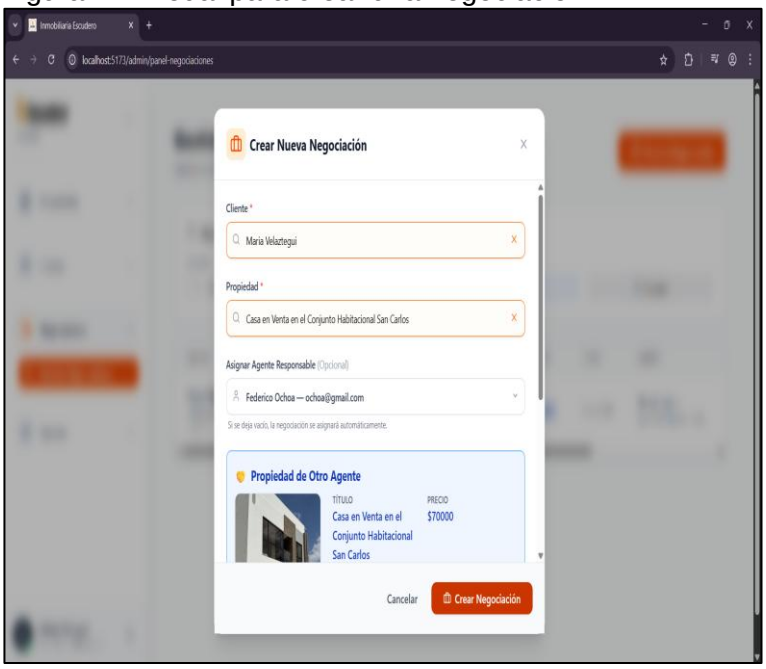


Figura 72. Modal para crear una negociación



4.3.5.2.2. Historia de usuario 17: Actualización de etapa negociación

El componente *ActualizarEtapaForm.jsx* (figura 73), renderiza un selector con botones para cada etapa de una negociación siendo estas: Interés, Negociación, Cierre, Finalizada y Cancelada, con una breve descripción e indicaciones, tal como se muestra en la figura 74. Dicho componente se presenta dentro del archivo *ModalActualizarEtapa Negociacion.jsx*. Ambos componentes, aplican validación de permisos en el *frontend* donde las opciones "Cierre" y "Finalizada" aparecen deshabilitadas, si el usuario no es administrador ni agente captador.

Al pulsar el botón "Confirmar Cambio", se ejecuta el *endpoint PUT /api/negociaciones/:id*, definido en *negociacion.routes.js* y gestionado por la función *actualizarNegociacion* del archivo *negociacion.controller.js* (figura 75), que actualiza el estado de la propiedad vinculada. Por ejemplo, si se selecciona la etapa “Cierre” de una negociación, la propiedad vinculada cambia al estado de “Reservada” o de actualizar la etapa de una negociación a “Finalizada” el estado de la propiedad será “Vendida” o “Arrendada” que esto lo determinará el tipo de transacción (Venta, Arriendo) de la propiedad en cuestión.

Figura 73. Código de ActualizarEtapaForm.jsx

Figura 74. Modal para actualizar etapa de una negociación

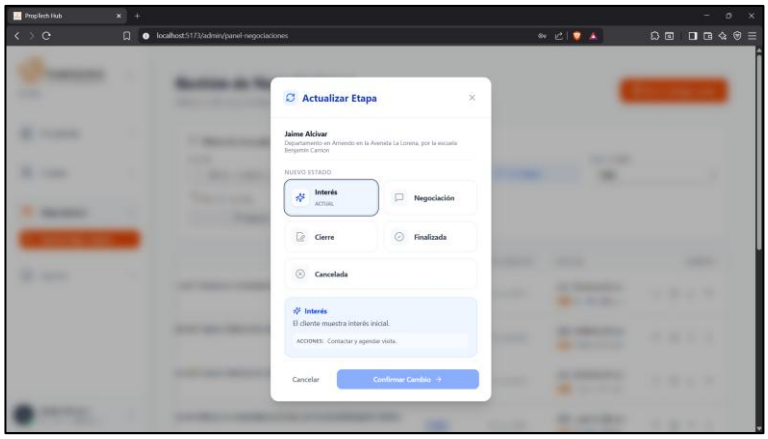
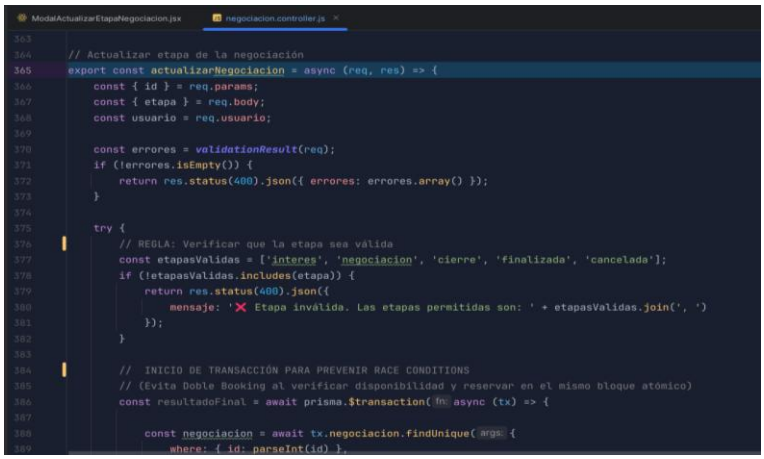


Figura 75. Función actualizarNegociacion en negociacion.controller.js



4.3.5.2.3. Historia de usuario 18: Historial de seguimiento negociación

Se creó el componente *HistorialSeguimientos.jsx* (figura 76) que permite registrar y consultar el historial de seguimiento de una determinada negociación. Al cargar el historial, este obtiene los registros de los diferentes eventos mediante el *endpoint GET /api/seguimientos/:negociacionId*, definido en *seguimiento.routes.js* y gestionado por la

función *obtenerSeguimientos* en el archivo *seguimiento.controller.js* (figura 77), que permite únicamente el acceso al agente dueño de la negociación y al administrador. El nuevo seguimiento requiere un comentario (máximo de 1000 caracteres) y un tipo de evento (llamada, visita, mensaje, email, reunión, documento u otro). El botón "Guardar" permanece deshabilitado hasta que se ingrese contenido en el nuevo evento, al enviar, se realiza una petición *POST /api/seguimientos/:negociacionId* procesado por la función *crearSeguimiento* del controlador correspondiente, que hace las respectivas validaciones. Además, se crea el registro en el modelo de seguimiento que relaciona los modelos *Negociacion* y *Usuario* en el archivo *schema.prisma*. La vista del historial de seguimientos de una negociación se observa en la figura 78.

Figura 76. Código de HistorialSeguimientos.jsx

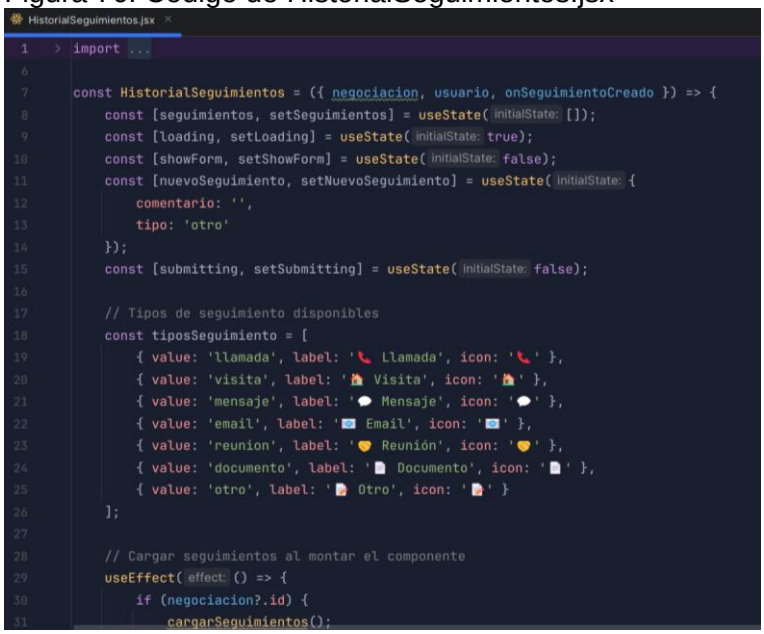


Figura 77. Función obtenerSeguimientos en seguimiento.controller.js

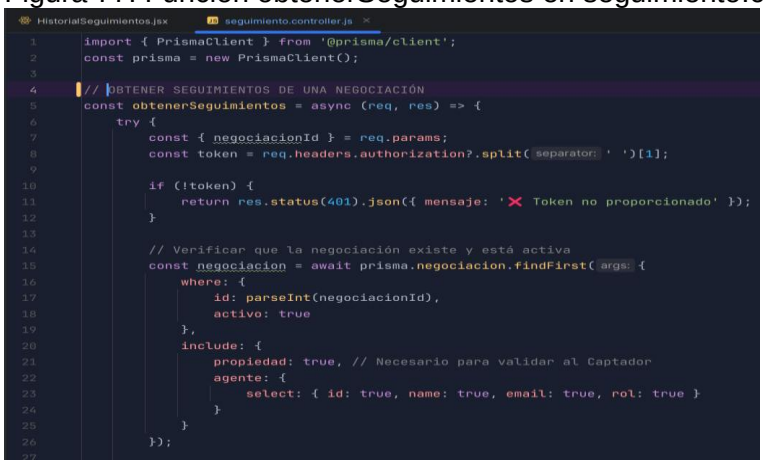
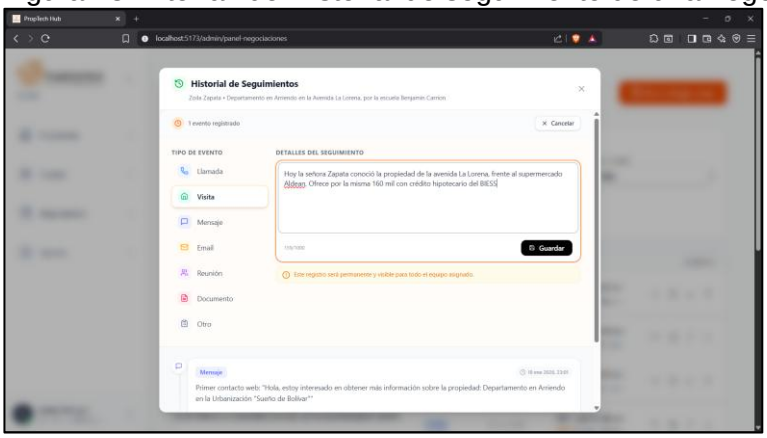


Figura 78. Interfaz del historial de seguimiento de una negociación



4.3.5.2.4. Historia de *usuario* 19: Notas internas por negociación

Se elaboró el componente *NotasInternas.jsx* (figura 79) que permite al agente dueño de una negociación, registrar y consultar notas privadas de aquella relación comercial. Antes de cargar las notas privadas, la función *canViewNotas()* verifica que el usuario autenticado tenga el rol de agente y que *negociacion.agenteId* coincida con su identificador, de no cumplirse dicha condición no se permite el acceso. Las notas se obtienen mediante el *endpoint GET /api/notas-internas/:negociacionId* y se registran mediante *POST /api/notas-internas/:negociacionId*, los cuales se encuentran definidos en *notaInterna.routes.js.* Y, por último, procesados por las funciones *obtenerNotasInternas* y *crearNotaInterna* (figura 80) en el archivo *notaInterna.controller.js*, que validan que el contenido no esté vacío y que la negociación a la que está vinculada exista y esté activa. Asimismo, se creó el modelo *NotaInterna* con relación de los modelos *Negociacion* y *Usuario* definido en el archivo *schema.prisma*.

Figura 79. Código de NotasInternas.jsx

Figura 80. Función crearNotaInterna en notaInterna.controller.js

Además, con el fin de proteger la confidencialidad de las notas ante accesos directos a la base de datos, el contenido se almacena cifrado mediante el algoritmo *AES-256-GCM* (implementado en *cifrado.js* usando el módulo nativo de *Node.js* llamado "*crypto"*). La interfaz de las notas privadas de un agente inmobiliario sobre una negociación específica se observa en la figura 81.

Figura 81. Interfaz de las notas privadas de un agente en una negociación

4.3.5.2.5. Historia de usuario 20: Adjuntar archivos por negociación

Se desarrollaron los componentes *ModalArchivosAdjuntos.jsx* y *ArchivosAdjuntos.jsx* (figura 82) que permiten a los agentes dueños de la negociación y administradores subir, listar y descargar archivos vinculados a una negociación determinada. El formulario de subida de archivos admite archivos *PDF*, *JPG* y *PNG* con un límite de 5 *MB*, visible solo para el agente a cargo de la negociación o el administrador. Al confirmar la subida, se realiza una petición *POST /api/archivos-negociacion/subir* con el archivo y el tipo de documento seleccionado (reserva, promesa, minuta, pago u otros).

Por otro lado, la lista de archivos adjuntos se carga mediante *GET /api/archivos-negociacion/negociacion/:negociacionId* y la descarga se realiza a través de *GET /api/archivos-negociacion/descargar/:archivoId*, todos definidos en *archivoNegociacion.routes.js* y procesados por las funciones *subirArchivo*, *obtenerArchivos* y *descargarArchivo* en el archivo *archivoNegociacion.controller.js* (figura 83). La función *subirArchivo* recibe el archivo desde *req.file* mediante la configuración de *multer.js* y crea el registro en el modelo *ArchivoNegociacion* definido en el archivo *schema.prisma*. El apartado visual para adjuntar archivos en una negociación específica se refleja en la figura 84.

Figura 82. Código de ArchivosAdjuntos.jsx

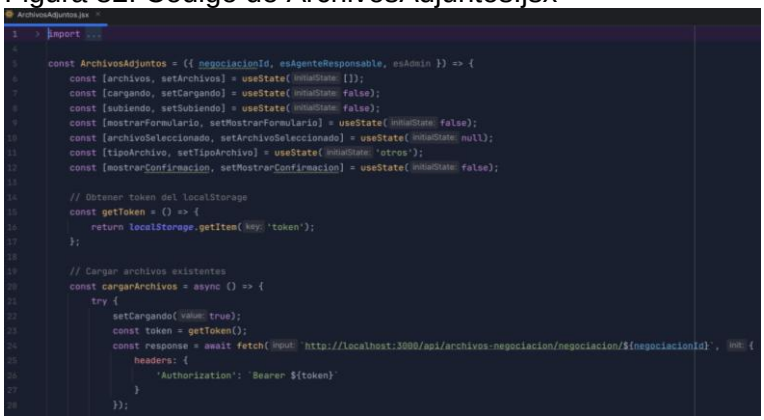
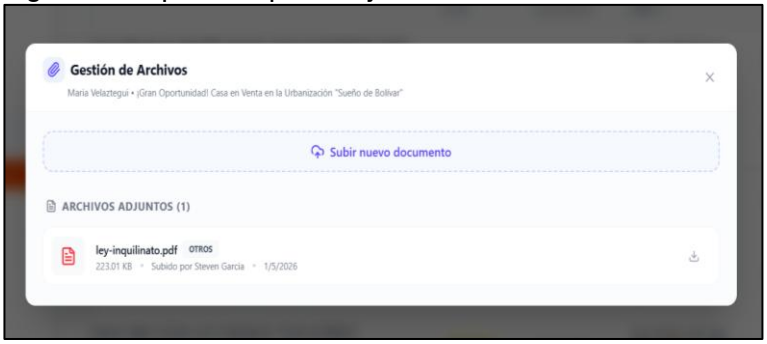


Figura 83. Función subirArchivo en archivoNegociacion.controller.js

Figura 84. Apartado para adjuntar archivos



4.3.5.2.6. Historia de usuario 21: Registro de agentes

La página *RegistrarAgente.jsx* (figura 85), renderiza el formulario que permite a los administradores crear nuevas cuentas de agentes inmobiliarios en el sistema. El formulario incluye campos obligatorios de nombre, cédula, email, teléfono y contraseña (mínimo 8 caracteres, al menos una mayúscula y un número). Además, en la sección de carga de documentos del agente, como se observa en la figura 86. Al enviar los datos del formulario, se ejecuta una petición *POST /api/agentes/crear*, definida en *agentes.routes.js* y procesada en la función *crearAgente* en el archivo *agentes.controller.js* (figura 87), que hace la validación de los datos y crea el usuario con rol de agente.

Figura 85. Código de RegistrarAgente.jsx

```
RegistrarAgente.jsx
export default function RegistrarAgente() {
    // Estados del formulario
    const [datos, setDatos] = useState({
        name: '',
        email: '',
        telefono: '',
        cedula: '',     // Nuevo campo
        direccion: '',  // Nuevo campo
        password: '',
        confirmPassword: ''
    });

    const [errores, setErrores] = useState({});

    // Visibilidad de contraseñas
    const [showPassword, setShowPassword] = useState(false);
    const [showConfirmPassword, setShowConfirmPassword] = useState(false);

    // Documentos
    const [documentos, setDocumentos] = useState({
        identificacion: [],
        contrato: [],
        hoja_vida: [],
        certificado: [],
        otro: []
    });

    // Referencia para comparación de cambios
    const initialDatos = {
        name: '',
        email: '',
```

Figura 86. Interfaz del formulario de registro de un agente

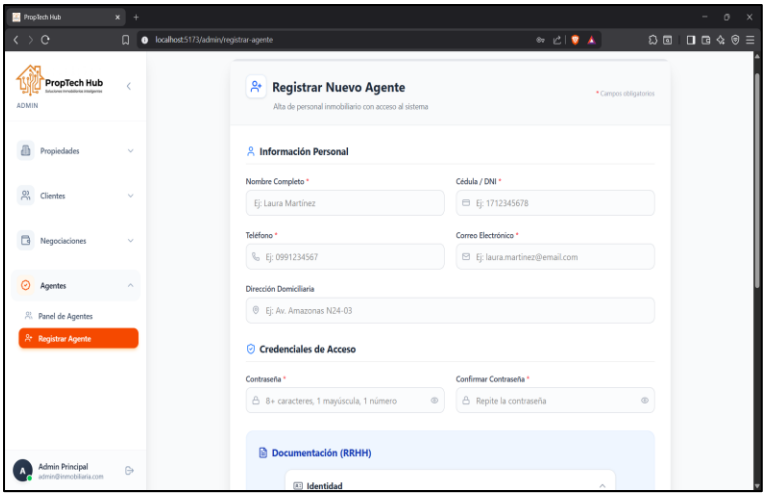


Figura 87. Función crearAgente en agentes.controller.js

4.3.5.2.7. Historia de usuario 22: Visualización de agentes

Se elaboró el componente *PanelAgentes.jsx* (figura 88), el cual permite a los administradores visualizar el listado completo de los agentes registrados en el sistema con columnas de nombre, email, teléfono, cantidad de propiedades, clientes y negociaciones activas, estado (activo/inactivo), así como se observa en la figura 89. Los datos de dichos agentes se cargan mediante el *endpoint GET /api/agentes*, definido en *agentes.routes.js* y procesado por la función *obtenerAgentes* de *agentes.controller.js* mostrada en la figura 90.

Figura 88. Código de PanelAgentes.jsx

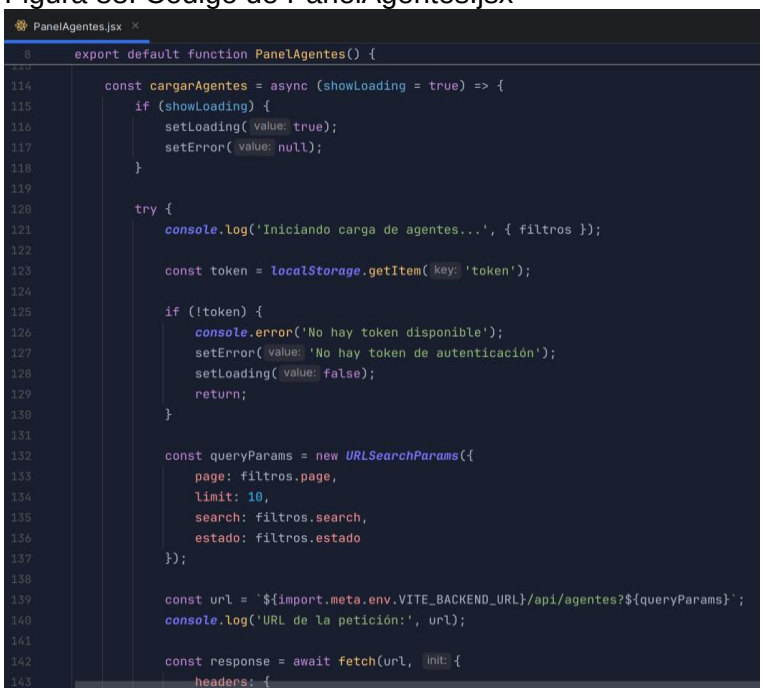


Figura 89. Interfaz del panel de agentes inmobiliarios registrados

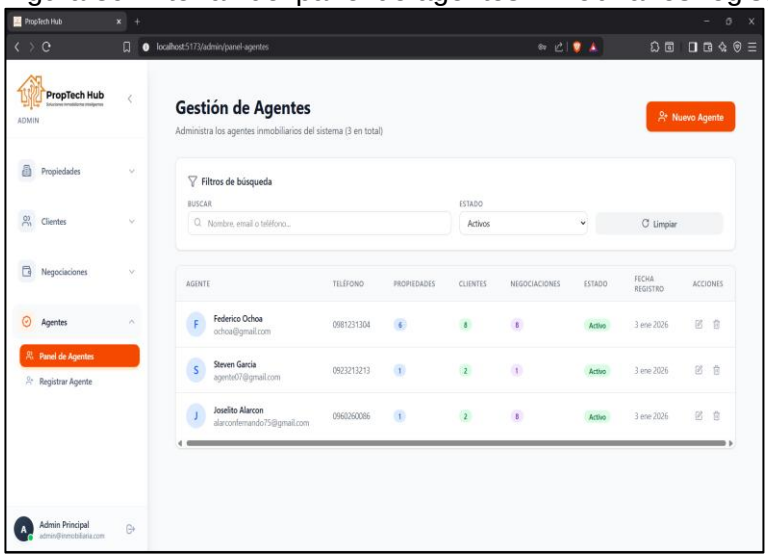


Figura 90. Función obtenerAgentes en agentes.controller.js

4.3.5.2.8. Historia de usuario 23: Edición de agentes

La página *EditarAgente.jsx* (figura 91), carga los datos actuales del agente seleccionado mediante *GET /api/agentes/:id*, definido en *agentes.routes.js* y procesado por la función *obtenerAgentePorId* de *agentes.controller.js*, que verifica la existencia del agente y retorna sus datos relacionados. El formulario pre-rellenado con esos datos, permite modificar nombre, cédula, email y teléfono, también incluye la subida y descarga de documentos del agente, como se observa en la figura 92. Al guardar, se envía una petición *PUT /api/agentes/:id* y se ejecuta la función controladora *actualizarAgente* (figura 93) que valida los campos y verifica que el correo y cédula sean únicos.

Figura 91. Código de EditarAgente.jsx

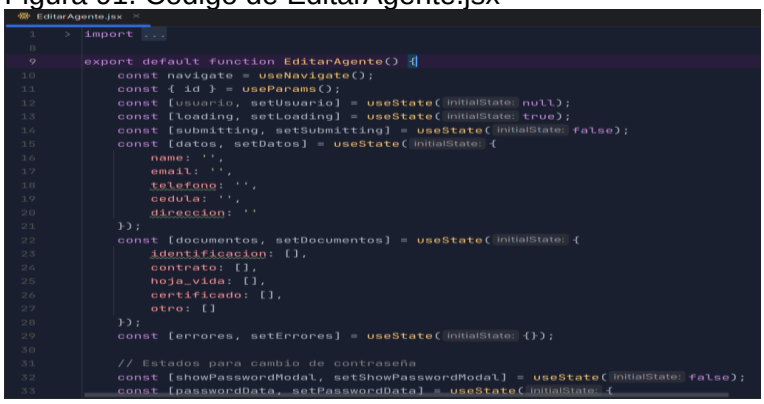


Figura 92. Interfaz del formulario de edición de un agente

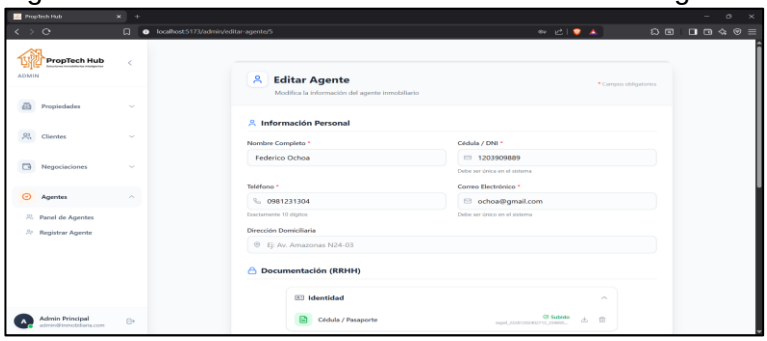
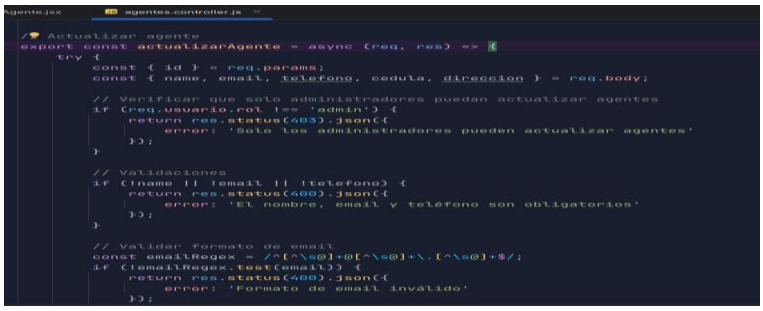


Figura 93. Función actualizarAgente en agentes.controller.js

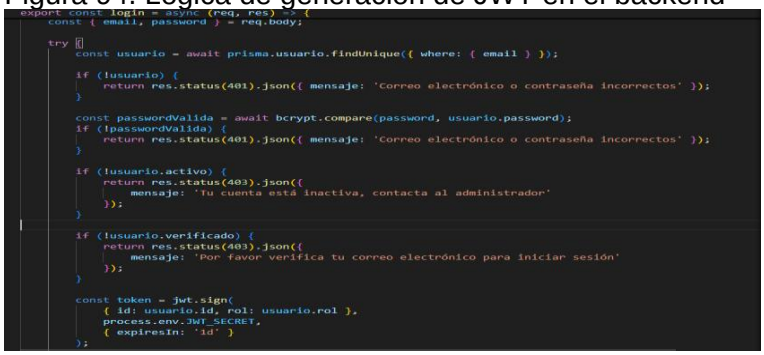


4.3.5.2.9. Historia Técnica 1: Autenticación y seguridad

La autenticación de la aplicación, se basa en *JSON Web Tokens* (JWT) para verificar la identidad de los usuarios en cada petición protegida. Cuando un usuario inicia sesión correctamente en la función *login* del archivo *auth.controller.js*, se genera un *token JWT* firmado mediante la librería *jsonwebtoken* utilizando una llave secreta (*JWT_SECRET*), tal como se observa en la figura 94.

El *token* contiene el *id* del usuario y su rol, permitiendo que el sistema identifique quién realiza la petición y qué permisos tiene asociados. El *token* tiene una expiración de 24 horas (1 día) para equilibrar seguridad y experiencia de usuario.

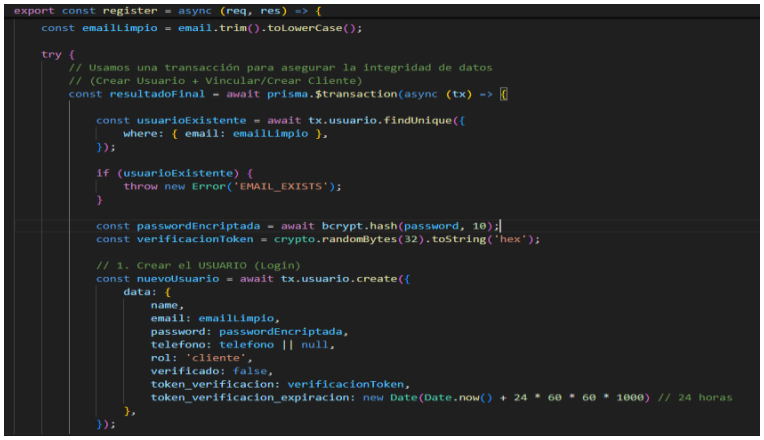
Figura 94. Lógica de generación de JWT en el backend



```
export const login = async (req, res) => {
    const { email, password } = req.body;

    try {
        const usuario = await prisma.usuario.findUnique({ where: { email } });

        if (!usuario) {
            return res.status(401).json({ mensaje: 'Correo electrónico o contraseña incorrectos' });
        }

        const passwordValida = await bcrypt.compare(password, usuario.password);
        if (!passwordValida) {
            return res.status(401).json({ mensaje: 'Correo electrónico o contraseña incorrectos' });
        }

        if (!usuario.activo) {
            return res.status(403).json({
                mensaje: 'Tu cuenta está inactiva, contacta al administrador'
            });
        }

        if (!usuario.verificado) {
            return res.status(403).json({
                mensaje: 'Por favor verifica tu correo electrónico para iniciar sesión'
            });
        }

        const token = jwt.sign(
            { id: usuario.id, rol: usuario.rol },
            process.env.JWT_SECRET,
            { expiresIn: '1d' }
        );
```

Para la protección de las credenciales (figura 95), las contraseñas nunca se almacenan en texto plano. En su lugar, se utiliza la librería *bcrypt* para generar un *hash* seguro. En la función *register* (y también en el restablecimiento de contraseñas), se utiliza *bcrypt*.*hash*(*password*, 10), donde 10 representa el número de rondas de procesamiento (*cost factor*), lo que garantiza una alta resistencia ante ataques de fuerza bruta o tablas de arcoíris.

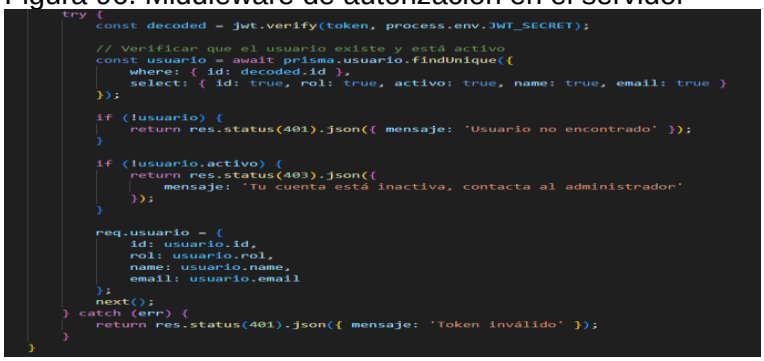
Figura 95. Lógica de encriptación de contraseñas en el registro de usuarios



```
export const register = async (req, res) => {
    const emailLimpio = email.trim().toLowerCase();

    try {
        // Usamos una transacción para asegurar la integridad de datos
        // (Crear Usuario + Vincular/Crear Cliente)
        const resultadoFinal = await prisma.$transaction(async (tx) => {

            const usuarioExistente = await tx.usuario.findUnique({
                where: { email: emailLimpio },
            });

            if (usuarioExistente) {
                throw new Error('EMAIL_EXISTS');
            }

            const passwordEncriptada = await bcrypt.hash(password, 10);
            const verificacionToken = crypto.randomBytes(32).toString('hex');

            // 1. Crear el USUARIO (Login)
            const nuevoUsuario = await tx.usuario.create({
                data: {
                    name,
                    email: emailLimpio,
                    password: passwordEncriptada,
                    telefono: telefono || null,
                    rol: 'cliente',
                    verificado: false,
                    token_verificacion: verificacionToken,
                    token_verificacion_expiracion: new Date(Date.now() + 24 * 60 * 60 * 1000) // 24 horas
                },
            });
```

En el *backend*, se definió el *middleware verificarToken.j*s que intercepta las peticiones enviadas a rutas protegidas. Este *middleware* extrae el token del encabezado *Authorization*, lo verifica y valida que el usuario asociado exista y esté activo en la base de datos. Una vez validado, adjunta la información del usuario al objeto *req* para que los controladores puedan aplicar reglas de negocio específicas según el rol, así como se muestra en la figura 96. Además, se cuenta con *middlewares* especializados como *esAdmin.js* y *esAgenteOAdmin.js* para restringir el acceso a *endpoints* específicos.

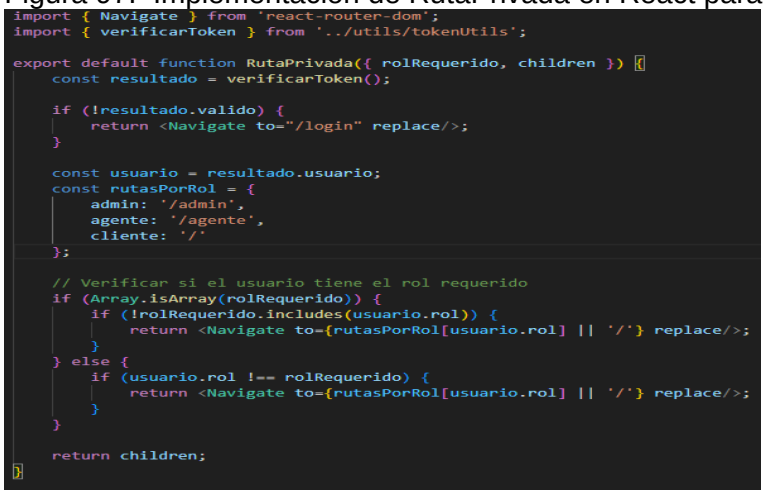
Figura 96. Middleware de autorización en el servidor



```
    try {
        const decoded = jwt.verify(token, process.env.JWT_SECRET);

        // Verificar que el usuario existe y está activo
        const usuario = await prisma.usuario.findUnique({
            where: { id: decoded.id },
            select: { id: true, rol: true, activo: true, name: true, email: true }
        });

        if (!usuario) {
            return res.status(401).json({ mensaje: 'Usuario no encontrado' });
        }

        if (!usuario.activo) {
            return res.status(403).json({
                mensaje: 'Tu cuenta está inactiva, contacta al administrador'
            });
        }

        req.usuario = {
            id: usuario.id,
            rol: usuario.rol,
            name: usuario.name,
            email: usuario.email
        };
        next();
    } catch (err) {
        return res.status(401).json({ mensaje: 'Token inválido' });
    }
}
```

En el *frontend (React)*, se implementó un componente de orden superior llamado *RutaPrivada.jsx*. Este componente actúa como un guardián de navegación global, evalúa si el usuario tiene un *token* válido almacenado en el *localStorage* mediante la utilidad *verificarToken* de *tokenUtils.js*. Si un usuario intenta acceder a una ruta protegida (como el Panel de Administración o la Gestión de Negociaciones) sin estar autenticado, es redirigido automáticamente a la página del *login*. Asimismo, se verifica que el rol del usuario coincida con el requerido por la ruta (por ejemplo, impidiendo que un "agente" acceda a funciones exclusivas de "*admin*"), como se refleja en la figura 97.

Figura 97. Implementación de RutaPrivada en React para protección de vistas



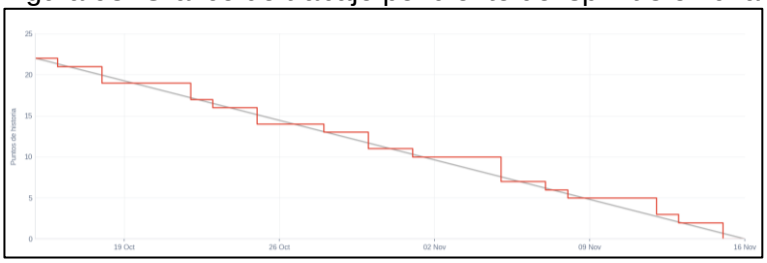
```
import { Navigate } from 'react-router-dom';
import { verificarToken } from '../utils/tokenUtils';

export default function RutaPrivada({ rolRequerido, children }) {
    const resultado = verificarToken();

    if (!resultado.valido) {
        return <Navigate to="/login" replace/>;
    }

    const usuario = resultado.usuario;
    const rutasPorRol = {
        admin: '/admin',
        agente: '/agente',
        cliente: '/'
    };

    // Verificar si el usuario tiene el rol requerido
    if (Array.isArray(rolRequerido)) {
        if (!rolRequerido.includes(usuario.rol)) {
            return <Navigate to={rutasPorRol[usuario.rol] || '/'} replace/>;
        }
    } else {
        if (usuario.rol !== rolRequerido) {
            return <Navigate to={rutasPorRol[usuario.rol] || '/'} replace/>;
        }
    }

    return children;
}
```

4.3.5.3. Sprint 3 - Gráfico de trabajo pendiente

En el *sprint* 3, el gráfico de trabajo pendiente generado por *Jira* (figura 98), representó el progreso de las historias de usuario: registro y actualización de etapa por negociación, historial de seguimiento, notas internas, adjuntar archivos por negociación, registro, visualización y edición de agentes inmobiliarios registrados. La gráfica demostró la

disminución de los puntos de historia restantes. Esta visualización permitió identificar los cuellos de botellas y tomar acciones correctivas para completar los objetivos del *sprint*.

Figura 98. Gráfico de trabajo pendiente del sprint 3 en Jira



4.3.5.4. Sprint 3 – Revisión

Al finalizar el *sprint* 3, se hizo la respectiva revisión del entregable con el *Product Owner*. Se presentaron los avances del módulo de negociaciones y gestión de agentes, y se verificó el progreso. Se comprobó los escenarios de pruebas y las pruebas de aceptación, dando como resultado que las funcionalidades cumplían con los requisitos y la aprobación del cierre del *sprint.*

4.3.5.5. Sprint 3 – Retrospectiva

Se realizaron tres preguntas fundamentales con el fin de obtener una perspectiva clara, completa y concisa, además de las dificultados al culminar el s*print* 3, se puede apreciar en la tabla 26.

Tabla 26. Sprint 3 - Retrospectiva

Aspectos exitosos en el Sprint	Mejoras que se podrían aplicar en el siguiente Sprint (recomendaciones para la mejora continua)	Errores en el Sprint
Durante el desarrollo del sprint 3 se logró implementar la gestión de agentes inmobiliarios y módulo de negociaciones con su historial de seguimiento y notas internas, el cual este último representa el núcleo del proceso comercial de la inmobiliaria	Para futuros sprint, se sugiere implementar un sistema de notificaciones en tiempo real para alertar a los agentes sobre cambios importantes en las negociaciones que gestionan. Adicionalmente, se recomienda optimizar el almacenamiento de archivos mediante servicios en la nube como AWS S3 o Cloudinary para mejorar la escalabilidad del sistema.	Durante el desarrollo del sprint 3 se presentaron dificultades con la funcionalidad de adjuntar archivos por negociación, la cual presentaba errores al momento de descargar archivos que superaban los 3 MB de tamaño, generando errores de timeout en el servidor. Este problema se resolvió implementando streaming de archivos y aplicando límite de tiempo de respuesta del servidor.

4.3.6. Sprint 4

4.3.6.1. Sprint 4 – Planificación

Para la planificación del *sprint* 4, se seleccionó las historias de usuario 24 al 27 del *Product Backlog* que sumaron 18 puntos de historia totales para culminar este *sprint*, de igual manera se elaboró el *sprint backlog* que corresponde a este *sprint* reflejado en la tabla 27.

Tabla 27. Sprint Backlog 4

Historia Usuario	Est. His.	Categoría	Tarea	Est. Tarea	Responsable	Estado
HU24 – Desactivación de agentes	2 pts	Backend	Implementación de endpoint PATCH /api/agentes/:id/estado para desactivar/reactivar agente y bloqueo si tiene responsabilidades activas (sin reasignación). (agentes.controller.js, agentes.routes.js)	1.5 pts	Jhery Alarcon	Completado
		Frontend	Panel con filtros por estado (activo/inactivo), etiqueta de estado y acción desactivar/reactivar con modales. (PanelAgentes.jsx)	0.5 pts	Romario Flores	Completado
HU25 – Recomendación de propiedades personalizada	8 pts	Base de datos	Creación de modelo InteraccionUsuario para capturar interacciones de usuario. (schema.prisma)	0.5 pts	Jhery Alarcon	Completado
		Backend	Implementación de endpoint POST /api/interacciones para registrar y ponderar vistas, favoritos y contactos. (interaccion.controller.js, interaccion.routes.js)	1.5 pts	Jhery Alarcon	Completado
		Backend	Implementación de endpoint GET /api/propiedades/recomendaciones que consolida perfil y llama al microservicio de machine learning (propiedad.controller.js, propiedad.routes.js)	1 pt	Jhery Alarcon	Completado
		Machine Learning	Creación de microservicio aislado con Flask y recepción de cargas útiles en JSON. (app.py)	1 pt	Romario Flores	Completado
		Machine Learning	Elaboración de modelo predictivo basado en contenido con TruncatedSVD y similitud coseno. (recomendaciones_service.py)	1.5 pts	Jhery Alarcon	Completado
		Backend	Desarrollo de bloque de tolerancia a fallos en JavaScript (Algoritmo K-Nearest Neighbors -	1 pt	Romario Flores	Completado

		KNN) para garantizar recomendaciones en caso de caída del servicio de IA de Python. (propiedad.controller.js)			
	Frontend	Creación del de componente visual para mostrar listado de propiedades recomendadas. (Recomendaciones.jsx)	1 pt	Romario Flores	Completado
	Frontend	Inyección de llamados a la API en el detalle de propiedad y formularios para registrar la interacción del usuario. (DetallePropiedad.jsx, ContactoForm.jsx)	0.5 pts	Romario Flores	Completado
HU26 – Recuperación de contraseña de la cuenta	5 pts	Backend	Implementación de endpoint POST /api/auth/forgot-password con token temporal y envío de correo (auth.controller.js)	2 pts	Jhery Alarcon	Completado
		Backend	Implementación de endpoint POST /api/auth/reset-password/:token con validación de token/expiración y política de contraseña (auth.controller.js)	1.5 pts	Jhery Alarcon	Completado
		Frontend	Creación de formulario para recuperación de contraseña. (RecuperarPassword.jsx)	0.75 pts	Romario Flores	Completado
		Frontend	Creación de formulario para poner una nueva contraseña. (NuevaPassword.jsx)	0.75 pts	Romario Flores	Completado
HU27 – Detalle de propiedad (panel administrativo)	3 pts	Backend	Implementación de endpoint GET /api/propiedades/:id que trae detalle con imágenes (propiedad.controller.js, propiedad.routes.js)	1.5 pts	Jhery Alarcon	Completado
		Frontend	Creación de vista administrativa con pestañas, galería, acordeones y mapa. (DetallePropiedadAdmin.jsx)	1.5 pts	Romario Flores	Completado

4.3.6.2. Sprint 4 - Reuniones diarias

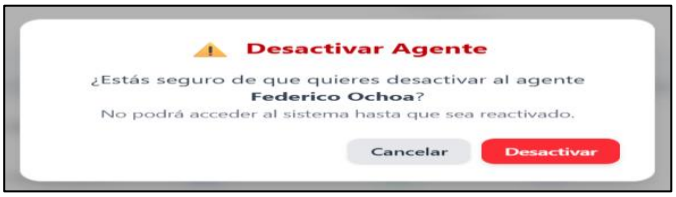
Durante el desarrollo del *sprint* 4, se realizaron reuniones diarias en complemento del uso de la herramienta de gestión de incidencias *Jira* para verificar los avances diarios de cada integrante del equipo. Asimismo, se llevó un control de las tareas finalizadas de acuerdo con la planificación.

4.3.6.2.1. Historia de usuario 24: Desactivación de agentes

La desactivación de agentes se implementó como una eliminación lógica (*soft delete*) para evitar la pérdida de historial en la inmobiliaria. En el *frontend*, dentro del

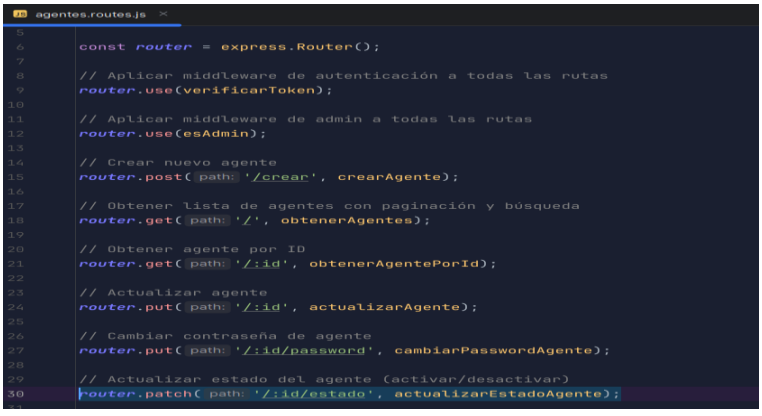
componente *PanelAgentes.jsx*, el usuario con rol de administrador visualiza el listado de agentes con su estado que está representado mediante una etiqueta de color y un botón con ícono de papelera para desactivar (si está activo) o reactivar (si está inactivo). Al hacer clic en el botón de desactivar de la fila del agente a suspender, no se ejecuta la acción de inmediato, sino que se despliega un modal de confirmación *showDesactivarModal* el cual se renderiza como se muestra en la figura 99.

Figura 99. Modal para la confirmación de la desactivación de un agente



Una vez que el administrador confirma la desactivación en el modal abierto, el *frontend* realiza una petición *PATCH* al *backend* hacia la ruta */api/agentes/:id/estado*, definida en *agentes.routes.js* (figura 100). Esta ruta está protegida por el *middleware esAdmin.js*, el cual intercepta la petición y asegura que solo un administrador autenticado pueda ejecutarla.

Figura 100. Ruta para activar o desactivar un agente en agentes.routes.js



Superada la validación, la acción llega a la función *actualizarEstadoAgente* que está dentro del controlador *agentes.controller.js* (figura 101). Usando *Prisma*, el *backend* realiza tres consultas a la base de datos para asegurar si el agente a ser desactivado aún tiene

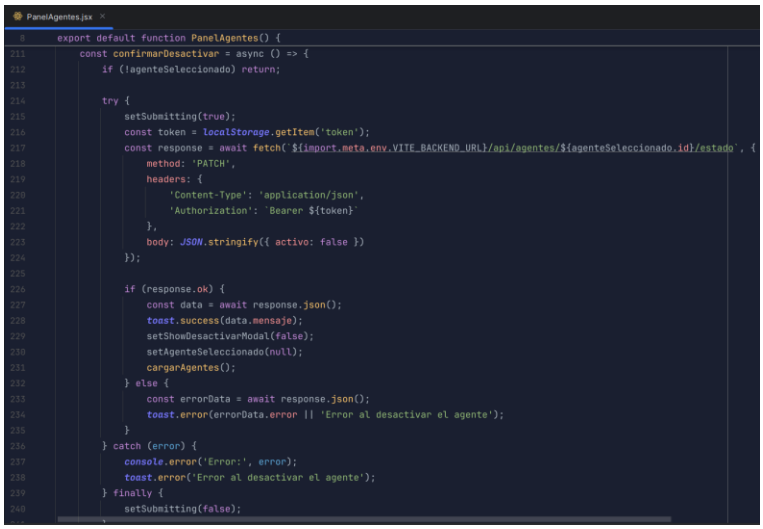
responsabilidades vigentes como clientes activos asignados, propiedades disponibles/reservadas, o negociaciones en curso.

Figura 101. Función actualizarEstadoAgente en agente.controller.js

Si la suma de estos pendientes es mayor a cero, el controlador impide la desactivación y retorna un error detallando qué responsabilidades se tienen que gestionar, ya que el propósito es forzar al administrador a reasignar las carteras manualmente antes de inhabilitar al agente.

Finalmente, el *frontend* recibe esta respuesta. Si es un error por responsabilidades activas, la función *confirmarDesactivar* en *PanelAgentes.jsx* (figura 102) captura el mensaje y lanza una alerta explicando el fallo. Si, por el contrario, el *backend* determina que la cartera está vacía, procede a actualizar el campo activo a *false*. El *frontend* recibe el éxito, cierra el modal, muestra una alerta de confirmación y recarga automáticamente la tabla de agentes para reflejar que el agente ahora está inactivo.

Figura 102. Código de la función confirmarDesactivar en PanelAgentes.jsx



4.3.6.2.2. Historia de usuario 25: Recomendación de Propiedades

La presente historia tiene como objetivo mostrar al cliente registrado un listado de propiedades recomendadas de forma personalizada, basadas en su comportamiento de navegación dentro de la aplicación. Tal como se observa en las figuras 103 y 104.

Figura 103. Sección de recomendaciones personalizadas al cliente con historial

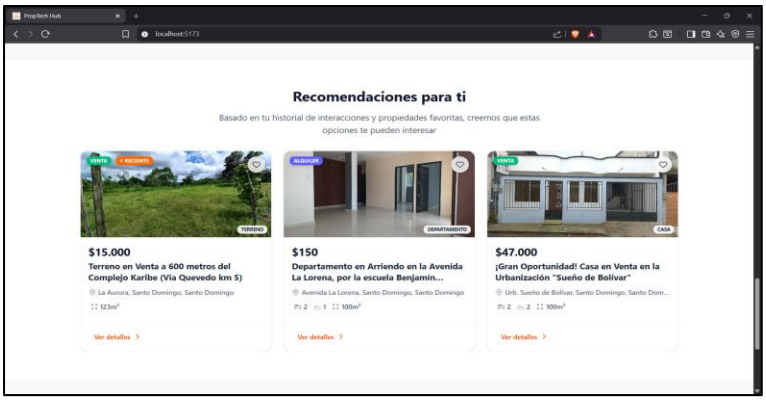
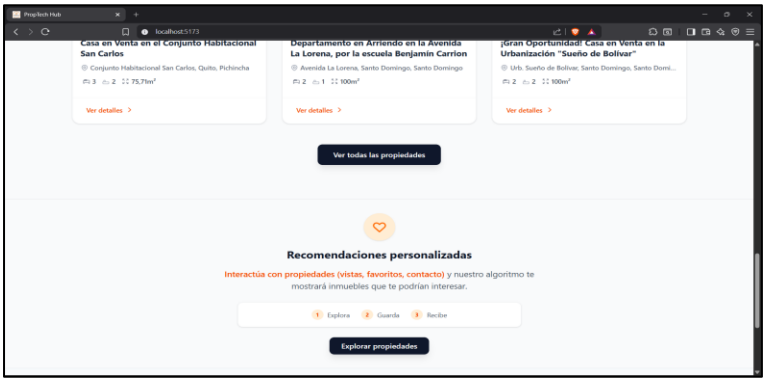


Figura 104. Sección de recomendaciones personalizadas al cliente sin historial



Para la creación del sistema de recomendación, se empleó un servicio *backend* en *Node.js* responsable de recopilar las interacciones del usuario, y un microservicio en *Python* con *Flask* que ejecuta el modelo de *machine* learning. El motor de recomendación desarrollado pertenece al campo del aprendizaje automático no supervisado, empleando específicamente técnicas de Reducción de Dimensionalidad. El algoritmo seleccionado fue la Descomposición en Valores Singulares Truncada (*TruncatedSVD*) aplicada sobre la matriz de características de las propiedades, combinada con la métrica vectorial de Similitud del Coseno. Este modelo opera bajo un enfoque de Filtrado Basado en Contenido, gestionado a través de Retroalimentación Implícita Ponderada. Este diseño permite

identificar patrones entre las características de las propiedades y en base a ello construir un perfil de las preferencias del usuario.

Para capturar el comportamiento del usuario, se implementó un modelo de datos enfocado en registrar acciones de intención (*Implicit Feedback*). Esto se diseñó a nivel de base de datos relacional mediante *ORM Prisma*, creando modelos específicos para las interacciones (*InteraccionUsuario*) y las listas de inmuebles favoritas (*Favorito*), tal como se observa en la figura 105. Por otro lado, en el *frontend* los archivos *DetallePropiedad.jsx* y *ContactoForm.jsx* registran las interacciones del usuario (vistas, favoritos y contactos) enviándolas al *backend* para entrenar al modelo. Por su parte, *Recomendaciones.jsx* se encarga de mostrar en pantalla las recomendaciones personalizadas de inmuebles que el algoritmo *SVD* daría para ese usuario.

Figura 105. Modelos Favorito e InteraccionUsuario para interacciones de usuario

```
model Favorito {
  id          Int       @id @default(autoincrement())
  usuarioId   Int
  propiedadId Int
  createdAt   DateTime  @default(now())
  propiedad   Propiedad @relation(fields: [propiedadId], references: [id])
  usuario     Usuario   @relation(fields: [usuarioId], references: [id])

  @@unique([usuarioId, propiedadId])
}

model InteraccionUsuario {
  id          Int              @id @default(autoincrement())
  usuarioId   Int
  propiedadId Int
  tipo        TipoInteraccion
  valor_peso  Int              // VISTA=1, FAVORITO=5, CONTACTO=10
  fecha       DateTime         @default(now())

  usuario     Usuario          @relation(fields: [usuarioId], references: [id])
  propiedad   Propiedad        @relation(fields: [propiedadId], references: [id])

  @@unique([usuarioId, propiedadId, tipo])
  @@index([usuarioId])
  @@index([propiedadId])
}
```

El sistema registra tres tipos de interacciones, cada una con un peso que refleja el nivel de intención de conversión del cliente, reflejado en la tabla 28.

Tabla 28. Sistema de pesos para las interacciones del usuario en la plataforma

Tipo de interacción	Peso asignado
Vista – Análisis del detalle de una propiedad	1
Favorito – Almacenamiento de la propiedad como favorita para seguimiento posterior	5
Contacto – Envío del formulario de contacto	10

Las interacciones son procesadas a través del *endpoint POST /api/interacciones* y *GET /api/propiedades/recomendaciones*, cuyas rutas están definidas en el *backend* (*interaccion.routes.js, propiedad.routes.js*) que son procesadas por las funciones

registrarInteraccion en el archivo *interaccion.controller.js* y *obtenerRecomendaciones* en el archivo *propiedad.controller.js*, respectivamente. Tal como se observa en las figuras 106 y 107.

Figura 106. Función registrarInteraccion en interaccion.controller.js

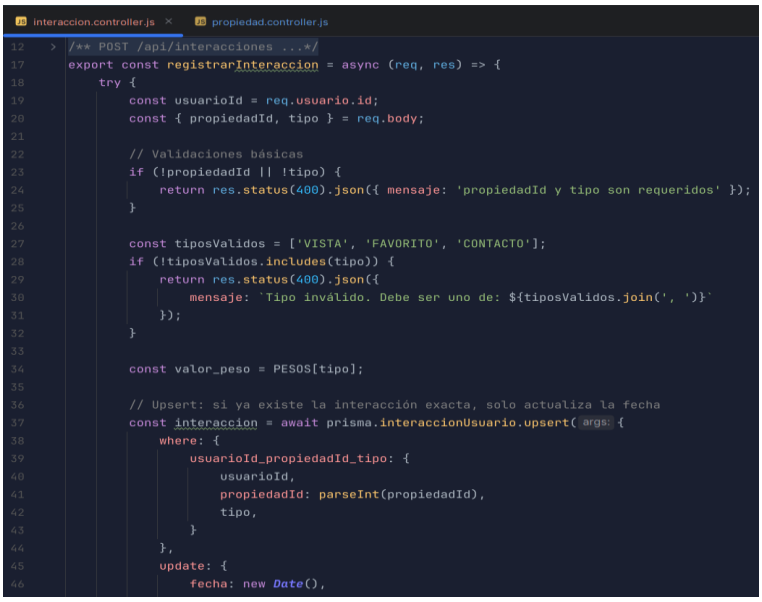
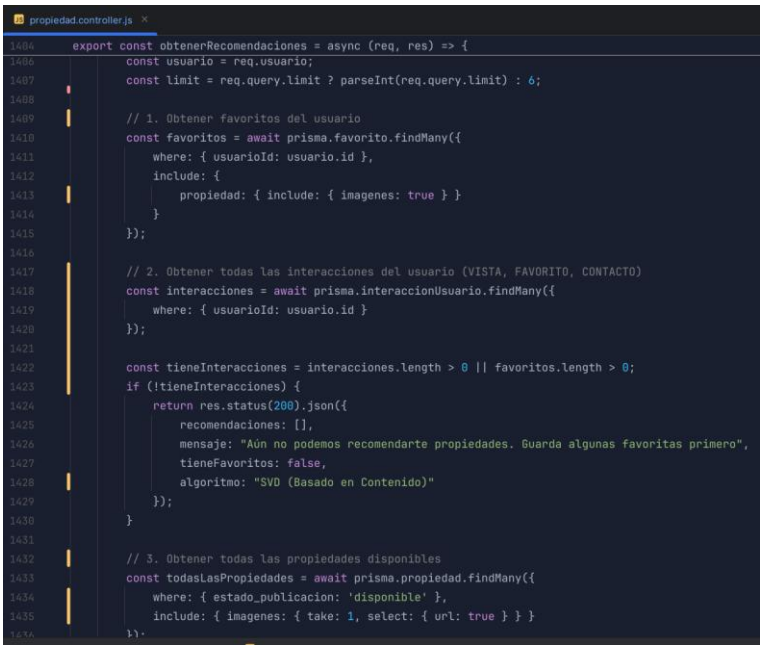


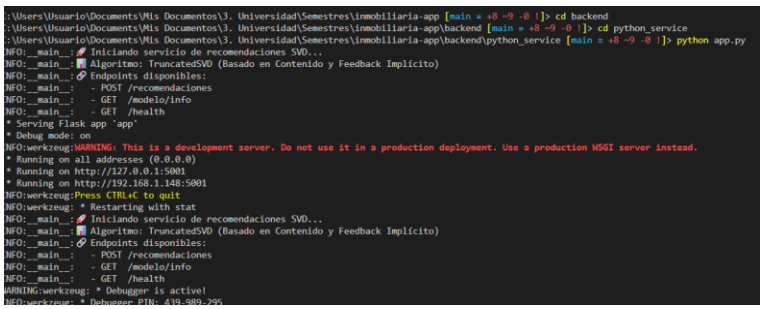
Figura 107. Función obtenerRecomendaciones en propiedad.controller.js



La lógica central del algoritmo en el microservicio de *Python,* se encuentra en el archivo *recomendaciones_service.py* dentro de la función *procesar_recomendaciones*. En el caso de que el motor de *Python* experimenta algún fallo temporal, se cuenta con un

mecanismo de tolerancia a fallos (*fallback*) en *JavaScript*, que aplica un algoritmo *K-Nearest Neighbors* (*KNN*) simplificado utilizando distancias euclidianas ponderadas para garantizar que el usuario siempre reciba recomendaciones de manera ininterrumpida. Asimismo, las recomendaciones se mantienen en ejecución mediante *python app.py*, levantando un servidor *Flask*, como se ejemplifica en la figura 108.

Figura 108. Servidor Flask activo en microservicio Python



Por otra parte, el proceso de recomendación sigue cuatro pasos: primero, convierte cada propiedad en un vector numérico de 29 dimensiones que codifica sus atributos como precio, superficie, número de habitaciones y baños, tipo de inmueble, modalidad de negocio (venta o alquiler) y amenidades disponibles como piscina, seguridad, ascensor, entre otras. Segundo, se construye el perfil del cliente como la suma ponderada de los vectores de las propiedades con las que interactuó, normalizada por el peso total acumulado. Tercero, se aplica *TruncatedSVD* (Descomposición en Valores Singulares Truncada, 15 componentes) para reducir la dimensionalidad del espacio y descubrir patrones en el catálogo. Finalmente, se calcula la similitud coseno entre el perfil del cliente y cada inmueble candidato, devolviendo los 6 con mayor afinidad que el cliente aún no haya visitado.

Para poder medir qué tan bien funciona el sistema de recomendación, se necesita contar con un conjunto de datos suficientemente amplio y variado de clientes, propiedades e interacciones entre ambos. Dado que la base de datos real del sistema no cuenta con ese volumen durante la etapa de desarrollo, se generaron datos sintéticos de manera controlada mediante tres *scripts* de *Python* ubicados en el directorio b*ackend/python_service/data/*.

Propiedades: El *script generar_propiedades_sinteticas.py* construye un catálogo de 150 inmuebles con *np.random.seed(42)* para garantizar reproducibilidad. Cada propiedad se genera respetando coherencia interna: el precio sigue una distribución normal parametrizada por tipo (casas ~USD 120.000, departamentos ~USD 75.000, terrenos ~USD 40.000), las habitaciones y baños se correlacionan con el tipo de inmueble. Además, la ubicación se distribuye entre siete ciudades del Ecuador ponderadas por actividad inmobiliaria real (Quito 30%, Guayaquil 25%, Cuenca 15%, entre otras), y las amenidades tienen probabilidades condicionadas al precio y tipo del inmueble. Tal como se ilustra en la figura 109.

Figura 109. Script de datos sintéticos de propiedades

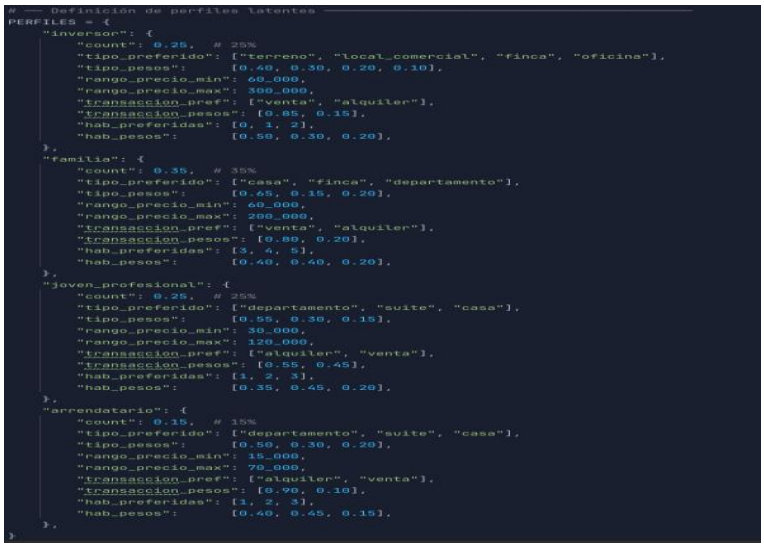
```
np.random.seed(42)

RAW_DIR = Path(__file__).parent / "raw"
RAW_DIR.mkdir(parents=True, exist_ok=True)

# ── Distribuciones de tipo de propiedad ──────────────────────────────
TIPOS = {
    "casa":           {"peso": 0.35, "precio_mu": 120_000, "precio_sigma": 50_000, "hab_mu": 3.5, "ban_mu": 2.5},
    "departamento":   {"peso": 0.28, "precio_mu":  75_000, "precio_sigma": 30_000, "hab_mu": 2.5, "ban_mu": 1.8},
    "suite":          {"peso": 0.08, "precio_mu":  55_000, "precio_sigma": 20_000, "hab_mu": 1.0, "ban_mu": 1.0},
    "terreno":        {"peso": 0.12, "precio_mu":  40_000, "precio_sigma": 25_000, "hab_mu": 0.0, "ban_mu": 0.0},
    "local_comercial": {"peso": 0.09, "precio_mu":  90_000, "precio_sigma": 40_000, "hab_mu": 0.0, "ban_mu": 1.0},
    "finca":          {"peso": 0.05, "precio_mu": 180_000, "precio_sigma": 80_000, "hab_mu": 3.0, "ban_mu": 2.0},
    "oficina":        {"peso": 0.03, "precio_mu":  60_000, "precio_sigma": 25_000, "hab_mu": 0.0, "ban_mu": 1.0},
}

# ── Ciudades y provincias de Ecuador ─────────────────────────────────
CIUDADES = [
    ("Quito",         "Pichincha",     0.30),
    ("Guayaquil",     "Guayas",        0.25),
    ("Cuenca",        "Azuay",         0.15),
    ("Santo Domingo", "Santo_Domingo", 0.10),
    ("Ambato",        "Tungurahua",    0.08),
    ("Manta",         "Manabi",        0.06),
    ("Loja",          "Loja",          0.06),
]

TRANSACCIONES = [("venta", 0.62), ("alquiler", 0.38)]


def _tipo_aleatorio():
```

Clientes: El *script generar_clientes_sinteticos.py* produce 60 perfiles de cliente organizados en cuatro arquetipos de comprador: inversor (25%), orientado a terrenos y locales comerciales con presupuesto elevado; familia (35%), enfocado en casas de tres o más habitaciones en rango de precio medio; joven profesional (25%), que prefiere departamentos o *suites* en venta o alquiler; y arrendatario (15%), concentrado en propiedades en modalidad de arriendo a bajo costo, tal como se ilustra en la figura 110.

Las interacciones: El script *generar_datos_sinteticos.py* evalúa los 9.000 pares posibles entre los 60 clientes y las 150 propiedades, calculando para cada par un puntaje

de afinidad (0 a 1) compuesto por cuatro criterios: coincidencia del tipo preferido (30%), ajuste al rango presupuestario (30%), modalidad de negocio concordante (25%) y amenidades de interés (15%).

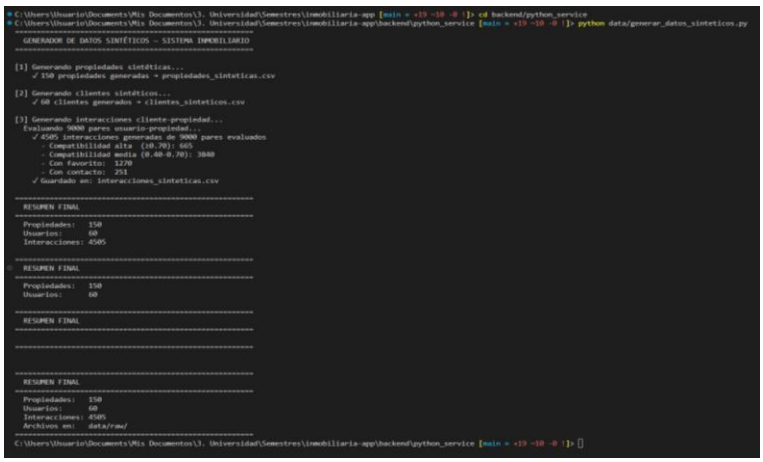
Figura 110. Script de datos sintéticos de clientes



También, es importante mencionar aquellos con afinidad alta (≥0.70), que generan visualizaciones, favorito probable y posible contacto; afinidad media (0.40–0.69) origina una o dos visitas, afinidad baja descarta la interacción. El resultado es un conjunto de 4.505 interacciones, con 1.270 favoritos y 251 contactos, exportadas en tres archivos *CSV* en *data/raw/.* Tal como se ilustra en las figuras 111 y 112.

Figura 111. Script de interacciones clientes – propiedades

Figura 112. Ejecución de script para generar datos sintéticos



Para que el algoritmo pueda comparar propiedades entre sí y con el perfil del cliente, cada inmueble se convierte en una lista de 29 valores numéricos. Los primeros seis capturan atributos físicos como precio, superficie y número de habitaciones. Los siguientes diez indican el tipo de propiedad mediante un esquema donde solo uno de los valores vale 1 y los demás 0; los tres siguientes hacen lo mismo con el tipo de transacción; y los últimos diez señalan la presencia o ausencia de amenidades con un peso mayor (valor 2 cuando está presente) para darle más relevancia frente a los demás atributos. Esta transformación permite al sistema trabajar matemáticamente con los datos sin perder información sobre las características del inmueble, tal como se ilustra en la figura 113.

Figura 113. Script de vectorización de propiedades

```python
def _vectorizar_propiedad(prop: Dict) -> np.ndarray:
    # ── Características numéricas normalizadas ──────────────────
    precio = float(prop.get('precio') or 0)
    feats.append(min(precio / 500_000, 3.0))                    # Precio
    feats.append(float(prop.get('nro_habitaciones') or 0) / 6.0)
    feats.append(float(prop.get('nro_banos') or 0) / 5.0)
    feats.append(float(prop.get('nro_parqueaderos') or 0) / 4.0)
    feats.append(min(float(prop.get('area_construccion') or 0) / 500.0, 2.0))
    feats.append(min(float(prop.get('area_terreno') or 0) / 1000.0, 2.0))

    # ── Tipo de propiedad (one-hot) ──────────────────────────────
    tipos = ['casa', 'departamento', 'suite', 'local_comercial', 'terreno',
             'finca', 'quinta', 'oficina', 'bodega_galpon', 'edificio']
    tipo_actual = prop.get('tipo_propiedad', '')
    feats.extend([1.0 if tipo_actual == t else 0.0 for t in tipos])

    # ── Tipo de transacción (one-hot) ────────────────────────────
    transacciones = ['venta', 'alquiler', 'alquiler_opcion_compra']
    trans_actual = prop.get('transaccion', '')
    feats.extend([1.0 if trans_actual == t else 0.0 for t in transacciones])

    # ── Amenidades (peso 2x para reforzar su importancia) ────────
    amenidades = [
        'tiene_piscina', 'tiene_seguridad', 'tiene_ascensor', 'tiene_area_bbq',
        'tiene_terraza', 'tiene_balcon', 'tiene_patio', 'tiene_bodega',
        'tiene_areas_comunales', 'amoblado'
    ]
    for am in amenidades:
        feats.append(2.0 if prop.get(am) else 0.0)

    return np.array(feats, dtype=np.float32)
```

Para evaluar la calidad del sistema de recomendación basado en contenido, se utilizaron las métricas de *ranking* estándar de los de sistemas de recomendación. El proceso de evaluación toma el 80% de las interacciones de cada cliente como su historial conocido, construye su perfil vectorial y genera recomendaciones aplicando el mismo algoritmo de producción. Luego compara esas recomendaciones contra el 20% restante de interacciones (que el sistema no conocía) para medir su acierto. Se considera relevante toda propiedad con la que el cliente tuvo una interacción de peso mayor o igual a 6, es decir, la marcó como favorita o envió el formulario. La evaluación se calcula para K = 1, 2, 3, 5 y 10 recomendaciones. Tal como se ilustra en la figura 114.

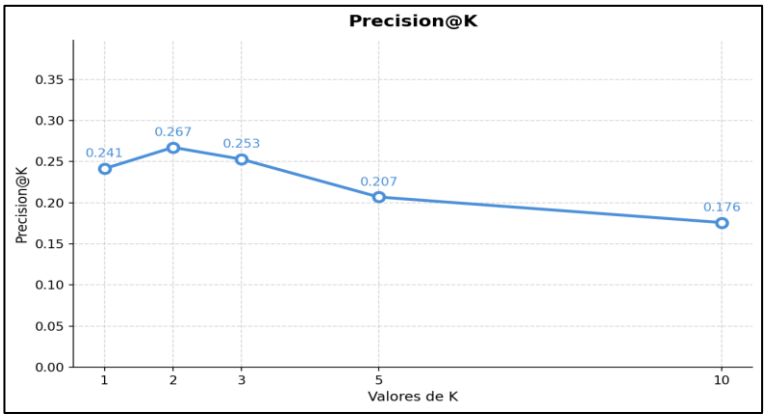
Figura 114. Script de evaluación de métricas

```python
def evaluate():
    print("=" * 57)
    print("  EVALUACION DE METRICAS — SISTEMA BASADO EN CONTENIDO")
    print("=" * 57)

    # ── 1. Cargar datos ──────────────────────────────────────────
    inter_path = DATA_DIR / "interacciones_sinteticas.csv"
    props_path = DATA_DIR / "propiedades_sinteticas.csv"

    for p in [inter_path, props_path]:
        if not p.exists():
            print(f"\n[ERROR] No se encontro: {p}")
            print("  Ejecuta primero: python data/generar_datos_sinteticos.py")
            return

    inter_df = pd.read_csv(inter_path)
    props_df  = pd.read_csv(props_path)

    # Puntaje de interacción: vistas×1 + favorito×5 + contacto×10
    inter_df["peso"] = (
        inter_df["clicks"] * 1 +
        inter_df["favorito"] * 5 +
        inter_df["contacto"] * 10
    )

    print(f"\n[1] {len(inter_df)} interacciones | {len(props_df)} propiedades")

    # ── 2. Split 80/20 ──────────────────────────────────────────
    train_df, test_df = train_test_split(inter_df, test_size=0.20, random_state=RANDOM_STATE)
    print(f"[2] Split 80/20 -> Historial (train): {len(train_df)} | Prueba (test): {len(test_df)}")
```

Las métricas de *ranking* aplicadas son: *Precision@K*, proporción de los *K* inmuebles recomendados que resultaron relevantes para el cliente. Responde a "¿qué tan acertadas son las propiedades que el sistema presenta primero?", tal como se ilustra en la figura 115, se observa que la precisión alcanza su valor óptimo (26.7%) en las primeras dos recomendaciones (K=2). Posteriormente, a medida que la lista mostrada se amplía (K=5 y K=10), la métrica disminuye de forma natural hasta un 17.6%. Esto ocurre porque el

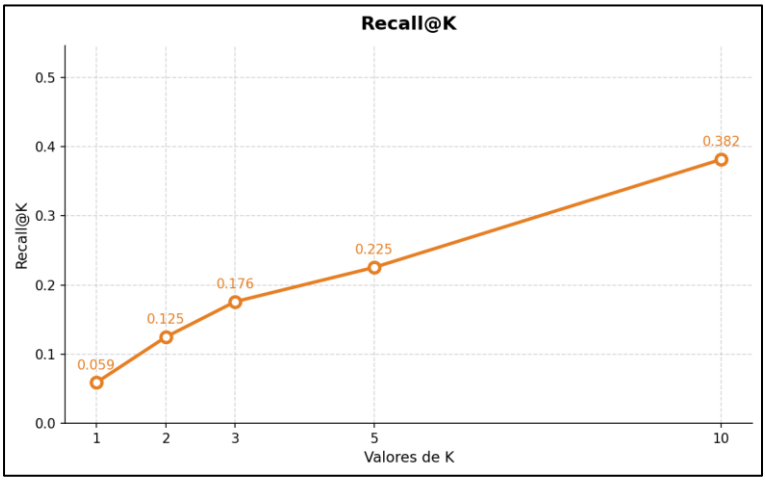
algoritmo se ve obligado a incluir propiedades con menor relevancia relativa para completar el total de sugerencias exigidas.

Figura 115. Gráfica Precision@K



Recall@K: fracción del total de propiedades relevantes para el cliente que el sistema logró incluir dentro de sus primeras *K* recomendaciones. Responde a "¿cuántas de las propiedades que le interesarían al cliente aparecen en la lista?", tal como se ilustra en la figura 116, esta métrica presenta un crecimiento continuo y positivo, logrando su punto máximo de 0.382 (38.2%) en K=10. Esto demuestra un alto nivel de efectividad al mostrar apenas 10 propiedades, el portal público consigue entregarle al usuario casi el 40% de todo el inventario que realmente le interesa.

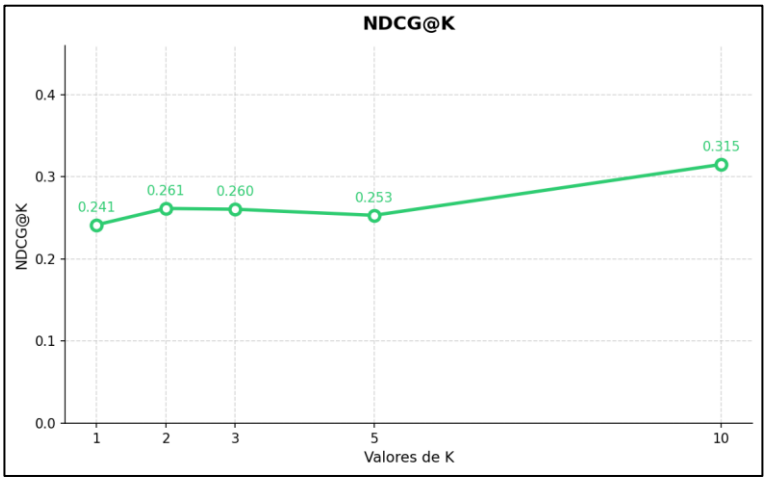
Figura 116. Gráfica Recall@K



NDCG@K (*Normalized Discounted Cumulative Gain*): pondera la posición de cada acierto dentro de la lista, los inmuebles relevantes que aparecen en las primeras posiciones

contribuyen más que los ubicados al final. Es la métrica más exigente de las tres, pues evalúa simultáneamente la relevancia y el orden de presentación. Tal como se ilustra en la figura 117, el modelo presenta un rendimiento muy estable en sus primeras posiciones (manteniendo valores entre 0.24 y 0.26 desde K=1 hasta K=5). Esto demuestra que el algoritmo logra ordenar las propiedades de mayor interés en los primeros lugares de la lista. Finalmente, el incremento de la métrica a 0.315 (31.5%) en K=10 confirma que la estructura general de las recomendaciones es de alta calidad y cumple satisfactoriamente con el propósito de sugerir opciones útiles al usuario.

Figura 117. Gráfica NDCG@K

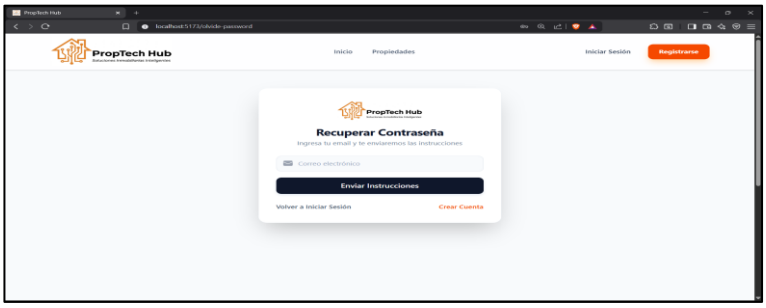


4.3.6.2.3. Historia de usuario 26: Recuperación de contraseña

La aplicación cuenta con la funcionalidad de recuperación de contraseña, la cual permite a los usuarios restablecer su contraseña. La página *RecuperarPassword.jsx* (figura 118) renderiza un pequeño formulario. Como se visualiza en la figura 119, que captura el correo electrónico del usuario y al pulsar el botón “Enviar Instrucciones” realiza una petición *POST /api/auth/forgot-password*, implementada en *auth.controller.js*, que verifica que el correo electrónico exista, genera un *token* único con un tiempo de expiración de una hora por motivos de seguridad y lo envía al usuario por correo electrónico.

Figura 118. Código de RecuperarPassword.jsx

Figura 119. Interfaz de recuperación de contraseña



Una vez recibido el enlace *web*, el usuario accede al formulario bajo la página *NuevaPassword.jsx* (figura 120) que, muestra la interfaz que corresponde a la figura 121, donde ingresa y confirma una nueva contraseña bajo la política de contraseña de seguridad (mínimo 8 caracteres, al menos una letra mayúscula y un número). Al enviar, se ejecuta una petición de tipo *POST* a la ruta */api/auth/reset-password/:token*, también en *auth.controller.js*, que valida la existencia y vigencia del *token*, además almacena la nueva clave cifrada mediante algoritmo *hash*.

Figura 120. Código de NuevaPassword.jsx

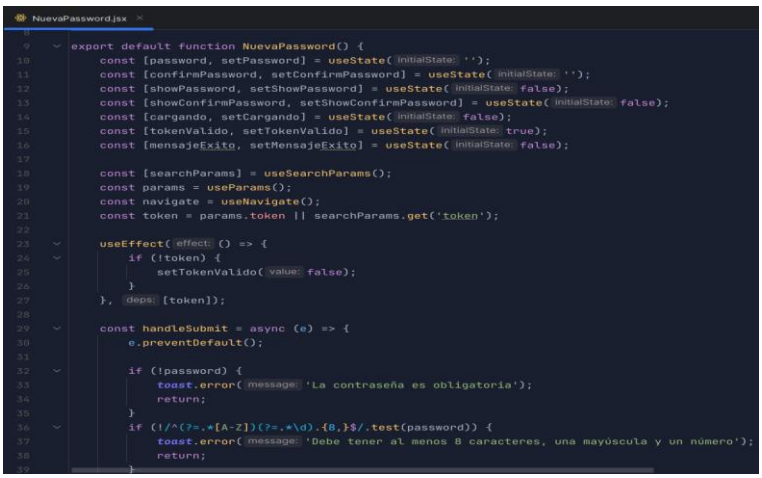
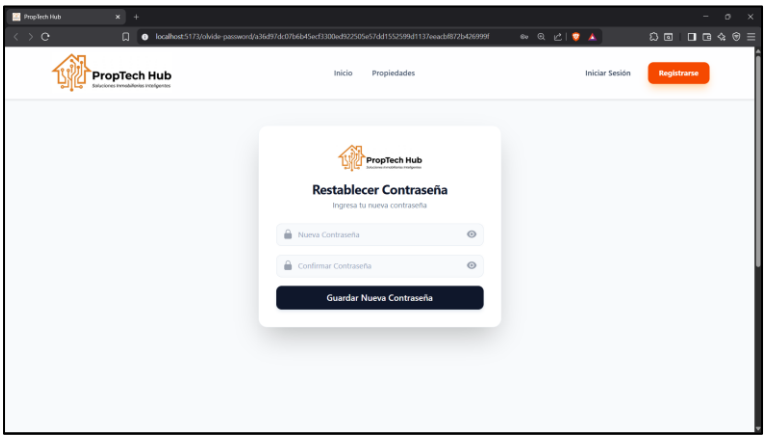


Figura 121. Interfaz para ingresar y confirmar una nueva contraseña



4.3.6.2.4. Historia de usuario 27: Detalle de propiedad (administrativo)

Se elaboró la página *DetallePropiedadAdmin.jsx* (figura 122), que permite a administradores y agentes visualizar la ficha técnica completa de una determinada propiedad, incluyendo información la cual no es visible en el portal público tales como precio mínimo de venta, porcentaje de comisión e información concerniente al contrato realizado con un cliente vendedor/arrendador.

Figura 122. Código de DetallePropiedadAdmin.jsx

```
DetallePropiedadAdmin.jsx
export default function DetallePropiedadAdmin() {

    const scroll = (direction) => {
        if (scrollContainerRef.current) {
            const { current } = scrollContainerRef;
            const scrollAmount = 300;
            if (direction === 'left') {
                current.scrollBy({ left: -scrollAmount, behavior: 'smooth' });
            } else {
                current.scrollBy({ left: scrollAmount, behavior: 'smooth' });
            }
        }
    };

    useEffect(() => {
        const token = localStorage.getItem('token');
        if (!token) {
            navigate('/login');
            return;
        }

        try {
            const decoded = jwtDecode(token);
            setUsuario(decoded);

            // Verificar que sea admin o agente
            if (decoded.rol !== 'admin' && decoded.rol !== 'agente') {
                toast.error('No tienes permisos para acceder a esta página');
                navigate('/');
                return;
            }
```

En la interfaz organiza la información mediante pestañas y acordeones, los datos como imágenes del inmueble, agente asignado o captador, propietarios con los porcentajes de su cuota de participación y documentos legales, se obtienen mediante el *endpoint GET*

/api/propiedades/:id, definido en *propiedad.routes.js* y gestionado por la función *obtenerPropiedadPorId* (figura 123) en *propiedad.controller.js*.

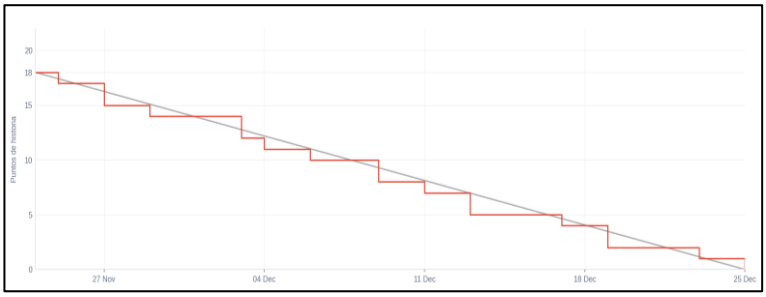
Figura 123. Función obtenerPropiedadPorId en propiedad.controller.js

```
propiedad.controller.js
export const obtenerPropiedadPorId = async (req, res) => {
    const { id } = req.params;
    const usuario = req.usuario;

    try {
        const propiedad = await prisma.propiedad.findUnique({
            where: { id: parseInt(id) },
            include: {
                imagenes: true,
                agente: {
                    select: { id: true, name: true, email: true },
                },
                propietarios: {
                    include: { cliente: true }
                },
                negociaciones: {
                    where: { activo: true },
                    select: { id: true, etapa: true }
                },
                documentos: true
            },
        });

        if (!propiedad) {
            return res.status(404).json({ mensaje: 'Propiedad no encontrada' });
        }

        if (usuario.rol === 'cliente') {
            return res.status(403).json({ mensaje: 'Acceso denegado' });
        }
```

4.3.6.2.5. Sprint 4 - Gráfico de trabajo pendiente

En el *sprint* 4, el gráfico de trabajo pendiente generado por *Jira* (figura 124), representó el progreso de las historias de usuario: Desactivación de agentes, Recomendación de propiedades, Recuperación de contraseña de la cuenta y Detalle de propiedad (panel administrativo). La gráfica demostró la disminución de los puntos de historia restantes. Esta visualización permitió identificar los cuellos de botellas y tomar acciones correctivas para completar los objetivos del *sprint*.

Figura 124. Gráfico de trabajo pendiente del sprint 4 en Jira



4.3.6.3. Sprint 4 – Revisión

Al finalizar el *sprint* 4, el *Product Owner* verificó el incremento en conjunto de las pruebas de aceptación las cuales portaban los escenarios de prueba. En dicho incremento, se revisaron las historias de usuario: Desactivación de agentes, Recomendación de propiedades, Recuperación de contraseña de la cuenta y Detalle de propiedad (panel administrativo) dando como resultado que las funcionalidades cumplían con los requisitos y la aprobación del cierre del *sprint*.

4.3.6.4. Sprint 4 – Retrospectiva

Se realizaron tres preguntas fundamentales con el fin de obtener una perspectiva clara, completa y concisa, además de las dificultados al culminar el s*print* 4, se puede apreciar en la tabla 29.

Tabla 29: Sprint 4 - Retrospectiva

Aspectos exitosos en el Sprint	Mejoras que se podrían aplicar en el siguiente Sprint (recomendaciones para la mejora continua)	Errores en el Sprint
Se completó las historias y tareas asignadas en el tiempo establecido. Se implemento el machine learning y la recuperación de contraseña de la cuenta.	Para futuros sprint, se sugiere ampliar los conocimientos de las herramientas utilizadas durante el proyecto.	Durante el proceso de desarrollo del sprint 4, se encontró varias dificultades al implementar machine learning en la aplicación dado la poca experiencia del Development Team en esta área.

4.4. Resultado general

Resultado general, consistió en la implementación del sistema, la validación de la propuesta y la comprobación de la hipótesis.

4.4.1. Implementación del sistema

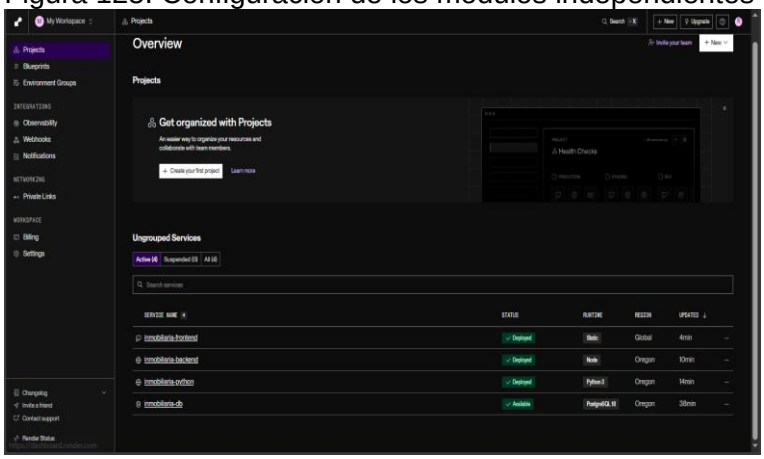
La implementación de la aplicación *web* de gestión inmobiliaria consistió en el despliegue e integración de sus tres componentes principales: la base de datos relacional, los servicios lógicos del sistema (*Backend API REST*) y la aplicación cliente encargada de la interfaz de usuario (*Frontend*).

Con el objetivo de garantizar la disponibilidad, integridad y accesibilidad permanente del sistema, se optó por un modelo de infraestructura en la nube basado en servicios de Infraestructura como Servicio y Plataforma como Servicio (*IaaS/PaaS*), proporcionados por el proveedor tecnológico *Render*. Este enfoque permitió desplegar cada componente del sistema de forma independiente, facilitando su mantenimiento, escalabilidad y administración.

4.4.1.1. Arquitectura de Despliegue en la Nube

La arquitectura tecnológica implementada, se diseñó bajo un enfoque de servicios modulares independientes (figura 125). Este paradigma permite separar adecuadamente las responsabilidades de cada componente del sistema y facilita la escalabilidad de la plataforma, así como su adaptación ante futuras mejoras o ampliaciones.

Figura 125. Configuración de los módulos independientes



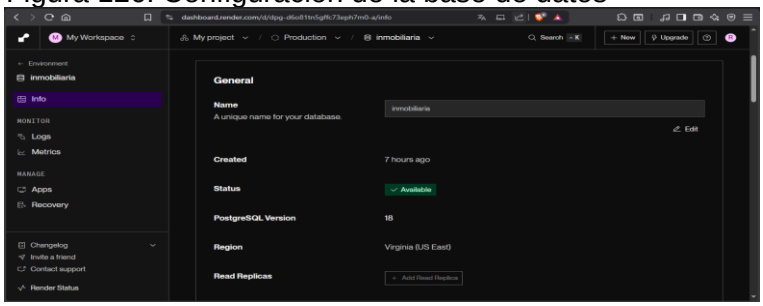
4.4.1.2. Configuración de la base de datos PostgreSQL

Para garantizar la persistencia y gestión estructurada de la información, se implementó una instancia administrada de la base de datos *PostgreSQL* en la infraestructura de *Render*, localizada en la región correspondiente (*US West*).

Durante el proceso de configuración, se realizaron diversas operaciones fundamentales, entre ellas la creación del proyecto de la base de datos, la habilitación de los mecanismos de la conectividad entre servicios y la estructuración del esquema de datos del sistema (figura 126). Para la definición y sincronización de las entidades de la base de

datos se utilizó la herramienta *Prisma ORM* (Object-Relational Mapping), la cual permitió automatizar los procesos de migración y generación del esquema relacional.

Figura 126. Configuración de la base de datos



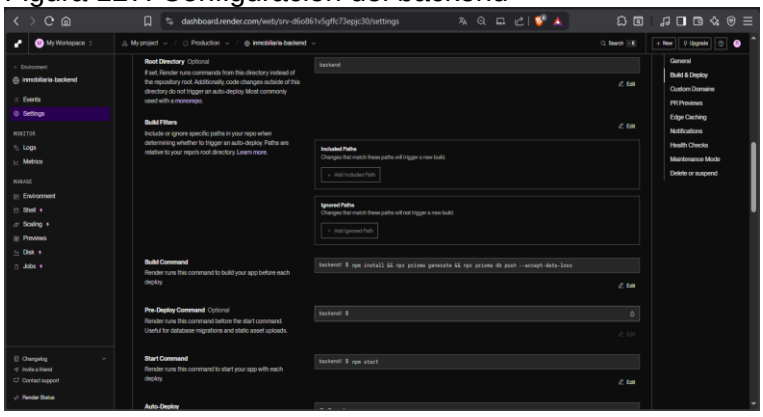
A través de estas migraciones, se crearon las diferentes entidades requeridas por el sistema, tales como usuarios administradores, clientes, propiedades inmobiliarias y registros de negociaciones. De igual manera, se establecieron las relaciones entre las entidades, incluyendo claves primarias, claves foráneas y restricciones de integridad referencial, con el propósito de mantener la consistencia y confiabilidad de la información almacenada.

4.4.1.3. Despliegue del Servicio Lógico (*Backend*)

El componente encargado de la lógica de negocio del sistema (*Backend*) fue desarrollado utilizando el entorno de ejecución *Node.js* y desplegado en la plataforma Render como un servicio *web* independiente (figura 127).

Este servicio centraliza el procesamiento de las solicitudes provenientes del *frontend*, gestionando operaciones de validación de datos, autenticación de usuarios mediante tokens seguros *JSON Web Token* (*JWT*), así como la comunicación con la base de datos y otros servicios requeridos por la aplicación.

Figura 127. Configuración del backend

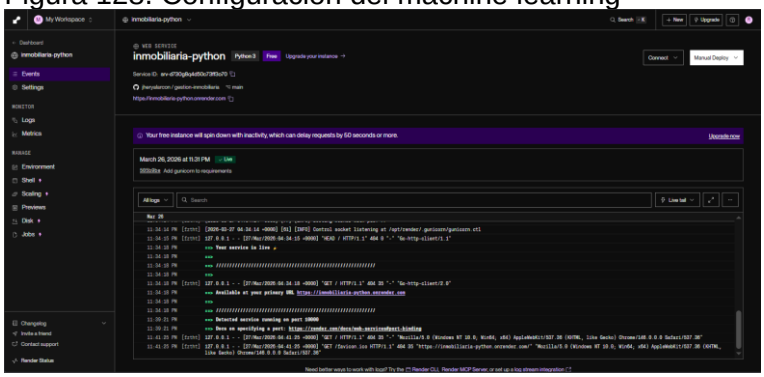


Para la puesta en producción del *backend* fue necesario configurar diversos parámetros técnicos dentro de la plataforma de despliegue:

1. **Integración Continua (CI):** Se estableció la vinculación directa con el repositorio del proyecto alojado en *GitHub*, permitiendo automatizar los procesos de construcción y despliegue cada vez que se realizan actualizaciones en el código fuente.
2. **Variables de Entorno (*Environment Variables*):** Se configuraron variables de entorno para la gestión segura de parámetros sensibles del sistema, incluyendo la cadena de conexión a la base de datos (*DATABASE_URL*), la clave secreta utilizada para la generación de tokens de autenticación (*JWT_SECRET*) y variables relacionadas con el entorno de ejecución del servidor (*NODE_ENV*).
3. **Procesos de Construcción (*Build Commands*):** Durante el despliegue se ejecutaron comandos necesarios para preparar el entorno productivo del sistema, tales como *npx prisma generate y npx prisma db push*, los cuales permiten generar dinámicamente los modelos de acceso a la base de datos y sincronizar la estructura del esquema relacional en el entorno de producción.

Por otro lado, se desplegó un segundo servicio independiente (*Web Service*) en *Python,* destinado exclusivamente al módulo de *Machine Learning*. Este microservicio, ejecutado en producción a través del servidor *Gunicorn*, se encarga del procesamiento algorítmico de recomendaciones basado en *TruncatedSVD* y de exponer los *endpoints* necesarios para comunicarse con la *API* principal. La separación del motor inteligente en un servicio aislado permite una mayor modularidad arquitectónica, asegurando que el mantenimiento, la carga computacional y la escalabilidad del sistema de inteligencia artificial se gestionen de forma completamente independiente al componente transaccional del *backend*, tal como se observa en la figura 128.

Figura 128. Configuración del machine learning



4.4.1.4. Despliegue de la Aplicación Cliente (*Frontend*)

El componente de interfaz de usuario fue desarrollado utilizando la biblioteca *React,* y optimizado mediante el empaquetador *Vite*. Este módulo fue desplegado como un sitio *web* estático (*Static Site*) dentro de la plataforma *Render*, permitiendo publicar la aplicación web accesible a los usuarios finales (figuras 129 y 130).

El *frontend* constituye la capa de interacción entre el sistema y los usuarios, proporcionando las funcionalidades necesarias para la gestión de propiedades, clientes, negociaciones y demás operaciones disponibles dentro de la plataforma.

Figura 129. Configuración del Frontend

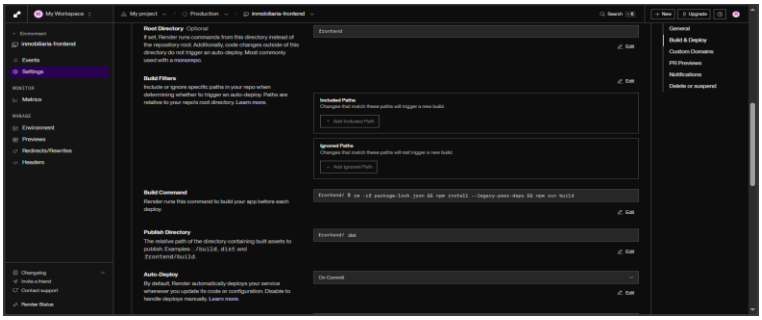
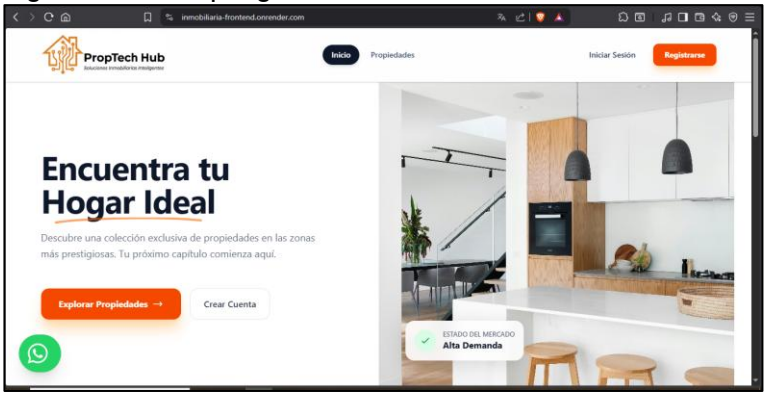


Figura 130. Despliegue del Frontend



Finalmente, la correcta integración entre el *frontend* y el *backend* requirió la configuración de dos elementos fundamentales para la comunicación del sistema:

1. **Asignación de la dirección del servidor:** Se definieron variables de entorno dentro del *frontend*, como *VITE_BACKEND_URL*, las cuales permiten establecer la dirección del servicio *backend* encargado de procesar las solicitudes del sistema.
2. **Reglas de redirección HTTP:** Se configuraron directivas de reescritura de rutas en *Render* (*Rewrites* /* to */index.html*), las cuales permiten garantizar el correcto funcionamiento del enrutamiento interno de la aplicación de una sola página (*Single Page Application - SPA*). Esta configuración evita errores de navegación como el conocido *HTTP 404 Not Found*, permitiendo que las rutas internas del sistema se gestionen adecuadamente desde el cliente.

Como último punto, la aplicación estuvo en producción desde el 16/01/2026. Asimismo, el *pretest* se realizó el 19/11/2025, después de tres meses se realizó el *postest* el 18/02/2026.

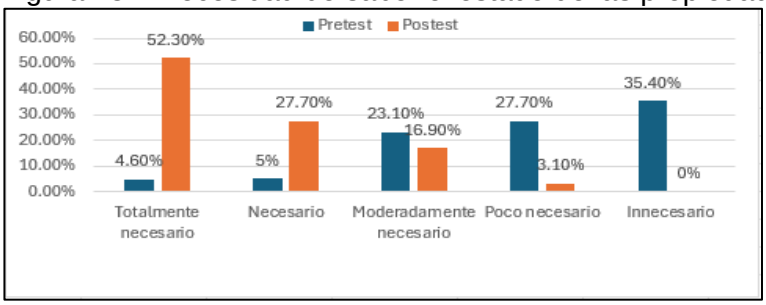
4.4.2. Validación de la propuesta

Por medio de la ejecución de las encuestas a una muestra de 65 clientes de la Empresa Escudero del Cantón Santo Domingo, se presentó el nivel de satisfacción de los clientes respecto a la aplicación *web*, tomando como referencia las preguntas incluidas en el instrumento de recolección de datos en el anexo III.

Para ello, se aplicó un *pretest* y un *postest* a una muestra de 65 clientes que utilizaron la aplicación *web* para la gestión inmobiliaria. Los resultados evidenciaron un cambio en la percepción de las respuestas relacionadas a la aplicación *web* para la gestión inmobiliaria. En la comparación realizada, se analizaron 5 preguntas específicas de las 49 que conformaban la encuesta.

Pregunta 1: ¿Qué tan necesario le resulta saber si una propiedad inmobiliaria se encuentra disponible?

Figura 131. Necesidad de saber el estado de las propiedades

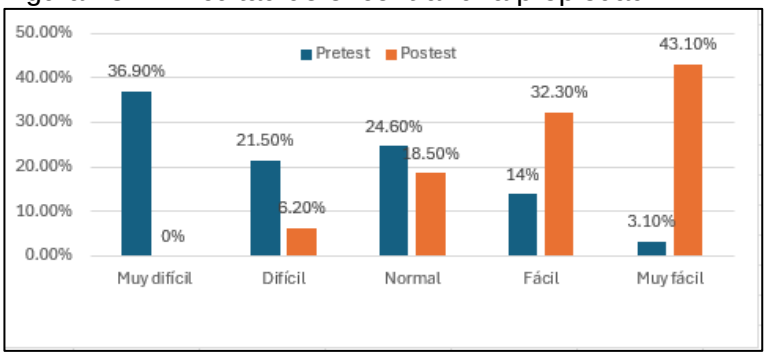


Análisis e interpretación: Los resultados del *pretest* y *postest* en relación con la necesidad de conocer si una propiedad inmobiliaria está disponible, se evidencian en la figura 131. En el *pretest*, la categoría “totalmente necesario” registra un 4.60% y “necesario” un 5%, mientras que en el *postest* estos valores aumentan a 52.30% y 27.70% respectivamente. En contraste, “moderadamente necesario” disminuye de 23.10% a 16.90%, “poco necesario” de 27.70% a 3.10% e “Innecesario” de 35.40% a 0%.

Los resultados indican que, tras la implementación de la solución, se incrementa el 47.70 % en “totalmente necesario” y el 22.70% en “necesario”. Por otro lado, se evidencia una reducción de 24.60% en “poco necesario” e “Innecesario” en 35.40%. En conjunto, se observa una tendencia clara en la percepción de los clientes en conocer si una propiedad inmobiliaria está disponible.

Pregunta 2: ¿Qué tan difícil le resulta encontrar una propiedad inmobiliaria de su interés?

Figura 132. Dificultad de encontrar una propiedad

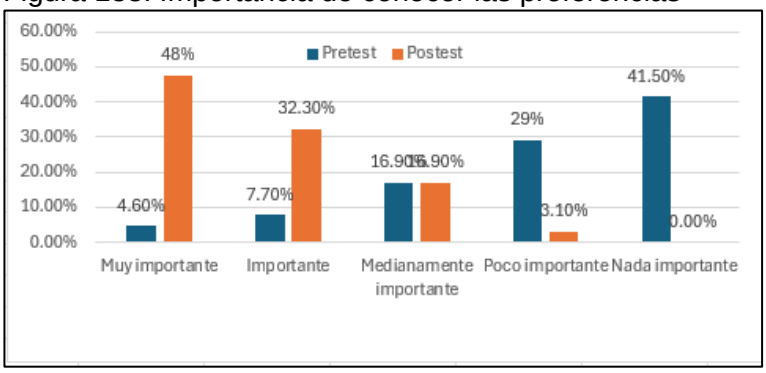


Análisis e Interpretación: Los datos obtenidos en el *pretest* y *postest* respecto al nivel de dificultad para encontrar una propiedad inmobiliaria, se presentan en la figura 132. Durante el *pretest*, la categoría “muy difícil” alcanza el 36.90% y “difícil” el 21.50%, en comparación con el 0% y 6.20% respectivamente en el *postest*. Por otro lado, “normal” disminuye de 24.60% a 18.50%, mientras que “fácil” incrementa del 14% a 32.30% y “muy fácil” de 3.10% a 43.10%.

Los resultados evidencian que, tras la implementación de la solución propuesta, los usuarios perciben una notable mejora en la facilidad para encontrar propiedades inmobiliarias de su interés. Esto se refleja en la reducción de las categorías “muy difícil” en un 36.90% y “difícil” en un 15.30%. Por lo contrario, el incremento de “fácil” en un 18.30%, y “muy fácil” en 40%. En conjunto, se observa una clara transición hacia niveles de menor dificultad y mayor accesibilidad en la búsqueda de propiedades.

Pregunta 3: ¿Qué tan importante es para usted que el agente inmobiliario conozca sus preferencias?

Figura 133. Importancia de conocer las preferencias

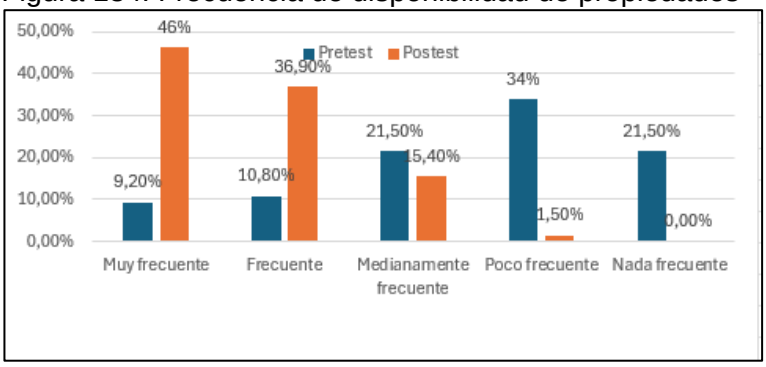


Análisis e Interpretación: Los datos obtenidos en el *pretest* y *postest* con respecto a la importancia de que el agente inmobiliario conozca las preferencias del cliente se presentan en la figura 133. Durante el pretest, la categoría “muy importante” registra un 4.60% e “importante” un 7.70%, en contraste, se incrementa con el 48% y 32.30% respectivamente en el *postest*. Por otro lado, “medianamente importante” se mantiene en 16.90% en ambas fases, mientras que “poco importante” disminuye de 29% a 3.10% y “nada importante” de 41.50% a 0%.

Los resultados evidencian que, tras la implementación de la solución, los usuarios perciben la importancia de que un agente conozca sus preferencias. Esto se refleja en el incremento en la percepción de los clientes en “muy importante” en un 43.40%, e “importante” en un 24.60%, junto con la reducción de “poco importante” en un 25.90%, y “nada importante” en 41.50%. En consecuencia, se observa una tendencia clara hacia una mayor valoración a que el agente inmobiliario conozca las preferencias del cliente.

Pregunta 4: ¿Con qué frecuencia encuentra propiedades disponibles?

Figura 134. Frecuencia de disponibilidad de propiedades

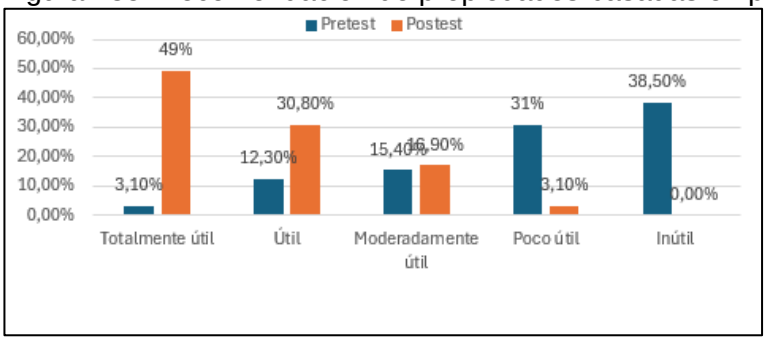


Análisis e Interpretación: Los resultados obtenidos en el *pretest* y *postest* con respecto a la frecuencia con la que los usuarios encuentran propiedades disponibles se presentan en la figura 134. Durante el *pretest*, la categoría “muy frecuente” registra un 9.20% y “frecuente” un 10.80%, en comparación con el 46% y 36.90% respectivamente en el *postest*. Por otro lado, “medianamente frecuente” disminuye de 21.50% a 15.40%, mientras que “poco frecuente” desciende de 34% a 1.50% y “nada frecuente” de 21.50% a 0%.

Los resultados indican que, tras la implementación de la solución, los usuarios perciben que, con mayor frecuencia encuentran propiedades disponibles. Esto se refleja en el incremento de “muy frecuente” en un 36.80% y “frecuente” en un 26.10%, junto con la disminución de “poco frecuente” de 32.50% y “nada frecuente” en un 21.50%. Por consecuencia, se observa una mejora clara en la percepción de disponibilidad de propiedades dentro de la plataforma.

Pregunta 5: ¿Considera útil que se le muestre propiedades basadas en las que ha consultado o marcado como favoritas previamente?

Figura 135. Recomendación de propiedades basadas en preferencias



Análisis e Interpretación: La información recopilada en el *pretest* y *postest* con respecto a la utilidad de mostrar las propiedades basadas en las consultas o marcaciones como favoritos, se presentan en la figura 135. Durante el *pretest*, la categoría “totalmente útil” alcanza el 3.10% y “útil” el 12.30%, en comparación con el 49% y 30.80% respectivamente en el *postest*. Por otro lado, “moderadamente útil” pasa de 15.40% a 16.90%, mientras que “poco útil” disminuye de 31% a 3.10% e “inútil” de 38.50% a 0%.

Los resultados demuestran que, tras la implementación de la solución, los usuarios perciben como totalmente útil que se muestre las propiedades basadas en las consultas o marcaciones como favoritos. Esto se refleja en el incremento de “totalmente útil” en un 45.90% y “útil” en un 18.50%, junto con la reducción de “poco útil” de 27.90%. En conjunto, se observa una tendencia clara hacia una mayor valoración en la percepción de mostrar las propiedades favoritas dentro de la plataforma.

4.4.3. Validación de la Hipótesis

El proceso de validación de la hipótesis requiere realizar la recodificación de los escenarios correspondientes al *pretest* y al *postest*. Asimismo, es necesario ponderar las respuestas obtenidas en las preguntas establecidas dentro del instrumento de recolección de datos, con el propósito de facilitar su análisis estadístico. La codificación de dichos escenarios se presenta en la tabla 30.

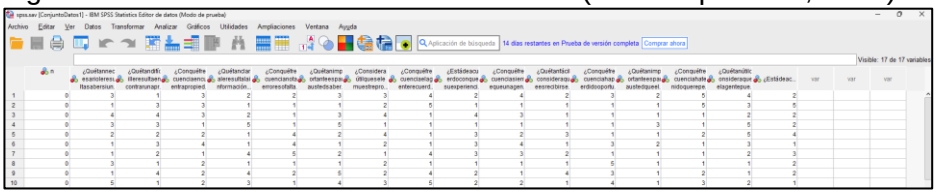
Tabla 30. Recodificación de Escenarios

Recodificación	Escenarios
0	Sin aplicación web y sin Machine Learning
1	Con aplicación web y Machine Learning

Los resultados obtenidos en las encuestas fueron recodificados para las preguntas basadas en la escala de *Likert*, y estos valores recodificados se presentan en el anexo X. Posteriormente, dicha información fue utilizada para realizar el análisis estadístico mediante el software *IBM SPSS*, con el propósito de facilitar la interpretación y validación de los datos obtenidos. Este procedimiento se puede observar en las figuras 136 y 137.

Figura 136. Proceso de Análisis en el SPSS (IBM Corporation, 2026)

Figura 137. Proceso de Análisis en el SPSS (IBM Corporation, 2026)



Adicionalmente, se realizó un análisis estadístico mediante regresión logística binaria, considerando un grado de libertad (gl) igual a 1, así como el nivel de significancia de probabilidad (p) correspondiente, a las preguntas incluidas en el instrumento de recolección de datos. Los resultados obtenidos se presentan en la tabla 31.

En ese análisis se evaluaron variables como: la necesidad de saber el estado de las propiedades, la dificultad de encontrar una propiedad de su interés, la importancia de conocer las preferencias de los usuarios, la frecuencia de disponibilidad de propiedades, recomendación de propiedades basadas en preferencias, entre otros.

Los resultados evidencian que todas presentan un nivel de significancia estadística menor a 0.05 ($p < 0.05$). Por lo tanto, los resultados expuestos en la tabla 31 permiten rechazar la hipótesis (H_0), evidenciando que la implementación de la aplicación *web* con

Machine Learning influyen de manera significativa en la gestión inmobiliaria de la empresa Escudero del cantón Santo Domingo.

Tabla 31. Estudio cruzado en base a la aplicación web

Recodificación	Puntuación	gl	Sig.
¿Qué tan necesario le resulta saber si una propiedad inmobiliaria se encuentra disponible o no?	66.985	1	0.001
¿Qué tan difícil le resulta encontrar una propiedad inmobiliaria de su interés?	57.341	1	0.001
¿Qué tan importante es para usted que el agente inmobiliario conozca sus preferencias?	70.949	1	0.001
¿Con qué frecuencia encuentra propiedades disponibles?	57.001	1	0.001
¿Considera útil que se le muestre propiedades basadas en las que ha consultado o marcado como favoritas previamente?	69.610	1	0.001

5. DISCUSIÓN

Con el propósito de determinar los requerimientos en la gestión inmobiliaria, se desarrolló una sesión de planificación con el gerente de la empresa. En dicha instancia, se confirmó su aceptación hacia la aplicación *web* con *machine learning* propuesta, debido a que contribuye a agilizar las operaciones y fortalecer la interacción entre clientes y agentes inmobiliarios. Este planteamiento coincide con lo expuesto por Choy y Ho (2023)., con respecto a la utilización del aprendizaje automático en el sector inmobiliario, para mejorar la toma de decisiones (p. 2).

Adicionalmente, se aplicó una encuesta a 65 clientes de la empresa Escudero, cuyos resultados demuestran que la implementación de la aplicación *web* incrementa en un 65.40% la comunicación sobre la disponibilidad y los procesos relacionados. Asimismo, se observa un aumento del 60% en el involucramiento de los usuarios durante la búsqueda y seguimiento, así como un 55% en el acceso autónomo a la información inmobiliaria. De igual manera, se evidencia un incremento del 22% en la confianza hacia las funcionalidades basadas en machine learning para la recomendación y predicción de propiedades. Este criterio se relaciona con lo propuesto por Mohri et al. (2018), quienes señalan que el aprendizaje automático permite realizar predicciones precisas a partir de datos, lo que respalda la importancia de integrar soluciones tecnológicas en la gestión inmobiliaria (p. 1).

Por otro lado, el segundo objetivo consiste en seleccionar las tecnologías más adecuadas para implementar en la plataforma *web* destinada a la gestión inmobiliaria. Para ello, se llevó a cabo comparaciones entre las distintas herramientas que cumplan con los requisitos del sistema. Se optó por servicios en la nube, que facilitó el despliegue de la aplicación desarrollada con *React* para el *frontend, Express* para el *backend* y *PostgreSQL* como base de datos principal. Con base en un análisis comparativo, se decidió utilizar *Render* debido a que simplifica el proceso de despliegue y permitiendo concentrarse en las funcionalidades del sistema sin atender la gestión de la infraestructura. Esta idea comparte

Mamani (2024), que menciona que una aplicación *cloud native* es un *software* diseñado para ser ejecutado en la nube, con un enfoque distribuido, elástico, y ayuda a un escalado horizontal (p. 61), y, por ende, simplifica el proceso de poner en producción un software.

Del mismo modo, para lograr alcanzar el tercer objetivo, el desarrollo de la plataforma *web* con *machine learning* para la gestión inmobiliaria, se optó por emplear el marco ágil *Scrum*. Este enfoque permitió acelerar el desarrollo del sistema, optimizando tiempo mediante la estimación y asignación eficiente de tareas técnicas, además de la gestión de incidencias, y al mismo tiempo posibilidades de incorporación de cambios durante el proceso de ejecución. De esta forma, se facilitó la evolución continua del sistema entre ciclos (*sprints*), adaptándose a nuevos requerimientos o mejoras. Esta adopción de *Scrum* coincide con lo señalado por sus creadores Schwaber y Sutherland (2020), quienes destacan que este marco agiliza la organización del trabajo en equipos, favoreciendo la entrega de valor mediante soluciones flexibles y adaptativas ante desafíos complejos (p. 1).

6. CONCLUSIONES Y RECOMENDACIONES

6.1. Conclusión

Se concluye que, tras la aplicación de las encuestas a los usuarios del sistema, se obtuvo información valiosa que evidencia un alto interés en integrar herramientas inteligentes dentro de los procesos de la gestión inmobiliaria. Los resultados permiten identificar con claridad los requisitos planteados en el primer objetivo del proyecto, destacando la necesidad de contar con una plataforma *web* que ofrezca visualización en tiempo real la disponibilidad de propiedades. Esta demanda por parte de los usuarios abre la oportunidad de potenciar las funciones del sistema mediante la incorporación de *machine learning*, las cuales demuestran una mejor precisión y rapidez en la presentación de información relevante.

En cuanto a la selección de herramientas y tecnologías empleadas en el proyecto, el análisis y la comparación permite identificar las soluciones más adecuadas para la plataforma *web*. A partir de los cuadros comparativos, se determina que la utilización de servicios en la nube es la mejor opción para el despliegue y operación del sistema. Además, el proveedor *Render* facilita la implementación de la arquitectura tecnológica compuesta por *React* en el *frontend*, *Express* en el *backend* y *PostgreSQL* como base de datos principal. Esta elección permite optimizar el tiempo de configuración, reduciendo la carga operativa relacionada con la infraestructura y concentrando el esfuerzo en el desarrollo de las funcionalidades claves.

Para cumplir el tercer objetivo, el desarrollo de la plataforma *web* con *machine learning* para la gestión inmobiliaria se realiza utilizando el marco ágil *Scrum*. Este enfoque permite agilizar el proceso mediante una asignación organizadas de tareas e incidencias, y facilita además la incorporación de la duración entre los ciclos (*sprints*). De este modo, el sistema evoluciona de forma continua y adaptable.

Por último, se realiza el análisis estadístico mediante la comparación de los datos obtenidos en el *pretest* y *postest*, lo cual permite evidenciar que la aplicación *web* con *Machine Learning* genera un cambio significativo en la gestión inmobiliaria de la empresa Escudero. Esto no solo contribuye a optimizar los procesos de búsqueda, organización y acceso a la información de propiedades, sino que también facilita un mejor seguimiento de los clientes y sus preferencias, además, permite un mayor control y eficiencia en cada una de las etapas que conforman el proceso de compra o arrendamiento.

6.2. Recomendaciones

Con el paso del tiempo, las necesidades y expectativas de los usuarios pueden transformarse, por lo que se recomienda aplicar periódicamente instrumentos de recolección de datos como encuestas, reuniones de planificación del *sprint* con el gerente y prueba de usabilidad. Esto permite identificar nuevos requerimientos y evaluar la satisfacción con las funciones implementadas, haciendo esto ayuda a mejorar los modelos de *machine learning*, garantizando una experiencia más precisa, personalizada y alineada al comportamiento del usuario.

Además, a medida que la plataforma evoluciona, es indispensable fortalecer la arquitectura mediante practicas *cloud native* que aseguren la escalabilidad, estabilidad y seguridad. Por ello, se sugiere integrar herramientas de monitoreo, automatizando los despliegues y realizando respaldos con frecuencia, lo que permite tener un mejor rendimiento y continuidad operativa.

En correspondencia con la dinámica del desarrollo tecnológico, se recomienda mantener el uso del marco ágil *Scrum* para iteraciones del sistema. Este enfoque facilita la adaptación del proyecto a nuevos requerimientos, permite introducir mejoras durante cada *sprint* y ordenando las tareas e incidencias. Además, fortalecer con la retrospectiva y las revisiones permite mejorar la comunicación del equipo y promover una evolución continua del producto.

Para finalizar, es fundamental seleccionar adecuadamente el análisis estadístico, asegurando que este se encuentre alineado con la naturaleza y el alcance del estudio. Asimismo, se recomienda que los instrumentos de medición sean previamente evaluados por profesionales con experiencia en el área, con el fin de garantizar la validez de su contenido y su pertinencia, antes de ser aplicados de manera definitiva.

7. REFERENCIAS

Aguirre, S. (2021a). *Librería React. Simplifica el desarrollo Front-end—Vol.1: Primeros pasos—Entorno de trabajo—Formularios—Tu primer proyecto*. RedUsers. https://books.google.com.ec/books?id=bQs1EAAAQBAJ

Aguirre, S. (2021b). *PHP - Programación Orientada a Objetos—Vol.3: API de PHP*. RedUSERS. https://books.google.com.ec/books?id=UqwlEAAAQBAJ

Ahmad, I. (2024). *50 Algoritmos Que Todo Programador Debe Conocer* (1st ed). Marcombo, S.A.

Ahumada, T. (2023). *Real estate prospecting: Create a million-dollar life through relationships, online leads, technology, and social media*. John Wiley & Sons, Inc.

Angular Team. (2026). *Angular*. Angular. https://angular.dev/

Apache Software Foundation. (2026). *Apache Subversion*. Subversion.apache.org. https://subversion.apache.org/

Appraisal Institute. (2013). *The appraisal of real estate* (14th ed). Appraisal Institute.

Armijos, S. M., & Chamba, W. (2022). Desarrollo de una aplicación informática para la búsqueda de oferta de alquiler de inmuebles en la ciudad de Loja. *CEDAMAZ*, *12*(1), 68-76. https://doi.org/10.54753/cedamaz.v12i1.1267

Asamblea Nacional del Ecuador (2008). Constitución de la República del Ecuador, Registro Oficial 449.

Asamblea Nacional del Ecuador (2016). Ley Orgánica de Ordenamiento Territorial, Uso y Gestión de Suelo, Registro Oficial Suplemento No. 790.

Asamblea Nacional del Ecuador (2023). Ley Orgánica para la Transformación Digital y Audiovisual, Registro Oficial Tercer Suplemento No. 245.

Asamblea Nacional del Ecuador (2024). Proyecto de Ley Orgánica de Regulación y Promoción de la Inteligencia Artificial en Ecuador, Memorando Nro. AN-NRSP-2024-0101-M.

Asociación de Corredores de Bienes Raíces del Guayas & Asociación de Corredores de Bienes Raíces del Azuay. (2021). *Disrupciones 2021: Memorias del I Congreso Virtual Inmobiliario Ecuador* (1). ACBIR.

Assembla. (2026). *Assembla—Enterprise Subversion Hosting & Management*. Assembla.Com. https://get.assembla.com/subversion/

Ayllapan, W. U. (2025). *Manual de Vue.js de Principiante a Avanzado*. https://www.google.com/search?q=https://books.google.com.ec/books%3Fid%3DP1uKEQAAQBAJ

Baudean, M. (2015). *Introducción a la investigación aplicada*. Universidad ORT, Facultad de Administración y Ciencias Sociales (FACS).

Beckman, M. D., Çetinkaya-Rundel, M., Horton, N. J., Rundel, C. W., Sullivan, A. J., & Tackett, M. (2020). Implementing Version Control With Git and GitHub as a Learning Objective in Statistics and Data Science Courses. *Journal of Statistics and Data Science Education*, *29*(sup1), S132-S144. https://doi.org/10.1080/10691898.2020.1848485

Bemthuis, R. H., Govers, R. R., & Asadi, A. (2025). A CRISP-DM-based methodology for assessing agent-based simulation models using process mining. *Journal of Simulation*, 1-22. https://doi.org/10.1080/17477778.2025.2508245

Blount, J., & Hunter, M. (2018). *Objections: The ultimate guide for mastering the art and science of getting past no*. Wiley.

Bobadilla, J. (2020). *Machine learning y deep learning: Usando Python, Scikit y Keras*. Ra-Ma.

Bottini, C. (2025). *Python para IA - Vol. 3: Machine Learning—Aprendizaje supervisado y no supervisado—Algoritmos ML* (1.ª ed.). Creative Andina Corp. / USERS.

Bradshaw, S., Chodorow, K., & Brazil, E. (2020). *MongoDB: The definitive guide: powerful and scalable data storage* (Third edition). O'Reilly Media, Inc.

Cantelon, M., Harter, M., Holowaychuk, T. J., Rajlich, N., & Schlueter, I. Z. (Eds.). (2014). *Node.js in action*. Manning.

Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., & Wirth, R. (2000). *CRISP-DM 1.0: Step-by-step data mining guide*. SPSS.

Choy, L. H. T., & Ho, W. K. O. (2023). The Use of Machine Learning in Real Estate Research. *Land*, *12*(4), 740. https://doi.org/10.3390/land12040740

Comisión Económica para América Latina y el Caribe - CEPAL (2021). *Tecnologías digitales para un nuevo futuro* (LC/TS.2021/43). Naciones Unidas.

Comisión Económica para América Latina y el Caribe - CEPAL (2022). *Agenda digital para América Latina y el Caribe (eLAC2024)* (LC/CMSI.8/5). Naciones Unidas. https://hdl.handle.net/11362/48497

Composer Team. (2026). *Composer Documentation*. Getcomposer.org. https://getcomposer.org/doc/

Contreras, L., Tarazona, G. M., & Alemán, A. P. (2023). *Machine Learning Aplicado Al Rendimiento Académico en Educación Superior: Factores, Variables y Herramientas* (1st ed). Universidad Distrital Francisco Jose de Caldas.

Dash, N. (2024). *Ultimate parallel and distributed computing with Julia for data science: Excel in data analysis, statistical modeling and machine learning by*. Orange Education PVT LTD.

Dessler, G., & Varela, R. (2011). *Administración de recursos humanos: Enfoque latinoamericano*. Pearson Educación.

Dintsis, D. (2020). *Customer Relationship Management and IT*. IntechOpen.

Django Software Foundation. (2026). *Django documentation | Django documentation*. Django Project. https://docs.djangoproject.com/en/5.1/

Domínguez, J. B. (2015). *Manual de metodología de la investigación científica* (3.ª ed.). Universidad Católica Los Ángeles de Chimbote.

Elman, J., & Lavin, M. (2015). *Lightweight Django: Using REST, WebSockets & Backbone* (1. ed). O'Reilly.

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). From Data Mining to Knowledge Discovery in Databases. *AI Mag.*, *17*(3), 37-54. https://doi.org/10.1609/aimag.v17i3.1230

Fernández, P. E. (2023). *Creación de componentes en JavaScript: Curso práctico*. RA-MA Editorial.

Ferrer, J. (2012). *Implantación de aplicaciones Web*. RA-MA Editorial.

Fisher, R., Ury, W., & Patton, B. (1998). *Obtenga el sí: El arte de negociar sin ceder* (1a edición). Gestión 2000.

Fitzsimmons, J. A., & Fitzsimmons, M. J. (2011). *Service management: Operations, strategy, information technology* (7th ed., international ed). McGraw-Hill.

Glavinich, T. E. (2008). *Contractor's guide to green building construction: Management, project delivery, documentation, and risk reduction*. John Wiley. https://doi.org/10.1002/9780470259979

Google Gemini. (2026). Google Gemini (versión utilizada) [Modelo de lenguaje de gran tamaño]. Google. https://gemini.google.com

Grove, S. K., & Gray, J. (2019). *Investigación en enfermería: Desarrollo de la práctica enfermera basada en evidencia* (Edition 7). Elsevier.

Hamilton, D. (2006). *Real estate marketing and sales essentials: Steps for success* (1. ed). Thomson/South-Western.

Han, J., Kamber, M., & Pei, J. (2012). *Data mining: Concepts and techniques* (3rd ed). Elsevier/Morgan Kaufmann.

Haupt, K. J., Faraz, I., Henry, D., Jarman, D., & Reiner, J. (2017). *Property management* (2nd ed). Rockwell Pub.

Heesen, B. (2024). *Artificial Intelligence and Machine Learning with R: Applications in the Field of Business Analytics*. Springer Fachmedien Wiesbaden. https://doi.org/10.1007/978-3-658-45392-3

Henríquez, C., & Sánchez, G. (2024). Implementation and Evaluation of a Hybrid Recommendation System for the Real Estate Market. *Data and Metadata*, *3*. https://doi.org/10.56294/dm2024.426

Hernández, R., Fernández, C., & Baptista, P. (2010). *Metodología de la investigación* (5a ed). McGraw-Hill.

Hernández, R., Fernandez, C. F., & Baptista, P. (2014). *Metodología de la investigación* (Sexta edición). McGraw-Hill Education.

Hernández, R., & Mendoza, C. P. (2018). *Metodología de la investigación: Las rutas cuantitativa, cualitativa y mixta* (First edition). McGraw-Hill Education.

Hinkel, D. F. (2008). *Essentials of practical real estate law* (4th ed). Thomson/Delmar Learning.

Huyen, C. (2023). *Diseño de Sistemas de Machine Learning: Un Proceso Iterativo para Aplicaciones Listas para Funcionar* (1st ed). Marcombo, S.A.

Johnson, C. (1989). *The recruiting revolution in real estate: Finding & keeping top-quality agents*. Real Estate Education Co.

Kazanjy, P. (2020). *Founding sales: The early stage go-to-market handbook : sales for founders (and others) in first time sales roles*. Peter Kazanjy.

Kumar, V., & Reinartz, W. J. (2018). *Customer Relationship Management: Concept, Strategy, and Tools* (Third edition). Springer. https://doi.org/10.1007/978-3-662-55381-7

Laguyás, N., Vivanco, F., Carrasco, C., Piedrafita, C., & De Ferrari, C. (2022). *PropTech en América Latina y el Caribe: ¿Cómo la tecnología puede ayudar a reducir el déficit de vivienda?* Banco Interamericano de Desarrollo. https://doi.org/10.18235/0004483

Laravel Team. (2026). *Laravel—The clean stack for Artisans and agents*. Laravel. https://laravel.com/

Mamani, Z. (2024). Aplicación cloud native en el contexto de una ingeniería de software continua. *Interfases*, (019), 61-76. https://doi.org/10.26439/interfases2024.n19.7038

Martín, C., Urquía, A., & Rubio, M. Á. (2021). *Lenguajes de programación*. UNED - Universidad Nacional de Educación a Distancia.

McAdams, L. D., Cyr, J. E., & Sobeck, J. M. (2004). *Real Estate Brokerage: A management guide* (6. ed). Dearborn Real Estate Education.

MDN Web Docs. (2025, abril 11). *Client-side web framework main features*. MDN Web Docs. https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Frameworks_libraries/Main_features

Mendoza, C., & Campoverde, M. (2025). Desarrollo de un prototipo de software para la gestión de la inmobiliaria INMOCAÑARI. *MQRInvestigar*, *9*(1), e133. https://doi.org/10.56048/MQR20225.9.1.2025.e133

Mercurial Development Team. (2026). *Mercurial—Source Control Management*. Mercurial-scm.org. https://mercurial-scm.org/

Meta Platforms Inc. (2026). *React*. React. https://react.dev/

Mettling, S., & Cusic, D. (2019). *Principles of real estate practice* (6th edition). Performance Programs Company.

Mohri, M., Rostamizadeh, A., & Talwalkar, A. (2018). *Foundations of machine learning* (Second edition). The MIT Press.

MongoDB Inc. (2026). *MongoDB: The World's Leading Modern Data Platform | MongoDB*. MongoDB. https://www.mongodb.com/

Naciones Unidas. (2018). *La Agenda 2030 y los objetivos de desarrollo sostenible una oportunidad para América Latina y Caribe* (LC/G.2681-P/Rev.3). Naciones Unidas.

Obinna, W. K., & Udo, M. J. (2022). Improving online Real Estate Management System using data analytics. *Journal of Emerging Technologies*, *2*(2), 66-75. https://doi.org/10.57040/jet.v2i2.207

Observatorio Nacional de Tecnología y Sociedad - ONTSI (2023). *Informe de digitalización de las pymes 2023–Una visión por sectores: Actividades inmobiliarias, administrativas y servicios auxiliares*. Red.es; Ministerio para la Transformación Digital y la Función Pública.

Octobus & Clever Cloud. (2026). *Heptapod*. Heptapod.net. https://heptapod.net/

OpenJS Foundation. (2026). *Express—Node.js web application framework*. Expressjs.Com. https://expressjs.com/

Ortega, J. M. (2022). *Big data, machine learning y data science en Python*. RA-MA Editorial.

O'Sullivan, B. (2009). *Mercurial: The definitive guide* (1st ed). O'Reilly Media.

Paredes, M. (2025). *Cómo convertirte en Desarrollador Frontend: La guia completa*. Independently published / Autopublicado.

Peppers, D., & Rogers, M. (2017). *Managing customer experience and relationships: A strategic framework* (Third edition). Wiley. https://doi.org/10.1002/9781119239833

Pfnür, A. (2011). *Modernes Immobilienmanagement*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-79468-4

Pilato, C. M., Collins-Sussman, B., & Fitzpatrick, B. W. (2008). *Version control with subversion* (2nd ed). O'Reilly.

Ponuthorai, P. K., & Loeliger, J. (2021). *Version control with Git: Powerful tools and techniques for collaborative software development* (Third edition). O'Reilly Media, Inc.

PostgreSQL Global Development Group. (2026). *PostgreSQL: The world's most advanced open source database*. PostgreSQL.org. https://www.postgresql.org/

Presidencia Constitucional de la República del Ecuador (2019). Reglamento General a la Ley Orgánica de Ordenamiento Territorial, Uso y Gestión de Suelo, Legislation No. Decreto No. 680, Registro Oficial Suplemento No. 460.

Presidencia Constitucional de la República del Ecuador (2023). Reglamento General a la Ley Orgánica para la Transformación Digital y Audiovisual, Legislation No. Decreto No. 813, Registro Oficial Segundo Suplemento No. 350.

Prisma Data Inc. (2026). *Express & Prisma | Next-Generation ORM for SQL DBs* [Prisma.io]. Prisma. https://www.prisma.io/express

Programa de las Naciones Unidas para los Asentamientos Humanos - UN-Habitat (2022). *World Cities Report 2022* (HS/004/22E; 2022).

Puciarelli, L. (2020). *Angular: TypeScript. Arquitectura. Instalación. Directivas y bindings. Forms. Ruteo y más* (1.ª ed.). Six Ediciones.

Ramírez, W., & Ramírez, C. (2023). *Programación de Inteligencia Artificial: Curso práctico*. RA-MA Editorial.

Ramos, A., & Ramos, M. Jesús. (2007). *Operaciones con bases de datos ofimáticas y corporativas*. Thomson Paraninfo.

Raschka, S. (2024). *El Machine Learning y la Inteligencia Artificial: 30 Preguntas y Respuestas Sobre el Aprendizaje Automático y la IA* (1st ed). Marcombo, S.A.

Redis Ltd. (2026). *Redis—The Real-time Data Platform*. Redis.Io. https://redis.io/

Render, B., Hanna, M. E., & Stair, R. M. (2006). *Métodos cuantitativos para los negocios* (9a ed.). Pearson Educación.

RhodeCode GmbH. (2026). *Enterprise Code Management for Hg, Git, SVN*. RhodeCode. https://rhodecode.com/

Richards, M. (2022). *Software Architecture Patterns* (2.ª ed.). O'Reilly Media.

Richards, M., & Ford, N. (2020). *Fundamentals of software architecture: An engineering approach* (First edition). O'Reilly.

Salafranca, L., Sierra, V., Núñez, I., Solanas, A., & Leiva, D. (2005). *Análisis estadístico mediante aplicaciones informáticas. SPSS, Statgraphics, Minitab y Excel*. Edicions Universitat Barcelona.

Sarasa, A. (2019). *Introducción a Las Bases de Datos NSQL Clave-Valor Usando Redis*. Editorial UOC.

SAS Institute Inc. (2018). *Data Mining Using SAS® Enterprise Miner™: A Case Study Approach, Fourth Edition*. SAS Institute Inc.

Schwaber, K., & Sutherland, J. (2020). *La Guía de Scrum: La Guía Definitiva de Scrum: Las Reglas del Juego*. Scrum Guides.

Sebesta, R. W. (2015). *Programming the World Wide Web* (8. ed). Pearson.

Sequelize Team. (2026). *Sequelize | Feature-rich ORM for modern TypeScript & JavaScript*. Sequelize.org. https://sequelize.org/

Smarandache, F., & Leyva, M. (2022). *Neutrosophic Computing and Machine Learning, Vol. 24*. Infinite Study.

Software Freedom Conservancy. (2025). *Git*. Git-scm.com. https://git-scm.com/

Stack Exchange Inc. (2026, marzo 20). The Stack Overflow Blog—Stack Overflow. *Stack Overflow Blog*. https://stackoverflow.blog/

Stauffer, M. (2017). *Laravel: Up and Running: A Framework for Building Modern PHP Apps* (First edition, 2017-02-03: Second release). O'Reilly UK Ltd.

Sznajdleder, P. A. (2024). *Java a Fondo. Curso de Programación* (5th ed). Marcombo, S.A.

Tracy, B. (2007). *El arte de cerrar la venta: La clave para hacer más dinero rápidamente en el mundo de las ventas profesionales*. Grupo Nelson.

TypeORM Team. (2026). *Getting Started | TypeORM*. TypeORM. https://typeorm.io/docs/getting-started/

Vanitha, & Kasthuri. (2024). *Basic Guide for Machine Learning Algorithms and Models*. SK Research Group of Companies.

Velasco, J. (2024). *Machine Learning: Fundamentos, Algoritmos y Aplicaciones para Los Negocios, Industria y Finanzas* (1st ed). Ediciones Diaz de Santos S.A.

Vue.js Core Team. (2026). *Vue.js*. Vue.Js. https://vuejs.org/

Zea, M. P., Molina, J. R., & Redrován Castillo, F. F. (2017). *Administración de base de datos con PostgreSQL*. Editorial Científica 3Ciencias.

8. ANEXOS

Anexo I: Carta de asignación, tabla de recursos y cronograma

Pontificia Universidad Católica del Ecuador | SANTO DOMINGO

DIRECCIÓN DE INVESTIGACIÓN
VINCULACIÓN E INNOVACIÓN

Santo Domingo, 18 de noviembre del 2025

Sr.
Cristian Escudero
Propietario de la Constructora Inmobiliaria Escudero
Presente. –

Reciba un atento saludo de quienes formamos parte de la Pontificia Universidad Católica del Ecuador, Sede Santo Domingo.

Por medio del presente informo a usted que el estudiante ALARCON MORILLO JHERY FERNANDO con C.C: 2300485105 y el estudiante FLORES VELEZ ROMARIO ARMANDO con C.C: 0803669688 son estudiantes del OCTAVO nivel, asignatura INTEGRACIÓN CURRICULAR, Escuela de SISTEMAS. Actualmente se encuentran elaborando su Trabajo de Titulación denominado: **APLICACIÓN WEB CON MACHINE LEARNING PARA LA GESTIÓN INMOBILIARIA DE LA EMPRESA ESCUDERO DEL CANTÓN SANTO DOMINGO PERÍODO 2025-2026**; por tal motivo, solicito muy comedidamente se le permita desarrollar en su Institución las siguientes actividades:

ACTIVIDADES:

1. Solicitud de información referente al manejo de procesos de gestión inmobiliaria en la empresa.
2. Aplicación de las encuestas dirigidas a los clientes del negocio.
3. Aplicación de la entrevista al gerente de la Constructora Inmobiliaria Escudero.
4. Entrega de aplicación y documentación del sistema desarrollado.

Los estudiantes deberán mantener la disciplina durante su estancia en la Institución, así como acogerse a las normativas necesarias para su desempeño.

Es necesario aclarar que los resultados de la investigación serán difundidos, mismos que previamente deberán ser comunicados a su persona a partir de lo cual le emitirá una carta que avale el impacto generado.

De antemano le agradezco su amable atención y ayuda en formar profesionales idóneos para servir a la sociedad.

Cordialmente,

PhD. Mikel Ugando Peñate
RESPONSABLE DE INVESTIGACIÓN FORMATIVA PUCESD
(02)3702868 Ext. 204

fe, razón y corazón

REGISTRO DE RECURSOS DEL PROYECTO			
GASTOS			
Recursos	**Cantidad**	**alor unitario (US**	**Valor total USD**
Humano			
Estudiantes	2	$0.00	$0.00
Tecnología			
Computadora	1	$800.00	$800.00
Servidor	1	$15.00	$15.00
Dominio	1	$10.00	$10.00
Materiales			
Impresora	42	$0.10	$4.20
Carpeta	1	$1.00	$1.00
Transporte	5	$0.39	$1.95
Gestión			
Internet	2	$40.00	$80.00
Luz	2	$37.25	$74.50
		TOTAL GASTOS	**$986.65**
INGRESOS			
Recursos	**Cantidad**	**alor unitario (US**	**Valor total USD**
Recursos propios			$200.00
		TOTAL INGRESOS	**$200.00**

PRODUCT BACKLOG DEL TRABAJO DE TITULACIÓN DE GRADO

N°	Actividades	Estado	2025										2026		
			Producto Mínimo Viable I = 8° nivel						Producto Mínimo Viable II = 9° nivel						
			MESES												
			Marzo	Abril	Mayo	Junio	Julio	Agosto	Septiembre	Octubre	Noviembre	Diciembre	Enero	Febrero	Marzo
ÉPICA 1	Desarrollo de la propuesta	HECHO													
ÉPICA 2	Ejecución del Proyecto	HECHO													
ÉPICA 3	Revisión literaria	HECHO													
ÉPICA 4	Metodología de la investigación	HECHO													
ÉPICA 5	Resultados	HECHO													
ÉPICA 5.1	Del primer objetivo específico	HECHO													
ÉPICA 5.2	Del segundo objetivo específico	HECHO													
ÉPICA 5.3	Del tercer objetivo específico	HECHO													
ÉPICA 6	Desarrollo de la propuesta de intervención (Gestión adaptativa con incremento iterativo "Scrum")	HECHO													
ÉPICA 6.1	Prototipo de diseño operativo, llamado solución en punta	HECHO													
ÉPICA 6.2	Sprint 1	HECHO													
ÉPICA 6.3	Sprint 2	HECHO													
ÉPICA 7	Conclusiones y Recomendaciones	HECHO													
ÉPICA 8	Informe Final del Trabajo de Titulación de Grado	HECHO													
ÉPICA 8.1	Correcciones de Lectores	HECHO													
ÉPICA 8.2	Anexos	HECHO													
ÉPICA 9	Disertación de Grado	HECHO													

Nota: PMV=Producto mínimo viable; = Relaciona la semana con "Done"

Anexo II. Carta de impacto, consentimiento informado, acta de entrega de recepción

Pontificia Universidad Católica del Ecuador
Seréis mis testigos
SANTO DOMINGO
DIRECCIÓN DE INVESTIGACIÓN VINCULACIÓN E INNOVACIÓN

Santo Domingo, 21 de enero de 2026

Mg. Yullio Cano de la Cruz
Director de Postgrados
Pontificia Universidad Católica del Ecuador Sede Santo Domingo
Presente. -

De mi consideración:

Reciba un cordial saludo y deseos de éxitos en sus delicadas funciones.

Por medio del presente, pongo en su conocimiento que el Trabajo de Titulación denominado **Aplicación Web con Machine Learning para la Gestión Inmobiliaria de la Empresa Escudero del Cantón Santo Domingo;** elaborado por los estudiantes, **Jhery Alarcón y Romario Flores, estudiantes de Ingeniería en Tecnologías de la Información** ha favorecido al desarrollo organizacional de la empresa generando un impacto en el **usuario interno, externo y en la calidad de los servicios de intermediación inmobiliaria ofrecidos a la comunidad.**

Por la atención dada a la presente, me suscribo de usted.

Afentamente,

Sr. Cristian Escudero
Propietario de la Constructora Inmobiliaria Escudero

Pontificia Universidad Católica del Ecuador
Seréis mis testigos
SANTO DOMINGO
DIRECCIÓN DE INVESTIGACIÓN VINCULACIÓN E INNOVACIÓN

Consentimiento informado

La Constructora Inmobiliaria Escudero ubicado en Santo Domingo, libre y voluntariamente participa en el proyecto de Trabajo de Titulación de Grado de la Pontificia Universidad Católica del Ecuador Sede Santo Domingo, con el título **"Aplicación Web con Machine Learning para la Gestión Inmobiliaria de la empresa Escudero del Cantón Santo Domingo"**, elaborado por Jhery Alarcón y Romario Flores, estudiantes de la carrera de Ingeniería de Tecnologías de la Información.

Luego de firmar este documento certifico lo siguiente:

- Recibimos una copia de este documento de consentimiento informado.
- Estamos de acuerdo en que los datos recopilados, fotografías y resultados de este proyecto de Trabajo de Titulación de Grado se publiquen en artículos académicos, conferencias, en páginas web institucionales y en otros medios de comunicación.
- No esperamos recibir beneficios o pago por la participación.

Y a los efectos que procedan, firmamos el presente consentimiento informado.
Santo Domingo, 21 de enero de 2026.

Firma del propietario de la Constructora Inmobiliaria Escudero

Sr. Cristian Escudero

Firma de los autores del Trabajo de Titulación de Grado:

Sr. Jhery Fernando Alarcón Morillo

Sr. Romario Armando Flores Vélez

Pontificia Universidad Católica del Ecuador
Seréis mis testigos
SANTO DOMINGO
DIRECCIÓN DE INVESTIGACIÓN VINCULACIÓN E INNOVACIÓN

ACTA DE ENTREGA RECEPCIÓN

En la ciudad de Santo Domingo de la Provincia de Santo Domingo de los Tsáchilas - Ecuador, siendo las 11 horas del día 21 de enero de 2026, en las instalaciones de la Constructora Inmobiliaria Escudero, comparecen:

El Sr. Cristian Escudero y los señores Jhery Alarcón y Romario Flores, estudiantes de la carrera de Ingeniería de Tecnologías de la Información de la Pontificia Universidad Católica del Ecuador sede Santo Domingo, y una vez culminado el Trabajo de Titulación de Grado **"Aplicación Web con Machine Learning para la Gestión Inmobiliaria de la empresa Escudero del Cantón Santo Domingo"** se procede a la entrega de:

- Aplicación Web con Machine Learning
- Manual de usuario
- Manual Técnico
- Capacitación a los Usuarios

Para constancia de lo actuado, en conformidad y aceptación, firman los intervinientes la presente acta de entrega-recepción en dos ejemplares.

Entrega:

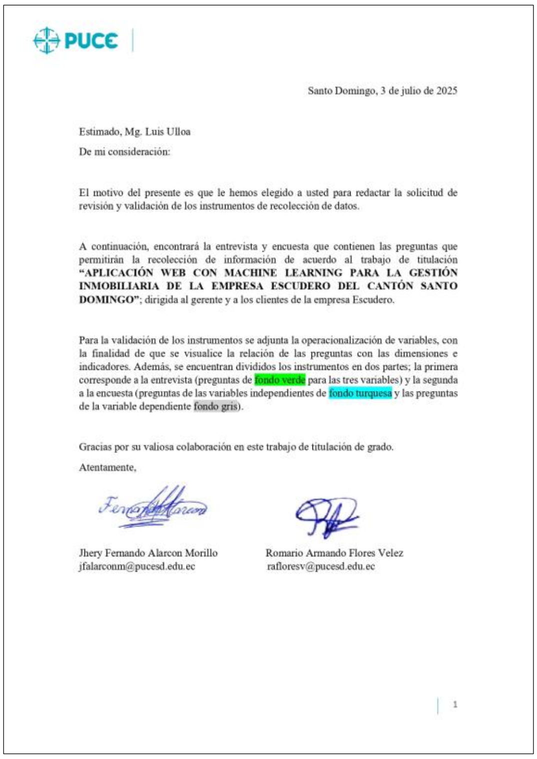
Sr. Jhery Alarcón
Estudiante PUCE-SD

Sr. Romario Flores
Estudiante PUCE-SD

Recibe:

Sr. Cristian Escudero
Propietario de la Constructora Inmobiliaria Escudero

Anexo III. Validación de instrumentos de recolección de datos



PUCE

Santo Domingo, 3 de julio de 2025

Estimado, Mg. Luis Ulloa

De mi consideración:

El motivo del presente es que le hemos elegido a usted para redactar la solicitud de revisión y validación de los instrumentos de recolección de datos.

A continuación, encontrará la entrevista y encuesta que contienen las preguntas que permitirán la recolección de información de acuerdo al trabajo de titulación **"APLICACIÓN WEB CON MACHINE LEARNING PARA LA GESTIÓN INMOBILIARIA DE LA EMPRESA ESCUDERO DEL CANTÓN SANTO DOMINGO"**; dirigida al gerente y a los clientes de la empresa Escudero.

Para la validación de los instrumentos se adjunta la operacionalización de variables, con la finalidad de que se visualice la relación de las preguntas con las dimensiones e indicadores. Además, se encuentran divididos los instrumentos en dos partes; la primera corresponde a la entrevista (preguntas de fondo verde para las tres variables) y la segunda a la encuesta (preguntas de las variables independientes de fondo turquesa y las preguntas de la variable dependiente fondo gris).

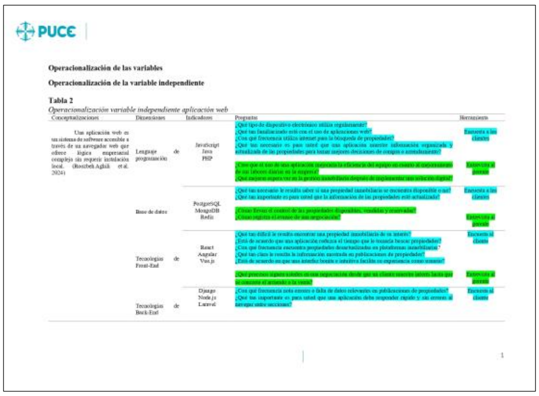
Gracias por su valiosa colaboración en este trabajo de titulación de grado.

Atentamente,

Jhery Fernando Alarcon Morillo
jfalarconm@pucesd.edu.ec

Romario Armando Flores Velez
rafloresv@pucesd.edu.ec

1



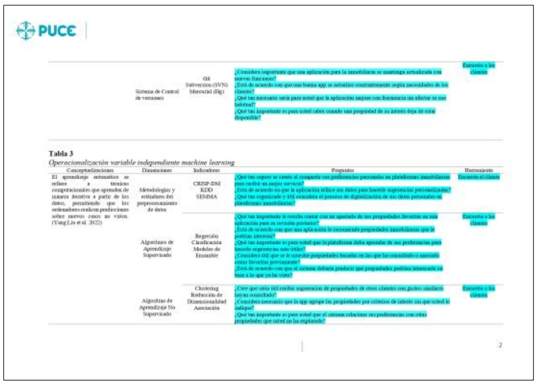
PUCE

Operacionalización de las variables

Operacionalización de la variable independiente

Tabla 2

Operacionalización variable independiente aplicación web

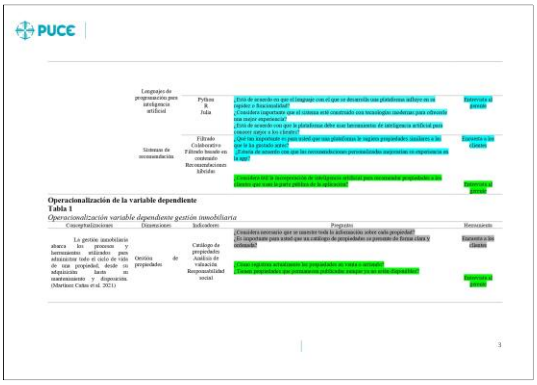


PUCE

Tabla 3

Operacionalización variable independiente machine learning

2



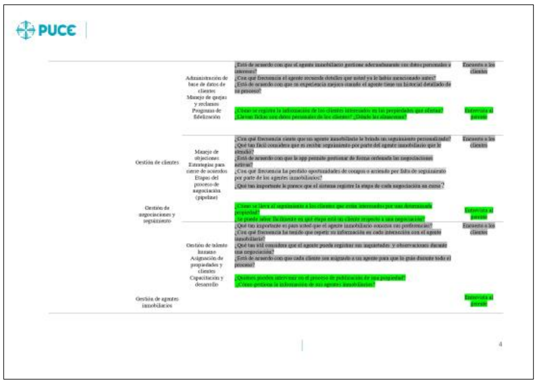
PUCE

Operacionalización de la variable dependiente

Tabla 1

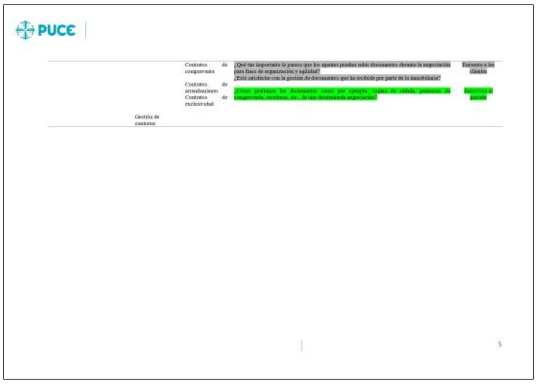
Operacionalización variable dependiente gestión inmobiliaria

3



PUCE

4



PUCE

5

PUCE

Preguntas

Objetivo:

Identificar los procesos de gestión inmobiliaria en la empresa inmobiliaria Escudero y fortalecerlos mediante la incorporación de una aplicación web con machine learning.

Baremo:

Claridad: Se refiere si la pregunta está comprendida por los destinatarios.

Pertinencia: Se refiere si la pregunta corresponde con lo que se quiere indagar

Las preguntas en cuanto a su claridad y pertinencia se encuentran bajo la escala valorativa Likert del 1 al 5 (donde 1 es el menor valor y 5 el mayor). Podrá añadir una formulación alternativa y observación, en caso que considere necesario

Marque con una cruz (X) el tramo del baremo que exprese mejor su juicio "Claridad" y "Pertinencia" sobre los ítems propuestos:

Entrevista dirigida gerente de la empresa

Variable independiente: Aplicación web (**Dimensión:** Lenguaje de programación)

1. ¿Cree que el uso de una app mejoraría la eficiencia del equipo en cuanto al mejoramiento de sus labores diarias en la empresa?

CLARIDAD					PERTINENCIA					FORMULACIÓN ALTERNATIVA	OBSERVACIÓN
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Aplicación web (**Dimensión:** Lenguaje de programación)

2. ¿Qué mejoras espera ver en la gestión inmobiliaria después de implementar una solución digital?

CLARIDAD					PERTINENCIA					FORMULACIÓN ALTERNATIVA	OBSERVACIÓN
1	2	3	4	5	1	2	3	4	5		

PUCE

Variable independiente: Aplicación web (**Dimensión:** Base de datos)

3. ¿Cómo llevan el control de las propiedades disponibles, vendidas y reservadas?

CLARIDAD					PERTINENCIA					FORMULACIÓN ALTERNATIVA	OBSERVACIÓN
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Aplicación web (**Dimensión:** Base de datos)

4. ¿Cómo registra el avance de una negociación?

CLARIDAD					PERTINENCIA					FORMULACIÓN ALTERNATIVA	OBSERVACIÓN
1	2	3	4	5	1	2	3	4	5		

Variable dependiente: Aplicación web (**Dimensión:** Tecnologías de Front-End)

5. ¿Qué procesos siguen ustedes en una negociación desde que un cliente muestra interés hasta que se concrete el arriendo o la venta?

CLARIDAD					PERTINENCIA					FORMULACIÓN ALTERNATIVA	OBSERVACIÓN
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Machine Learning (**Dimensión:** Sistemas de Recomendación)

6. ¿Considera útil la incorporación de inteligencia artificial para recomendar propiedades a los clientes que usan la parte pública de la aplicación?

CLARIDAD					PERTINENCIA					FORMULACIÓN ALTERNATIVA	OBSERVACIÓN
1	2	3	4	5	1	2	3	4	5		

Variable dependiente: Gestión inmobiliaria (**Dimensión:** Gestión de propiedades)

PUCE

7. ¿Cómo registran actualmente las propiedades en venta o arriendo?

CLARIDAD					PERTINENCIA					FORMULACIÓN ALTERNATIVA	OBSERVACIÓN
1	2	3	4	5	1	2	3	4	5		

Variable dependiente: Gestión inmobiliaria (**Dimensión:** Gestión de propiedades)

8. ¿Tienen propiedades que permanecen publicadas aunque ya no estén disponibles?

CLARIDAD					PERTINENCIA					FORMULACIÓN ALTERNATIVA	OBSERVACIÓN
1	2	3	4	5	1	2	3	4	5		

Variable dependiente: Gestión inmobiliaria (**Dimensión**: Gestión de clientes)

9. ¿Cómo se registra la información de los clientes interesados en las propiedades que ofertan?

CLARIDAD					PERTINENCIA					FORMULACIÓN ALTERNATIVA	OBSERVACIÓN
1	2	3	4	5	1	2	3	4	5		

Variable dependiente:Gestión inmobiliaria (**Dimensión**: Gestión de clientes)

10. ¿Llevan fichas con datos personales de los clientes? ¿Dónde las almacenan?

CLARIDAD					PERTINENCIA					FORMULACIÓN ALTERNATIVA	OBSERVACIÓN
1	2	3	4	5	1	2	3	4	5		

Variable dependiente: Gestión inmobiliaria (**Dimensión**: Gestión de negociaciones y seguimiento)

11. ¿Cómo se lleva el seguimiento a los clientes que están interesados por una determinada propiedad?

CLARIDAD					PERTINENCIA					FORMULACIÓN ALTERNATIVA	OBSERVACIÓN
1	2	3	4	5	1	2	3	4	5		

PUCE

Variable dependiente: Gestión inmobiliaria (**Dimensión**: Gestión de negociaciones y seguimiento)

12. ¿Se puede saber fácilmente en qué etapa está un cliente respecto a una negociación?

CLARIDAD					PERTINENCIA					FORMULACIÓN ALTERNATIVA	OBSERVACIÓN
1	2	3	4	5	1	2	3	4	5		

Variable dependiente: Gestión inmobiliaria (**Dimensión**: Gestión de agentes inmobiliarios)

13. ¿Quiénes pueden intervenir en el proceso de publicación de una propiedad?

CLARIDAD					PERTINENCIA					FORMULACIÓN ALTERNATIVA	OBSERVACIÓN
1	2	3	4	5	1	2	3	4	5		

Variable dependiente: Gestión inmobiliaria (**Dimensión**: Gestión de agentes inmobiliarios)

14. ¿Cómo gestiona la información de sus agentes inmobiliarios?

CLARIDAD					PERTINENCIA					FORMULACIÓN ALTERNATIVA	OBSERVACIÓN
1	2	3	4	5	1	2	3	4	5		

Variable dependiente: Gestión inmobiliaria (**Dimensión**: Gestión de contratos)

15. ¿Cómo gestionan los documentos como, por ejemplo, copias de cédula, promesas de compraventa, escrituras, etc. de una determinada negociación?

CLARIDAD					PERTINENCIA					FORMULACIÓN ALTERNATIVA	OBSERVACIÓN
1	2	3	4	5	1	2	3	4	5		

PUCE

Encuesta dirigida a los clientes

Tema del Trabajo de Titulación de grado: Aplicación web con machine learning para la gestión inmobiliaria de la empresa Escudero, del cantón Santo Domingo

Objetivo: Recolectar información para validar la propuesta de intervención enfocada al fortalecimiento y mejora de los procesos de gestión inmobiliaria de la empresa Escudero.

Instrucciones al público objetivo: La encuesta está dirigida a los clientes de la empresa Escudero, en base a la información obtenida, permitirá conocer el manejo de nuevas tendencias tecnológicas en el manejo de procesos de gestión inmobiliaria.

PREGUNTAS

Variable independiente: Aplicación web (**Dimensión**: Lenguaje de programación)

1. ¿Qué tipo de dispositivo electronico utiliza regularmente?
   a) Teléfono móvil
   b) Computadora de escritorio
   c) Laptop
   d) Tablet
   e) Otro

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Aplicación web (**Dimensión**: Lenguaje de programación)

2. ¿Qué tan familiarizado está con el uso de aplicaciones web?
   a) Totalmente familiarizado
   b) Familiarizado
   c) Moderadamente familiarizado
   d) Poco familiarizado
   e) Nada familiarizado

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

PUCE

Variable independiente: Aplicación web (**Dimensión**: Lenguaje de programación)

3. ¿Con qué frecuencia utiliza internet para la búsqueda de propiedades?
   a) Muy frecuente
   b) Frecuente
   c) Medianamente frecuente
   d) Poco frecuente
   e) Nada frecuente

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Aplicación web (**Dimensión**: Lenguaje de programación)

4. ¿Qué tan necesario es para usted que una aplicación muestre información organizada y actualizada de las propiedades para tomar mejores decisiones de compra o arrendamiento?
   a) Totalmente necesario
   b) Necesario
   c) Moderadamente necesario
   d) Poco necesario
   e) Innecesario

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Aplicación web (**Dimensión**: Base de datos)

5. ¿Qué tan necesario le resulta saber si una propiedad inmobiliaria se encuentra disponible o no?
   a) Totalmente necesario
   b) Necesario
   c) Moderadamente necesario
   d) Poco necesario
   e) Innecesario

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Aplicación web (**Dimensión**: Base de datos)

6. ¿Qué tan importante es para usted que la información de las propiedades esté actualizada?
   a) Muy importante
   b) Importante
   c) Medianamente importante

PUCE

   d) Poco importante
   e) Nada importante

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Aplicación web (**Dimensión:** Tecnologías de Front-End)

7. ¿Qué tan difícil le resulta encontrar una propiedad inmobiliaria de su interés?
   a) Muy difícil
   b) Difícil
   c) Normal
   d) Fácil
   e) Muy fácil

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Aplicación web (**Dimensión:** Tecnologías de Front-End)

8. ¿Está de acuerdo que una aplicación reduzca el tiempo que le tomaría buscar propiedades?
   a) Totalmente de acuerdo
   b) Bastante de acuerdo
   c) Neutral
   d) Poco de acuerdo
   e) Totalmente en desacuerdo

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Aplicación web (**Dimensión:** Tecnologías de Front-End)

9. ¿Con qué frecuencia encuentra propiedades disponibles?
   a) Muy frecuente
   b) Frecuente
   c) Medianamente frecuente
   d) Poco frecuente
   e) Nada frecuente

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

PUCE

Variable independiente: Aplicación web (**Dimensión:** Tecnologías de Front-End)

10. ¿Qué tan clara le resulta la información mostrada en publicaciones de propiedades?
    a) Muy claro
    b) Claro
    c) Medianamente claro
    d) Poco claro
    e) Nada claro

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Aplicación web (**Dimensión:** Tecnologías de Front-End)

11. ¿Está de acuerdo en que una interfaz bonita e intuitiva facilita su experiencia como usuario?
    a) Totalmente de acuerdo
    b) De acuerdo
    c) Moderadamente de acuerdo
    d) Poco de acuerdo
    e) Nada de acuerdo

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Aplicación web (**Dimensión:** Tecnologías de Back-End)

12. ¿Con qué frecuencia nota errores o falta de datos relevantes en publicaciones de propiedades?
    a) Muy frecuente
    b) Frecuente
    c) Medianamente frecuente
    d) Poco frecuente
    e) Nada frecuente

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Aplicación web (**Dimensión:** Tecnologías de Back-End)

13. ¿Qué tan importante es para usted que una aplicación deba responder rápido y sin errores al navegar entre secciones?

a) Muy importante
b) Importante
c) Medianamente importante
d) Poco importante
e) Nada importante

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Aplicación web (**Dimensión:** Sistema de Control de Versiones)

14. ¿Considera importante que una aplicación para la inmobiliaria se mantenga actualizada con nuevas funciones?
a) Muy importante
b) Importante
c) Medianamente importante
d) Poco importante
e) Nada importante

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Aplicación web (**Dimensión:** Sistema de Control de Versiones)

15. ¿Está de acuerdo con que una buena app se actualice constantemente según necesidades de los clientes?
a) Totalmente de acuerdo
b) De acuerdo
c) Moderadamente de acuerdo
d) Poco de acuerdo
e) Nada de acuerdo

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Aplicación web (**Dimensión:** Sistema de Control de Versiones)

16. ¿Qué tan necesario sería para usted que la aplicación mejore con frecuencia sin afectar su uso habitual?

a) Totalmente necesario
b) Necesario
c) Moderadamente necesario
d) Poco necesario
e) Innecesario

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Aplicación web (**Dimensión:** Sistema de Control de Versiones)

17. ¿Qué tan importante es para usted saber cuando una propiedad de su interés deja de estar disponible?
a) Muy importante
b) Importante
c) Medianamente importante
d) Poco importante
e) Nada importante

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Machine Learning (**Dimensión**: Metodologías y estándares del preprocesamiento de datos)

18. ¿Qué tan seguro se siente al compartir sus preferencias personales en plataformas inmobiliarias para recibir un mejor servicio?
a) Totalmente seguro
b) seguro
c) Moderadamente seguro
d) Poco seguro
e) Totalmente inseguro

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Machine Learning (**Dimensión**: Metodologías y estándares del preprocesamiento de datos)

19. ¿Está de acuerdo en que la aplicación utilice sus datos para hacerle sugerencias personalizadas?
a) Totalmente de acuerdo
b) De acuerdo
c) Moderadamente de acuerdo
d) Poco de acuerdo

e) Nada de acuerdo

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Machine Learning (**Dimensión**: Metodologías y estándares del preprocesamiento de datos)

20. ¿Qué tan organizado y útil considera el proceso de digitalización de sus datos personales en plataformas inmobiliarias?
a) Totalmente útil
b) Útil
c) Moderadamente útil
d) Poco útil
e) Inutil

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Machine Learning (**Dimensión**: Algoritmos de Aprendizaje Supervisado)

21. ¿Qué tan importante le resulta contar con un apartado de sus propiedades favoritas en una aplicación para su revisión posterior?
a) Muy importante
b) Importante
c) Medianamente importante
d) Poco importante
e) Nada importante

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Machine Learning (**Dimensión**: Algoritmos de Aprendizaje Supervisado)

22. ¿Está de acuerdo con que una aplicación le recomiende propiedades inmobiliarias que le podrían interesar?
a) Totalmente de acuerdo
b) De acuerdo
c) Moderadamente de acuerdo
d) Poco de acuerdo
e) Nada de acuerdo

Relevancia	Claridad	Formulación alternativa:	Observación:

1	2	3	4	5	1	2	3	4	5		

Variable independiente: Machine Learning (**Dimensión**: Algoritmos de Aprendizaje Supervisado)

23. ¿Qué tan importante es para usted que la plataforma deba aprender de sus preferencias para hacerle sugerencias más útiles?
a) Muy importante
b) Importante
c) Medianamente importante
d) Poco importante
e) Nada importante

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Machine Learning (**Dimensión**: Algoritmos de Aprendizaje Supervisado)

24. ¿Considera útil que se le muestre propiedades basadas en las que ha consultado o marcado como favoritas previamente?
a) Totalmente útil
b) Útil
c) Moderadamente útil
d) Poco útil
e) Inutil

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Machine Learning (**Dimensión**: Algoritmos de Aprendizaje Supervisado)

25. ¿Está de acuerdo con que el sistema debería predecir qué propiedades podrían interesarle en base a lo que ya ha visto?
a) Totalmente de acuerdo
b) De acuerdo
c) Moderadamente de acuerdo
d) Poco de acuerdo
e) Nada de acuerdo

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

PUCE

Variable independiente: Machine Learning (**Dimensión**: Algoritmos de Aprendizaje No Supervisado)

26. ¿Cree que sería útil recibir sugerencias de propiedades de otros clientes con gustos similares hayan consultado?
    a) Totalmente útil
    b) Útil
    c) Moderadamente útil
    d) Poco útil
    e) Inutil

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Machine Learning (**Dimensión**: Algoritmos de Aprendizaje No Supervisado)

27. ¿Considera necesario que la app agrupe las propiedades por criterios de interés sin que usted lo indique?
    a) Totalmente necesario
    b) Necesario
    c) Moderadamente necesario
    d) Poco necesario
    e) Innecesario

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Machine Learning (**Dimensión**: Algoritmos de Aprendizaje No Supervisado)

28. ¿Qué tan importante es para usted que el sistema relacione sus preferencias con otras propiedades que usted no ha explorado?
    a) Muy importante
    b) Importante
    c) Medianamente importante
    d) Poco importante
    e) Nada importante

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Machine Learning (**Dimensión**: Lenguajes de programación para inteligencia artificial)

29. ¿Está de acuerdo en que se desarrolle con tecnologías modernas?

PUCE

    a) Totalmente de acuerdo
    b) De acuerdo
    c) Moderadamente de acuerdo
    d) Poco de acuerdo
    e) Nada de acuerdo

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Machine Learning (**Dimensión**: Lenguajes de programación para inteligencia artificial)

30. ¿Considera importante que el sistema esté construido con tecnologías modernas para ofrecerle una mejor experiencia?
    a) Muy importante
    b) Importante
    c) Medianamente importante
    d) Poco importante
    e) Nada importante

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Machine Learning (**Dimensión**: Lenguajes de programación para inteligencia artificial)

31. ¿Está de acuerdo con que la plataforma debe usar herramientas de inteligencia artificial para conocer mejor a los clientes?
    a) Totalmente de acuerdo
    b) De acuerdo
    c) Moderadamente de acuerdo
    d) Poco de acuerdo
    e) Nada de acuerdo

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Machine Learning (**Dimensión**: Sistemas de Recomendación)

32. ¿Qué tan importante es para usted que una plataforma le sugiera propiedades similares a las que le ha gustado antes?
    a) Muy importante
    b) Importante
    c) Medianamente importante
    d) Poco importante
    e) Nada importante

PUCE

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable independiente: Machine Learning (**Dimensión**: Sistemas de Recomendación)

33. ¿Estaría de acuerdo con que las recomendaciones personalizadas mejorarían su experiencia en la app?
    f) Totalmente de acuerdo
    g) De acuerdo
    h) Moderadamente de acuerdo
    i) Poco de acuerdo
    j) Nada de acuerdo

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable dependiente: Gestión inmobiliaria (**Dimensión**: Gestión de propiedades)

34. ¿Considera necesario que se muestre toda la información sobre cada propiedad?
    a) Totalmente necesario
    b) Necesario
    c) Moderadamente necesario
    d) Poco necesario
    e) Innecesario

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable dependiente: Gestión inmobiliaria (**Dimensión**: Gestión de propiedades)

35. ¿Es importante para usted que un catálogo de propiedades se presente de forma clara y ordenada?
    a) Muy importante
    b) Importante
    c) Medianamente importante
    d) Poco importante
    e) Nada importante

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable dependiente: Gestión inmobiliaria (**Dimensión**: Gestión de clientes)

36. ¿Está de acuerdo con que el agente inmobiliario gestione adecuadamente sus datos personales e intereses?
    a) Totalmente de acuerdo

PUCE

    b) De acuerdo
    c) Moderadamente de acuerdo
    d) Poco de acuerdo
    e) Nada de acuerdo

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable dependiente: Gestión inmobiliaria (**Dimensión**: Gestión de clientes)

37. ¿Con qué frecuencia el agente recuerda detalles que usted ya le había mencionado antes?
    a) Muy frecuente
    b) Frecuente
    c) Medianamente frecuente
    d) Poco frecuente
    e) Nada frecuente

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable dependiente: Gestión inmobiliaria (**Dimensión**: Gestión de clientes)

38. ¿Está de acuerdo con que su experiencia mejora cuando el agente tiene un historial detallado de su proceso?
    a) Totalmente de acuerdo
    b) De acuerdo
    c) Moderadamente de acuerdo
    d) Poco de acuerdo
    e) Nada de acuerdo

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable dependiente: Gestión inmobiliaria (**Dimensión**: Gestión de negociaciones)

39. ¿Con qué frecuencia siente que un agente inmobiliario le brinda un seguimiento personalizado
    a) Muy frecuente
    b) Frecuente
    c) Medianamente frecuente
    d) Poco frecuente
    e) Nada frecuente

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

PUCE

Variable dependiente: Gestión inmobiliaria (**Dimensión**: Gestión de negociaciones)

40. ¿Qué tan fácil considera que es recibir seguimiento por parte del agente inmobiliario que le atendió?
    a) Muy difícil
    b) Difícil
    c) Normal
    d) Fácil
    e) Muy fácil

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable dependiente: Gestión inmobiliaria (**Dimensión**: Gestión de negociaciones)

41. ¿Está de acuerdo con que la app permite gestionar de forma ordenada las negociaciones activas?
    f) Totalmente de acuerdo
    g) De acuerdo
    h) Moderadamente de acuerdo
    i) Poco de acuerdo
    j) Nada de acuerdo

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable dependiente: Gestión inmobiliaria (**Dimensión**: Gestión de negociaciones)

42. ¿Con qué frecuencia ha perdido oportunidades de compra o arriendo por falta de seguimiento por parte de los agentes inmobiliarios?
    a) Muy frecuente
    b) Frecuente
    c) Medianamente frecuente
    d) Poco frecuente
    e) Nada frecuente

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable dependiente: Gestión inmobiliaria (**Dimensión**: Gestión de negociaciones)

43. ¿Qué tan importante le parece que el sistema registre la etapa de cada negociación en curso?
    a) Muy importante
    b) Importante
    c) Medianamente importante
    d) Poco importante

PUCE

e) Nada importante

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable dependiente: Gestión inmobiliaria (**Dimensión**: Gestión de agentes inmobiliarios)

44. ¿Qué tan importante es para usted que el agente inmobiliario conozca sus preferencias?
    a) Muy importante
    b) Importante
    c) Medianamente importante
    d) Poco importante
    e) Nada importante

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable dependiente: Gestión inmobiliaria (**Dimensión**: Gestión de agentes inmobiliarios)

45. ¿Con qué frecuencia ha tenido que repetir su información en cada interacción con el agente inmobiliario?
    a) Muy frecuente
    b) Frecuente
    c) Medianamente frecuente
    d) Poco frecuente
    e) Nada frecuente

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable dependiente: Gestión inmobiliaria (**Dimensión**: Gestión de agentes inmobiliarios)

46. ¿Qué tan útil considera que el agente pueda registrar sus inquietudes y observaciones durante una negociación?
    a) Totalmente útil
    b) Útil
    c) Moderadamente útil
    d) Poco útil
    e) Inutil

Relevancia	Claridad	Formulación alternativa:	Observación:

PUCE

1	2	3	4	5	1	2	3	4	5		

Variable dependiente: Gestión inmobiliaria (**Dimensión**: Gestión de agentes inmobiliarios)

47. ¿Está de acuerdo con que cada cliente sea asignado a un agente para que lo guíe durante todo el proceso?
    f) Totalmente de acuerdo
    g) De acuerdo
    h) Moderadamente de acuerdo
    i) Poco de acuerdo
    j) Nada de acuerdo

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable dependiente: Gestión inmobiliaria (**Dimensión**: Gestión de contratos)

48. ¿Qué tan importante le parece que los agentes puedan subir documentos durante la negociación para fines de organización y agilidad?
    a) Muy importante
    b) Importante
    c) Medianamente importante
    d) Poco importante
    e) Nada importante

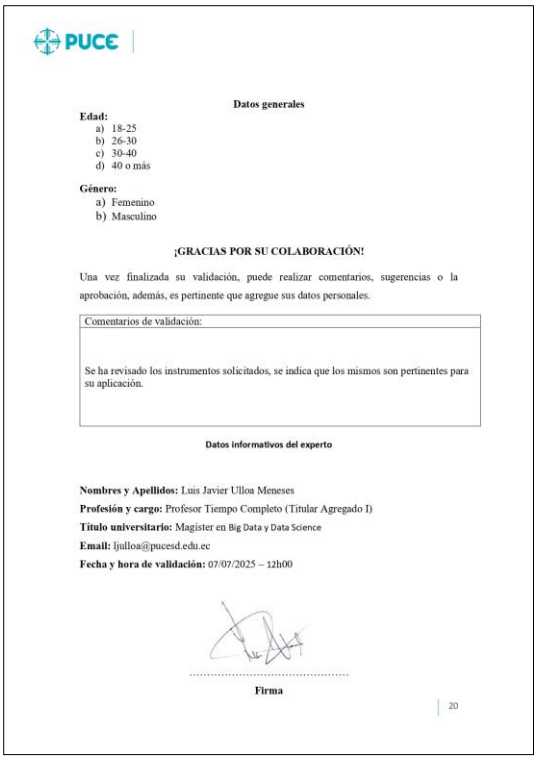
Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		

Variable dependiente: Gestión inmobiliaria (**Dimensión**: Gestión de contratos)

49. ¿Está satisfecho con la gestión de documentos que ha recibido por parte de la inmobiliaria?

    a) Totalmente satisfecho
    b) Satisfecho
    c) Moderadamente satisfecho
    d) Poco satisfecho
    e) Nada satisfecho

Relevancia					Claridad					Formulación alternativa:	Observación:
1	2	3	4	5	1	2	3	4	5		



PUCE

Datos generales

Edad:
a) 18-25
b) 26-30
c) 30-40
d) 40 o más

Género:
a) Femenino
b) Masculino

¡GRACIAS POR SU COLABORACIÓN!

Una vez finalizada su validación, puede realizar comentarios, sugerencias o la aprobación, además, es pertinente que agregue sus datos personales.

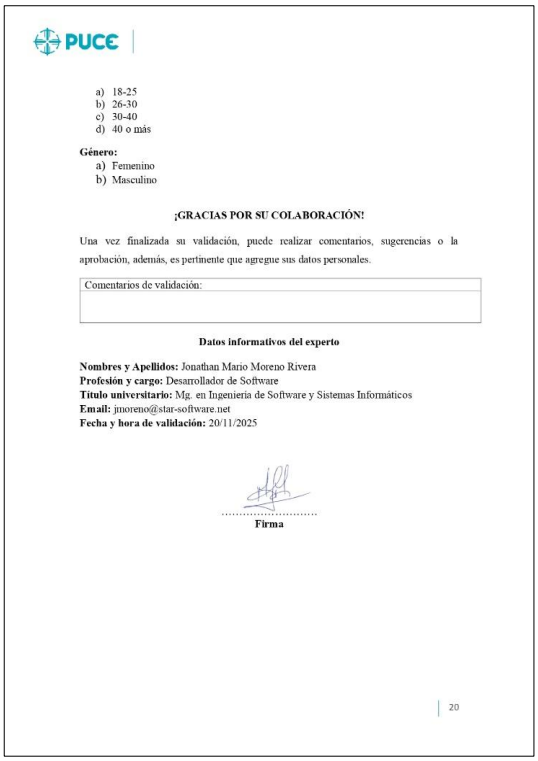
Comentarios de validación:
Se ha revisado los instrumentos solicitados, se indica que los mismos son pertinentes para su aplicación.

Datos informativos del experto

Nombres y Apellidos: Luis Javier Ulloa Meneses
Profesión y cargo: Profesor Tiempo Completo (Titular Agregado I)
Título universitario: Magíster en Big Data y Data Science
Email: ljulloa@pucesd.edu.ec
Fecha y hora de validación: 07/07/2025 – 12h00

Firma

20



PUCE

a) 18-25
b) 26-30
c) 30-40
d) 40 o más

Género:
a) Femenino
b) Masculino

¡GRACIAS POR SU COLABORACIÓN!

Una vez finalizada su validación, puede realizar comentarios, sugerencias o la aprobación, además, es pertinente que agregue sus datos personales.

Comentarios de validación:

Datos informativos del experto

Nombres y Apellidos: Jonathan Mario Moreno Rivera
Profesión y cargo: Desarrollador de Software
Título universitario: Mg. en Ingeniería de Software y Sistemas Informáticos
Email: jmoreno@star-software.net
Fecha y hora de validación: 20/11/2025

Firma

20

Anexo IV: Historias de Usuario e Historia Técnica

Historia de Usuario	
Número: 1	**Usuario:** Cliente, Agente, Administrador
Nombre historia: Inicio de Sesión con Roles	
Prioridad en negocio: 100	**Riesgo en desarrollo:** ALTO
Puntos estimados: 5	**Sprint:** 1
Programador responsable: Jhery Alarcón y Romario Flores	
Descripción: **Como** Cliente, Agente, Administrador **Quiero** ingresar mis credenciales **Para** acceder a la plataforma y a las funcionalidades correspondientes según mi rol.	
Escenario de prueba: • **Dado** el ingreso de las credenciales correctas de un usuario con rol Cliente activo y verificado **Cuando** pulse el botón "Iniciar Sesión" **Entonces** se permite el acceso y se redirige a la vista principal del portal público. • **Dado** el ingreso de las credenciales correctas de un usuario con rol Agente activo y verificado **Cuando** pulse el botón "Iniciar Sesión" **Entonces** se permite el acceso y se redirige al panel de propiedades del agente. • **Dado** el ingreso de las credenciales correctas de un usuario con rol Administrador activo y verificado **Cuando** pulse el botón "Iniciar Sesión" **Entonces** se permite el acceso y se redirige al panel de propiedades del administrador. • **Dado** el ingreso de un correo electrónico o contraseña que no corresponden a ningún usuario registrado **Cuando** pulse el botón "Iniciar Sesión" **Entonces** se muestra el mensaje: "Correo electrónico o contraseña incorrectos". • **Dado** que el usuario deja el campo correo o contraseña vacíos **Cuando** pulse el botón "Iniciar Sesión" **Entonces** el formulario muestra los mensajes de validación correspondientes. • **Dado** el ingreso de credenciales correctas de una cuenta con estado inactivo **Cuando** pulse el botón "Iniciar Sesión" **Entonces** se muestra el mensaje: "Tu cuenta está inactiva, contacta al administrador". • **Dado** el ingreso de credenciales correctas de una cuenta que aún no ha verificado su correo electrónico **Cuando** pulse el botón "Iniciar Sesión" **Entonces** se muestra el mensaje: "Por favor verifica tu correo electrónico para iniciar sesión".	

Historia de Usuario	
Número: 2	**Usuario:** Cliente, Agente, Administrador
Nombre historia: Cierre de sesión	
Prioridad en negocio: 98	**Riesgo en desarrollo:** BAJO
Puntos estimados: 1	**Sprint:** 1
Programador responsable: Jhery Alarcón y Romario Flores	
Descripción: **Como** Cliente, Agente, Administrador **Quiero** cerrar la sesión de mi cuenta **Para** proteger la misma y evitar accesos no autorizados.	
Escenario de prueba: • **Dado** que existe una sesión activa de un usuario con rol Cliente **Cuando** pulse la opción "Cerrar Sesión" disponible en el menú desplegable de su perfil en la barra de navegación **Entonces** la sesión se cierra, el menú de navegación vuelve a mostrar los botones "Iniciar Sesión" y "Registrarse", y se muestra la página de inicio del portal público. • **Dado** que existe una sesión activa de un usuario con rol Agente **Cuando** pulse el ícono de "Cerrar sesión" ubicado en la parte inferior del panel lateral **Entonces** la sesión se cierra y se muestra la página de inicio del portal público. • **Dado** que existe una sesión activa de un usuario con rol Administrador **Cuando** pulse el ícono de "Cerrar sesión" ubicado en la parte inferior del panel lateral **Entonces** la sesión se cierra y se muestra la página de inicio del portal público. • **Dado** que un usuario acaba de cerrar sesión **Cuando** intente acceder directamente a una página restringida (por ejemplo, el panel del agente o del administrador) **Entonces** el sistema no permite el acceso y lo redirige a la pantalla de inicio de sesión.	

Historia de Usuario

Número: 3	**Usuario:** Visitante
Nombre historia: Registro de cuenta y verificación (cliente externo)	
Prioridad en negocio: 96	**Riesgo en desarrollo:** BAJO
Puntos estimados: 3	**Sprint:** 1
Programador responsable: Jhery Alarcón y Romario Flores	

Descripción:

Como Visitante **Quiero** crear una cuenta con mis datos y verificarla mediante el enlace enviado a mi correo electrónico **Para** navegar las diferentes propiedades de la plataforma, guardar mis favoritas y recibir recomendaciones personalizadas de inmuebles que me podrían interesar.

Escenario de prueba:

- **Dado** el ingreso de datos válidos en los campos obligatorios del formulario de registro de cuenta (nombre, correo electrónico, contraseña y confirmación de contraseña) **Cuando** pulse el botón "Registrarse" **Entonces** aparece una notificación indicando que se envió un enlace de verificación al correo electrónico ingresado, y se redirige a la página de inicio de sesión.
- **Dado** que el usuario deja en blanco el campo nombre, correo electrónico, contraseña o confirmación de contraseña **Cuando** pulse el botón "Registrarse" **Entonces** se muestran mensajes de advertencia en los campos correspondientes indicando que son obligatorios.
- **Dado** que el usuario ingresa un correo electrónico con formato incorrecto **Cuando** pulse el botón "Registrarse" **Entonces** se muestra un mensaje de correo inválido en el campo respectivo.
- **Dado** el ingreso de un correo electrónico que ya pertenece a una cuenta existente **Cuando** pulse el botón "Registrarse" **Entonces** se muestra el mensaje "El correo ya está registrado".
- **Dado** el ingreso de una contraseña que no cumple los requisitos mínimos (mínimo 8 caracteres, una mayúscula y un número) **Cuando** pulse el botón "Registrarse" **Entonces** se muestra el mensaje "Debe tener 8 caracteres, al menos una mayúscula y un número".
- **Dado** que el campo "Confirmar contraseña" contiene un valor diferente al campo "Contraseña" **Cuando** pulse el botón "Registrarse" **Entonces** se muestra el mensaje "Las contraseñas no coinciden" bajo el campo de confirmación.
- **Dado** que el usuario decide ingresar un número de teléfono (opcional) y lo hace con un formato incorrecto (diferente a 10 dígitos numéricos) **Cuando** pulse el botón "Registrarse" **Entonces** se muestra el mensaje "El teléfono debe tener 10 dígitos" bajo el campo de teléfono.
- **Dado** que el usuario recibió el correo de verificación **Cuando** pulse el botón "Verificar Cuenta" o en el enlace incluido en el correo electrónico y se cargue la página de verificación **Entonces** se muestra el mensaje "¡Tu cuenta ha sido verificada exitosamente!" y un botón "Ir a Iniciar Sesión", y en unos segundos redirige automáticamente al inicio de sesión.
- **Dado** que el usuario intenta verificar su cuenta usando un enlace cuyo plazo de validez ha vencido (más de 24 horas desde el registro) **Cuando** se cargue la página de verificación **Entonces** se muestra el mensaje "El enlace de verificación ha expirado. Por favor solicita uno nuevo." y un botón para volver al inicio.
- **Dado** que el usuario accede a la página de verificación con un enlace alterado o incorrecto **Cuando** se cargue la página de verificación **Entonces** se muestra un ícono rojo de error con el mensaje "Token inválido" y un botón para volver al inicio.

Historia de Usuario

Número: 4	**Usuario:** Agente, Administrador
Nombre historia: Registro de propiedades	
Prioridad en negocio: 94	**Riesgo en desarrollo:** MEDIO
Puntos estimados: 5	**Sprint:** 1
Programador responsable: Jhery Alarcón y Romario Flores	

Descripción:

Como Agente, Administrador **Quiero** registrar una propiedad con sus características, imágenes y documentos **Para** gestionarla en el sistema y publicarla en la plataforma.

Escenario de prueba:

- **Dado** que estoy en un paso determinado (exceptuando el 5) con todos los campos obligatorios completos **Cuando** pulse el botón "Siguiente" **Entonces** avanzo al paso siguiente.
- **Dado** que estoy en un paso determinado (exceptuando el 5) con campos obligatorios vacíos o inválidos **Cuando** pulse el botón "Siguiente" **Entonces** se muestran mensajes de error en los campos obligatorios correspondientes y no se permite avanzar al paso siguiente.
- **Dado** que estoy en cualquier paso posterior al Paso 1 **Cuando** pulse el botón "Anterior" **Entonces** regreso al paso anterior sin perder los datos ingresados previamente.
- **Dado** que ya hay imágenes cargadas y el total superaría las 15 imágenes permitidas **Cuando** seleccione archivos adicionales en el área de carga de imágenes del Paso 4 **Entonces** se muestra una notificación: "Solo puedes subir un máximo de 15 imágenes" y se impide la acción.
- **Dado** el intento de avanzar al paso 5 sin seleccionar ninguna imagen **Cuando** pulse el botón "Siguiente" **Entonces** se muestra el mensaje "Debe subir al menos una imagen" bajo el área de carga y no se avanza.
- **Dado** el intento de subida de archivos con formato no permitido en el área de carga de imágenes del paso 4 **Cuando** seleccione los archivos **Entonces** se muestra un error indicando que solo se permiten imágenes (JPG, PNG, WEBP).
- **Dado** la selección de un tipo de contrato determinado sin subir el documento de comercialización correspondiente en el paso 5 **Cuando** pulse el botón "Registrar Propiedad" **Entonces** se muestra un error indicando que falta el documento obligatorio y no se guarda la propiedad.
- **Dado** que he completado todos los pasos correctamente como Agente **Cuando** pulse el botón "Registrar Propiedad" en el Paso 5 **Entonces** la propiedad se guarda asignada automáticamente a la cuenta del agente, se sube la documentación adjunta y se redirige al panel de propiedades del agente.
- **Dado** que he completado todos los pasos correctamente como Administrador y he seleccionado un agente responsable **Cuando** pulse el botón "Registrar Propiedad" en el Paso 5 **Entonces** la propiedad se guarda asignada al agente seleccionado y se redirige al panel de propiedades del administrador.
- **Dado** que estoy en el Paso 5 como Administrador y no he seleccionado un agente responsable **Cuando** pulse el botón "Registrar Propiedad" **Entonces** se muestra el mensaje de error "Seleccione un agente" bajo el campo correspondiente y no se guarda la propiedad.
- **Dado** el ingreso de datos incompletos y/o inválidos en el paso 5 **Cuando** pulse el botón "Registrar Propiedad" **Entonces** se muestran mensajes de error en los campos obligatorios correspondientes.
- **Dado** que he comenzado a ingresar datos en el formulario sin haber registrado la propiedad **Cuando** intente navegar a otra sección de la plataforma o cerrar la pestaña del navegador **Entonces** se muestra una advertencia solicitando confirmación antes de abandonar el formulario.

Historia de Usuario

Número: 5	**Usuario:** Agente, Administrador
Nombre historia: Visualización de propiedades registradas	
Prioridad en negocio: 92	**Riesgo en desarrollo:** BAJO
Puntos estimados: 3	**Sprint:** 1
Programador responsable: Jhery Alarcón y Romario Flores	

Descripción:

Como Agente, Administrador **Quiero** ver las propiedades registradas según mi rol **Para** buscarlas, filtrarlas y gestionarlas fácilmente.

Escenario de prueba:

- **Dado** que el usuario con rol Agente accede al panel de propiedades **Cuando** la página cargue con la opción "Todas" activa (por defecto) **Entonces** se muestra la lista completa de propiedades del sistema mediante tarjetas, barra de búsqueda, una variedad de filtros y los controles "Todas" y "Mis Propiedades" visibles en la parte superior.
- **Dado** que el usuario con rol Agente está en el panel de propiedades **Cuando** pulse el botón "Mis Propiedades" del selector en la parte superior **Entonces** la lista se actualiza mostrando únicamente las propiedades asignadas o registradas por ese agente.
- **Dado** que el usuario con rol Administrador accede al panel de propiedades **Cuando** la página cargue **Entonces** se muestra la lista de todas las propiedades del sistema mediante tarjetas, barra de búsqueda, una variedad de filtros y un selector desplegable para filtrar por agente.
- **Dado** que el usuario con rol Administrador está en el panel de propiedades **Cuando** seleccione un agente específico del desplegable de agentes en la parte superior **Entonces** la lista se actualiza mostrando únicamente las propiedades asignadas a ese agente.
- **Dado** que no hay propiedades registradas o los filtros aplicados no producen resultados **Cuando** se cargue el panel o se apliquen los filtros **Entonces** se muestra el mensaje "No se encontraron propiedades" con el texto de apoyo "Intenta ajustar los filtros de búsqueda".
- **Dado** que el panel muestra las propiedades en vista de tarjetas **Cuando** se carguen las tarjetas **Entonces** cada tarjeta muestra: Imagen principal de la propiedad (o placeholder si no tiene), código interno en la esquina superior izquierda de la imagen, etiqueta de estado con color en la esquina superior derecha: disponible (verde), reservada (amarillo), vendida (morado), arrendada (azul) o inactiva (gris), precio en formato de moneda y etiqueta de Venta o Alquiler, título de la publicación, características disponibles (número de habitaciones, baños, área de terreno (m²) y área de construcción (m²)), ciudad y tipo de propiedad, nombre del agente asignado, botones de acción (ver negociaciones, editar propiedad, cambiar estado (si tiene permiso) y desactivar (solo Admin)).
- **Dado** que el agente o administrador está en el panel de propiedades **Cuando** seleccione alguna de las opciones de los filtros rápidos "Operación", "Tipo" o "Estado" en la barra de filtros **Entonces** el listado se actualiza automáticamente mostrando solo las propiedades que coinciden con la selección.
- **Dado** que el agente o administrador escribe texto en la barra de búsqueda **Cuando** ingrese un término (título, código interno, descripción o nombre de propietario) **Entonces** el sistema filtra y muestra las propiedades que coincidan.
- **Dado** que el agente o administrador está en el panel de propiedades **Cuando** pulse el botón "Filtros Globales" en la barra de filtros **Entonces** se abre un modal con opciones avanzadas de filtrado: rango de precio, ubicación por texto, número mínimo de habitaciones/baños/garajes, año de construcción, rangos de área de terreno y construcción, y selección de amenidades (piscina, seguridad, ascensor, balcón, terraza, patio, bodega, área BBQ, áreas comunales, gas centralizado, cisterna, lavandería, amoblado); al aplicar estos filtros el listado se actualiza mostrando solo las propiedades que cumplen todos los criterios seleccionados.
- **Dado** que el resultado actual contiene más de 9 propiedades **Cuando** se cargue la vista **Entonces** se muestra la barra de paginación con los botones "Anterior" y "Siguiente", y el indicador de página actual.
- **Dado** que el usuario está viendo el panel de propiedades **Cuando** seleccione una opción del selector de ordenamiento (Más recientes, Más antiguas, Precio bajo a alto, Precio alto a bajo, A-Z, Z-A) **Entonces** las propiedades se reorganizan en pantalla según el criterio seleccionado
- **Dado** que el usuario tiene al menos un filtro activo (los filtros activos cambian de color) **Cuando** pulse el ícono "X" que aparece junto a los filtros activos **Entonces** se limpian todos los filtros y la búsqueda, y se vuelven a mostrar todas las propiedades.

Historia de Usuario

Número: 6	**Usuario:** Agente, Administrador
Nombre historia: Edición de propiedades	
Prioridad en negocio: 91	**Riesgo en desarrollo:** MEDIO
Puntos estimados: 5	**Sprint:** 1
Programador responsable: Jhery Alarcón y Romario Flores	

Descripción:

Como Agente, Administrador **Quiero** editar una propiedad existente **Para** corregir o actualizar su información, imágenes y documentos de comercialización.

Escenario de prueba:

- **Dado** que un Agente o Administrador modifica datos de una propiedad que le corresponde **Cuando** pulse el botón "Guardar Cambios" **Entonces** se muestra el mensaje "Propiedad actualizada correctamente" y el sistema redirige al panel de propiedades.
- **Dado** que un Agente intenta acceder a la página de edición de una propiedad que no le fue asignada **Cuando** la página cargue **Entonces** se muestra una pantalla con el mensaje "Acceso denegado" y el texto "No tienes permisos para editar esta propiedad", con un botón para volver.
- **Dado** que hay campos obligatorios vacíos o con valores inválidos **Cuando** pulse el botón "Guardar Cambios" **Entonces** se muestra la notificación "Por favor, corrige los errores en el formulario" y los campos con error se marcan con su mensaje correspondiente.
- **Dado** que se eliminaron todos los propietarios de la propiedad **Cuando** pulse el botón "Guardar Cambios" **Entonces** se muestra la notificación "Debes asignar al menos un propietario" y no se guarda la propiedad.
- **Dado** que hay propietarios asignados pero sus porcentajes de participación no suman exactamente 100% **Cuando** pulse el botón "Guardar Cambios" **Entonces** se muestra la notificación "La suma de porcentajes debe ser 100% (Actual: X%)" y no se guarda.
- **Dado** que hay propietarios asignados, pero ninguno está marcado como propietario principal **Cuando** pulse el botón "Guardar Cambios" **Entonces** se muestra la notificación "Debe marcar un propietario como principal" y no se guarda.
- **Dado** el cambio de tipo de contrato sin subir el documento correspondiente **Cuando** pulse el botón "Guardar Cambios" **Entonces** se muestra la notificación de error correspondiente indicando el documento faltante y no se guarda.
- **Dado** que se eliminaron todas las imágenes existentes y no se subieron imágenes nuevas **Cuando** pulse el botón "Guardar Cambios" **Entonces** se muestra el error "Debe subir al menos una imagen de la propiedad" y no se guarda.
- **Dado** que el usuario ha modificado algún campo del formulario sin pulsar "Guardar Cambios" **Cuando** intente navegar a otra sección de la plataforma o cerrar/recargar la pestaña del navegador **Entonces** se muestra una advertencia solicitando confirmación antes de abandonar el formulario.

Historia de Usuario	
Número: 7	**Usuario:** Agente, Administrador
Nombre historia: Actualización del Estado de Publicación de Propiedades	
Prioridad en negocio: 90	**Riesgo en desarrollo:** MEDIO
Puntos estimados: 3	**Sprint:** 2
Programador responsable: Jhery Alarcón y Romario Flores	

Descripción:

Como Agente, Administrador **Quiero** actualizar el estado administrativo de una propiedad (disponible o inactiva) **Para** reflejar si está activa o no.

Escenario de prueba:

- **Dado** que un Agente ve en el panel una propiedad que le fue asignada y no tiene restricciones por negociaciones **Cuando** pulse el ícono de "Cambiar Estado", seleccione "Disponible" o "Inactiva" y pulse "Guardar" **Entonces** el modal muestra el mensaje "Estado actualizado correctamente", se cierra automáticamente y la etiqueta de estado en la tarjeta cambia al nuevo valor.
- **Dado** que un Administrador ve cualquier propiedad en el panel (propia o de otro agente) y no tiene restricciones por negociaciones **Cuando** pulse el ícono de "Cambiar Estado", seleccione "Disponible" o "Inactiva" y pulse "Guardar" **Entonces** el modal muestra el mensaje "Estado actualizado correctamente", se cierra automáticamente y la etiqueta de estado en la tarjeta cambia al nuevo valor.
- **Dado** que un Agente está en el panel de propiedades viendo una propiedad que no le fue asignada **Cuando** revise los botones de acción de esa tarjeta **Entonces** el ícono de "Cambiar Estado" no está visible.
- **Dado** que un Agente propietario o Administrador abre el modal "Cambiar estado de la propiedad" **Cuando** se despliegue el selector de estados **Entonces** únicamente aparecen las opciones "Disponible" e "Inactiva", y el modal muestra un aviso informativo que indica que para marcar la propiedad como Vendida, Reservada o Arrendada es necesario finalizar una Negociación.
- **Dado** que una propiedad tiene una negociación activa en etapa "Cierre" **Cuando** el Agente propietario o el Administrador seleccione "Inactiva" y pulse "Guardar" **Entonces** el modal muestra el mensaje de error: " No se puede desactivar la propiedad. Hay una negociación en etapa de CIERRE (Reserva activa)" y el estado no cambia.
- **Dado** que una propiedad tiene negociaciones en etapa "Interés", "Negociación", "Cancelada" o "Finalizada" (pero ninguna activa en "Cierre") **Cuando** el Agente propietario o el Administrador seleccione "Inactiva" y pulse "Guardar" **Entonces** el modal muestra el mensaje "Estado actualizado correctamente", se cierra automáticamente y la etiqueta de la tarjeta refleja "inactiva".
- **Dado** que una propiedad tiene estado "Reservada", "Vendida" o "Arrendada" y hay una negociación activa en etapa "Cierre" o "Finalizada" **Cuando** el Agente propietario o el Administrador seleccione "Disponible" y pulse "Guardar" **Entonces** el modal muestra el mensaje "No se puede marcar como disponible. Hay una negociación en Cierre/Finalizada que bloquea la propiedad" y el estado no cambia.
- **Dado** que una propiedad tiene estado "Reservada", "Vendida" o "Arrendada" pero ya no tiene negociaciones activas en etapa Cierre o Finalizada **Cuando** el Agente propietario o el Administrador seleccione "Disponible" y pulse "Guardar" **Entonces** el modal muestra el mensaje "Estado actualizado correctamente" y la etiqueta de la tarjeta refleja "disponible".

Historia de Usuario	
Número: 8	**Usuario:** Agente, Administrador
Nombre historia: Desactivación de propiedades	
Prioridad en negocio: 89	**Riesgo en desarrollo:** MEDIO
Puntos estimados: 3	**Sprint:** 2
Programador responsable: Jhery Alarcón y Romario Flores	

Descripción:

Como Agente, Administrador **Quiero** desactivar una propiedad registrada **Para** retirarla de los listados públicos y mantener las propiedades organizadas.

Escenario de prueba:

- **Dado** que un Agente propietario o Administrador ve una tarjeta de propiedad en el panel de propiedades **Cuando** pulse el ícono de papelera y confirme pulsando el botón "Desactivar" en el modal de confirmación **Entonces** el modal muestra el mensaje "Propiedad desactivada correctamente", la propiedad cambia a estado "inactiva" y deja de aparecer en el portal público; en el panel interno (admin/agente) sigue visible con la etiqueta "inactiva".
- **Dado** que un Agente visualiza en el panel una propiedad que no le fue asignada **Cuando** revise los botones de acción de esa tarjeta **Entonces** el ícono de papelera no aparece.
- **Dado** que un Agente visualiza en el panel una propiedad que le fue asignada **Cuando** revise los botones de acción de esa tarjeta **Entonces** el ícono de papelera es visible junto a los demás botones de gestión.
- **Dado** que un Agente propietario o Administrador abre el modal de confirmación de desactivación **Cuando** pulse el botón "Cancelar" **Entonces** el modal se cierra y la propiedad permanece en su estado actual sin cambios.
- **Dado** que la propiedad tiene una negociación activa en etapa "Cierre" **Cuando** el Agente propietario o el Administrador pulse el botón "Desactivar" en el modal de confirmación **Entonces** el modal muestra el mensaje: "No se puede desactivar la propiedad. Hay una negociación en etapa de CIERRE (Reserva activa)" y el estado de la propiedad no cambia.
- **Dado** que la propiedad tiene negociaciones únicamente en etapas "Interés", "Negociación", "Cancelada" o "Finalizada" (pero ninguna en "Cierre") **Cuando** el Agente propietario o el Administrador pulse el botón "Desactivar" en el modal de confirmación **Entonces** la desactivación se realiza correctamente y la propiedad pasa a estado "inactiva".

Historia de Usuario	
Número: 9	**Usuario:** Visitante, Cliente
Nombre historia: Búsqueda y Filtrado de Propiedades en el Portal Público	
Prioridad en negocio: 88	**Riesgo en desarrollo:** BAJO
Puntos estimados: 3	**Sprint:** 2
Programador responsable: Jhery Alarcón y Romario Flores	

Descripción:

Como Visitante, Cliente **Quiero** buscar y filtrar propiedades disponibles por diferentes criterios **Para** encontrar inmuebles que se adapten a mis necesidades.

Escenario de prueba:

- **Dado** que el visitante o cliente está en el portal de propiedades **Cuando** seleccione una de las pestañas "Todos", "Comprar" o "Alquilar", elija un tipo de propiedad, una ciudad y un rango de precio mínimo/máximo en los selectores **Entonces** el listado se actualiza mostrando únicamente las propiedades que cumplen todos los criterios, y el contador superior refleja la cantidad de resultados.
- **Dado** que el visitante o cliente está en el portal de propiedades **Cuando** escriba en el campo "Ubicación o Palabra Clave" (soporta título, ubicación, código o descripción) **Entonces** el listado se actualiza mostrando las propiedades que coincidan con el texto ingresado
- **Dado** que el visitante o cliente desea usar criterios más específicos **Cuando** pulse el ícono de ajustes para desplegar los filtros avanzados y seleccione valores de habitaciones, baños, parqueaderos, estado físico, pisos, área de construcción, área de terreno, año mínimo de construcción o amenidades **Entonces** el listado se actualiza con las propiedades que cumplen todos los criterios combinados.
- **Dado** que los filtros activos no producen ningún resultado **Cuando** el listado se actualice
- **Entonces** se muestra el mensaje "No encontramos coincidencias" con el texto de apoyo "Intenta ajustar los filtros o buscar términos más generales" y un botón "Ver todas las propiedades" que limpia todos los filtros y la búsqueda.
- **Dado** que hay propiedades en el listado **Cuando** seleccione un criterio en el selector de ordenamiento (Precio, Reciente o A-Z) y/o pulse el botón de dirección para alternar entre ascendente y descendente **Entonces** las propiedades se reorganizan según el criterio y la dirección seleccionada.
- **Dado** que el usuario tiene filtros avanzados activos y el panel de filtros avanzados está visible **Cuando** pulse el botón "Borrar Filtros" que aparece en la parte inferior del panel avanzado **Entonces** se limpian todos los filtros y la búsqueda, y el listado vuelve a mostrar todas las propiedades disponibles
- **Dado** que el visitante o cliente ingresa al portal sin aplicar ningún filtro y no hay propiedades publicadas **Cuando** se cargue la página **Entonces** se muestra el mensaje "Aún no hay propiedades disponibles" con el texto de apoyo "Pronto publicaremos nuevas propiedades. ¡Vuelve a visitarnos!".

Historia de Usuario	
Número: 10	**Usuario:** Visitante, Cliente
Nombre historia: Visualización del detalle de la propiedad	
Prioridad en negocio: 87	**Riesgo en desarrollo:** BAJO
Puntos estimados: 3	**Sprint:** 2
Programador responsable: Jhery Alarcón y Romario Flores	

Descripción:

Como Visitante, Cliente **Quiero** visualizar la información detallada de una propiedad **Para** conocer sus características, precio, ubicación, amenidades y tener opciones directas de contacto comercial.

Escenario de prueba:

- **Dado** que el visitante o cliente ve una tarjeta de propiedad en el portal público **Cuando** haga clic en la tarjeta **Entonces** se carga la página de detalle mostrando: título, precio, tipo de transacción, galería de imágenes, características con iconos (habitaciones, baños, parqueaderos, área de construcción, área de terreno), descripción, detalles (tipo de propiedad, estado físico, tipo de transacción, código), ubicación (provincia, ciudad, dirección) y amenidades.
- **Dado** que la ficha de detalle muestra miniaturas en la galería **Cuando** haga clic directamente en una miniatura **Entonces** se abre un visor de imágenes a pantalla completa con navegación y se puede cerrar para volver a la ficha.
- **Dado** que un Cliente autenticado accede a la ficha de detalle **Cuando** la página se cargue **Entonces** el formulario de contacto aparece pre-rellenado con su nombre, email y el mensaje predefinido: "Hola, estoy interesado en obtener más información sobre la propiedad: [título]".
- **Dado** que el formulario de contacto tiene los campos nombre, email, teléfono y mensaje completos y válidos **Cuando** pulse el botón "Enviar Mensaje" **Entonces** aparece la notificación "¡Mensaje enviado con éxito! Un agente te contactará pronto.", el botón cambia a "Enviado" y se deshabilita para evitar reenvíos.
- **Dado** que el formulario tiene campos obligatorios vacíos o un email con formato incorrecto **Cuando** pulse "Enviar Mensaje" **Entonces** aparece la notificación "Por favor, corrige los errores en el formulario.", se enmarcan y se mencionan los errores en los campos con problemas y el mensaje no se envía.

Historia Técnica

Título: Autenticación y Seguridad | **Estimación:** 8

Descripción:

Se implementó un sistema de seguridad de tres capas sobre la plataforma inmobiliaria. En el *backend* *auth.controller.js*, el inicio de sesión verifica credenciales bloqueando cuentas inactivas y sin verificar, y genera un token *JWT* firmado con *JWT_SECRET* que contiene el *id* y rol del usuario con vigencia de 24 horas. Las contraseñas se protegen encriptándolas con *bcrypt* antes de persistirlas, aplicándose también al restablecer la clave. Para el control de acceso, el *middleware verificarToken.js* intercepta cada petición protegida, valida el *token* y comprueba en la base de datos que el usuario exista y permanezca activo; los *middlewares esAdmin.js* y *esAgenteOAdmin.js* restringen operaciones críticas según el rol. Adicionalmente, se implementó recuperación de contraseña mediante *tokens* de uso único con expiración de 1 hora, generados con el módulo nativo *crypto*. En el *frontend RutaPrivada.jsx*, se protegen las rutas validando el *token* almacenado en el *localStorage* y redirigiendo a */login* si el usuario no está autenticado o no tiene el rol requerido.

Criterio de aceptación:

- El login bloquea el acceso si la cuenta está inactiva o sin verificación de correo
- Las contraseñas se almacenan siempre encriptadas con bcrypt (nunca en texto plano)
- Toda ruta protegida verifica el token y comprueba el estado activo del usuario en la BD
- Los middlewares de rol impiden que un agente acceda a funciones exclusivas de administrador
- El flujo de recuperación de contraseña genera un token de un solo uso que expira en 1 hora
- Las rutas del frontend redirigen automáticamente a /login si no hay sesión activa válida

Beneficio de justificación:

La plataforma gestiona datos sensibles de propietarios, clientes y negociaciones de alto valor económico. Este sistema garantiza que únicamente usuarios legítimos y con el rol apropiado puedan acceder o modificar la información, protegiendo la integridad del negocio y la privacidad de los involucrados.

Impacto en el producto o proyecto:

Sin esta implementación, cualquier persona con acceso a la URL podría consultar o modificar propiedades, clientes y negociaciones sin restricción; las contraseñas almacenadas en texto plano serían vulnerables ante cualquier acceso indebido a la base de datos; y un agente podría ejecutar acciones administrativas como desactivar cuentas o acceder a estadísticas globales del sistema.

Dependencias:

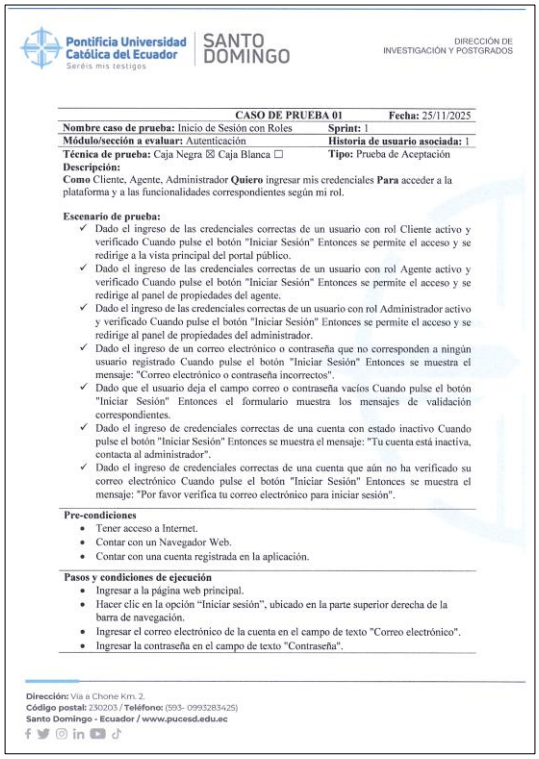
- Librerías jsonwebtoken, bcrypt, crypto (Node.js nativo) en el backend
- Variables de entorno JWT_SECRET y DATABASE_URL
- Servicio de correo configurado en email.js (para verificación y recuperación de contraseña)
- react-router-dom en el frontend para el componente
- RutaPrivada.jsx

Tareas o subtareas:

- Autenticación con JWT con validaciones de estado de cuenta (auth.controller.js)
- Encriptación de contraseñas con bcrypt (auth.controller.js) (registro y restablecimiento)
- Middlewares de autorización por roles (verificarToken.js, esAdmin.js, esAgenteOAdmin.js)
- Protección de rutas en React (RutaPrivada.jsx, tokenUtils.js)
- Verificación de cuenta por correo electrónico (auth.controller.js, verifyEmail)
- Recuperación de contraseña con token seguro y expiración (auth.controller.js, forgotPassword, resetPassword, RecuperarPassword.jsx)

Historias de usuario completas: [Historias de Usuario]

Anexo V: Pruebas de aceptación



Pontificia Universidad Católica del Ecuador — Sedes mis testigos | SANTO DOMINGO — DIRECCIÓN DE INVESTIGACIÓN Y POSTGRADOS

CASO DE PRUEBA 01 — **Fecha:** 25/11/2025

Nombre caso de prueba: Inicio de Sesión con Roles | **Sprint:** 1

Módulo/sección a evaluar: Autenticación | **Historia de usuario asociada:** 1

Técnica de prueba: Caja Negra ☒ Caja Blanca ☐ | **Tipo:** Prueba de Aceptación

Descripción:

Como Cliente, Agente, Administrador **Quiero** ingresar mis credenciales **Para** acceder a la plataforma y a las funcionalidades correspondientes según mi rol.

Escenario de prueba:

- ✓ Dado el ingreso de las credenciales correctas de un usuario con rol Cliente activo y verificado Cuando pulse el botón "Iniciar Sesión" Entonces se permite el acceso y se redirige a la vista principal del portal público.
- ✓ Dado el ingreso de las credenciales correctas de un usuario con rol Agente activo y verificado Cuando pulse el botón "Iniciar Sesión" Entonces se permite el acceso y se redirige al panel de propiedades del agente.
- ✓ Dado el ingreso de las credenciales correctas de un usuario con rol Administrador activo y verificado Cuando pulse el botón "Iniciar Sesión" Entonces se permite el acceso y se redirige al panel de propiedades del administrador.
- ✓ Dado el ingreso de un correo electrónico o contraseña que no corresponden a ningún usuario registrado Cuando pulse el botón "Iniciar Sesión" Entonces se muestra el mensaje: "Correo electrónico o contraseña incorrectos".
- ✓ Dado que el usuario deja el campo correo o contraseña vacíos Cuando pulse el botón "Iniciar Sesión" Entonces el formulario muestra los mensajes de validación correspondientes.
- ✓ Dado el ingreso de credenciales correctas de una cuenta con estado inactivo Cuando pulse el botón "Iniciar Sesión" Entonces se muestra el mensaje: "Tu cuenta está inactiva, contacta al administrador".
- ✓ Dado el ingreso de credenciales correctas de una cuenta que aún no ha verificado su correo electrónico Cuando pulse el botón "Iniciar Sesión" Entonces se muestra el mensaje: "Por favor verifica tu correo electrónico para iniciar sesión".

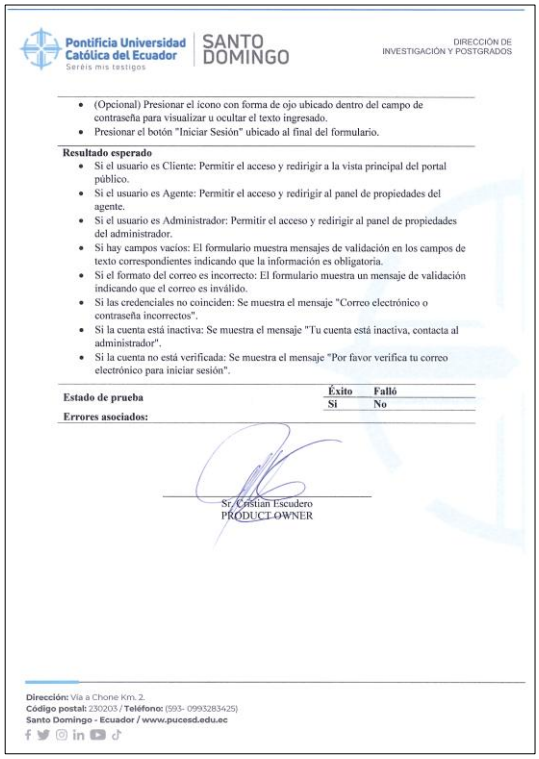
Pre-condiciones

- Tener acceso a Internet.
- Contar con un Navegador Web.
- Contar con una cuenta registrada en la aplicación.

Pasos y condiciones de ejecución

- Ingresar a la página web principal.
- Hacer clic en la opción "Iniciar sesión", ubicado en la parte superior derecha de la barra de navegación.
- Ingresar el correo electrónico de la cuenta en el campo de texto "Correo electrónico".
- Ingresar la contraseña en el campo de texto "Contraseña".

Dirección: Vía a Chone Km. 2.
Código postal: 230203 / Teléfono: (593- 0993283425)
Santo Domingo - Ecuador / www.pucesd.edu.ec



Pontificia Universidad Católica del Ecuador — Sedes mis testigos | SANTO DOMINGO — DIRECCIÓN DE INVESTIGACIÓN Y POSTGRADOS

- (Opcional) Presionar el ícono con forma de ojo ubicado dentro del campo de contraseña para visualizar u ocultar el texto ingresado.
- Presionar el botón "Iniciar Sesión" ubicado al final del formulario.

Resultado esperado

- Si el usuario es Cliente: Permitir el acceso y redirigir a la vista principal del portal público.
- Si el usuario es Agente: Permitir el acceso y redirigir al panel de propiedades del agente.
- Si el usuario es Administrador: Permitir el acceso y redirigir al panel de propiedades del administrador.
- Si hay campos vacíos: El formulario muestra mensajes de validación en los campos de texto correspondientes indicando que la información es obligatoria.
- Si el formato del correo es incorrecto: El formulario muestra un mensaje de validación indicando que el correo es inválido.
- Si las credenciales no coinciden: Se muestra el mensaje "Correo electrónico o contraseña incorrectos".
- Si la cuenta está inactiva: Se muestra el mensaje "Tu cuenta está inactiva, contacta al administrador".
- Si la cuenta no está verificada: Se muestra el mensaje "Por favor verifica tu correo electrónico para iniciar sesión".

Estado de prueba	**Éxito**	**Falló**
	Si	**No**

Errores asociados:

Sr. Cristian Escudero
PRODUCT OWNER

Dirección: Vía a Chone Km. 2.
Código postal: 230203 / Teléfono: (593- 0993283425)
Santo Domingo - Ecuador / www.pucesd.edu.ec

CASO DE PRUEBA 02	Fecha: 25/11/2025
Nombre caso de prueba: Cierre de Sesión	**Sprint:** 1
Módulo/sección a evaluar: Autenticación	**Historia de usuario asociada:** 2
Técnica de prueba: Caja Negra ☒ Caja Blanca ☐	**Tipo:** Prueba de Aceptación

Descripción:
Como Cliente, Agente, Administrador **Quiero** cerrar la sesión de mi cuenta **Para** proteger la misma y evitar accesos no autorizados.

Escenario de prueba:
- ✓ Dado que existe una sesión activa de un usuario con rol Cliente Cuando pulse la opción "Cerrar Sesión" disponible en el menú desplegable de su perfil en la barra de navegación Entonces la sesión se cierra, el menú de navegación vuelve a mostrar los botones "Iniciar Sesión" y "Registrarse", y se muestra la página de inicio del portal público.
- ✓ Dado que existe una sesión activa de un usuario con rol Agente Cuando pulse el ícono de "Cerrar sesión" ubicado en la parte inferior del panel lateral Entonces la sesión se cierra y se muestra la página de inicio del portal público.
- ✓ Dado que existe una sesión activa de un usuario con rol Administrador Cuando pulse el ícono de "Cerrar sesión" ubicado en la parte inferior del panel lateral Entonces la sesión se cierra y se muestra la página de inicio del portal público.
- ✓ Dado que un usuario acaba de cerrar sesión Cuando intente acceder directamente a una página restringida (por ejemplo, el panel del agente o del administrador) Entonces el sistema no permite el acceso y lo redirige a la pantalla de inicio de sesión.

Pre-condiciones
- Tener acceso a Internet.
- Contar con un Navegador Web.
- Tener una sesión activa en la aplicación.

Pasos y condiciones de ejecución

Para usuario con rol Cliente:
- Hacer clic en el botón del perfil (que muestra la inicial del nombre del usuario y el correo electrónico), ubicado en la parte superior derecha de la barra de navegación, para desplegar el menú de opciones.
- Hacer clic en la opción " Cerrar Sesión".

Para usuario con rol Agente o Administrador:
- Ubicarse en el panel lateral de navegación (menú izquierdo).
- Dirigirse a la parte inferior de dicho panel, donde se muestra el perfil del usuario.
- Hacer clic en el botón con el ícono de salida (Cerrar sesión).

Resultado esperado
- El sistema cierra la cuenta de inmediato y redirige al usuario a la página principal del portal público.

Pontificia Universidad Católica del Ecuador | SANTO DOMINGO
Seréis mis testigos
DIRECCIÓN DE INVESTIGACIÓN Y POSTGRADOS

- Tras el cierre de sesión, la barra de navegación superior se actualiza mostrando las opciones "Iniciar sesión" y "Registrarse".

Estado de prueba	Éxito	Falló
	Si	No

Errores asociados:

Sr. Cristian Escudero
PRODUCT OWNER

Dirección: Vía a Chone Km. 2.
Código postal: 230203 / **Teléfono:** (593- 0993283425)
Santo Domingo - Ecuador / www.pucesd.edu.ec

Pontificia Universidad Católica del Ecuador | SANTO DOMINGO
Seréis mis testigos
DIRECCIÓN DE INVESTIGACIÓN Y POSTGRADOS

CASO DE PRUEBA 03	Fecha: 25/11/2025
Nombre caso de prueba: Registro de Cuenta y Verificación (cliente externo)	**Sprint:** 1
Módulo/sección a evaluar: Autenticación/ Registro	**Historia de usuario asociada:** 3
Técnica de prueba: Caja Negra ☒ Caja Blanca ☐	**Tipo:** Prueba de Aceptación

Descripción:
Como Visitante **Quiero** crear una cuenta con mis datos y verificarla mediante el enlace enviado a mi correo electrónico **Para** navegar las diferentes propiedades de la plataforma, guardar mis favoritas y recibir recomendaciones personalizadas de inmuebles que me podrían interesar.

Escenario de prueba:
- ✓ Dado el ingreso de datos válidos en los campos obligatorios del formulario de registro de cuenta (nombre, correo electrónico, contraseña y confirmación de contraseña) Cuando pulse el botón "Registrarse" Entonces aparece una notificación indicando que se envió un enlace de verificación al correo electrónico ingresado, y se redirige a la página de inicio de sesión.
- ✓ Dado que el usuario deja en blanco el campo nombre, correo electrónico, contraseña o confirmación de contraseña Cuando pulse el botón "Registrarse" Entonces se muestran mensajes de advertencia en los campos correspondientes indicando que son obligatorios.
- ✓ Dado que el usuario ingresa un correo electrónico con formato incorrecto Cuando pulse el botón "Registrarse" Entonces se muestra un mensaje de correo inválido en el campo respectivo.
- ✓ Dado el ingreso de un correo electrónico que ya pertenece a una cuenta existente Cuando pulse el botón "Registrarse" Entonces se muestra el mensaje "El correo ya está registrado".
- ✓ Dado el ingreso de una contraseña que no cumple los requisitos mínimos (mínimo 8 caracteres, una mayúscula y un número) Cuando pulse el botón "Registrarse" Entonces se muestra el mensaje "Debe tener 8 caracteres, al menos una mayúscula y un número".
- ✓ Dado que el campo "Confirmar contraseña" contiene un valor diferente al campo "Contraseña" Cuando pulse el botón "Registrarse" Entonces se muestra el mensaje "Las contraseñas no coinciden" bajo el campo de confirmación.
- ✓ Dado que el usuario decide ingresar un número de teléfono (opcional) y lo hace con un formato incorrecto (diferente a 10 dígitos numéricos) Cuando pulse el botón "Registrarse" Entonces se muestra el mensaje "El teléfono debe tener 10 dígitos" bajo el campo de teléfono.
- ✓ Dado que el usuario recibió el correo de verificación Cuando pulse el botón "Verificar Cuenta" o en el enlace incluido en el correo electrónico y se cargue la página de verificación Entonces se muestra el mensaje "¡Tu cuenta ha sido verificada exitosamente!" y un botón "Ir a Iniciar Sesión", y en unos segundos redirige automáticamente al inicio de sesión.
- ✓ Dado que el usuario intenta verificar su cuenta usando un enlace cuyo plazo de validez ha vencido (más de 24 horas desde el registro) Cuando se cargue la página de

Dirección: Vía a Chone Km. 2.
Código postal: 230203 / **Teléfono:** (593- 0993283425)
Santo Domingo - Ecuador / www.pucesd.edu.ec

Pontificia Universidad Católica del Ecuador | SANTO DOMINGO
Seréis mis testigos
DIRECCIÓN DE INVESTIGACIÓN Y POSTGRADOS

verificación Entonces se muestra el mensaje "El enlace de verificación ha expirado. Por favor solicita uno nuevo." y un botón para volver al inicio.
- ✓ Dado que el usuario accede a la página de verificación con un enlace alterado o incorrecto Cuando se cargue la página de verificación Entonces se muestra un ícono rojo de error con el mensaje "Token inválido" y un botón para volver al inicio.

Pre-condiciones
- Tener acceso a Internet.
- Contar con un Navegador Web.
- Contar con una cuenta de correo electrónico válida y accesible.

Pasos y condiciones de ejecución

Fase 1: Registro de Cuenta
- Ingresar a la página web principal.
- Hacer clic en la opción "Registrarse", ubicado en la parte superior derecha de la barra de navegación.
- Ingresar los nombres en el campo de texto "Nombre completo".
- Ingresar el correo electrónico en el campo de texto "Correo electrónico".
- (Opcional) Ingresar el número telefónico en el campo de texto "Teléfono".
- Ingresar la contraseña en el campo de texto "Contraseña".
- Ingresar nuevamente la contraseña en el campo de texto "Confirmar contraseña".
- (Opcional) Presionar el ícono con forma de ojo ubicado dentro de los campos de contraseña para visualizar u ocultar el texto ingresado.
- Presionar el botón "Registrarse" ubicado al final del formulario.

Fase 2: Verificación de Correo
- Abrir la bandeja de entrada del correo electrónico ingresado en el formulario de registro de cuenta.
- Buscar y abrir el correo electrónico de verificación enviado por la plataforma.
- Hacer clic en el enlace de verificación o el botón "Verificar Cuenta" incluido en el cuerpo del mensaje.
- Esperar a que se abra la página de comprobación en el navegador web.

Resultado esperado

Para el Registro de Cuenta:
- Si el registro es exitoso: El botón principal cambia su texto a "¡Cuenta Creada! Redirigiendo...". El sistema muestra una notificación con el texto "¡Registro exitoso!" y redirige automáticamente a la pantalla de inicio de sesión tras unos segundos.
- Si hay campos vacíos: Se muestran mensajes indicando la información que hace falta.
- Si el formato del correo es incorrecto: Se muestra el mensaje "Correo inválido".
- Si la contraseña es débil: Se muestra el mensaje "Debe tener 8 caracteres, al menos una mayúscula y un número".
- Si las contraseñas no coinciden: Se muestra el mensaje "Las contraseñas no coinciden".

Dirección: Vía a Chone Km. 2.
Código postal: 230203 / **Teléfono:** (593- 0993283425)
Santo Domingo - Ecuador / www.pucesd.edu.ec

- Si el teléfono es inválido: Se muestra el mensaje "El teléfono debe tener 10 dígitos".
- Si el correo ya pertenece a otra cuenta: Se muestra el mensaje de advertencia "El correo ya está registrado".

Para la Verificación de Correo:

- Si la verificación es exitosa: Se muestra en pantalla el mensaje "¡Tu cuenta ha sido verificada exitosamente!" y un botón "Ir a Iniciar Sesión", y en unos segundos redirige automáticamente al inicio de sesión.
- Si el enlace expiró (Error): Se muestra en pantalla el mensaje "El enlace de verificación ha expirado. Por favor solicita uno nuevo." y un botón para volver al inicio.
- Si el enlace está alterado o incorrecto (Error): Se muestra en pantalla el mensaje "Token inválido" y un botón para volver al inicio.

Estado de prueba	Éxito	Falló
	Si	No

Errores asociados:

Sr. Cristian Escudero
PRODUCT OWNER

Pontificia Universidad Católica del Ecuador | SANTO DOMINGO
Seréis mis testigos
DIRECCIÓN DE INVESTIGACIÓN Y POSTGRADOS

CASO DE PRUEBA 04 **Fecha:** 25/11/2025

Nombre caso de prueba: Registro de propiedades **Sprint:** 1
Módulo/sección a evaluar: Gestión de propiedades **Historia de usuario asociada:** 4
Técnica de prueba: Caja Negra ☒ Caja Blanca ☐ **Tipo:** Prueba de Aceptación

Descripción:
Como Agente, Administrador **Quiero** registrar una propiedad con sus características, imágenes y documentos **Para** gestionarla en el sistema y publicarla en la plataforma.

Escenario de prueba:

- ✓ Dado que estoy en un paso determinado (exceptuando el 5) con todos los campos obligatorios completos Cuando pulse el botón "Siguiente" Entonces avanzo al paso siguiente.
- ✓ Dado que estoy en un paso determinado (exceptuando el 5) con campos obligatorios vacíos o inválidos Cuando pulse el botón "Siguiente" Entonces se muestran mensajes de error en los campos obligatorios correspondientes y no se permite avanzar al paso siguiente.
- ✓ Dado que estoy en cualquier paso posterior al Paso 1 Cuando pulse el botón "Anterior" Entonces regreso al paso anterior sin perder los datos ingresados previamente.
- ✓ Dado que ya hay imágenes cargadas y el total superaría las 15 imágenes permitidas Cuando seleccione archivos adicionales en el área de carga de imágenes del Paso 4 Entonces se muestra una notificación: "Solo puedes subir un máximo de 15 imágenes" y se impide la acción.
- ✓ Dado el intento de avanzar al paso 5 sin seleccionar ninguna imagen Cuando pulse el botón "Siguiente" Entonces se muestra el mensaje "Debe subir al menos una imagen" bajo el área de carga y no se avanza.
- ✓ Dado el intento de subida de archivos con formato no permitido en el área de carga de imágenes del paso 4 Cuando seleccione los archivos Entonces se muestra un error indicando que solo se permiten imágenes (JPG, PNG, WEBP).
- ✓ Dado la selección de un tipo de contrato determinado sin subir el documento de comercialización correspondiente en el paso 5 Cuando pulse el botón "Registrar Propiedad" Entonces se muestra un error indicando que falta el documento obligatorio y no se guarda la propiedad.
- ✓ Dado que he completado todos los pasos correctamente como Agente Cuando pulse el botón "Registrar Propiedad" en el Paso 5 Entonces la propiedad se guarda asignada automáticamente a la cuenta del agente, se sube la documentación adjunta y se redirige al panel de propiedades del agente.
- ✓ Dado que he completado todos los pasos correctamente como Administrador y he seleccionado un agente responsable Cuando pulse el botón "Registrar Propiedad" en el Paso 5 Entonces la propiedad se guarda asignada al agente seleccionado y se redirige al panel de propiedades del administrador.
- ✓ Dado que estoy en el Paso 5 como Administrador y no he seleccionado un agente responsable Cuando pulse el botón "Registrar Propiedad" Entonces se muestra el

Dirección: Vía a Chone Km. 2.
Código postal: 230203 / **Teléfono:** (593- 0993283425)
Santo Domingo - Ecuador / www.pucesd.edu.ec

Pontificia Universidad Católica del Ecuador | SANTO DOMINGO
Seréis mis testigos
DIRECCIÓN DE INVESTIGACIÓN Y POSTGRADOS

mensaje de error "Seleccione un agente" bajo el campo correspondiente y no se guarda la propiedad.

- ✓ Dado el ingreso de datos incompletos y/o inválidos en el paso 5 Cuando pulse el botón "Registrar Propiedad" Entonces se muestran mensajes de error en los campos obligatorios correspondientes.
- ✓ Dado que he comenzado a ingresar datos en el formulario sin haber registrado la propiedad Cuando intente navegar a otra sección de la plataforma o cerrar la pestaña del navegador Entonces se muestra una advertencia solicitando confirmación antes de abandonar el formulario.

Pre-condiciones

- Tener acceso a Internet.
- Contar con un Navegador Web.
- Tener una sesión activa en el sistema con rol de Agente o Administrador.
- Disponer de la información, fotografías y documentos legales de la propiedad a registrar.

Pasos y condiciones de ejecución

Navegación general por los pasos:

- Ubicarse en el panel lateral de navegación (menú izquierdo).
- Desplegar la sección "Propiedades" haciendo clic sobre ella.
- Hacer clic en la opción "Registrar Propiedad" para abrir el formulario.
- Llenar la información solicitada en cada campo de texto, lista desplegable o selector, según el paso en el que se encuentre.
- Presionar el botón "Siguiente" ubicado en la parte inferior derecha para avanzar de sección.
- (Opcional) Presionar el botón "Anterior" ubicado en la parte inferior izquierda para retroceder y modificar datos previos.

Interacción con Multimedia (Paso 4):

- Ubicarse en la zona de carga indicada para subir fotografías de la propiedad.
- Hacer clic en el área de carga para abrir el explorador de archivos y seleccionar las imágenes correspondientes.

Finalización y Guardado (Paso 5):

- Desplazarse a la zona de "Documentación Legal y Técnica".
- Hacer clic en los botones de carga correspondientes a los documentos exigidos según el tipo de contrato seleccionado.
- (Solo aplicable para Administradores) Seleccionar el agente inmobiliario responsable en el campo de asignación provisto.
- Presionar el botón "Registrar Propiedad" ubicado al final del formulario del último paso.

Resultado esperado

Dirección: Vía a Chone Km. 2.
Código postal: 230203 / **Teléfono:** (593- 0993283425)
Santo Domingo - Ecuador / www.pucesd.edu.ec

Pontificia Universidad Católica del Ecuador | SANTO DOMINGO
Seréis mis testigos
DIRECCIÓN DE INVESTIGACIÓN Y POSTGRADOS

Navegación y Guardado Parcial:

- Al completar un paso correctamente: El sistema avanza automáticamente a la siguiente sección del formulario.
- Al intentar avanzar con información faltante: El sistema impide el avance y muestra mensajes de validación indicando qué datos son obligatorios o inválidos.
- Al presionar "Anterior": El sistema retrocede a la sección previa cargando toda la información ya introducida.
- Al intentar abandonar la página sin guardar: Se emite una advertencia solicitando confirmación del usuario para evitar la pérdida de los datos rellenados.

Validación de Multimedia (Paso 4):

- Al intentar superar el límite de imágenes: El sistema bloquea la acción y emite una notificación con el texto "Solo puedes subir un máximo de 15 imágenes".
- Al subir formatos no admitidos: El sistema bloquea el archivo y emite una notificación indicando que no es una imagen válida y que solo se permiten formatos como JPG, PNG o WEBP.
- Al intentar avanzar sin evidencia visual: Se muestra un mensaje de validación advirtiendo que "Debe subir al menos una imagen".

Finalización del Registro (Paso 5):

- Falta de documentación: Si no se sube el archivo correspondiente al contrato comercial elegido, el sistema impide guardar y notifica "Falta Contrato de Exclusividad" o "Falta Autorización de Venta" según corresponda.
- Falta de responsable (Administrador): Si no se le ha asignado la propiedad a un agente, el sistema bloquea el guardado y muestra el mensaje "Seleccione un agente".
- Datos insuficientes en el último paso: Si falta rellenar precios, comisiones o agregar a los propietarios, el sistema avisa con mensajes de validación sobre qué información es obligatoria.
- Registro Exitoso: El sistema guarda la propiedad mostrando la confirmación "Propiedad y documentos registrados correctamente", y en pocos segundos redirige al panel de propiedades.

Estado de prueba	Éxito	Falló
	Si	No

Errores asociados:

Sr. Cristian Escudero
PRODUCT OWNER

Dirección: Vía a Chone Km. 2.
Código postal: 230203 / **Teléfono:** (593- 0993283425)
Santo Domingo - Ecuador / www.pucesd.edu.ec

CASO DE PRUEBA 05 **Fecha:** 25/11/2025

Nombre caso de prueba: Visualización de Propiedades	**Sprint:** 1
Módulo/sección a evaluar: Gestión de Propiedades	**Historia de usuario asociada:** 5
Técnica de prueba: Caja Negra ☒ Caja Blanca ☐	**Tipo:** Prueba de Aceptación

Descripción:
Como Agente, Administrador **Quiero** ver las propiedades registradas según mi rol **Para** buscarlas, filtrarlas y gestionarlas fácilmente.

Escenario de prueba:

- ✓ Dado que el usuario con rol Agente accede al panel de propiedades Cuando la página cargue con la opción "Todas" activa (por defecto) Entonces se muestra la lista completa de propiedades del sistema mediante tarjetas, barra de búsqueda, una variedad de filtros y los controles "Todas" y "Mis Propiedades" visibles en la parte superior.
- ✓ Dado que el usuario con rol Agente está en el panel de propiedades Cuando pulse el botón "Mis Propiedades" del selector en la parte superior Entonces la lista se actualiza mostrando únicamente las propiedades asignadas o registradas por ese agente.
- ✓ Dado que el usuario con rol Administrador accede al panel de propiedades Cuando la página cargue Entonces se muestra la lista de todas las propiedades del sistema mediante tarjetas, barra de búsqueda, una variedad de filtros y un selector desplegable para filtrar por agente.
- ✓ Dado que el usuario con rol Administrador está en el panel de propiedades Cuando seleccione un agente específico del desplegable de agentes en la parte superior Entonces la lista se actualiza mostrando únicamente las propiedades asignadas a ese agente.
- ✓ Dado que no hay propiedades registradas o los filtros aplicados no producen resultados Cuando se cargue el panel o se apliquen los filtros Entonces se muestra el mensaje "No se encontraron propiedades" con el texto de apoyo "Intenta ajustar los filtros de búsqueda".
- ✓ Dado que el panel muestra las propiedades en vista de tarjetas Cuando se carguen las tarjetas Entonces cada tarjeta muestra: Imagen principal de la propiedad (o placeholder si no tiene), código interno en la esquina superior izquierda de la imagen, etiqueta de estado con color en la esquina superior derecha: disponible (verde), reservada (amarillo), vendida (morado), arrendada (azul) o inactiva (gris), precio en formato de moneda y etiqueta de Venta o Alquiler, título de la publicación, características disponibles (número de habitaciones, baños, área de terreno (m^2) y área de construcción (m^2)), ciudad y tipo de propiedad, nombre del agente asignado, botones de acción (ver negociaciones, editar propiedad, cambiar estado (si tiene permiso) y desactivar (solo Admin)).
- ✓ Dado que el agente o administrador está en el panel de propiedades Cuando seleccione alguna de las opciones de los filtros rápidos "Operación", "Tipo" o "Estado" en la barra de filtros Entonces el listado se actualiza automáticamente mostrando solo las propiedades que coinciden con la selección.
- ✓ Dado que el agente o administrador escribe texto en la barra de búsqueda Cuando ingrese un término (título, código interno, descripción o nombre de propietario) Entonces el sistema filtra y muestra las propiedades que coincidan.

Pontificia Universidad Católica del Ecuador
Seréis mis testigos
SANTO DOMINGO
DIRECCIÓN DE INVESTIGACIÓN Y POSTGRADOS

- ✓ Dado que el agente o administrador está en el panel de propiedades Cuando pulse el botón "Filtros Globales" en la barra de filtros Entonces se abre un modal con opciones avanzadas de filtrado: rango de precio, ubicación por texto, número mínimo de habitaciones/baños/garajes, año de construcción, rangos de área de terreno y construcción, y selección de amenidades (piscina, seguridad, ascensor, balcón, terraza, patio, bodega, área BBQ, áreas comunales, gas centralizado, cisterna, lavandería, amoblado); al aplicar estos filtros el listado se actualiza mostrando solo las propiedades que cumplen todos los criterios seleccionados.
- ✓ Dado que el resultado actual contiene más de 9 propiedades Cuando se cargue la vista Entonces se muestra la barra de paginación con los botones "Anterior" y "Siguiente", y el indicador de página actual.
- ✓ Dado que el usuario está viendo el panel de propiedades Cuando seleccione una opción del selector de ordenamiento (Más recientes, Más antiguas, Precio bajo a alto, Precio alto a bajo, A-Z, Z-A) Entonces las propiedades se reorganizan en pantalla según el criterio seleccionado
- ✓ Dado que el usuario tiene al menos un filtro activo (los filtros activos cambian de color) Cuando pulse el ícono "X" que aparece junto a los filtros activos Entonces se limpian todos los filtros y la búsqueda, y se vuelven a mostrar todas las propiedades.

Pre-condiciones

- Tener acceso a Internet.
- Contar con un Navegador Web.
- Tener una sesión activa en el sistema con rol de Agente o Administrador.

Pasos y condiciones de ejecución

Acceso inicial:

- Ubicarse en el panel lateral de navegación (menú izquierdo).
- Desplegar la sección "Propiedades" haciendo clic sobre ella.
- Hacer clic en el submenú "Mis Propiedades" (si es Agente) o "Panel de Propiedades" (si es Administrador).

Interacción con Filtros Rápidos (Agentes):

- Presionar el botón "Mis Propiedades" del selector situado en la parte superior (para ver las propiedades registradas o asignadas al agente autenticado).
- Para visualizar todas las propiedades registradas en el sistema, presionar el botón "Todas" ubicado junto al botón descrito anteriormente.

Interacción con Filtros Rápidos (Administradores):

- Ubicarse en la lista desplegable de la parte superior derecha de la pantalla (donde aparece inicialmente la opción "Todas las propiedades").
- Seleccionar a un agente específico de la lista.

Interacción con la Barra de Búsqueda y Ordenamiento:

Dirección: Vía a Chone Km. 2.
Código postal: 230203 / Teléfono: (593- 0993283425)
Santo Domingo - Ecuador / www.pucesd.edu.ec

Pontificia Universidad Católica del Ecuador
Seréis mis testigos
SANTO DOMINGO
DIRECCIÓN DE INVESTIGACIÓN Y POSTGRADOS

- Hacer clic en la barra de búsqueda y escribir un término.
- Clicar en la lista desplegable de ordenamiento (situada encima de la barra de búsqueda) y seleccionar una opción (Más recientes, Más antiguas, Precio: Bajo a Alto, Precio: Alto a Bajo, A-Z, Z-A).
- Seleccionar cualquiera de las opciones desplegables de filtro situadas también junto a la barra de búsqueda (Operación, Tipo o Estado).

Interacción con Filtros Globales (Avanzados):

- Ubicarse en la barra de filtros y presionar el botón "Filtros Globales".
- En la ventana emergente, introducir los rangos de precio, áreas (construcción/terreno), o número de habitaciones/baños/garajes requeridos.
- Seleccionar una o varias amenidades (ej. Piscina, Seguridad, Ascensor, etc.) haciendo clic directamente sobre las opciones.
- Presionar el botón negro "Ver Resultados" al final de la ventana emergente.
- (Opcional) Para deshacer la selección, presionar el ícono en forma de "X" que aparece junto al botón "Filtros Globales" o usar el botón "Limpiar todo" de la ventana emergente.

Resultado esperado

Carga Inicial y Vistas según Rol:

- Al ingresar como Agente el sistema carga todas las propiedades por defecto, mostrando los botones "Todas" / "Mis Propiedades".
- Al cambiar a "Mis Propiedades" (Agente): La lista se actualiza automáticamente enseñando únicamente las tarjetas de las propiedades asociadas a la cuenta del agente.
- Al ingresar como Administrador: El sistema carga todas las propiedades por defecto y la lista desplegable correspondiente a los agentes.
- Al usar el filtro por agente (Administrador): La vista muestra solo las propiedades que están a cargo del agente seleccionado.

Visualización de las Tarjetas de Propiedad:

- Contenido Individual: El sistema muestra cada propiedad en formato de tarjeta donde resalta: la fotografía principal clickeable que al presionarla redirige al detalle de dicha propiedad, el precio en dólares, el tipo de operación (Venta o Alquiler), el título, las características principales representadas mediante íconos (cantidades de habitaciones, baños, áreas), ubicación (ciudad y tipo de propiedad), y etiquetas superpuestas en la fotografía principal sobre el estado de la propiedad y su código interno.
- Controles y Botones de la Tarjeta: Al pie de cada tarjeta se muestra al agente encargado y los botones de acción permitidos (ver negociaciones, edición, cambiar estado de publicación o desactivar propiedad, según los permisos activos de la cuenta autenticada).

Dirección: Vía a Chone Km. 2.
Código postal: 230203 / Teléfono: (593- 0993283425)
Santo Domingo - Ecuador / www.pucesd.edu.ec

Pontificia Universidad Católica del Ecuador
Seréis mis testigos
SANTO DOMINGO
DIRECCIÓN DE INVESTIGACIÓN Y POSTGRADOS

- Si no hay propiedades o resultados que se acoplen a los filtros seleccionados se muestra el mensaje "No se encontraron propiedades" y "Intenta ajustar los filtros de búsqueda".

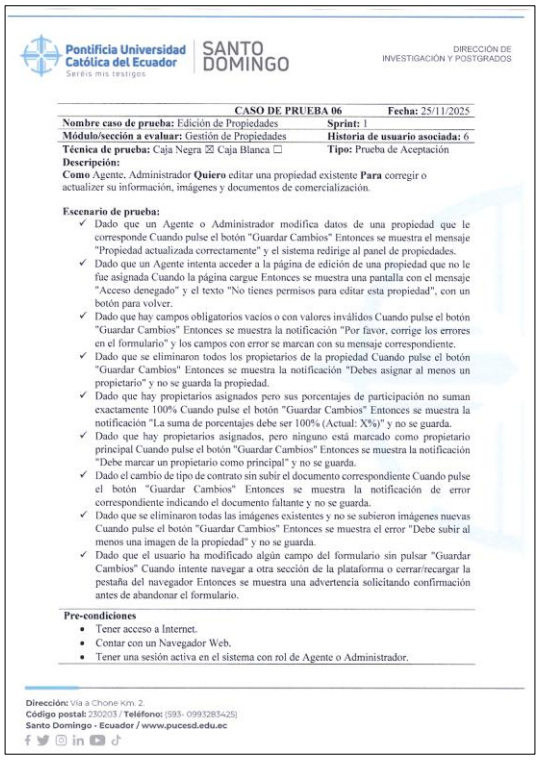
Filtrado, Búsqueda y Ordenamiento:

- Al escribir texto en la barra de búsqueda, seleccionar filtros rápidos (Estado, Tipo u Operación) o escoger opciones del filtro global, el listado actualiza los resultados automáticamente.
- Al escoger opciones del desplegable de ordenamiento, las tarjetas de las propiedades se reordenan respetando el criterio seleccionado.
- Al presionar la "X" al lado de los botones de filtro o el botón "Limpiar todo" de la ventana emergente que surge a partir del botón de "Filtros Globales", el sistema resetea al instante todos los campos y listas de la ventana mostrando nuevamente todo el inventario de propiedades general.

	Éxito	Falló
Estado de prueba	**Si**	**No**
Errores asociados:		

Sr. Cristian Escudero
PRODUCT OWNER

Dirección: Vía a Chone Km. 2.
Código postal: 230203 / Teléfono: (593- 0993283425)
Santo Domingo - Ecuador / www.pucesd.edu.ec



Pontificia Universidad Católica del Ecuador
Seréis mis testigos
SANTO DOMINGO
DIRECCIÓN DE INVESTIGACIÓN Y POSTGRADOS

CASO DE PRUEBA 06	Fecha: 25/11/2025
Nombre caso de prueba: Edición de Propiedades	**Sprint:** 1
Módulo/sección a evaluar: Gestión de Propiedades	**Historia de usuario asociada:** 6
Técnica de prueba: Caja Negra ☒ Caja Blanca ☐	**Tipo:** Prueba de Aceptación

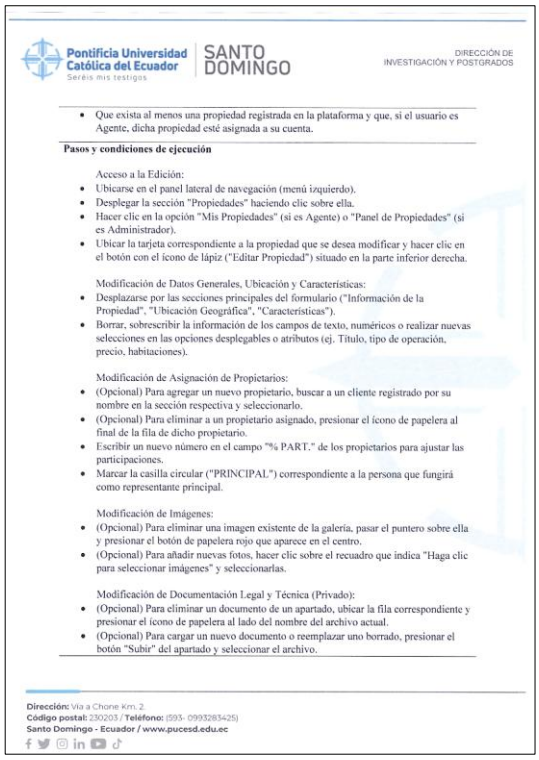
Descripción:
Como Agente, Administrador **Quiero** editar una propiedad existente **Para** corregir o actualizer su información, imágenes y documentos de comercialización.

Escenario de prueba:

- ✓ Dado que un Agente o Administrador modifica datos de una propiedad que le corresponde Cuando pulse el botón "Guardar Cambios" Entonces se muestra el mensaje "Propiedad actualizada correctamente" y el sistema redirige al panel de propiedades.
- ✓ Dado que un Agente intenta acceder a la página de edición de una propiedad que no le fue asignada Cuando la página cargue Entonces se muestra una pantalla con el mensaje "Acceso denegado" y el texto "No tienes permisos para editar esta propiedad", con un botón para volver.
- ✓ Dado que hay campos obligatorios vacíos o con valores inválidos Cuando pulse el botón "Guardar Cambios" Entonces se muestra la notificación "Por favor, corrige los errores en el formulario" y los campos con error se marcan con su mensaje correspondiente.
- ✓ Dado que se eliminaron todos los propietarios de la propiedad Cuando pulse el botón "Guardar Cambios" Entonces se muestra la notificación "Debes asignar al menos un propietario" y no se guarda la propiedad.
- ✓ Dado que hay propietarios asignados pero sus porcentajes de participación no suman exactamente 100% Cuando pulse el botón "Guardar Cambios" Entonces se muestra la notificación "La suma de porcentajes debe ser 100% (Actual: X%)" y no se guarda.
- ✓ Dado que hay propietarios asignados, pero ninguno está marcado como propietario principal Cuando pulse el botón "Guardar Cambios" Entonces se muestra la notificación "Debe marcar un propietario como principal" y no se guarda.
- ✓ Dado el cambio de tipo de contrato sin subir el documento correspondiente Cuando pulse el botón "Guardar Cambios" Entonces se muestra la notificación de error correspondiente indicando el documento faltante y no se guarda.
- ✓ Dado que se eliminaron todas las imágenes existentes y no se subieron imágenes nuevas Cuando pulse el botón "Guardar Cambios" Entonces se muestra el error "Debe subir al menos una imagen de la propiedad" y no se guarda.
- ✓ Dado que el usuario ha modificado algún campo del formulario sin pulsar "Guardar Cambios" Cuando intente navegar a otra sección de la plataforma o cerrar/recargar la pestaña del navegador Entonces se muestra una advertencia solicitando confirmación antes de abandonar el formulario.

Pre-condiciones

- Tener acceso a Internet.
- Contar con un Navegador Web.
- Tener una sesión activa en el sistema con rol de Agente o Administrador.



Pontificia Universidad Católica del Ecuador
Seréis mis testigos
SANTO DOMINGO
DIRECCIÓN DE INVESTIGACIÓN Y POSTGRADOS

- Que exista al menos una propiedad registrada en la plataforma y que, si el usuario es Agente, dicha propiedad esté asignada a su cuenta.

Pasos y condiciones de ejecución

Acceso a la Edición:

- Ubicarse en el panel lateral de navegación (menú izquierdo).
- Desplegar la sección "Propiedades" haciendo clic sobre ella.
- Hacer clic en la opción "Mis Propiedades" (si es Agente) o "Panel de Propiedades" (si es Administrador).
- Ubicar la tarjeta correspondiente a la propiedad que se desea modificar y hacer clic en el botón con el ícono de lápiz ("Editar Propiedad") situado en la parte inferior derecha.

Modificación de Datos Generales, Ubicación y Características:

- Desplazarse por las secciones principales del formulario ("Información de la Propiedad", "Ubicación Geográfica", "Características").
- Borrar, sobrescribir la información de los campos de texto, numéricos o realizar nuevas selecciones en las opciones desplegables o atributos (ej. Título, tipo de operación, precio, habitaciones).

Modificación de Asignación de Propietarios:

- (Opcional) Para agregar un nuevo propietario, buscar a un cliente registrado por su nombre en la sección respectiva y seleccionarlo.
- (Opcional) Para eliminar a un propietario asignado, presionar el ícono de papelera al final de la fila de dicho propietario.
- Escribir un nuevo número en el campo "% PART." de los propietarios para ajustar las participaciones.
- Marcar la casilla circular ("PRINCIPAL") correspondiente a la persona que fungirá como representante principal.

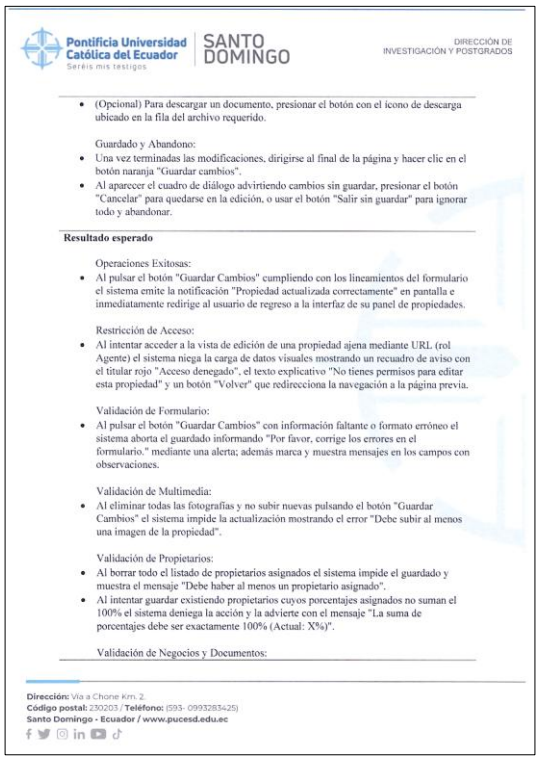
Modificación de Imágenes:

- (Opcional) Para eliminar una imagen existente de la galería, pasar el puntero sobre ella y presionar el botón de papelera rojo que aparece en el centro.
- (Opcional) Para añadir nuevas fotos, hacer clic sobre el recuadro que indica "Haga clic para seleccionar imágenes" y seleccionarlas.

Modificación de Documentación Legal y Técnica (Privado):

- (Opcional) Para eliminar un documento de un apartado, ubicar la fila correspondiente y presionar el ícono de papelera al lado del nombre del archivo actual.
- (Opcional) Para cargar un nuevo documento o reemplazar uno borrado, presionar el botón "Subir" del apartado y seleccionar el archivo.

Dirección: Vía a Chone Km. 2.
Código postal: 230203 / **Teléfono:** (593- 0993283425)
Santo Domingo - Ecuador / www.pucesd.edu.ec



Pontificia Universidad Católica del Ecuador
Seréis mis testigos
SANTO DOMINGO
DIRECCIÓN DE INVESTIGACIÓN Y POSTGRADOS

- (Opcional) Para descargar un documento, presionar el botón con el ícono de descarga ubicado en la fila del archivo requerido.

Guardado y Abandono:

- Una vez terminadas las modificaciones, dirigirse al final de la página y hacer clic en el botón naranja "Guardar cambios".
- Al aparecer el cuadro de diálogo advirtiendo cambios sin guardar, presionar el botón "Cancelar" para quedarse en la edición, o usar el botón "Salir sin guardar" para ignorar todo y abandonar.

Resultado esperado

Operaciones Exitosas:

- Al pulsar el botón "Guardar Cambios" cumpliendo con los lineamientos del formulario el sistema emite la notificación "Propiedad actualizada correctamente" en pantalla e inmediatamente redirige al usuario de regreso a la interfaz de su panel de propiedades.

Restricción de Acceso:

- Al intentar acceder a la vista de edición de una propiedad ajena mediante URL (rol Agente) el sistema niega la carga de datos visuales mostrando un recuadro de aviso con el titular rojo "Acceso denegado", el texto explicativo "No tienes permisos para editar esta propiedad" y un botón "Volver" que redirecciona la navegación a la página previa.

Validación de Formulario:

- Al pulsar el botón "Guardar Cambios" con información faltante o formato erróneo el sistema aborta el guardado informando "Por favor, corrige los errores en el formulario." mediante una alerta; además marca y muestra mensajes en los campos con observaciones.

Validación de Multimedia:

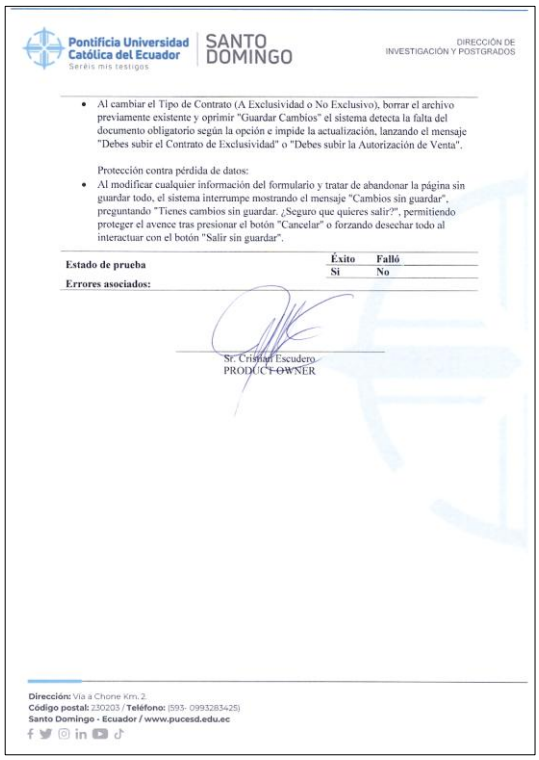
- Al eliminar todas las fotografías y no subir nuevas pulsando el botón "Guardar Cambios" el sistema impide la actualización mostrando el error "Debe subir al menos una imagen de la propiedad".

Validación de Propietarios:

- Al borrar todo el listado de propietarios asignados el sistema impide el guardado y muestra el mensaje "Debe haber al menos un propietario asignado".
- Al intentar guardar existiendo propietarios cuyos porcentajes asignados no suman el 100% el sistema deniega la acción y la advierte con el mensaje "La suma de porcentajes debe ser exactamente 100% (Actual: X%)".

Validación de Negocios y Documentos:

Dirección: Vía a Chone Km. 2.
Código postal: 230203 / **Teléfono:** (593- 0993283425)
Santo Domingo - Ecuador / www.pucesd.edu.ec



Pontificia Universidad Católica del Ecuador
Seréis mis testigos
SANTO DOMINGO
DIRECCIÓN DE INVESTIGACIÓN Y POSTGRADOS

- Al cambiar el Tipo de Contrato (A Exclusividad o No Exclusivo), borrar el archivo previamente existente y oprimir "Guardar Cambios" el sistema detecta la falta del documento obligatorio según la opción e impide la actualización, lanzando el mensaje "Debes subir el Contrato de Exclusividad" o "Debes subir la Autorización de Venta".

Protección contra pérdida de datos:

- Al modificar cualquier información del formulario y tratar de abandonar la página sin guardar todo, el sistema interrumpe mostrando el mensaje "Cambios sin guardar", preguntando "Tienes cambios sin guardar. ¿Seguro que quieres salir?", permitiendo proteger el avence tras presionar el botón "Cancelar" o forzando desechar todo al interactuar con el botón "Salir sin guardar".

Estado de prueba	Éxito	Falló
	Si	No

Errores asociados:

Sr. Cristian Escudero
PRODUCT OWNER

Dirección: Vía a Chone Km. 2.
Código postal: 230203 / **Teléfono:** (593- 0993283425)
Santo Domingo - Ecuador / www.pucesd.edu.ec

CASO DE PRUEBA 07	Fecha: 25/11/2025
Nombre caso de prueba: Actualización del Estado de Publicación de Propiedades	**Sprint:** 2
Módulo/sección a evaluar: Gestión de Propiedades	**Historia de usuario asociada:** 7
Técnica de prueba: Caja Negra ☒ Caja Blanca ☐	**Tipo:** Prueba de Aceptación

Descripción:
Como Agente, Administrador **Quiero** actualizar el estado administrativo de una propiedad (disponible o inactiva) **Para** reflejar si está activa o no.

Escenario de prueba:

- ✓ Dado que un Agente ve en el panel una propiedad que le fue asignada y no tiene restricciones por negociaciones Cuando pulse el ícono de "Cambiar Estado", seleccione "Disponible" o "Inactiva" y pulse "Guardar" Entonces el modal muestra el mensaje "Estado actualizado correctamente", se cierra automáticamente y la etiqueta de estado en la tarjeta cambia al nuevo valor.
- ✓ Dado que un Administrador ve cualquier propiedad en el panel (propia o de otro agente) y no tiene restricciones por negociaciones Cuando pulse el ícono de "Cambiar Estado", seleccione "Disponible" o "Inactiva" y pulse "Guardar" Entonces el modal muestra el mensaje "Estado actualizado correctamente", se cierra automáticamente y la etiqueta de estado en la tarjeta cambia al nuevo valor.
- ✓ Dado que un Agente está en el panel de propiedades viendo una propiedad que no le fue asignada Cuando revise los botones de acción de esa tarjeta Entonces el ícono de "Cambiar Estado" no está visible.
- ✓ Dado que un Agente propietario o Administrador abre el modal "Cambiar estado de la propiedad" Cuando se despliegue el selector de estados Entonces únicamente aparecen las opciones "Disponible" e "Inactiva", y el modal muestra un aviso informativo que indica que para marcar la propiedad como Vendida, Reservada o Arrendada es necesario finalizar una Negociación.
- ✓ Dado que una propiedad tiene una negociación activa en etapa "Cierre" Cuando el Agente propietario o el Administrador seleccione "Inactiva" y pulse "Guardar" Entonces el modal muestra el mensaje de error: " No se puede desactivar la propiedad. Hay una negociación en etapa de CIERRE (Reserva activa)" y el estado no cambia.
- ✓ Dado que una propiedad tiene negociaciones en etapa "Interés", "Negociación", "Cancelada" o "Finalizada" (pero ninguna activa en "Cierre") Cuando el Agente propietario o el Administrador seleccione "Inactiva" y pulse "Guardar" Entonces el modal muestra el mensaje "Estado actualizado correctamente", se cierra automáticamente y la etiqueta de la tarjeta refleja "inactiva".
- ✓ Dado que una propiedad tiene estado "Reservada", "Vendida" o "Arrendada" y hay una negociación activa en etapa "Cierre" o "Finalizada" Cuando el Agente propietario o el Administrador seleccione "Disponible" y pulse "Guardar" Entonces el modal muestra el mensaje "No se puede marcar como disponible. Hay una negociación en Cierre/Finalizada que bloquea la propiedad" y el estado no cambia.

Pontificia Universidad Católica del Ecuador | SANTO DOMINGO
Seréis mis testigos
DIRECCIÓN DE INVESTIGACIÓN Y POSTGRADOS

- ✓ Dado que una propiedad tiene estado "Reservada", "Vendida" o "Arrendada" pero ya no tiene negociaciones activas en etapa Cierre o Finalizada Cuando el Agente propietario o el Administrador seleccione "Disponible" y pulse "Guardar" Entonces el modal muestra el mensaje "Estado actualizado correctamente" y la etiqueta de la tarjeta refleja "disponible".

Pre-condiciones

- Tener acceso a Internet.
- Contar con un Navegador Web.
- Tener una sesión activa en el sistema con rol de Agente o Administrador.
- Que exista al menos una propiedad registrada en la plataforma.

Pasos y condiciones de ejecución

Verificación de Acceso al Control:

- Buscar una propiedad específica en el listado del panel de propiedades haciendo uso de la navegación visual o la barra de búsqueda/filtros.
- Revisar en la parte inferior de la tarjeta de dicha propiedad los íconos de acción.
- Hacer clic en el botón con el ícono de dos flechas circulares ("Cambiar Estado"). (Nota: Este paso solo se puede efectuar si el botón está visible, condición que depende de los permisos del usuario autenticado).

Interacción con la Ventana de Estado:

- Hacer clic sobre la lista desplegable central del formulario y visualizar las opciones permitidas.
- Seleccionar una de las opciones mostradas.
- Hacer clic en el botón "Guardar" ubicado en la esquina inferior derecha de la ventana emergente.
- (Opcional) Para deshacer la acción, seleccionar el botón contiguo "Cancelar".

Resultado esperado

Visibilidad y Exposición de Opciones:

- Al visualizar una tarjeta de propiedad asignada a otro agente (rol Agente): El sistema oculta el botón "Cambiar Estado" para evitar modificaciones ajenas.
- Al abrir la lista desplegable de cambio de estado (Administrador o Agente captador de la propiedad): El selector muestra las opciones "Disponible" e "Inactiva.

Actualizaciones Exitosas:

- Al pulsar "Guardar" seleccionando "Disponible" o "Inactiva" (Agente Captador de la propiedad) y no hay restricciones por negociaciones: Aparece el mensaje "Estado actualizado correctamente", la ventana se cierra automáticamente y la etiqueta de la propiedad se actualiza al nuevo estado.
- Al pulsar "Guardar" seleccionando "Disponible" o "Inactiva" (Administrador) y no hay restricciones por negociaciones: El sistema responde del mismo modo, confirmando

Dirección: Vía a Chone Km. 2.
Código postal: 230203 / **Teléfono:** (593- 099328342S)
Santo Domingo - Ecuador / www.pucesd.edu.ec

Pontificia Universidad Católica del Ecuador | SANTO DOMINGO
Seréis mis testigos
DIRECCIÓN DE INVESTIGACIÓN Y POSTGRADOS

con el mensaje "Estado actualizado correctamente", cerrando la ventana y actualizando visualmente la etiqueta.

Restricciones por Negociaciones:

- Al intentar marcar como "Inactiva" una propiedad en negociación con etapa "Cierre": El sistema bloquea el cambio y se alerta con el mensaje: "No se puede desactivar la propiedad. Hay una negociación en etapa de CIERRE (Reserva activa)", manteniendo el estado actual.
- Al marcar como "Inactiva" una propiedad con negociaciones en etapa de "Interés", "Negociación", "Cancelada" o "Finalizada": El sistema permite el guardado, confirma con "Estado actualizado correctamente", cierra la ventana y actualiza la etiqueta a "inactiva".
- Al intentar cambiar a "Disponible" una propiedad "Reservada", "Vendida" o "Arrendada", pero con una negociación en "Cierre" o "Finalizada": El sistema impide la acción y advierte: "No se puede marcar como disponible. Hay una negociación en Cierre/Finalizada que bloquea la propiedad".
- Al cambiar a "Disponible" una propiedad reservada/vendida/arrendada que ya no tiene negociaciones en etapa "Cierre" o "Finalizada": El sistema permite el guardado, confirma con "Estado actualizado correctamente" y actualiza la etiqueta a "disponible".

Estado de prueba	Éxito	Falló
	Si	No
Errores asociados:		

Sr. Cristian Escudero
PRODUCT OWNER

Dirección: Vía a Chone Km. 2.
Código postal: 230203 / **Teléfono:** (593- 099328342S)
Santo Domingo - Ecuador / www.pucesd.edu.ec

Pontificia Universidad Católica del Ecuador | SANTO DOMINGO
Seréis mis testigos
DIRECCIÓN DE INVESTIGACIÓN Y POSTGRADOS

CASO DE PRUEBA 08	Fecha: 25/11/2025
Nombre caso de prueba: Desactivación de propiedades	**Sprint:** 2
Módulo/sección a evaluar: Gestión de propiedades	**Historia de usuario asociada:** 8
Técnica de prueba: Caja Negra ☒ Caja Blanca ☐	**Tipo:** Prueba de Aceptación

Descripción:
Como Agente, Administrador **Quiero** desactivar una propiedad registrada **Para** retirarla de los listados públicos y mantener las propiedades organizadas.

Escenario de prueba:

- ✓ Dado que un Agente propietario o Administrador ve una tarjeta de propiedad en el panel de propiedades Cuando pulse el ícono de papelera y confirme pulsando el botón "Desactivar" en el modal de confirmación Entonces el modal muestra el mensaje "Propiedad desactivada correctamente", la propiedad cambia a estado "inactiva" y deja de aparecer en el portal público; en el panel interno (admin/agente) sigue visible con la etiqueta "inactiva".
- ✓ Dado que un Agente visualiza en el panel una propiedad que no le fue asignada Cuando revise los botones de acción de esa tarjeta Entonces el ícono de papelera no aparece.
- ✓ Dado que un Agente visualiza en el panel una propiedad que le fue asignada Cuando revise los botones de acción de esa tarjeta Entonces el ícono de papelera es visible junto a los demás botones de gestión.
- ✓ Dado que un Agente propietario o Administrador abre el modal de confirmación de desactivación Cuando pulse el botón "Cancelar" Entonces el modal se cierra y la propiedad permanece en su estado actual sin cambios.
- ✓ Dado que la propiedad tiene una negociación activa en etapa "Cierre" Cuando el Agente propietario o el Administrador pulse el botón "Desactivar" en el modal de confirmación Entonces el modal muestra el mensaje: "No se puede desactivar la propiedad. Hay una negociación en etapa de CIERRE (Reserva activa)" y el estado de la propiedad no cambia.
- ✓ Dado que la propiedad tiene negociaciones únicamente en etapas "Interés", "Negociación", "Cancelada" o "Finalizada" (pero ninguna en "Cierre") Cuando el Agente propietario o el Administrador pulse el botón "Desactivar" en el modal de confirmación Entonces la desactivación se realiza correctamente y la propiedad pasa a estado "inactiva".

Pre-condiciones

- Tener acceso a Internet.
- Contar con un Navegador Web.
- Tener una sesión activa en el sistema con rol de Agente o Administrador.
- Que exista al menos una propiedad registrada en la plataforma.

Pasos y condiciones de ejecución

Acceso al botón para desactivar una propiedad:

- Buscar una propiedad específica en el panel de propiedades.

Dirección: Vía a Chone Km. 2.
Código postal: 230203 / **Teléfono:** (593- 099328342S)
Santo Domingo - Ecuador / www.pucesd.edu.ec

Pontificia Universidad Católica del Ecuador | SANTO DOMINGO
Seréis mis testigos
DIRECCIÓN DE INVESTIGACIÓN Y POSTGRADOS

- Hacer clic en el botón con el ícono de papelera ("Desactivar Propiedad"). (Nota: Este botón es visible únicamente para el perfil Administrador o el Agente captador de la propiedad).

Interacción con la Ventana de Confirmación:
- Para confirmar el proceso en la ventana emergente, hacer clic en el botón rojo "Desactivar" situado en la esquina inferior derecha.
- (Opcional) Para anular el proceso, pulsar el botón gris "Cancelar" adyacente al botón de confirmación.

Resultado esperado

Visibilidad de Opciones de Gestión:
- Al visualizar una tarjeta de propiedad asignada a otro agente (siendo un Agente diferente al captador): El sistema oculta el botón con ícono de papelera.
- Al inspeccionar su propia propiedad o cualquier propiedad (siendo Agente captador o Administrador): El botón con el ícono de papelera se muestra habilitado.

Proceso de Desactivación y Efectos Visibles:
- Al presionar el botón "Desactivar" en el modal de confirmación (sobre una propiedad sin negociaciones en Cierre): El modal muestra el mensaje "Propiedad desactivada correctamente" y se cierra automáticamente. La propiedad cambia a estado "inactiva", se actualiza su etiqueta visual y deja de aparecer en los listados públicos de la plataforma.
- Al presionar el botón "Cancelar" en el modal de confirmación: El modal se cierra, el proceso se cancela y el estado de la propiedad no cambia.

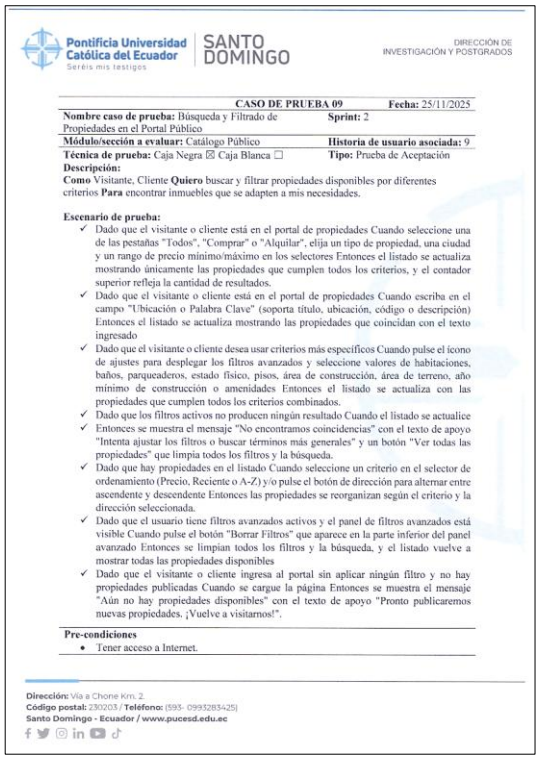
Restricciones por Negociaciones:
- Al intentar desactivar desde la ventana de confirmación una propiedad que mantiene una negociación con la etapa de "Cierre": El sistema impide la acción y emite la alerta: "No se puede desactivar la propiedad. Hay una negociación en etapa de CIERRE (Reserva activa)", reservando el estado original.
- Al intentar desactivar una propiedad con negociaciones en etapa "Interés", "Negociación", "Cancelada" o "Finalizada" (ninguna en "Cierre"): El sistema permite la acción, el modal muestra el mensaje "Propiedad desactivada correctamente" y la etiqueta de la propiedad se actualiza a "inactiva".

Estado de prueba	Éxito	Falló
	Si	No

Errores asociados:

Sr. Cristian Escudero
PRODUCT OWNER

Dirección: Vía a Chone Km. 2.
Código postal: 230203 / Teléfono: (593- 0993283425)
Santo Domingo - Ecuador / www.pucesd.edu.ec



Pontificia Universidad Católica del Ecuador | SANTO DOMINGO
Seréis mis testigos
DIRECCIÓN DE INVESTIGACIÓN Y POSTGRADOS

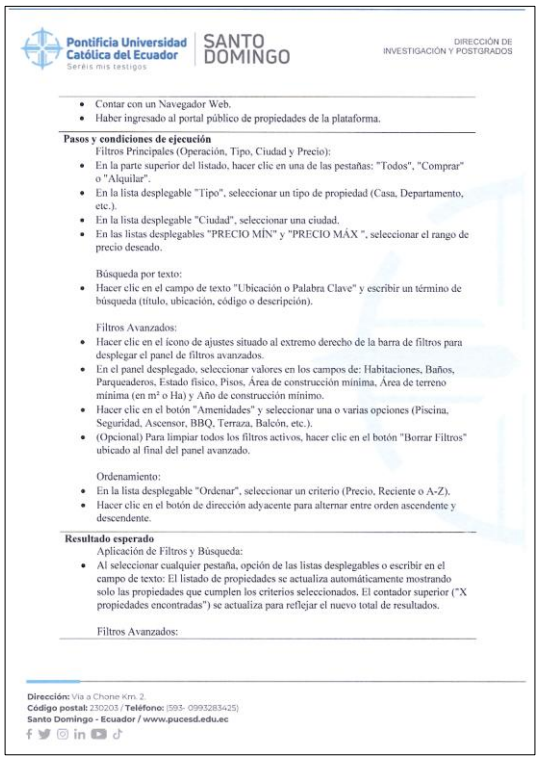
CASO DE PRUEBA 09 **Fecha:** 25/11/2025

Nombre caso de prueba: Búsqueda y Filtrado de Propiedades en el Portal Público | **Sprint:** 2
Módulo/sección a evaluar: Catálogo Público | **Historia de usuario asociada:** 9
Técnica de prueba: Caja Negra ☒ Caja Blanca ☐ | **Tipo:** Prueba de Aceptación
Descripción:
Como Visitante, Cliente **Quiero** buscar y filtrar propiedades disponibles por diferentes criterios **Para** encontrar inmuebles que se adapten a mis necesidades.

Escenario de prueba:
- ✓ Dado que el visitante o cliente está en el portal de propiedades Cuando seleccione una de las pestañas "Todos", "Comprar" o "Alquilar", elija un tipo de propiedad, una ciudad y un rango de precio mínimo/máximo en los selectores Entonces el listado se actualiza mostrando únicamente las propiedades que cumplen todos los criterios, y el contador superior refleja la cantidad de resultados.
- ✓ Dado que el visitante o cliente está en el portal de propiedades Cuando escriba en el campo "Ubicación o Palabra Clave" (soporta título, ubicación, código o descripción) Entonces el listado se actualiza mostrando las propiedades que coincidan con el texto ingresado
- ✓ Dado que el visitante o cliente desea usar criterios más específicos Cuando pulse el ícono de ajustes para desplegar los filtros avanzados y seleccione valores de habitaciones, baños, parqueaderos, estado físico, pisos, área de construcción, área de terreno, año mínimo de construcción o amenidades Entonces el listado se actualiza con las propiedades que cumplen todos los criterios combinados.
- ✓ Dado que los filtros activos no producen ningún resultado Cuando el listado se actualice
- ✓ Entonces se muestra el mensaje "No encontramos coincidencias" con el texto de apoyo "Intenta ajustar los filtros o buscar términos más generales" y un botón "Ver todas las propiedades" que limpia todos los filtros y la búsqueda.
- ✓ Dado que hay propiedades en el listado Cuando seleccione un criterio en el selector de ordenamiento (Precio, Reciente o A-Z) y/o pulse el botón de dirección para alternar entre ascendente y descendente Entonces las propiedades se reorganizan según el criterio y la dirección seleccionada.
- ✓ Dado que el usuario tiene filtros avanzados activos y el panel de filtros avanzados está visible Cuando pulse el botón "Borrar Filtros" que aparece en la parte inferior del panel avanzado Entonces se limpian todos los filtros y la búsqueda, y el listado vuelve a mostrar todas las propiedades disponibles
- ✓ Dado que el visitante o cliente ingresa al portal sin aplicar ningún filtro y no hay propiedades publicadas Cuando se cargue la página Entonces se muestra el mensaje "Aún no hay propiedades disponibles" con el texto de apoyo "Pronto publicaremos nuevas propiedades. ¡Vuelve a visitarnos!".

Pre-condiciones
- Tener acceso a Internet.

Dirección: Vía a Chone Km. 2.
Código postal: 230203 / Teléfono: (593- 0993283425)
Santo Domingo - Ecuador / www.pucesd.edu.ec



Pontificia Universidad Católica del Ecuador | SANTO DOMINGO
Seréis mis testigos
DIRECCIÓN DE INVESTIGACIÓN Y POSTGRADOS

- Contar con un Navegador Web.
- Haber ingresado al portal público de propiedades de la plataforma.

Pasos y condiciones de ejecución

Filtros Principales (Operación, Tipo, Ciudad y Precio):
- En la parte superior del listado, hacer clic en una de las pestañas: "Todos", "Comprar" o "Alquilar".
- En la lista desplegable "Tipo", seleccionar un tipo de propiedad (Casa, Departamento, etc.).
- En la lista desplegable "Ciudad", seleccionar una ciudad.
- En las listas desplegables "PRECIO MÍN" y "PRECIO MÁX ", seleccionar el rango de precio deseado.

Búsqueda por texto:
- Hacer clic en el campo de texto "Ubicación o Palabra Clave" y escribir un término de búsqueda (título, ubicación, código o descripción).

Filtros Avanzados:
- Hacer clic en el ícono de ajustes situado al extremo derecho de la barra de filtros para desplegar el panel de filtros avanzados.
- En el panel desplegado, seleccionar valores en los campos de: Habitaciones, Baños, Parqueaderos, Estado físico, Pisos, Área de construcción mínima, Área de terreno mínima (en m² o Ha) y Año de construcción mínimo.
- Hacer clic en el botón "Amenidades" y seleccionar una o varias opciones (Piscina, Seguridad, Ascensor, BBQ, Terraza, Balcón, etc.).
- (Opcional) Para limpiar todos los filtros activos, hacer clic en el botón "Borrar Filtros" ubicado al final del panel avanzado.

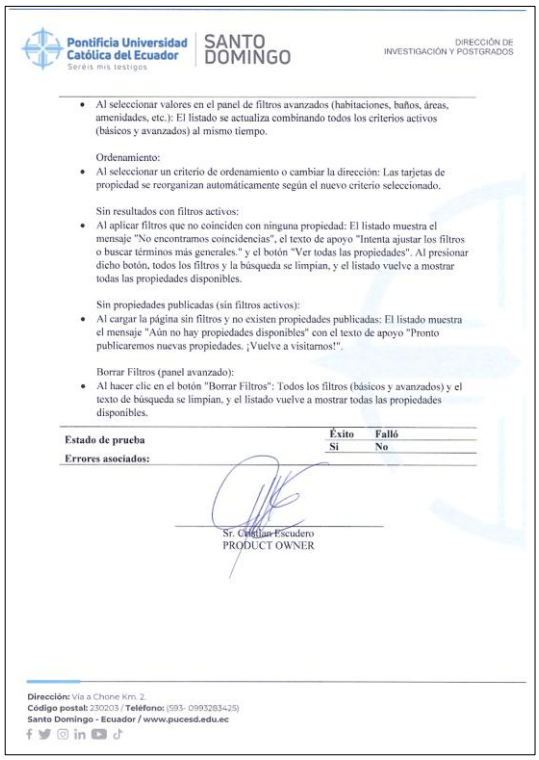
Ordenamiento:
- En la lista desplegable "Ordenar", seleccionar un criterio (Precio, Reciente o A-Z).
- Hacer clic en el botón de dirección adyacente para alternar entre orden ascendente y descendente.

Resultado esperado

Aplicación de Filtros y Búsqueda:
- Al seleccionar cualquier pestaña, opción de las listas desplegables o escribir en el campo de texto: El listado de propiedades se actualiza automáticamente mostrando solo las propiedades que cumplen los criterios seleccionados. El contador superior ("X propiedades encontradas") se actualiza para reflejar el nuevo total de resultados.

Filtros Avanzados:

Dirección: Vía a Chone Km. 2.
Código postal: 230203 / Teléfono: (593- 0993283425)
Santo Domingo - Ecuador / www.pucesd.edu.ec



Pontificia Universidad Católica del Ecuador | SANTO DOMINGO
Seréis mis testigos
DIRECCIÓN DE INVESTIGACIÓN Y POSTGRADOS

- Al seleccionar valores en el panel de filtros avanzados (habitaciones, baños, áreas, amenidades, etc.): El listado se actualiza combinando todos los criterios activos (básicos y avanzados) al mismo tiempo.

Ordenamiento:
- Al seleccionar un criterio de ordenamiento o cambiar la dirección: Las tarjetas de propiedad se reorganizan automáticamente según el nuevo criterio seleccionado.

Sin resultados con filtros activos:
- Al aplicar filtros que no coinciden con ninguna propiedad: El listado muestra el mensaje "No encontramos coincidencias", el texto de apoyo "Intenta ajustar los filtros o buscar términos más generales." y el botón "Ver todas las propiedades". Al presionar dicho botón, todos los filtros y la búsqueda se limpian, y el listado vuelve a mostrar todas las propiedades disponibles.

Sin propiedades publicadas (sin filtros activos):
- Al cargar la página sin filtros y no existen propiedades publicadas: El listado muestra el mensaje "Aún no hay propiedades disponibles" con el texto de apoyo "Pronto publicaremos nuevas propiedades. ¡Vuelve a visitarnos!".

Borrar Filtros (panel avanzado):
- Al hacer clic en el botón "Borrar Filtros": Todos los filtros (básicos y avanzados) y el texto de búsqueda se limpian, y el listado vuelve a mostrar todas las propiedades disponibles.

Estado de prueba	Éxito	Falló
	Si	No

Errores asociados:

Sr. Cristian Escudero
PRODUCT OWNER

Dirección: Vía a Chone Km. 2.
Código postal: 230203 / Teléfono: (593- 0993283425)
Santo Domingo - Ecuador / www.pucesd.edu.ec

Casos de prueba completo: Pruebas de Aceptación

Anexo VI: Evidencia de la reunión de planificación del sprint con el gerente

Evidencia reunión planificación sprint

Anexo VII: Manual de usuario

25 DE NOVIEMBRE DE 2025

MANUAL DE USUARIO

IEEE 1063-2001

JHERY FERNANDO ALARCÓN MORILLO – ROMARIO ARMANDO FLORES VELEZ
PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR
Santo Domingo

REGISTRO DE CAMBIOS

FECHA	USUARIO	VERSIÓN	ACCIONES
20-NOV-25	JHERY	1.0	CREACIÓN
20-NOV-25	ROMARIO	1.0	EDICIÓN

Tabla de Contenidos

1 INTRODUCCIÓN

En el dinámico y competitivo mercado inmobiliario actual, la agilidad en la gestión de propiedades y clientes es un factor determinante para el éxito de cualquier agencia. Es por esta razón que la implementación de soluciones tecnológicas juega un papel crucial, permitiendo centralizar la información, optimizar los tiempos de respuesta y mantener un control efectivo sobre las operaciones de venta y alquiler. En este contexto, contar con una plataforma web diseñada específicamente para potenciar la productividad de los agentes y mejorar la experiencia del cliente se convierte en una ventaja estratégica indispensable.

El presente manual detalla las características, funcionalidades y especificaciones para el uso de la aplicación web "Gestión Inmobiliaria", desarrollada con el propósito de modernizar y facilitar los procesos comerciales de la agencia. Desde la administración de inventario de propiedades hasta el seguimiento de negociaciones y clientes, esta herramienta busca integrar todas las áreas operativas en un entorno digital eficiente.

Este documento contiene la información técnica y operativa necesaria para garantizar el aprovechamiento máximo de la plataforma, teniendo como objetivo principal ofrecer al usuario una guía clara y estructurada que asegure el uso correcto y eficiente de la aplicación web en sus labores diarias.

2 CONCEPTO DE LAS OPERACIONES

Para garantizar el correcto funcionamiento y la mejor experiencia de usuario en la aplicación web PropTech Hub, el dispositivo de acceso debe cumplir con los siguientes requerimientos mínimos:

Hardware:

- Dispositivo: Computadora de escritorio, portátil (Laptop).
- Procesador: Dual Core 1.6 GHz o superior.
- Memoria RAM: 1 GB o superior (recomendado para una navegación fluida).
- Pantalla: Resolución mínima de 1366x768 píxeles para escritorio; adaptable a pantallas móviles.

Software:

- Sistema Operativo:
  - Windows 10 o superior.
  - macOS 10.15 (Catalina) o superior.

- Navegador Web (Actualizado):
  - Google Chrome (Recomendado).
  - Mozilla Firefox.
  - Microsoft Edge.
  - Safari.
- Conectividad:
- Internet: Conexión de banda ancha estable (Fibra óptica, ADSL) o red móvil 4G/LTE/5G.
- Velocidad: Mínimo 5 Mbps de bajada.

3 PROCEDIMIENTOS

3.1 Ejecución

- Para proceder a iniciar la aplicación, ejecute los servicios *frontend* y *backend* del directorio del proyecto.

3.2 Uso de la aplicación

3.2.1 Inicio de Sesión

- Diríjase a la página de inicio de sesión, para ello de clic en el botón "Iniciar Sesión" ubicado en la parte derecha de la barra de navegación.

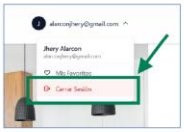
- Ingrese las credenciales (correo electrónico y contraseña) de su cuenta y pulse el botón "Iniciar Sesión".

3.2.2 Cierre de sesión

3.2.2.1 *Para clientes*

- Haga clic sobre el correo electrónico o el ícono circular color negro ubicado en la parte derecha de la barra de navegación.

- De clic en la opción "Cerrar Sesión".



3.2.2.2 *Para administradores y agentes inmobiliarios*

- En la parte inferior de la barra lateral izquierda de la interfaz presione el botón situado a la derecha del nombre y correo electrónico del usuario autenticado.

3.2.3 Recuperación de cuenta

- Si olvidó la contraseña de su cuenta, puede reestablecerla. Para ello, haga clic en la opción "Iniciar Sesión" de la parte derecha de la barra de navegación.

- En el formulario de inicio de sesión haga clic en el texto color naranja "¿Olvidaste tu contraseña?".

- Ingrese el correo electrónico del cual desea reestablecer su contraseña y de clic en el botón "Enviar Instrucciones".

- Ingrese a la bandeja de entrada del correo electrónico, busque aquel cuyo asunto sea "Restablecimiento de contraseña - PropTech Hub" y en el cuerpo del mensaje de clic en el botón "Reestablecer Contraseña".

- Se abrirá un formulario para ingresar la nueva contraseña, así como su confirmación, respetando la política de seguridad implementada (mínimo 8 caracteres, al menos un número y una mayúscula). Cuando lo haga, haga clic en la opción "Guardar Nueva Contraseña".

- Se mostrará la pantalla "¡Contraseña Guardada!" con una notificación de éxito, y tras pocos segundos le redirigirá a la página de inicio de sesión.

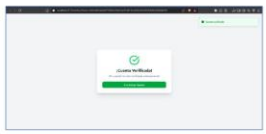
3.2.4 Registro de Cuenta

- Dirijase a la página de registro de cuenta, para ello de clic en el botón "Registrarse" ubicado en la parte derecha de la barra de navegación.

- En el formulario presente, ingrese el nombre completo, un correo electrónico no utilizado en el sistema y una contraseña con su confirmación en los campos correspondientes. Adicionalmente, puede ingresar de manera opcional un teléfono. Con los datos ingresados correctamente presione el botón "Registrarse".

- Ingrese a la bandeja de entrada del correo electrónico, busque aquel cuyo asunto sea "Verificación de cuenta - PropTech Hub" y en el cuerpo del mensaje de clic en el botón "Verificar Cuenta".

- Se mostrará la pantalla "¡Cuenta Verificada!" con una notificación de éxito, y tras pocos segundos le redirigirá a la página de inicio de sesión.



- En la página de inicio de sesión, ingresa las credenciales (correo y contraseña) que utilizó para registrarse y pulse el botón "Iniciar Sesión".

3.2.5 Funcionalidades para visitantes y clientes

3.2.5.1 *Visualizar propiedades disponibles en el portal público*

- Para ver las propiedades disponibles en la plataforma, de clic en la opción "Propiedades" de la barra de navegación.

- Para hacer una búsqueda más precisa, utilice los filtros básicos como toggle de tipo de operación (compra o alquiler), barra de búsqueda (titular, ubicación, código interno del inmueble, descripción), selectores de tipo de propiedad, ciudad, rango de precios, ordenamiento de inmuebles por antigüedad, precio o titular.

- Adicionalmente, puede utilizar más filtros, para ello, haga clic en el ícono de mezclador de audio (señalado en la imagen) y seleccione o ingrese los criterios adicionales que considere necesario para hacer una búsqueda más minuciosa.

- Para reestablecer los filtros aplicados y mostrar todas las propiedades disponibles, de clic en el botón "Borrar Filtros".

3.2.5.2 *Visualizar información de una propiedad en el portal público*

- Para ver la información completa de una propiedad en el portal público, localice la tarjeta de la propiedad que le interese y haga clic en su miniatura.

- En la parte superior de la página de información de dicho inmueble se mostrará el titular, precio y opciones para compartir por Facebook, correo electrónico o simplemente copiar su enlace web.

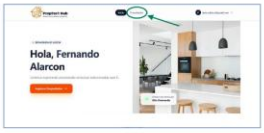
- Para visualizar a pantalla completa las diferentes imágenes de una propiedad, de clic en cualquiera de las imágenes.

- Para navegar entre las diferentes fotos, de clic en las flechas de navegación (izquierda y derecha).

- En la parte inferior de la ficha de información del inmueble verá su descripción, amenidades y un formulario de información en el cual puede enviar sus datos para que un agente le contacte y le brinde más información o agende una visita. Para esto, llene la información requerida y pulse el botón "Enviar Mensaje".

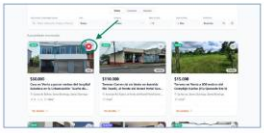
3.2.5.3 *Guardar una propiedad como favorita*

- Si desea guardar una propiedad como favorita, debe iniciar sesión con una cuenta de cliente. Para buscar propiedades que le pueden gustar, puede navegar en la página de "Inicio" o de "Propiedades".



- Para guardar una propiedad como favorita, pulse el ícono con forma de corazón ubicado en la esquina superior derecha de su tarjeta correspondiente. Dicho ícono tornará su relleno de color rojo.

- Si desea retirarla de su listado de favoritos, pulse nuevamente en el ícono de corazón. Se percatará que el corazón ya no tendrá relleno alguno.



Manual de Usuario: [Manual de Usuario Completo]

Anexo VIII: Manual técnico

27 DE DICIEMBRE DE 2025

MANUAL TÉCNICO

IEEE 1063-2001

JHERY FERNANDO ALARCÓN MORILLO – ROMARIO ARMANDO FLORES VÉLEZ
PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR
Santo Domingo

REGISTRO DE CAMBIOS

FECHA	USUARIO	VERSIÓN	ACCIONES
27-DIC-25	JHERY	1.0	CREACIÓN

Tabla de Contenidos

1 OBJETIVOS

1.1 General

Proporcionar una guía técnica que describa la estructura, configuración y funcionamiento de la aplicación web de gestión inmobiliaria basada en técnicas de machine learning, con el fin de facilitar su instalación, comprensión, mantenimiento y futura evolución por parte de desarrolladores o administradores del sistema.

1.2 Específico

- Describir la estructura del proyecto, incluyendo la organización de los componentes del frontend, backend y base de datos, con el propósito de comprender la arquitectura del sistema.
- Explicar el proceso de instalación y configuración del entorno de desarrollo, detallando los pasos necesarios para clonar el repositorio, instalar dependencias y configurar las variables de entorno requeridas para el funcionamiento de la aplicación.
- Documentar el proceso de ejecución y despliegue del sistema, incluyendo la conexión con la base de datos, la integración entre los diferentes componentes y la implementación de la aplicación en la infraestructura en la nube.

2 INTRODUCCIÓN

Este manual técnico presenta una descripción detallada de la estructura y funcionamiento de la aplicación web desarrollada para la gestión inmobiliaria con integración de técnicas de *machine learning*. En el documento se abordan los principales componentes tecnológicos utilizados en el desarrollo del sistema, así como la organización del proyecto dentro de su entorno de implementación.

3 PROCEDIMIENTOS

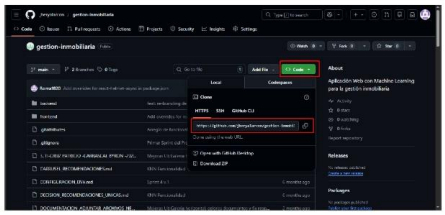
3.1 Clonar repositorios de GitHub

3.1.1 Repositorio backend

Para acceder al código fuente del sistema se utiliza el repositorio oficial del proyecto alojado en la plataforma GitHub. Este repositorio contiene todos los archivos necesarios para el funcionamiento de la aplicación, incluyendo el backend, el frontend y la configuración general del sistema.

El repositorio del proyecto se encuentra disponible en el siguiente enlace:

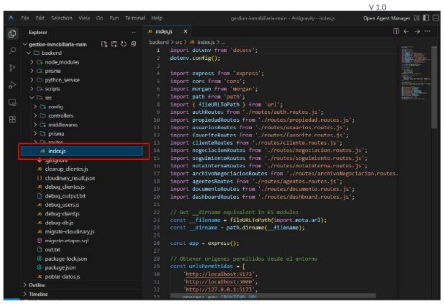
https://github.com/jheryalarcon/gestion-inmobiliaria



3.2 Archivos del backend

3.2.1 Archivo Principal

El archivo principal del backend corresponde a **src/index.js**, el cual funciona como el punto de entrada de la aplicación. Este archivo es responsable de iniciar el servidor, cargar las configuraciones necesarias y ejecutar los servicios del sistema.

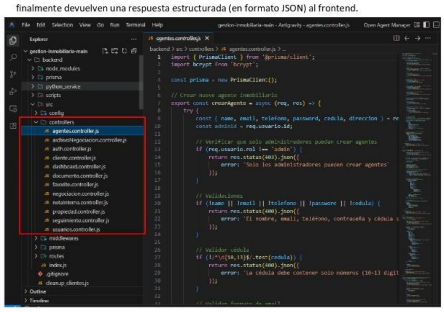


3.2.2 Carperta Config

Contiene la configuración de servicios externos, integraciones y utilidades compartidas necesarias para el funcionamiento global de la aplicación. Aquí se encuentra la configuración de almacenamiento de imágenes en la nube (Cloudinary), el envío de correos electrónicos (Nodemailer/Resend) y el uso de multer para procesar la subida de archivos (imágenes/documentos).

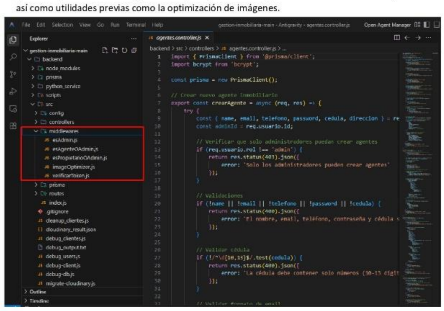
3.2.3 Carpeta controllers

Almacena la lógica de negocio de la aplicación. Los controladores son las funciones que reciben las peticiones HTTP desde las rutas, se comunican con la base de datos (vía Prisma) para leer o modificar datos (propiedades, clientes, negociaciones, agentes, etc.), y finalmente devuelven una respuesta estructurada (en formato JSON) al frontend.



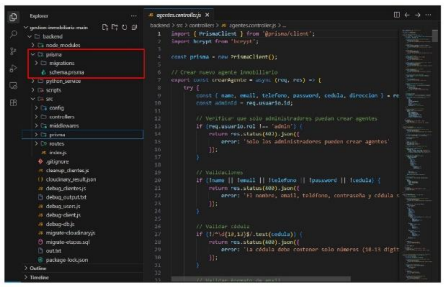
3.2.4 Carpeta middlewares

Contiene funciones intermedias que se ejecutan antes de que una petición llegue a su respectivo controlador. Su propósito principal es la validación y seguridad; por ejemplo, verificar que un usuario esté autenticado (verificarToken) o comprobar que tenga los permisos y roles correctos (ej. esAdmin, esAgenteOAdmin) para realizar ciertas acciones, así como utilidades previas como la optimización de imágenes.



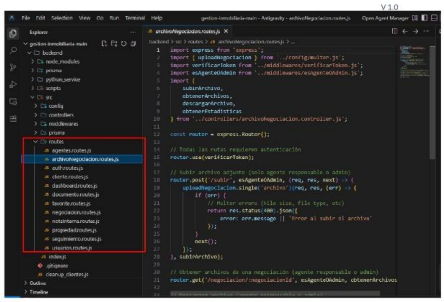
3.2.5 Carpeta prisma

Es el núcleo de la base de datos (el ORM). Contiene el archivo clave schema.prisma donde se modelan todas las tablas, relaciones y campos de la base de datos (PostgreSQL). También aloja las migraciones, que son los historiales de los cambios en la estructura de la base de datos a lo largo del tiempo.



3.2.6 Carpeta routes

Define los puntos de acceso (Endpoints) o URLs de la API REST (ej. /api/auth, /api/propiedades). Aquí se especifica qué método HTTP (GET, POST, PUT, DELETE) y ruta específica se enlaza con qué controller, además de aplicarle los middlewares necesarios de autenticación o verificación de rol en cada caso.



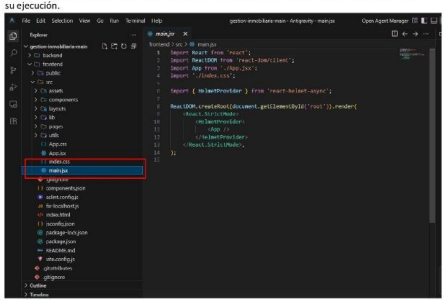
3.3 Archivos del Frontend

3.3.1 Carpeta Principal

El archivo principal del frontend corresponde a src/main.jsx, el cual actúa como el punto de inicio de la aplicación desarrollada con React. Este archivo se encarga de iniciar la interfaz gráfica del sistema y de renderizar los componentes principales dentro del navegador.

Una de sus funciones principales es establecer la conexión entre la aplicación de React y el archivo index.html, que contiene la estructura básica de la página web. En este archivo HTML se encuentra el contenedor principal , dentro del cual se carga toda la aplicación. Para realizar esta operación se utiliza la función ReactDOM.createRoot(), la cual permite renderizar la aplicación en dicho contenedor.

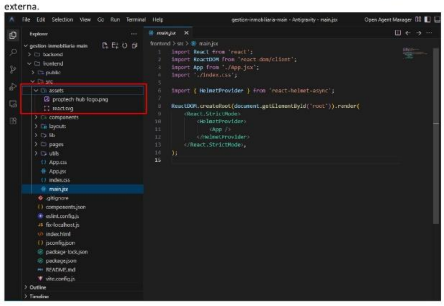
Además, dentro de main.jsx se realiza la importación de los estilos globales del sistema mediante el archivo index.css, asegurando que toda la aplicación cargue los estilos generales desde el inicio de su ejecución.



3.3.2 Carpeta assets

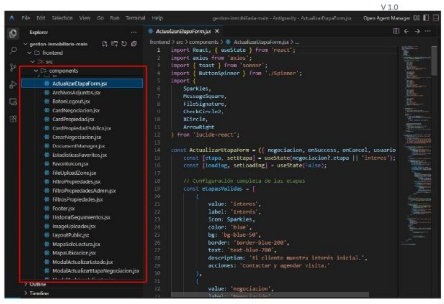
Almacena los recursos estáticos multimedia que se utilizan directamente en la aplicación y que son procesados durante el build (empaquetado). Generalmente contiene imágenes locales, iconos, logotipos (ej. proptech-hub-logo.png) y otros archivos estáticos que no provienen de una base de datos

externa.



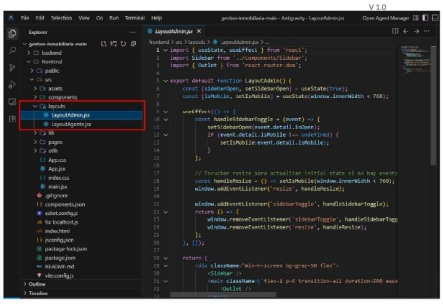
3.3.3 Carpeta components

Contiene los componentes reutilizables de la interfaz de usuario. Estos son los "bloques de construcción" de la aplicación (como botones, tarjetas de propiedades, menús de navegación, modales, formularios, etc.) que se pueden importar y usar múltiples veces en diferentes páginas sin tener que reescribir código. Es la carpeta más grande y modular del frontend.



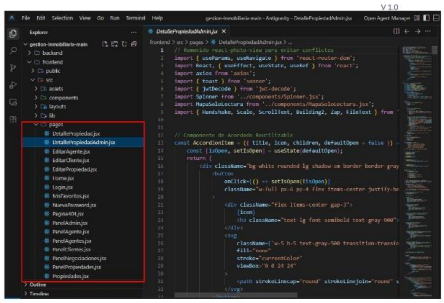
3.3.4 Carpeta layouts

Define las estructuras maestras o "plantillas" base de la interfaz. En lugar de repetir elementos comunes (como el Navbar superior o el Sidebar lateral) en cada página, los layouts (ej. LayoutAdmin, LayoutAgente) envuelven a las páginas específicas, garantizando que el diseño y la navegación sean consistentes según el tipo de usuario o sección de la web.



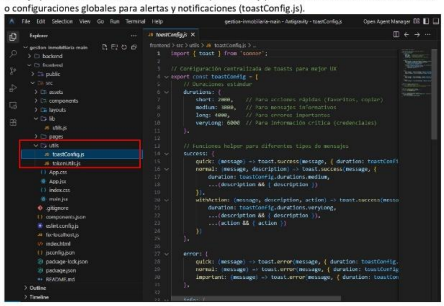
3.3.5 Carperta pages

Representa las vistas completas o "pantallas" principales a las que un usuario puede navegar a través del enrutador (React Router). Cada archivo aquí suele corresponder a una URL específica de tu aplicación (ej. /login mapea a Login.jsx, /panel-admin mapea a PanelAdmin.jsx). Estas páginas se componen ensamblando los diferentes components e integrando los layouts.



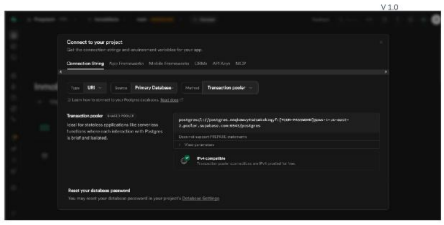
3.3.6 Carpeta utils

Almacena funciones auxiliares puras y configuraciones transversales que se repiten con frecuencia en el código. Aquí se encuentran utilidades de lógica pura que no devuelven interfaz gráfica, como el manejo y descifrado de tokens de autenticación (tokenUtils.js), formateadores de fechas, validadores, o configuraciones globales para alertas y notificaciones (toastConfig.js).

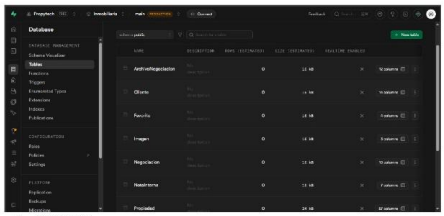


4 ACCESO A LOS DATOS

La base de datos del sistema fue implementada en la nube mediante el servicio BaaS (Backend as a Service) proporcionado por Supabase. Para establecer la conexión con la base de datos se utilizan las credenciales de acceso y la cadena de conexión que la plataforma ofrece dentro de su sección de configuración.



Para acceder y administrar la base de datos remota se emplea el **Table Editor** disponible en la interfaz de Supabase. Esta herramienta permite gestionar de forma visual las tablas y los registros almacenados, facilitando así la administración de la información dentro del sistema.



5 GLOSARIO

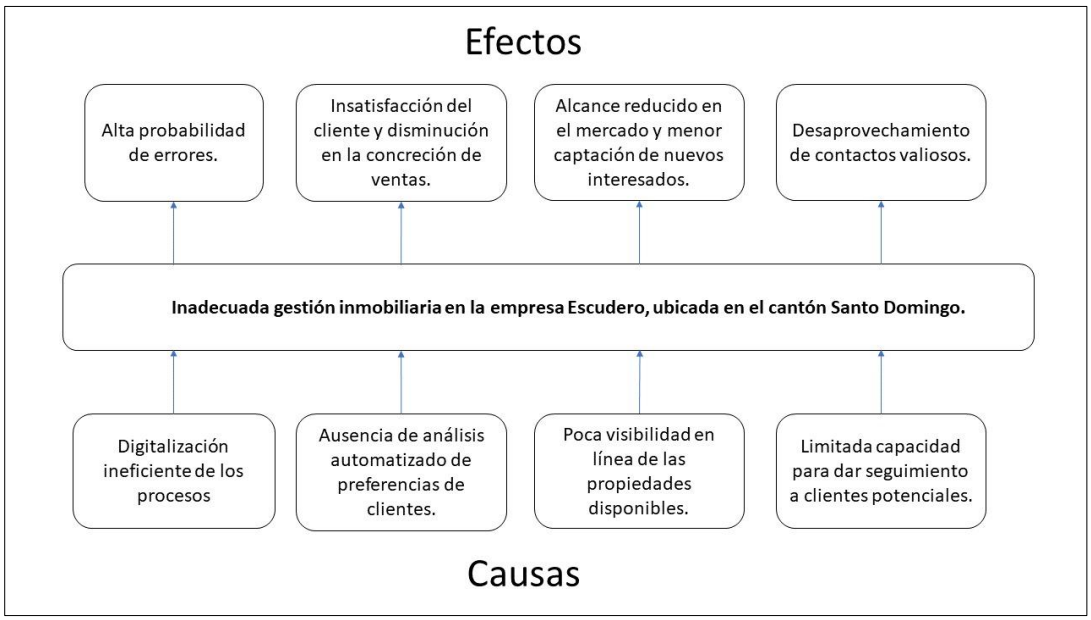
- **API REST:** Conjunto de servicios web que permiten la comunicación entre diferentes aplicaciones mediante solicitudes HTTP para consultar, crear, actualizar o eliminar información.
- **Backend:** Sistema utilizado para almacenar, organizar y gestionar la información del sistema, permitiendo realizar consultas y operaciones sobre los datos almacenados.

- **Clonación de repositorio:** Proceso mediante el cual se descarga una copia completa de un proyecto alojado en un repositorio remoto, generalmente utilizando la herramienta de control de versiones Git.
- **Frontend:** Capa de la aplicación que interactúa directamente con el usuario, encargada de mostrar la interfaz gráfica y permitir la interacción con el sistema.
- **Base de datos:** Sistema utilizado para almacenar, organizar y gestionar la información del sistema, permitiendo realizar consultas y operaciones sobre los datos almacenados.
- **GitHub:** Plataforma de desarrollo colaborativo que permite almacenar proyectos de software, controlar versiones del código y facilitar el trabajo en equipo.
- **Machine Learning:** Rama de la inteligencia artificial que permite a los sistemas aprender a partir de datos y realizar predicciones o análisis sin ser programados explícitamente para cada tarea.
- **Node.js:** Entorno de ejecución que permite ejecutar código JavaScript en el servidor, utilizado para desarrollar aplicaciones backend.
- **Prisma ORM:** Herramienta que permite interactuar con bases de datos relacionales mediante modelos de datos definidos en código, facilitando la gestión de consultas y migraciones.
- **React:** Biblioteca de JavaScript utilizada para construir interfaces de usuario dinámicas y basadas en componentes.
- **Repositorio:** Espacio donde se almacena el código fuente de un proyecto, junto con su historial de cambios y versiones.
- **Render:** Plataforma de computación en la nube utilizada para desplegar aplicaciones web, permitiendo alojar backend, frontend y bases de datos.
- **Supabase:** Plataforma de backend que proporciona servicios como base de datos PostgreSQL, autenticación y almacenamiento en la nube para aplicaciones web.
- **Variables de entorno:** Parámetros de configuración utilizados por una aplicación para definir valores importantes como credenciales de acceso, claves de seguridad o direcciones de servicios externos.
- **Vite:** Herramienta de desarrollo utilizada para construir aplicaciones web modernas de forma rápida, especialmente en proyectos basados en React.

6 REFERENCIAS

- IEEE (2001). *1063-2001 - IEEE Standard for Software User Documentation* [Internet]. Recuperado de: https://ieeexplore.ieee.org/document/974401

Anexo IX: Árbol del Problema



Anexo X: Recodificación del instrumento

Recodificación	Escenarios
0	Sin propuesta
1	Con propuesta

Recodificación escala de Likert

Valoración	Escala	Valoración	Escala	Valoración	Escala
	Utilidad		**Frecuencia**		**Seguridad**
5	Totalmente útil	5	Muy frecuente	5	Totalmente seguro
4	Útil	4	Frecuente	4	Seguro
3	Moderadamente útil	3	Medianamente frecuente	3	Moderadamente seguro
2	Poco útil	2	Poco frecuente	2	Poco seguro
1	Inútil	1	Nada frecuente	1	Totalmente inseguro
	Importancia		**Dificultad**		**Edad**
5	Muy importante	5	Muy fácil	4	18 - 25
4	Importante	4	Fácil	3	26 - 30
3	Medianamente importante	3	Normal	2	30 - 40
2	Poco importante	2	Difícil	1	40 o más
1	Nada importante	1	Muy difícil		**Genero**
	Necesidad		**Claridad**	2	Masculino
5	Totalmente necesario	5	Muy clara	1	Femenino
4	Necesario	4	Clara		
3	Moderadamente necesario	3	Medianamente clara		
2	Poco necesario	2	Poco clara		
1	Innecesario	1	Nada clara		
	De acuerdo		**Tipo**		
5	Totalmente de acuerdo	5	Teléfono móvil		
4	De acuerdo	4	Computadora de escritorio		
3	Moderadamente de acuerdo	3	Laptop		
2	Poco de acuerdo	2	Tablet		
1	Nada de acuerdo	1	Otro		
	Familiaridad		**Satisfacción**		
5	Totalmente familiarizado	5	Totalmente satisfecho		
4	Familiarizado	4	Satisfecho		
3	Moderadamente familiarizado	3	Moderadamente satisfecho		
2	Poco familiarizado	2	Poco satisfecho		
1	Nada familiarizado	1	Nada satisfecho		

Anexo XI: Transcripción de la reunión de planificación del sprint con el gerente

Pregunta 1: ¿Cree que el uso de una *app* mejoraría la eficiencia del equipo en cuanto al mejoramiento de sus labores diarias en la empresa?

Respuesta: Indudablemente, la tecnología es la que está ahora en auge y es lo que marca diferencia en cualquier tipo de negocio. Entonces, si, es muy importante y primordial mantener y tener tecnología en cualquier tipo de negocio.

Pregunta 2: ¿Qué mejoras espera ver en la gestión inmobiliaria después de implementar una solución digital?

Respuesta: Bueno, como la tecnología está el alcance de todos, y ahora está por todos lados, al tener un equipo o un programa que nos ayude con el tema de ventas, vamos a llegar a muchas más personas, a muchos más clientes y, por ende, vamos a tener más ventas.

Pregunta 3: ¿Cómo llevan el control de las propiedades disponibles, vendidas y reservadas?

Respuesta: Bueno, el control lastimosamente se lo lleva de una manera análoga, por medio de archivadores, en el tema de propiedades que tenemos a la venta. Las vendidas ya se las descartan, ya no se las ofrece ni se las publica.

Pregunta 4: ¿Cómo registra el avance de una negociación?

Respuesta: El avance de una negociación se lo hace por medio de vía WhatsApp, por medio de agenda también que tiene cada asesor o cada agente de aquí de la oficina.

Pregunta 5: ¿Qué procesos siguen ustedes en una negociación desde que un cliente muestre interés hasta que se concrete el arriendo o la venta?

Respuesta: Bueno, el proceso que se sigue es, más que todo, darle seguimiento al cliente hasta ver el interés de la propiedad que desea adquirir. Entonces, en este negocio hay que estar constante y persistente en la negociación, porque muchas veces los clientes no están decididos a las primeras ofertas que se les ofrece. Entonces, hay que dar seguimiento, pero en este caso lo hacemos por medio de vía telefónica, vía WhatsApp, por las redes sociales.

Pregunta 6: ¿Considera útil la incorporación de inteligencia artificial para recomendar propiedades a los clientes que usan la parte pública de la aplicación?

Respuesta: Indudablemente, la inteligencia artificial es la que ahora está destacando a nivel mundial. Y si, es muy importante mantenerla y tenerla ahora en el negocio.

Pregunta 7: ¿Cómo registran actualmente las propiedades en venta o arriendo?

Respuesta: Bueno, se registran por medio de la página que tenemos, donde se las publica, y también en forma manual, que es ya por medio de archivadores o carpetas donde se tienen todas las propiedades con contratos de corretaje.

Pregunta 8: ¿Tienen propiedades que permanecen publicadas, aunque ya no estén disponibles?

Respuesta: No, propiedad que se venda o se reserve, inmediatamente se le da de baja del sistema que tenemos.

Pregunta 9: ¿Cómo se registra la información de los clientes interesados en las propiedades que ofertan?

Respuesta: Bueno, cada agente o asesor de aquí de la oficina lo tiene registrado por medio de una agenda que cada uno tiene personalmente y también el teléfono, en una aplicación que tenemos y en WhatsApp.

Pregunta 10: ¿Llevan fichas con datos personales de los clientes? ¿Dónde las almacenan?

Respuesta: Eso no se lleva con un dato exacto. Lo que se almacena es también por medio de la agenda, los clientes que se atienden.

Pregunta 11: ¿Cómo se lleva el seguimiento a los clientes que están interesados por una determinada propiedad?

Respuesta: El seguimiento que se le lleva los clientes es de acuerdo al interés que tengan de la propiedad con fechas. Muchos aplazan días, muchos ponen un día determinado, y lo que se anota es en una agenda personal para saber qué día volver a llamarlos.

Pregunta 12: ¿Se puede saber fácilmente en qué etapa está un cliente respecto a una negociación?

Respuesta: Bueno, la negociación cuando se hace de contado, el trámite para hacer el traspaso al nuevo propietario es inmediato; lo hacemos en dos días, entonces no es mucho tiempo el que lleva hacer eso. Pero cuando es con tema bancario si nos toca depender del banco y pedir información cada día o cada tiempo, que el banco nos diga que le solicitemos en qué proceso está el trámite, porque eso ya es interno del banco.

Pregunta 13: ¿Quiénes pueden intervenir en el proceso de publicación de una propiedad?

Respuesta: Bueno, aquí los que intervenimos somos todos los agentes, pero en especial tenemos una persona que se dedica al marketing y la publicidad de aquí de la oficina. Pero en general publican todos.

Pregunta 14: ¿Cómo gestiona la información de sus agentes inmobiliarios?

Respuesta: Bueno, la información de cada agente que tenemos aquí es que cuando se solicita un agente le pedimos una hoja de vida. Nos traen el PDF, donde se la tiene archivada en el WhatsApp interno y también físicamente.

Pregunta 15: ¿Cómo gestionan los documentos como, por ejemplo, copias de cédula, promesas de compraventa, escrituras, etc., de una determinada negociación?

Respuesta: Bueno, a cada propiedad se le hace una carpeta. Se le hace una carpeta donde hay un archivador donde se mantiene a todas. Cuando se está ya negociando la propiedad, también por medio de carpeta, se lleva el seguimiento hasta culminar, y ya la carpeta se daría de baja cuando se da la venta.

Anexo XII: Certificados de participación

Pontificia Universidad Católica del Ecuador | SANTO DOMINGO

Otorga el siguiente

CERTIFICADO

A:

FLORES VÉLEZ ROMARIO ARMANDO

En reconocimiento a su destacada contribución en el ámbito de la Investigación e Innovación, mediante su participación activa con el tema:

APLICACIÓN WEB CON MACHINE LEARNING PARA LA GESTIÓN INMOBILIARIA DE LA EMPRESA ESCUDERO DEL CANTÓN SANTO DOMINGO

Su compromiso y excelencia representan un ejemplo inspirador para nuestra institución y futuras generaciones de profesionales.

Mg. Franklin Carrasco Ramírez
COORDINADOR DE LA CARRERA DE SISTEMAS DE INFORMACIÓN

QUITO - AMAZONAS - AMBATO - ESMERALDAS - IBARRA - MANABÍ - SANTO DOMINGO

Pontificia Universidad Católica del Ecuador | SANTO DOMINGO

Otorga el siguiente

CERTIFICADO

A:

ALARCÓN MORILLO JHERY FERNANDO

En reconocimiento a su destacada contribución en el ámbito de la Investigación e Innovación, mediante su participación activa con el tema:

APLICACIÓN WEB CON MACHINE LEARNING PARA LA GESTIÓN INMOBILIARIA DE LA EMPRESA ESCUDERO DEL CANTÓN SANTO DOMINGO

Su compromiso y excelencia representan un ejemplo inspirador para nuestra institución y futuras generaciones de profesionales.

Mg. Franklin Carrasco Ramírez
COORDINADOR DE LA CARRERA DE SISTEMAS DE INFORMACIÓN

QUITO - AMAZONAS - AMBATO - ESMERALDAS - IBARRA - MANABÍ - SANTO DOMINGO

Anexo XIII: Informe del Turnitin

TTG-TI-ALARCON-FLORES-202502FINAL..docx

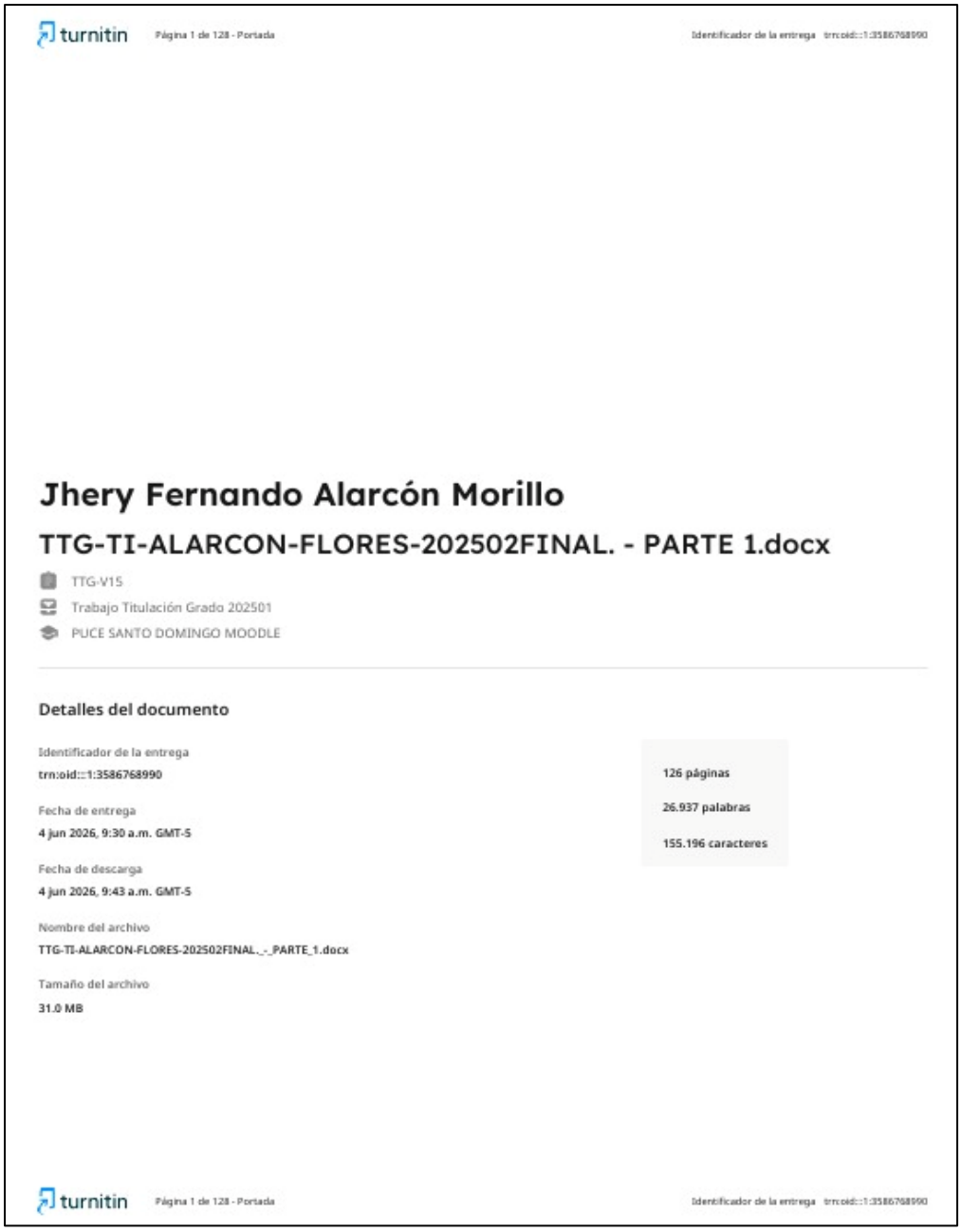
por Jhery Fernando Alarcón Morillo

Fecha de entrega: 04-jun-2026 09:20a. m. (UTC-0500)
Identificador de la entrega: 2976300877
Nombre del archivo: TTG-TI-ALARCON-FLORES-202502FINAL..docx (87.14M)
Total de palabras: 35718
Total de caracteres: 207867

TTG-TI-ALARCON-FLORES-202502FINAL..docx

INFORME DE ORIGINALIDAD

7%	6%	1%	2%
INDICE DE SIMILITUD	FUENTES DE INTERNET	PUBLICACIONES	TRABAJOS DEL ESTUDIANTE



turnitin Página 1 de 128 - Portada Identificador de la entrega trn:oid:::1:3586768990

Jhery Fernando Alarcón Morillo

TTG-TI-ALARCON-FLORES-202502FINAL. - PARTE 1.docx

TTG-V15
Trabajo Titulación Grado 202501
PUCE SANTO DOMINGO MOODLE

Detalles del documento

Identificador de la entrega
trn:oid:::1:3586768990

Fecha de entrega
4 jun 2026, 9:30 a.m. GMT-5

Fecha de descarga
4 jun 2026, 9:43 a.m. GMT-5

Nombre del archivo
TTG-TI-ALARCON-FLORES-202502FINAL_-_PARTE_1.docx

Tamaño del archivo
31.0 MB

126 páginas
26.937 palabras
155.196 caracteres

turnitin Página 1 de 128 - Portada Identificador de la entrega trn:oid:::1:3586768990

turnitin Página 2 de 128 - Descripción general de la escritura con IA Identificador de la entrega trn:oid:::1:3586768990

0 % detectado como IA

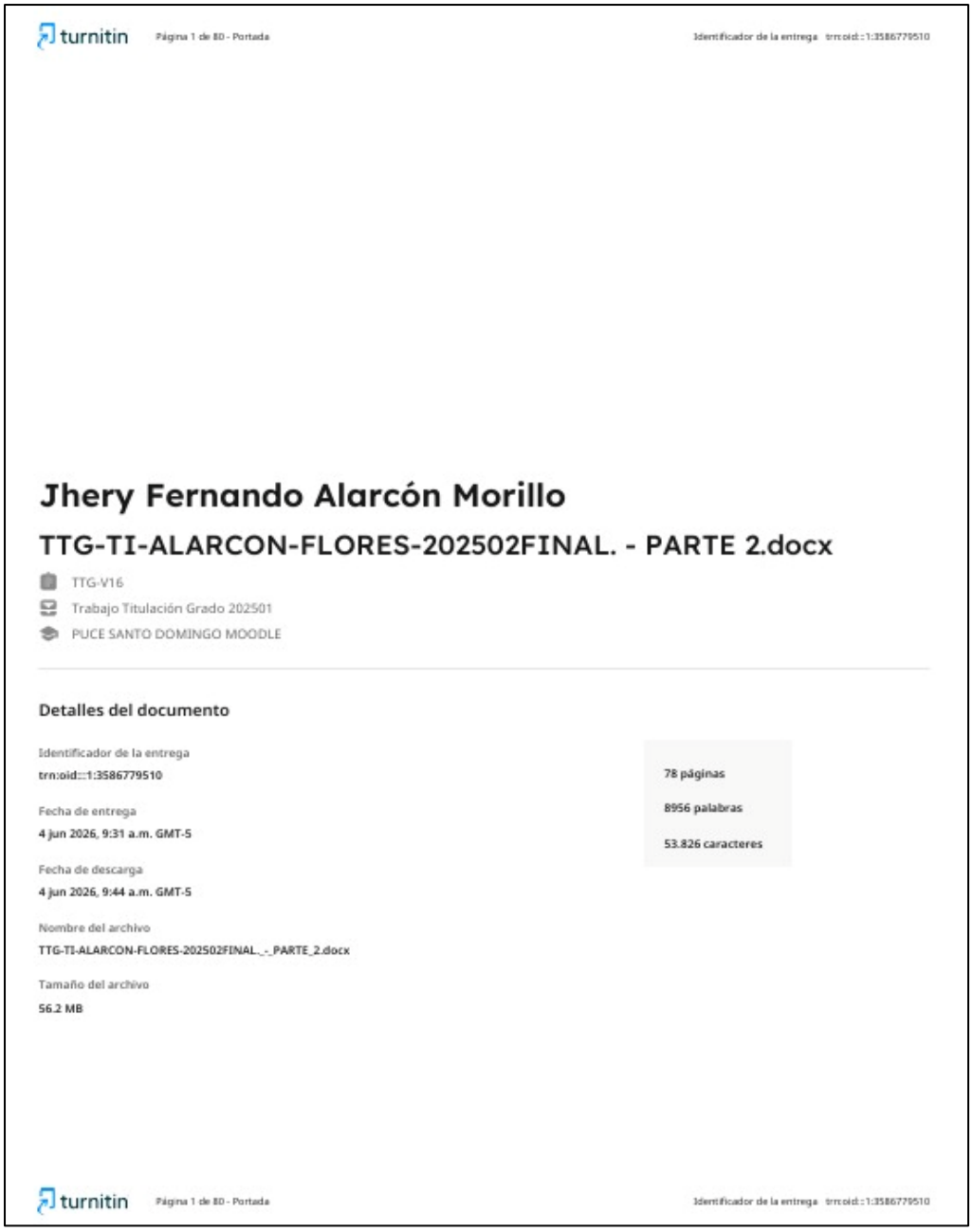
El porcentaje indica la cantidad de texto calificado en la entrega que probablemente se generó usando IA.

Precaución: Se necesita revisión.

Es esencial comprender los límites de la detección de IA antes de tomar decisiones acerca del trabajo del estudiante. Te alentamos a obtener más información acerca de las funciones de detección de IA de Turnitin antes de usar la herramienta.

2 Sólo generado con IA 0%
Texto generado probablemente con IA desde un modelo de lenguaje de gran tamaño.

Aviso legal
Nuestra evaluación de escritura con IA está diseñada para ayudar a los académicos a identificar texto que podrían haberse preparado mediante una herramienta de IA generativa. Es posible que nuestra evaluación de escritura con IA no siempre sea precisa (existe la posibilidad de que identifique erróneamente redacciones probablemente generadas por humanos como generadas por IA, y redacciones probablemente generadas por IA como generadas por humanos), por lo que no debe usarse como único fundamento para aplicar sanciones a un estudiante. Para determinar si es un caso de deshonestidad académica, se necesita de un escrutinio mayor y el juicio humano, junto con la aplicación de las políticas académicas específicas de la organización.



turnitin Página 1 de 80 - Portada Identificador de la entrega trn:oid:::1:3586779510

Jhery Fernando Alarcón Morillo

TTG-TI-ALARCON-FLORES-202502FINAL. - PARTE 2.docx

TTG-V16
Trabajo Titulación Grado 202501
PUCE SANTO DOMINGO MOODLE

Detalles del documento

Identificador de la entrega
trn:oid:::1:3586779510

Fecha de entrega
4 jun 2026, 9:31 a.m. GMT-5

Fecha de descarga
4 jun 2026, 9:44 a.m. GMT-5

Nombre del archivo
TTG-TI-ALARCON-FLORES-202502FINAL_-_PARTE_2.docx

Tamaño del archivo
56.2 MB

78 páginas
8956 palabras
53.826 caracteres

turnitin Página 1 de 80 - Portada Identificador de la entrega trn:oid:::1:3586779510

turnitin Página 2 de 80 - Descripción general de la escritura con IA Identificador de la entrega trn:oid:::1:3586779510

*% detectado como IA

La detección de IA incluye la posibilidad de que haya falsos positivos. Aunque cierto texto en esta entrega se generó probablemente con IA, los puntajes inferiores al umbral del 20 % no aparecen porque tienen una mayor probabilidad de falsos positivos.

Precaución: Se necesita revisión.

Es esencial comprender los límites de la detección de IA antes de tomar decisiones acerca del trabajo del estudiante. Te alentamos a obtener más información acerca de las funciones de detección de IA de Turnitin antes de usar la herramienta.

Aviso legal
Nuestra evaluación de escritura con IA está diseñada para ayudar a los académicos a identificar texto que podrían haberse preparado mediante una herramienta de IA generativa. Es posible que nuestra evaluación de escritura con IA no siempre sea precisa (existe la posibilidad de que identifique erróneamente redacciones probablemente generadas por humanos como generadas por IA, y redacciones probablemente generadas por IA como generadas por humanos), por lo que no debe usarse como único fundamento para aplicar sanciones a un estudiante. Para determinar si es un caso de deshonestidad académica, se necesita de un escrutinio mayor y el juicio humano, junto con la aplicación de las políticas académicas específicas de la organización.