

# **ESCUELA POLITÉCNICA NACIONAL**

**ESCUELA DE FORMACIÓN DE TECNÓLOGOS**

## **DESARROLLO FRONT-END PARA LA APLICACIÓN WEB DE COMPRAVENTA DE BIENES INMUEBLES**

**TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO  
REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE TECNÓLOGO SUPERIOR  
EN DESARROLLO DE SOFTWARE**

**RONNY PATRICIO CAJAS BENITEZ**

**DIRECTOR: ING. JUAN PABLO ZALDUMBIDE PROAÑO**

**DMQ, septiembre 2022**

## CERTIFICACIONES

Yo, RONNY PATRICIO CAJAS BENITEZ declaro que el trabajo de integración curricular aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.



---

**RONNY PATRICIO CAJAS BENITEZ**

**ronny.cajas@epn.edu.ec**

**pato1418@yahoo.com**

Certifico que el presente trabajo de integración curricular fue desarrollado por RONNY PATRICIO CAJAS BENITEZ, bajo mi supervisión.



**JUAN PABLO ZALDUMBIDE PROAÑO**

**DIRECTOR**

**juan.zaldumbide@epn.edu.ec**

## DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como el (los) producto(s) resultante(s) del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

A handwritten signature in black ink, appearing to read 'Ronny Cajas', with a stylized flourish underneath.

RONNY PATRICIO CAJAS BENITEZ

## **DEDICATORIA**

Este proyecto está dedicado a mi familia y a los docentes de la ESFOT ya que sin ellos no habría podido desempeñarme como estudiante y profesional.

**RONNY PATRICIO CAJAS BENITEZ**

## **AGRADECIMIENTO**

Quiero agradecer a mis padres por darme ánimos durante toda mi carrera y apoyarme a pesar de las dificultades.

Agradezco a mi pareja por estar a mi lado durante esta etapa tan importante de mi vida.

Finalmente agradezco a mis profesores por su enorme aporte en mi formación profesional.

**RONNY PATRICIO CAJAS BENITEZ**

## ÍNDICE DE CONTENIDO

CERTIFICACIONES .....	I
DECLARACIÓN DE AUTORÍA .....	II
DEDICATORIA.....	III
AGRADECIMIENTO .....	IV
ÍNDICE DE CONTENIDO .....	V
RESUMEN .....	VI
ABSTRACT .....	VII
1 DESCRIPCIÓN DEL COMPONENTE DESARROLLADO.....	1
1.1 Objetivo general .....	2
1.2 Objetivos específicos .....	2
1.3 Alcance .....	2
1.4 Marco Teórico .....	2
2 METODOLOGÍA .....	3
2.1 Programación extrema.....	3
2.2 Diseño de interfaces ( <i>mockups</i> ).....	7
2.3 Diseño de la arquitectura .....	10
2.4 Herramientas de desarrollo.....	12
3 RESULTADOS.....	14
Iteración 1. Conceptualización, diseño y configuración. ....	14
Iteración 2. Codificación de pantallas: inicio de sesión y registro de usuarios .....	16
Iteración 3. Codificación de pantallas: CRUD de publicaciones. ....	19
Iteración 4. Codificación de pantalla: Buscador de propiedades. ....	24
Iteración 5. Pruebas.....	27
Conclusiones.....	29
Recomendaciones .....	30
6 REFERENCIAS BIBLIOGRÁFICAS.....	31
7 ANEXOS .....	32
ANEXO II. Manual técnico .....	34

## RESUMEN

Este documento explica el proceso de desarrollo de una aplicación web progresiva que contendrá vistas para publicar anuncios de bienes inmuebles con la finalidad de encontrar compradores o vendedores según lo requieran los usuarios. La aplicación utilizará React como recurso principal para programar la lógica interactiva. Para crear una experiencia fluida se utilizarán librerías de nextjs que permiten implementar el renderizado del lado del servidor con el objetivo de reducir el tiempo de espera antes de que el usuario pueda visualizar información relevante [1]

**PALABRAS CLAVE:** Renderizado del lado del servidor, React, Nextjs, lógica interactiva.

## **ABSTRACT**

This document explains the process of developing a progressive web application that will contain views to publish real estate ads in order to find buyers or sellers as required by users. The application will use React as the main resource to program the interactive logic. To create a smooth experience, nextjs libraries will be used to implement server-side rendering in order to reduce the waiting time before the user can view relevant information [1]

**KEY WORDS:** Server-side rendering, React, Nextjs, interactive logic.



# 1 DESCRIPCIÓN DEL COMPONENTE DESARROLLADO

Se espera que con el paso del tiempo las personas se beneficien cada vez más de plataformas online cuando se propongan a comprar un inmueble [2].

Con la finalidad de generar una alternativa de código libre disponible para la comunidad involucrada en la comercialización de bienes inmuebles se ha desarrollado una plataforma web de compraventa de inmuebles. Esta plataforma se enfoca principalmente en proporcionar una forma de publicar anuncios que permitan contactar a posibles compradores y/o vendedores y que estos anuncios sean visibles para otros usuarios.

Este proyecto se ha conceptualizado como un servicio accesible por medios electrónicos, construido con tecnologías y herramientas modernas para proporcionar una experiencia de usuario similar al de otras plataformas de anuncios.

Se ha seleccionado este tipo de software por la facilidad de distribución y difusión que tiene. Basta con publicar un enlace en redes sociales para que un usuario pueda dar click y acceder a la página de bienvenida del sitio sin tener que descargar archivos adicionales como sucede con aplicaciones móviles nativas. Así mismo se le ha agregado responsividad a todas las pantallas por lo que pueden visualizarse correctamente en gran variedad de dispositivos móviles y computadores de escritorio.

Pensando en la facilidad de difusión se ha agregado también el soporte para los navegadores populares Chrome, Firefox y Safari.

Para proporcionar el servicio la plataforma accede a información resguardada bajo un método de autenticación con correo y contraseña por lo que los usuarios deberán identificarse utilizando las pantallas de registro e inicio de sesión antes de acceder.

Finalmente, pensando en mantener la simplicidad, se ha concentrado el proceso para publicar en un formulario accesible desde la pantalla inicial del sitio. Aquellos interesados en comprar tienen siempre visible el enlace al buscador en la barra de navegación. En caso de ser requerido, el comprador puede segregar resultados de búsqueda según sus preferencias. Cada anuncio muestra información de contacto suficiente para que las partes puedan concretar el negocio como deseen, esto incluye características del inmueble y un enlace de acceso directo que abrirá la aplicación de whatsapp, misma que debe mostrar una ventana de chat con el vendedor anunciante.

La plataforma no interviene en actividades relacionadas con la negociación, pago o trámites legales con instituciones públicas. La plataforma no incluye un servicio de mensajería

integrado ni almacena historiales de conversaciones entre los involucrados. La plataforma tampoco verifica la identidad real de los usuarios referentes a las leyes vigentes locales y tampoco almacena códigos de identificación regionales tales como como cédula de identidad o registro de contribuyente.

## **1.1 Objetivo general**

Desarrollar una aplicación web progresiva para la compraventa de bienes inmuebles

## **1.2 Objetivos específicos**

1. Determinar los requerimientos.
2. Diseñar las interfaces y arquitectura.
3. Implementar la aplicación.
4. Comprobar el funcionamiento de la aplicación.

## **1.3 Alcance**

El presente proyecto propone el desarrollo de una aplicación web progresiva que habilite a los usuarios para que publiquen anuncios de inmuebles de su propiedad y que de esta manera puedan contactarse directamente con potenciales compradores y/o vendedores.

La propuesta es crear una herramienta que le permita al dueño de una propiedad en venta participar de forma mas activa en el cierre del negocio. [3]

Para esto se utilizarán el *stack* de tecnologías de *front-end* React, Next y Vercel que permite crear interfaces modernas e intuitivas. [2] [3]

## **1.4 Marco Teórico**

En los meses posteriores a la pandemia el ecuador experimentó un aumento considerable en la demanda de bienes inmuebles [4]. Este contexto presenta la oportunidad de abrir una plataforma que permita publicar anuncios que sirva a la comunidad como alternativa a las opciones ya disponibles en el mercado.

## **Compra de inmuebles**

En la actualidad es posible publicar anuncios utilizando plataformas como inmoblex, plusvalía o mercado libre. Plusvalía por ejemplo goza de gran popularidad en el Ecuador puesto que en el 2014 ya tenía registrados a 450 agentes inmobiliarios [5].

## **El rol de las plataformas inmobiliarias**

Las páginas web inmobiliarias se han convertido en medios de acceso a negocios inmobiliarios de gran importancia. Según Market Watch, más del 60% de usuarios con acceso a internet utilizan estas plataformas para informarse sobre las ofertas de bienes inmuebles [6].

# **2 METODOLOGÍA**

## **2.1 Programación extrema**

Programación extrema o *XP* es una metodología ágil y disciplina que propone el uso riguroso de técnicas de desarrollo. [7]

XP fomenta la mejora continua del código fuente en cualquier etapa del desarrollo. Por este motivo es indispensable crear y mantener pruebas de software automatizadas. Éstas pruebas deben asegurar la disponibilidad del sistema. [7]

Puesto que el proyecto es ejecutado por un solo integrante que desempeña los roles de cliente y desarrollador se ha considerado como riesgos a considerar:

1. Retrasos
  - No cumplir con las fechas de entrega acordadas.
2. Exceso de errores.
  - El software es desplegado correctamente, pero es difícil de utilizar porque tiene demasiados errores.
3. Funcionalidad innecesaria
  - Desarrollar cosas entretenidas de programar pero que no aportan valor real al negocio.
4. Deserción
  - Acumular demasiado código ilegible y difícil de modificar hasta el punto de provocar el abandono del proyecto.

Siguiendo la metodología, dichos riesgos fueron mitigados como se describe a continuación:

1. Reducción de retrasos
  - Ciclos de una semana para implementar cada historia de usuario.
  - Enfoque del esfuerzo en las funcionalidades más importantes.
2. Mitigación de errores
  - Creación de pruebas automatizadas con *Cypress*.
  - Uso del lenguaje fuertemente tipado *Typescript* [8] para detectar de manera temprana errores comunes de tipado.
  - Uso del autocompletado de código *Typescript* incluido en *Visual Studio Code* por defecto. [9]
3. Identificación temprana de funcionalidad innecesaria
  - Se ha determinado el alcance y las características específicas de la aplicación.
4. Deserción
  - Configurar el formateo de código automático con *eslint* para mejorar la legibilidad del proyecto.
  - Configurar el despliegue automático de los cambios en el código fuente de la aplicación para agregar cambios y correcciones de forma ágil. Para esto se utilizaron los servicios de Vercel junto con el framework *nextjs* [10].

## **Roles**

XP también propone las actividades y responsabilidades para los roles del proyecto de manera que cada miembro pueda beneficiarse de la metodología mientras ejecuta su trabajo.

### **Programador**

Debe velar por el correcto funcionamiento del producto y comunicarse de manera efectiva con los clientes para asegurarse de que su trabajo está agregando valor al negocio. Esto hace necesario despejar cualquier duda con el cliente antes y durante el desarrollo de sus actividades. [7]

### **Cliente**

En XP el cliente se convierte en una parte integral del equipo. Debe utilizar el producto con frecuencia y conocer el negocio de cerca. Es deseable que el cliente exprese requerimientos del sistema con **historias de usuario** o codificando **casos de prueba**. [7]

## Artefactos

### Estrategia de Planificación

En XP el proceso de planificación involucra al dueño de negocio y desarrollador en actividades que deberán entregar software funcional minimizando el riesgo y costes de desarrollo.

Estas actividades incluyen:

**Exploración:** Definir que funcionalidades se agregarán y crear las historias de usuario necesarias para completarlas.

**Compromiso:** Decidir las fechas límite para el despliegue a producción.

### Estrategia para el Desarrollo

Uno de los fundamentos de XP es empezar con un diseño simple que pueda evolucionar con el tiempo [7].

### Historias de Usuario:

Para este proyecto se ha distribuido el trabajo en pantallas. Cada historia de usuario describe una pantalla junto con sus funcionalidades asociadas.

La pantalla para publicar propiedades permite al usuario llenar la información de su inmueble con campos de texto y fotos.

La TABLA II muestra la historia de usuario para publicar un inmueble.

**TABLA II Historia de Usuario No. 3**

HISTORIA DE USUARIO	
<b>Identificador (ID):</b> HU003	<b>Usuario:</b> Vendedor / Comprador
<b>Nombre Historia:</b> Publicar propiedad	
<b>Prioridad en Negocio:</b> Alta	<b>Riesgo en Desarrollo:</b> Bajo
<b>Iteración Asignada:</b> 4	
<b>Responsable:</b> Ronny Cajas	
<b>Descripción:</b> El usuario podrá ingresar los detalles de una propiedad en un formulario con los siguientes campos: Título de publicación Número de baños Número de dormitorios	

<p>Número de salas  Número de cocinas  Número de parqueaderos  Hasta 4 fotografías</p> <p>El botón para confirmar la edición tendrá el mensaje “Publicar”.</p>
<p><b>Observaciones:</b></p> <p>Después de que el usuario presione el botón “Publicar”, un componente de carga deberá ocupar la pantalla e impedir que el usuario interactúe con el formulario hasta que la petición al servidor se complete.</p>

La TABLA III muestra la historia de usuario para publicar un inmueble.

**TABLA III Historia de Usuario No. 4**

<b>HISTORIA DE USUARIO</b>	
<b>Identificador (ID):</b> HU004	<b>Usuario:</b> Vendedor / Comprador
<b>Nombre Historia:</b> Editar propiedad	
<b>Prioridad en Negocio:</b> Alta	<b>Riesgo en Desarrollo:</b> Bajo
<b>Iteración Asignada:</b> 4	
<b>Responsable:</b> Ronny Cajas	
<p><b>Descripción:</b></p> <p>El usuario podrá seleccionar una de sus propiedades previamente creadas con el formulario de la Historia HU003. Al momento de abrir esta pantalla, los campos se llenarán automáticamente con la información actual de la propiedad.</p> <p>El botón para confirmar la edición tendrá el mensaje “Guardar cambios”.</p> <p>Cuando el usuario accede a esta pantalla los campos deben precargar la información actual de la propiedad sin una petición al servidor de por medio.</p>	
<p><b>Observaciones:</b></p> <p>Después de que el usuario presione el botón “Guardar cambios”, un componente de carga deberá ocupar la pantalla e impedir que el usuario interactúe con el formulario hasta recibir una respuesta del servidor.</p>	

## 2.2 Diseño de interfaces (*mockups*)

Una página web puede ser simple o compleja en ambos casos es recomendable crear un prototipo o mockup.

Un prototipo detallado permite trabajar de manera ordenada y confiable puesto que puede ser usado como referencia durante la implementación previniendo la omisión o malinterpretación de características importantes de tamaño, color e interactividad.

### Herramienta de diseño

Para graficar las pantallas se ha utilizado el software de diseño vectorial *Figma*. *Figma* tiene una gran comunidad de desarrolladores y diseñadores que libera plantillas de código abierto.

En este proyecto se utilizó la plantilla *Ant Design Open Source*, basada en la librería *Ant Design*. [11]

Con esta plantilla fue posible crear un prototipo bastante parecido al resultado final.

### Diseño de pantallas

En *Figma* cada prototipo de pantalla empieza con un *Frame* vacío, los frames son contenedores que representan la pantalla de un dispositivo y son de utilidad principalmente para agrupar elementos de diseño como botones y figuras.

La Fig. 2 muestra los elementos del formulario para publicar anuncios dentro del *Frame PropertyForm*.



**Fig. 2: Prototipo de pantalla para publicar**

Todos los prototipos de pantalla se crearon utilizando varios *Frames* y elementos tomados de la plantilla *Ant Design Open Source*.

### Reutilización de componentes

Una de las técnicas de desarrollo más eficaces para reducir la carga de trabajo es la reutilización de código fuente [12], por este motivo en los prototipos se diseñaron componentes que fuesen de utilidad en varias pantallas de la aplicación.

Para planificar la reutilización de elementos visuales, *Figma* permite la creación de elementos reutilizables con características parametrizadas. [13]

En la **Fig. 3** tenemos al componente *PropertyCard* parametrizado y reutilizado en los prototipos **Fig. 4** y **Fig. 5**.




**Fig. 3: Componente de tipo tarjeta con información de la propiedad.**



## Mis publicaciones

PUBLICAR

4.769 Departamentos quito en alquiler en Ecuador




**USD 40 000**  
Casa - Bourgeois, Quito, Ecuador

Este hermosa y amplia propiedad se encuentra en la zona baja de cumbayá, con tiene acceso a jardines privados dentro de la Urbanización Unión...

200 m<sup>2</sup> 1 dormitorio

2 baños 1 parqueadero 1 cocina




**USD 40 000**  
Casa - Bourgeois, Quito, Ecuador

Este hermosa y amplia propiedad se encuentra en la zona baja de cumbayá, con tiene acceso a jardines privados dentro de la Urbanización Unión...

200 m<sup>2</sup> 1 dormitorio

2 baños 1 parqueadero 1 cocina



**USD 40 000**  
Casa - Bourgeois, Quito, Ecuador

Este hermosa y amplia propiedad se encuentra en la zona baja de cumbayá, con tiene acceso a jardines privados dentro de la Urbanización Unión...

200 m<sup>2</sup> 1 dormitorio

2 baños 1 parqueadero 1 cocina

Fig. 4: Prototipo con el listado de publicaciones propias.

## Buscar propiedades

Comprar Alquiler

Casa

Dormitorios

+1	+2	+3	+4	+5
----	----	----	----	----

Parqueaderos

+1	+2	+3	+4	+5
----	----	----	----	----

Baños

+1	+2	+3	+4	+5
----	----	----	----	----

Cocinas

+1	+2	+3	+4	+5
----	----	----	----	----

4.769 Departamentos quito en alquiler en Ecuado



**USD 40 000**  
Casa - Bourgeois, Quito, Ecuador

Este hermosa y amplia propiedad se encuentra en la zona baja de cumbayá, con tiene acceso a jardines privados dentro de la Urbanización Unión...

200 m<sup>2</sup> 1 dormitorio

2 baños 1 parqueadero 1 cocina

Contactar



**USD 40 000**  
Casa - Bourgeois, Quito, Ecuador

Este hermosa y amplia propiedad se encuentra en la zona baja de cumbayá, con tiene acceso a jardines privados dentro de la Urbanización Unión...

200 m<sup>2</sup> 1 dormitorio

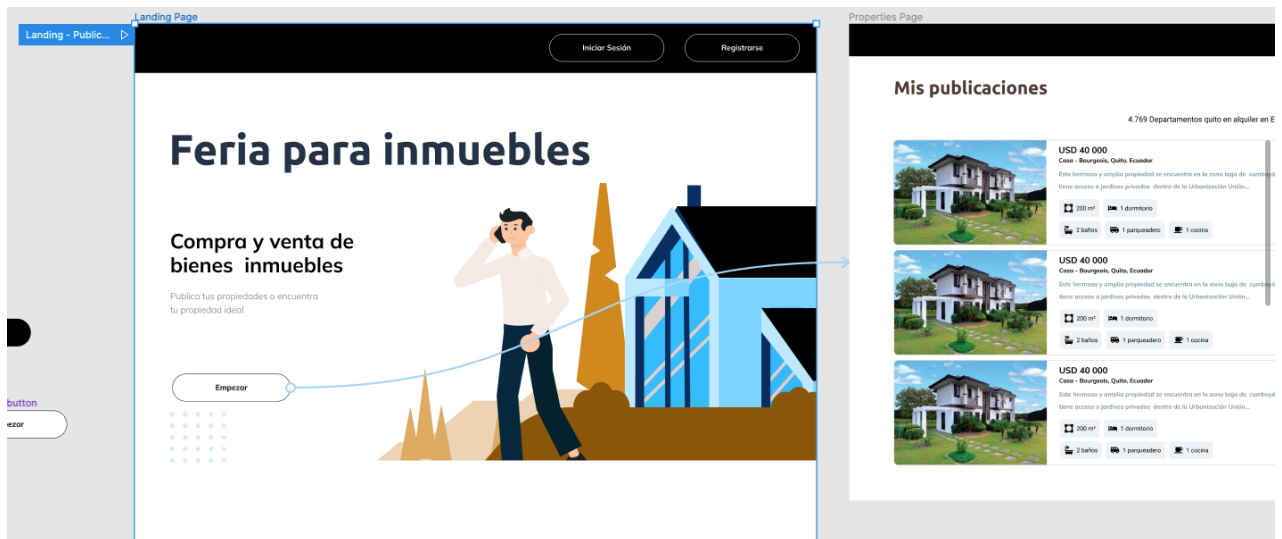
2 baños 1 parqueadero 1 cocina

Contactar

Fig. 5: Prototipo del buscador.

## Diseño de interactividad

En una página web es importante permitir que el usuario se desplace con facilidad. Por este motivo el diseño de la navegación fue esquematizado con la herramienta *Prototype* de *figma* **Fig. 6**, con el objetivo de diseñar una experiencia de navegación consistente y agradable.



**Fig. 6:** Flujo: “Landing – Publicación” creado con *Prototype*.

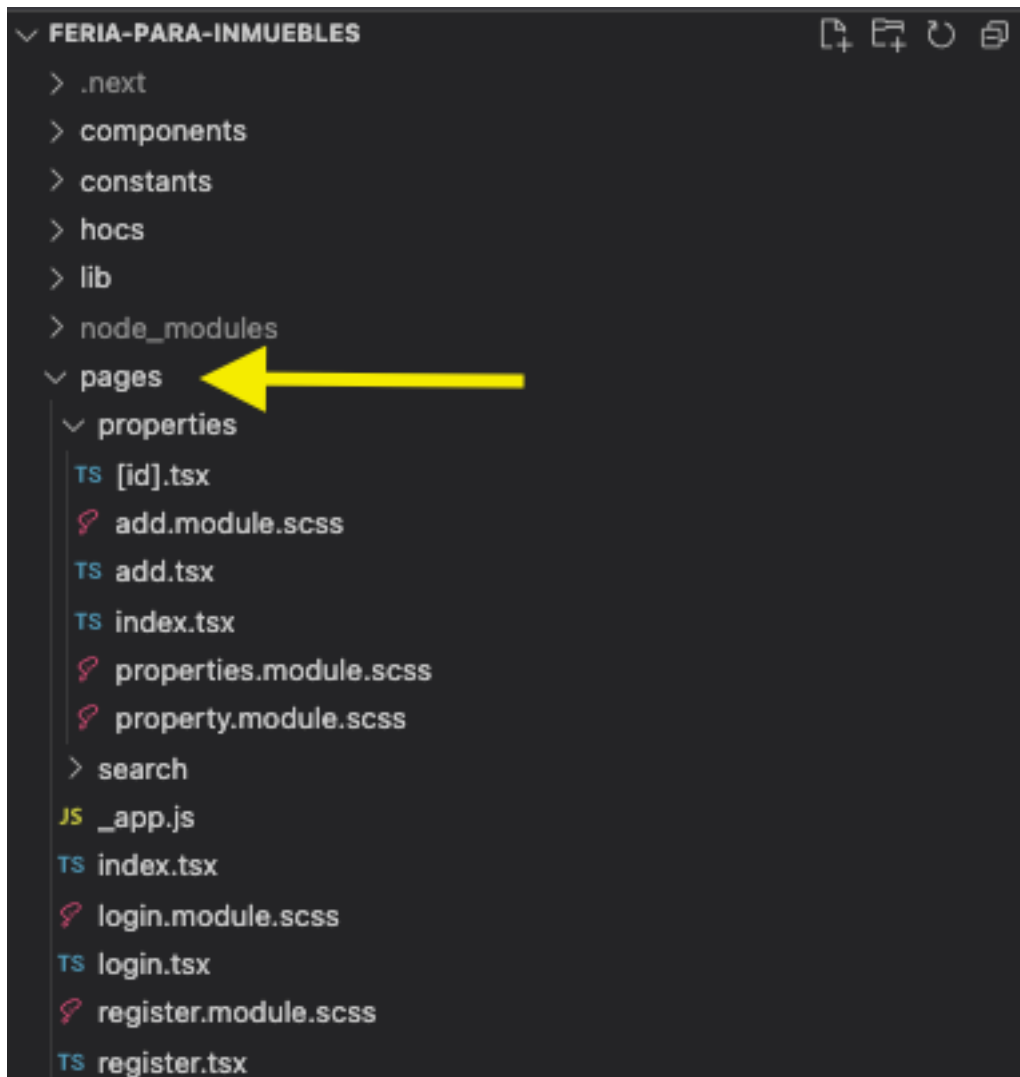
## 2.3 Diseño de la arquitectura

Este proyecto está constituido por una aplicación web que se ejecuta en el navegador. Esta aplicación está desarrollada con las librerías de *JavaScript React* y *Nextjs*.

Se ha utilizado *React* para programar pantallas reactivas formadas por *componentes* renderizados en el navegador en lenguaje *HTML*, *CSS* y *JavaScript*.

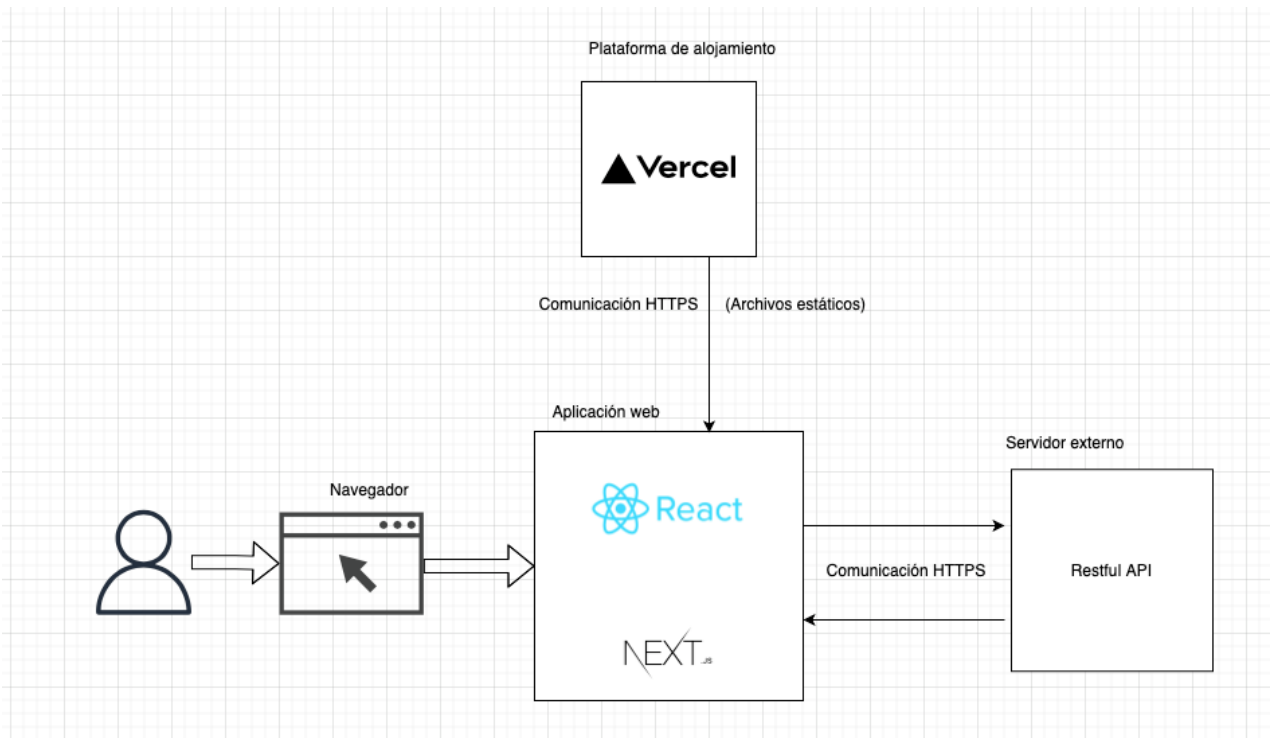
El uso de componentes de *React* permitió que el proyecto implementara paradigmas como composición y modularidad durante la codificación. [14].

*Nextjs* se utilizó para definir las rutas de navegación de la aplicación por medio de la estructura de directorios y archivos que establece la librería [15]. Véase la **Fig. 7**. La carpeta *Pages*.



**Fig. 7: Navegación basada en estructura de archivos.**

Adicionalmente, Nextjs es necesario para utilizar los servicios de alojamiento y despliegue automático de *Vercel*.



**Fig. 8: Diagrama de arquitectura.**

El elemento final de la arquitectura es una *API RESTful* externa de donde se obtiene la información de los usuarios y sus propiedades mediante peticiones *HTTPS*.

## 2.4 Herramientas de desarrollo

El código fuente de este proyecto fue escrito con *Visual Studio Code* con extensiones adicionales.

La TABLA V muestra cada herramienta utilizada con la justificación correspondiente.

**TABLA V Herramientas para el desarrollo de la aplicación web**

Herramienta	Justificación
<b>Visual Studio Code</b>	<p>Visual Studio Code es un editor de texto utilizado ampliamente para codificar aplicaciones web. En el espacio de trabajo de Visual Studio Code se instalaron las siguientes extensiones:</p> <p><i>Prettier</i>: formateo de código.</p> <p><i>Git Graph</i>: Visualizar el historial de versiones del sistema, navegar con facilidad por el historial de versiones del código.</p>

	<p>Eslint: Extensión oficial para el paquete de npm eslint, utilizado en este proyecto para mostrar alertas sobre el código fuente durante el desarrollo.</p>
<b>Github</b>	<p>Github Plataforma gratuita para alojar el código fuente.</p> <p>Se implementaron los servicios de Github en conjunto con <i>Vercel</i> para que los cambios realizados en el repositorio sean desplegados de forma automática al servidor de producción.</p>
<b>Inspector del navegador</b>	<p>Esta herramienta está disponible en todos los navegadores, webs populares, permite inspeccionar archivos estáticos y el código fuente de la aplicación.</p> <p>Herramientas del <i>Inspector</i> utilizadas.</p> <p><i>Visor de elementos:</i> Proporciona una ventana con el código fuente html que ha sido renderizado. Se utilizó principalmente para realizar ajustes de estilos y responsividad.</p> <p><i>Network:</i> Examinar las peticiones https para observar elementos como: cuerpo de peticiones y respuestas <i>http</i>, cabeceras y <i>url</i>.</p>
<b>Figma</b>	<p>Se seleccionó esta herramienta para aprovechar la plantilla de código abierto <i>Ant Design Open Source</i>.</p>

## 3 RESULTADOS

Este proyecto se realizó en iteraciones de dos semanas.

### Iteración 1. Conceptualización, diseño y configuración.

#### Conceptualización

Para publicar anuncios de manera segura y consistente, la aplicación debe incluir los siguientes elementos.

**Formulario de registro:** Debe contener campos de texto para almacenar la información personal del usuario, además de sus credenciales de acceso. Requiere validaciones con mensajes de ayuda.

**Formulario de inicio de sesión:** Permite autenticar al usuario si coloca correctamente sus credenciales.

**Formulario para publicar y modificar un inmueble:** Le permite al usuario colocar las características del inmueble. Requiere validaciones con mensajes de ayuda.

**Botón para eliminar inmueble:** Un usuario puede optar por remover su anuncio de la plataforma presionando un botón.

**Buscador:** Página que permite buscar propiedades con las características que el usuario prefiera. Esta página despliega los resultados de forma paginada.

#### Diseño

- Crear prototipos de pantallas
- Organizar elementos comunes en forma de componentes reutilizables.
- Agregar interactividad a los prototipos de pantalla.

#### Configuración

Crear el proyecto con la herramienta *create-next-app*. Esto generó la estructura de directorios plantilla. Sirvió como ejemplo y facilitó la comprensión sobre el patrón de navegación de *next* **Fig. 8**.

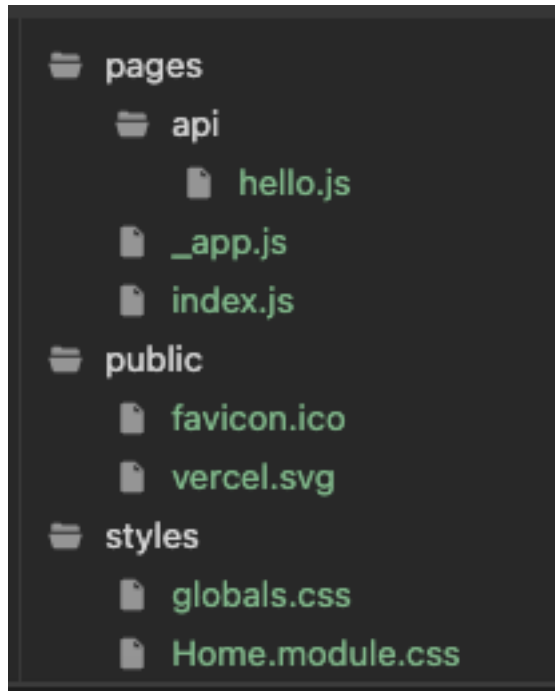


Fig. 8: Plantilla inicial

Crear el repositorio de *GitHub* Fig. 9.

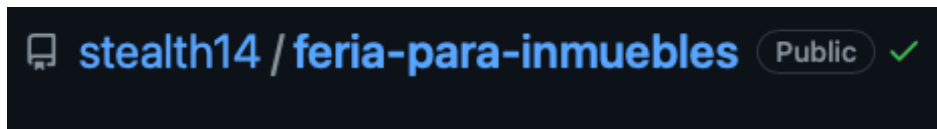


Fig. 9: Directorio público del repositorio.

Crear proyecto de la aplicación en *Vercel*. Esto incluye las configuraciones de despliegue y de dominio público.

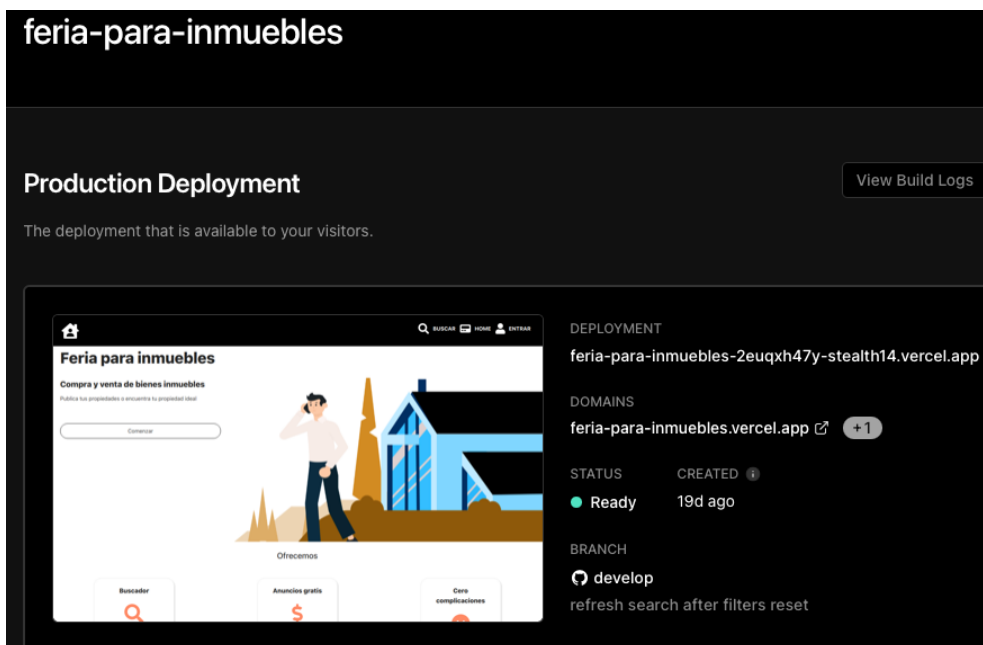


Fig. 10: Dashboard de proyecto en *Vercel*.

En la actualidad el servicio de alojamiento y despliegue es gratuito para proyectos de un integrante. Las configuraciones se hacen por completo desde la plataforma y solo es necesario asociar un repositorio público o privado. En el lado de *github* se configura automáticamente un *webhook* que envía el código fuente del proyecto a *vercel* después de recibir cambios en la rama principal del proyecto **Fig. 10**.

## Iteración 2. Codificación de pantallas: inicio de sesión y registro de usuarios

### Registro

Durante esta iteración implementó las pantallas para autenticar los usuarios. El usuario puede acceder al registro desde la página de bienvenida presionando el botón Comenzar, como se muestra en la **Fig. 11**

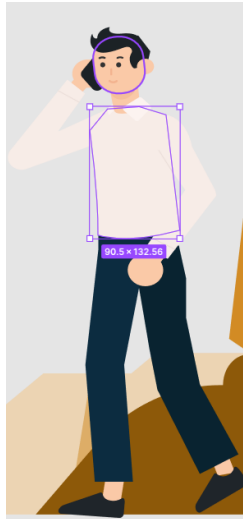


**Fig. 11: Página de bienvenida.**

Esta página es lo primero que se le presenta al usuario al ingresar. Se han cuidado los elementos de esta sección para contener imágenes temáticas y texto describiendo el servicio.

La imagen del usuario hablando por teléfono fue recuperada del repositorio público de *figma*, este elemento se modificó hasta lograr la postura con celular **Fig. 12**.





**Fig. 12: Grafico de usuario con vectores.**

En la **Fig. 13** se observa el formulario de registro con los campos de información que el usuario debe proporcionar para darse de alta.

🔍 BUSCAR 🏠 HOME 👤 ENTRAR

### Registrarse

\* Correo

\* Nombre

\* Apellido

\* Número celular

\* Contraseña

\* Confirmar contraseña

+  
Agregar

Registrarse

**Fig. 13: Página de registro.**

Esta pantalla utiliza los componentes de *antd* para construir formularios. El componente 'Form' de la librería permitió reducir el código fuente necesario para capturar todos los campos a la vez, aplicar validaciones de datos primitivos y a variables de estado **Fig. 14**.

```
const onFinish = async (user: User) => {  
  // photos validation  
  if (fileList.length < 1) {  
    alert(lang("photo_required"));  
    return;  
  }  
  
  handleLoading(true);  
  const [savedUser] = await register({  
    ...user,  
    avatar: fileList[0].originFileObj,  
  });  
  
  handleLoading(false);  
  
  if (savedUser) router.push("/login");  
};
```

**Fig. 14:** Función de *Submit* del formulario de registro.

## Inicio de sesión

Una vez registrado, el usuario es redireccionado al inicio de sesión. **Fig. 15**.



Navigation bar:  BUSCAR  HOME  ENTRAR

Iniciar Sesión

\* Correo

\* Contraseña

Entrar a mi cuenta

**Fig. 15:** Página de inicio de sesión.

### Iteración 3. Codificación de pantallas: CRUD de publicaciones.

#### Visualización

Si las credenciales ingresadas son correctas el usuario será redireccionado a la página de inicio, **Fig. 16**



**Fig. 16** Página de inicio

Como se puede observar en la **Fig. 16**, el usuario puede visualizar una lista de propiedades que ha publicado con anterioridad. Usuarios nuevos verán el mensaje “*No hay propiedades*” en lugar de la lista. Adicionalmente este listado se actualiza automáticamente con la estrategia de refresco de datos *SWR*.

*SWR* aprovecha la caché del navegador y realiza peticiones al servidor solo cuando detecta cambios en el listado.

SWR fue implementado con el patrón definido por la librería para *React* de *SWR*, un *hook* que encapsula los estados de error y petición exitosa **Fig. 17**.

```
export function useProperties() {    You, 9 months ago • path aliases,
  const fetcher = (url: string) => api.get(url).then((res) => res.data);

  const { data, error } = useSWR(
    process.env.NEXT_PUBLIC_API_BASE_URL + "properties",
    fetcher
  );

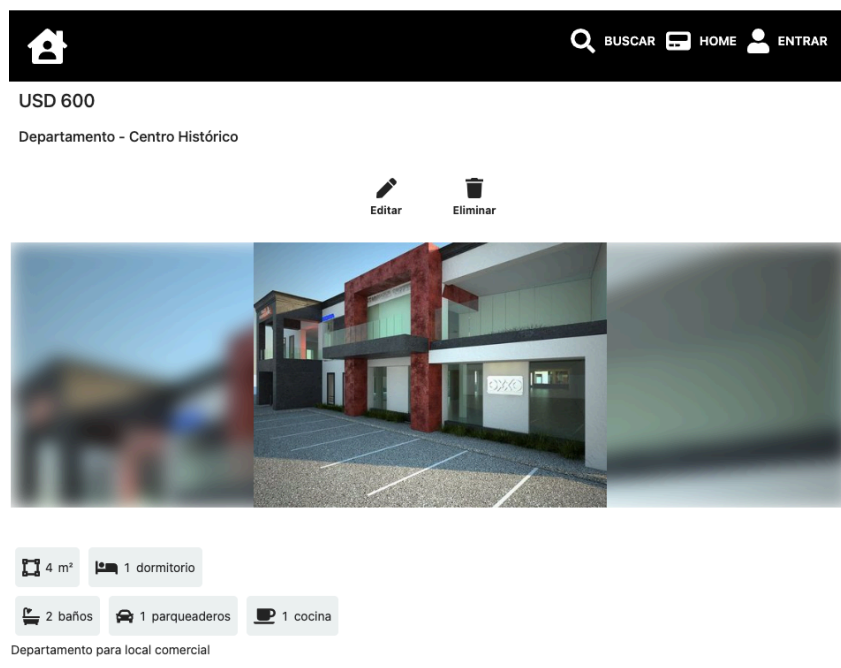
  return {
    properties: data as Property[],
    isLoading: !error && !data,
    isError: error,
  };
}
```

**Fig. 17 Hook SWR para listado de propiedades**

Esta implementación permite que el usuario tenga una lista actualizada de sus anuncios en todo momento.

### Edición y eliminación

Cada tarjeta del listado tiene un botón con icono de flecha. Al presionar el botón se abre una página con información detallada del inmueble como se muestra en la **Fig. 17**



**Fig. 18 Vista detallada del inmueble**

```

export const drop = async (property: Property) => {
  const path = `properties/${property.id}`;

  try {
    const response = await api.delete(
      process.env.NEXT_PUBLIC_API_BASE_URL + String(path)
    );

    const property: Property = response.data;

    return [property, null];
  } catch (error: any) {
    return [null, error];
  }
};

```

**Fig. 19** Función para eliminar propiedad

Se han colocado los botones *Editar* y *Eliminar* en la página de la **Fig. 18** en lugar de ubicarlos directamente en los ítems del listado, de esta manera el usuario no borrará propiedades por accidente al manipular la lista.

Para eliminar una propiedad se utiliza una función con una petición convencional. Esta función captura excepciones con un bloque de código *try catch* resolviendo la promesa en ambos escenarios, éxito o error. Este patrón de función asegura un valor de retorno predecible. Todas las peticiones que no requieren un refresco de datos en tiempo real han sido implementadas como describe la **Fig. 19**

## Edición

El botón editar despliega el formulario de edición. Como se observa en la **Fig. 20**, este formulario precarga automáticamente los datos de la propiedad.

Por favor, ingresa la información de la propiedad

\* Título

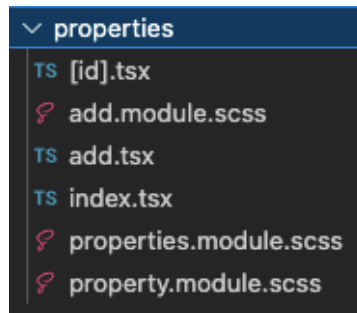
\* Descripción

**Fig. 20** Página de edición de propiedades

## Creación

Al presionar el botón *Publicar* de la **Fig. 16**. Se desplegará el formulario de la **Fig.20**, pero sin datos precargados por tratarse de una publicación nueva.

El desarrollo de esta pantalla se benefició de la planificación de componentes, puesto que se ha reutilizado por completo el código fuente de creación y actualización **Fig. 21**.



**Fig. 21** Página para añadir / editar propiedades

El formulario del componente 'add' llena automáticamente los campos del formulario cuando la variable de estado 'property' recibe objeto de tipo 'Property' válido **Fig. 22**.

```
const [property, setProperty] = useState<Property | null | undefined>(
  undefined
);
```

**Fig. 22** Página para añadir / editar propiedades

Fue necesario también el efecto secundario **Fig. 23** sincronizado con 'property' para prepoplar el componente de fotos **Fig. 24**.

```
/** Load current property photos */
useEffect(() => {
  if (!property) return;

  (async () => {
    const list: UploadFile[] = [];

    for (const photo of property.photos) {
      const blob = await resolveImage(photo as string);

      if (!blob) continue;

      list.push({
        url: photo,
        originFileObj: blob as RcFile,
      } as UploadFile);
    }

    setFileList(list);
  })();
}, [property]);
```

**Fig. 23** Efecto secundario para precargar selector de fotos.

\* Cocinas: 1 2 3 4 5

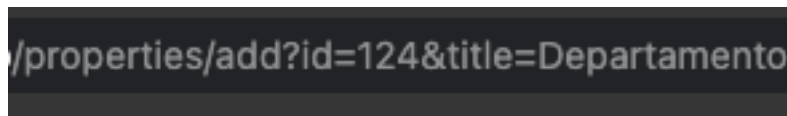
\* Parqueaderos: 1 2 3 4 5

Fotos

Five image thumbnails showing different house exteriors.

**Fig. 24** Componente selector de fotos.

De manera similar se precargan los demás campos del formulario **Fig. 27**. La diferencia con el mecanismo anterior es que en este caso los valores de los campos se extraen de los *query params* de la *url*.



**Fig. 25** Url con query params

El objeto *'query'*, propio de la librería *next* **Fig. 26** se inicializa como indefinido y después de unos instantes recibe los valores presentes en la *url*, por este motivo es necesario acceder a ellos mediante el *hook* de efectos secundarios de *React*.

```

/** Initialize uncontrolled fields*/
useEffect(() => {
  if (isReady) {
    if (Object.keys(query).length === 0) {
      setProperty(null);
      return;
    }
    // initialize uncontrolled fields
    const property = parseQuery(query) as Property;

    setProperty(property);
    form.setFieldsValue(property);
  }
}, [form, isReady, query]);

```

**Fig. 26** Efecto sincronizado con el objeto query.

El resultado es un formulario que le facilita la tarea de editar la propiedad, previniendo la tarea tediosa de llenar todos los campos de nuevo.

Por favor, ingresa la información de la propiedad

* Título :	Departamento grande
* Descripción :	zona céntrica de la ciudad
* Dirección :	12 de octubre
* Precio :	30000
* Area :	200
* Tipo de propiedad :	Terreno <input type="button" value="v"/>
* Baños :	<input type="button" value="1"/> <input type="button" value="2"/> <input type="button" value="3"/> <input type="button" value="4"/> <input type="button" value="5"/>
* Salas :	<input type="button" value="1"/> <input type="button" value="2"/> <input type="button" value="3"/> <input type="button" value="4"/> <input type="button" value="5"/>
* Dormitorios :	<input type="button" value="1"/> <input type="button" value="2"/> <input type="button" value="3"/> <input type="button" value="4"/> <input type="button" value="5"/>
* Cocinas :	<input type="button" value="1"/> <input type="button" value="2"/> <input type="button" value="3"/> <input type="button" value="4"/> <input type="button" value="5"/>

**Fig. 27 Campos precargados**

## **Iteración 4. Codificación de pantalla: Buscador de propiedades.**

### **Búsquedas**

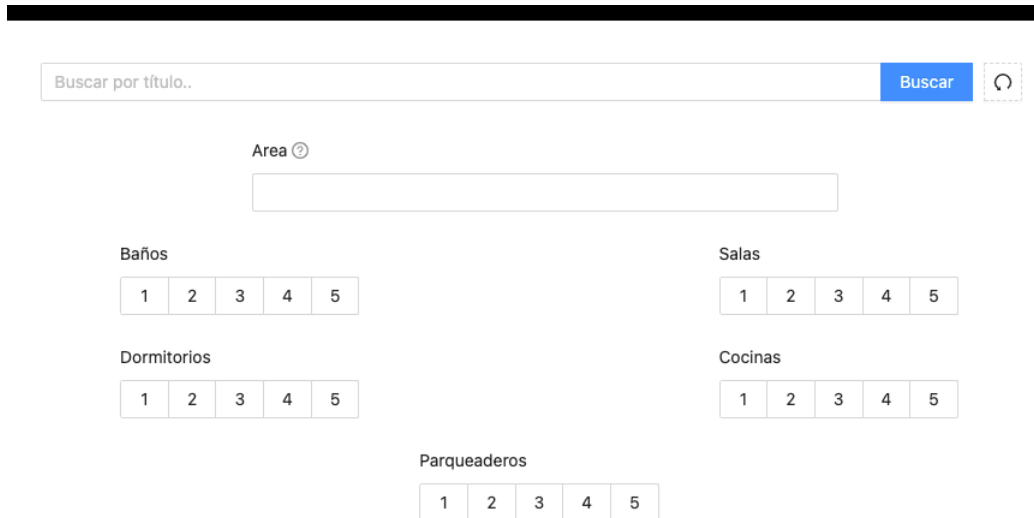
Todas las pantallas anteriores fueron necesarias para recopilar bienes inmuebles y hacerlos accesibles a los usuarios con la ayuda de una herramienta de búsquedas filtradas.

En la **Fig. 28** se observa el buscador a detalle. El campo de texto permite buscar coincidencias en títulos de publicaciones. El resto de los filtros permiten al usuario encontrar un inmueble que se adapte mejor a su presupuesto y necesidades.



## Petición de búsqueda

Los valores seleccionados mediante controles de filtrado se utilizan para construir una petición de tipo *get* con *query parameters*. Dejando como resultado la función de la **Fig. 29** que agrupa convenientemente los parámetros enviados al servidor en un objeto *'params'*.



The screenshot shows a search interface with the following elements:

- A search input field with the placeholder text "Buscar por título.." and a blue "Buscar" button.
- An "Area" label with a dropdown arrow and an empty input field below it.
- Five filter categories, each with a row of five buttons labeled 1 through 5:
  - Baños
  - Salas
  - Dormitorios
  - Cocinas
  - Parqueaderos

Fig. 28 Controles de filtrado

```
export async function search(params) {
  try {
    const url = "/search";

    const response = await api.get(url, {
      params,
    });

    const searchResult: SearchResult = response.data;

    return [searchResult, null];
  } catch (error: any) {
    return [null, error];
  }
}
```

Fig. 29 Petición de búsqueda

Una vez invocada la función 'search' Fig. 29, se asignará su valor de retorno a la variable de estado correspondiente para provocar un refresco del componente de resultados.

### Paginación de resultados

Teniendo en cuenta que los resultados son retornados con el formato de paginación establecido por *Laravel* Fig. 30, en el cliente se definió la estructura de la Fig. 31.

```
{
  "total": 50,
  "per_page": 15,
  "current_page": 1,
  "last_page": 4,
  "first_page_url": "http://laravel.app?page=1",
  "last_page_url": "http://laravel.app?page=4",
  "next_page_url": "http://laravel.app?page=2",
  "prev_page_url": null,
  "path": "http://laravel.app",
  "from": 1,
  "to": 15,
  "data": [
    {
      // Record...
    },
    {
      // Record...
    }
  ]
}
```

Fig. 30 Esquema de respuesta de *Laravel*

```
export interface SearchResult {
  current_page: number;
  data: Property[];
  total: number;
  per_page: number;
}
```

Fig. 31 Interfaz para resultados de búsqueda

Finalmente, se recogen los resultados con asignación *deestructurante* de *javascript* en constantes separadas Fig. 32 mismas que se utilizan para renderizar los controles de paginación y listado de resultados Fig. 33

```
const { data: properties, total, per_page } = searchResult as SearchResult;
```

Fig. 32 Respuesta de búsqueda

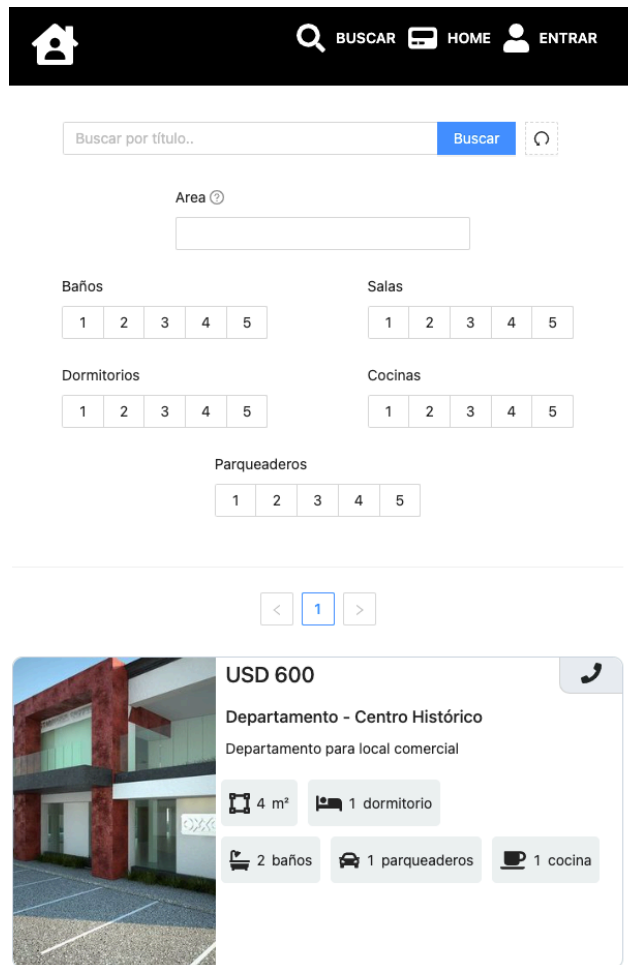


Fig. 33 Página con buscador de propiedades

## Iteración 5. Pruebas.

### Funcionales

Se realizaron también pruebas funcionales con frecuencia sobre los componentes desarrollados en búsqueda de fallos no implementados *tests* automáticos. Cada cambio en el código fuente se probó con usuarios temporales para procurar una presentación final completa y funcional del proyecto.

### Automatizadas

Con el objetivo de añadir cambios con frecuencia se implementaron pruebas automatizadas con *Cypress*.

Para eso fue necesario crear un proyecto de node desde cero e instalar *cypress* como única dependencia.

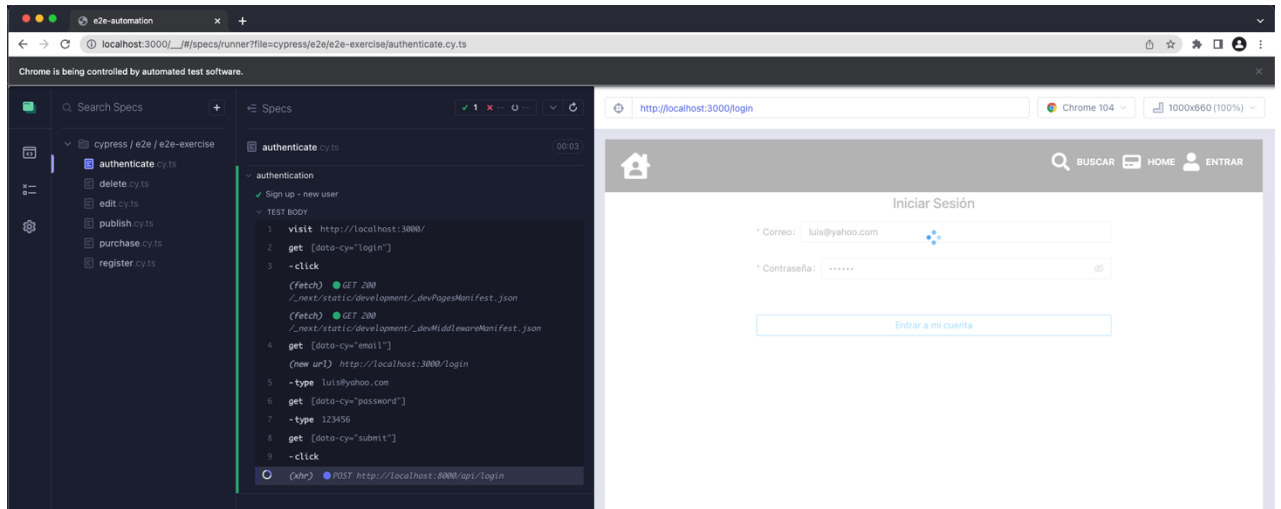


Fig. 34 Ejecución de prueba sobre el flujo de inicio de sesión

Como se observa en la **Fig. 34**, **cypress** ejecuta como una aplicación independiente con su propia instancia de navegador, en este caso Google Chrome.

Para el desarrollo de pruebas automatizadas se utilizaron selectores de *cypress* basados en selectores CSS. Estos selectores permiten ejecutar interacciones del usuario simuladas tales como *clicks* del ratón y *typeos* del teclado. Los selectores de *cypress* también permiten interceptar peticiones al servidor. Los interceptores son necesarios para pausar la ejecución de ciertas sentencias de código hasta que dichas peticiones hayan terminado de ejecutarse. Esto es indispensable para escribir pruebas verifiquen el correcto funcionamiento de un caso de uso.

## Conclusiones

- Se ha construido una aplicación con todas las funcionalidades propuestas demostrando la madurez de *React* con respecto a recursos de terceros disponibles y herramientas propias de la librería.
- Durante la implementación ha sido posible reducir errores de tipado gracias a la utilización de *Typescript*. También se ha logrado una base de código más comprensible. Estos resultados demuestran la importancia de expresar de manera explícita el tipo de dato que será alojado tanto en variables primitivas como en estructuras de datos.
- Para el funcionamiento correcto de la página fue de vital importancia agregar código para el manejo de excepciones. Al momento de consumir datos provenientes de un servicio http externo siempre habrá escenarios en los que el servidor devuelva datos incompletos o en formato diferente al esperado, por este motivo una aplicación web progresiva debe incluir código para manejar excepciones controladas y no controladas.
- Una excepción no capturada puede producir un error en el intérprete del navegador provocando que toda la aplicación deje de responder, forzando al usuario a recargar la página para volver a usar el sitio.
- Durante la carga de datos desde fuentes externas la aplicación debe dar *feedback* visual de este proceso para no dar la impresión de que el sitio dejó de responder. Esto requiere código que resuelva correctamente las promesas de petición de lo contrario la aplicación mostrará un *feedback* erróneo sobre la operación de carga generando confusión y frustración.
- Utilizar una librería reactiva permite al desarrollador despreocuparse de la lógica de renderizado para poder enfocarse por completo en de la lógica interactiva y apariencia del sitio.
- Cada cambio introducido al código fuente tiene una probabilidad de introducir errores a otras funcionalidades, por eso es importante probar constantemente el código después de introducir cambios.

## Recomendaciones

- Para estimar correctamente el esfuerzo de un desarrollo es muy importante evaluar minuciosamente las herramientas a utilizar, se deben tener en cuenta todos los elementos de esa implementación tales como datos necesarios, interacciones y apariencia de las pantallas. Por eso es recomendable disponer siempre de un prototipo interactivo y documentación sobre los servicios externos que se utilizarán.
- Para desarrollar aplicaciones del lado del cliente se deben tener bien identificados los códigos de petición errónea y de respuesta satisfactoria lo antes posible, puesto que una gran parte del código de *frontend* depende de dicha información.
- Se debe invertir esfuerzo en mantener una base de código legible y sin líneas innecesarias puesto que genera beneficios a corto y largo plazo.
- Es recomendable revisar minuciosamente el mantenimiento recibido por las dependencias que se agregan al proyecto. Añadir dependencias abandonadas producirá deuda técnica que se debe evitar de ser posible.
- Los componentes reutilizables deben definirse al principio de una implementación antes de programarse y con la ayuda de un prototipo. De esta manera se puede delimitar los casos de uso de cada componente y evitamos escribir código interdependiente, también conocido como código mal acoplado. Además, los componentes deben programarse con una buena comprensión de las opciones de composición y herencia proporcionados por las herramientas de desarrollo.
- Recordar que toda base de código va a volverse inviable de mantener eventualmente, las técnicas de desarrollo aplicadas en este proyecto solo permiten prolongar el tiempo de vida útil por el máximo tiempo posible.

## 6 REFERENCIAS BIBLIOGRÁFICAS

- [1] Heavy, 2020. [En línea]. Available: <https://www.heavy.ai/technical-glossary/server-side-rendering>.
- [2] F. O. Source, «Reactjs,» 2021. [En línea]. Available: <https://es.reactjs.org>.
- [3] Vercel, «Nextjs,» 2021. [En línea]. Available: <https://nextjs.org>.
- [4] Ekos, «El sector inmobiliario se dinamiza,» *Ekos*, 2021.
- [5] El Comercio, «Las plataformas digitales ayudan a quien busca comprar, vender o arrendar una propiedad,» *El Comercio*, 2014.
- [6] L. p. d. p. l. n. inmobiliarios. [En línea]. Available: <https://www.eltelegrafo.com.ec/noticias/economia/1/plataformasdigitales-negociosinmobiliarios-olx>.
- [7] K. Beck, *Extreme Programming Explained*, München: Addison-Wesley, 2005.
- [8] Typescript, «JavaScript With Syntax For Types.,» [En línea]. Available: <https://www.typescriptlang.org/docs/>. [Último acceso: 2 Junio 2022].
- [9] Visual Studio Code, «TypeScript in Visual Studio Code,» [En línea]. Available: <https://code.visualstudio.com/docs/languages/typescript>. [Último acceso: 4 Junio 2022].
- [10] Vercel, [En línea]. Available: <https://vercel.com/docs>. [Último acceso: 08 06 2022].
- [11] Figma, «[https://www.figma.com/community/file/831698976089873405,](https://www.figma.com/community/file/831698976089873405)» [En línea]. [Último acceso: 01 06 2022].
- [12] R. J. Anthony. [En línea]. Available: <https://www.sciencedirect.com/topics/computer-science/code-reuse>. [Último acceso: 08 06 2022].
- [13] Figma, [En línea]. Available: <https://help.figma.com/hc/en-us/articles/5579474826519-Create-and-use-component-properties>. [Último acceso: 08 06 2022].

[14] React, «Composición vs. herencia,» [En línea]. Available: <https://es.reactjs.org/docs/composition-vs-inheritance.html>. [Último acceso: 09 06 2022].

[15] Nextjs, [En línea]. Available: <https://nextjs.org/docs/routing/dynamic-routes>. [Último acceso: 01 06 2022].

[16] Ant Design, «Ant Design of React,» [En línea]. Available: <https://ant.design/docs/react/introduce>. [Último acceso: 01 06 2022].

## **7 ANEXOS**

ANEXO I. Porcentaje de plagio





**ESCUELA POLITÉCNICA NACIONAL  
ESCUELA DE FORMACIÓN DE TECNÓLOGOS  
CAMPUS POLITÉCNICO "ING. JOSÉ RUBÉN ORELLANA"**

**CERTIFICADO DE ORIGINALIDAD**

Quito, D.M. 10 de septiembre de 2022

De mi consideración:

Yo, Juan Pablo Zaldumbide Proaño, en calidad de Director del Trabajo de Integración Curricular titulado DESARROLLO FRONT-END PARA LA APLICACIÓN WEB DE COMPRAVENTA DE BIENES INMUEBLES asociado al proyecto DESARROLLO DE APLICACIÓN WEB PARA COMPRAVENTA DE BIENES INMUEBLES elaborado por el estudiante Ronny Patricio Cajas Benitez de la carrera en Tecnología Superior en Desarrollo de Software, certifico que he empleado la herramienta Turnitin para la revisión de originalidad del documento escrito completo, producto del Trabajo de Integración Curricular indicado.

El documento escrito tiene un índice de similitud del 9%.

Es todo cuanto puedo certificar en honor a la verdad, pudiendo el interesado hacer uso del presente documento para los trámites de titulación.

NOTA: Se adjunta el informe generado por la herramienta Turnitin.

Atentamente,

**Ing. Juan Pablo Zaldumbide Proaño  
Director  
ESFOT**

## ANEXO II. Manual técnico

### HISTORIAS DE USUARIO

Tabla I a la Tabla VII contienen las historias de usuario para esta aplicación.

**TABLA I Historia de usuario #1**

HISTORIA DE USUARIO	
<b>Identificador (ID):</b> HU001	<b>Usuario:</b> Vendedor / Comprador
<b>Nombre Historia:</b> Iniciar Sesión	
<b>Prioridad en Negocio:</b> Alta	<b>Riesgo en Desarrollo:</b> Alto
<b>Iteración Asignada:</b> 2	
<b>Responsable:</b> Ronny Cajas	
<b>Descripción:</b> El usuario llena los campos:  Correo Contraseña  Presiona el botón "Entrar a mi cuenta"  El sistema valida las credenciales  El sistema redirecciona al usuario a la página de inicio	
<b>Observaciones:</b> Mostrar alerta de excepción en caso de que falle la petición	

**TABLA II Historia de usuario #2**

<b>HISTORIA DE USUARIO</b>	
<b>Identificador (ID):</b> HU002	<b>Usuario:</b> Vendedor / Comprador
<b>Nombre Historia:</b> Registrarse	
<b>Prioridad en Negocio:</b> Alta	<b>Riesgo en Desarrollo:</b> Medio
<b>Iteración Asignada:</b> 2	
<b>Responsable:</b> Ronny Cajas	
<p><b>Descripción:</b></p> <p>El usuario:</p> <p>Accede a la página de bienvenida            Presiona el botón 'Comenzar'</p> <p>Llena los campos:</p> <p>Correo            Nombre            Apellido            Numero celular            Contraseña            Confirmar contraseña</p> <p>Agrega una foto. (opcional)            Presiona el botón "Registrarse"            Es redireccionado a la página de <b>Inicio de sesión.</b></p>	
<p><b>Observaciones:</b></p> <p>Los campos del formulario deben ir validados</p>	

**TABLA III Historia de usuario #3**

<b>HISTORIA DE USUARIO</b>	
<b>Identificador (ID):</b> HU003	<b>Usuario:</b> Vendedor / Comprador
<b>Nombre Historia:</b> Publicar propiedad	
<b>Prioridad en Negocio:</b> Alta	<b>Riesgo en Desarrollo:</b> Bajo
<b>Iteración Asignada:</b> 3	
<b>Responsable:</b> Ronny Cajas	
<p><b>Descripción:</b></p> <p>El usuario podrá ingresar los detalles de una propiedad en un formulario con los siguientes campos:</p> <ul style="list-style-type: none"> <li>Título de publicación</li> <li>Número de baños</li> <li>Número de dormitorios</li> <li>Número de salas</li> <li>Número de cocinas</li> <li>Número de parqueaderos</li> <li>Hasta 4 fotografías</li> </ul> <p>El botón para confirmar la edición tendrá el mensaje "Publicar".</p>	
<p><b>Observaciones:</b></p> <p>Después de que el usuario presione el botón "Publicar", un componente de carga deberá ocupar la pantalla e impedir que el usuario interactúe con el formulario hasta que la petición al servidor se complete.</p>	

**TABLA IV Historia de usuario #4**

<b>HISTORIA DE USUARIO</b>	
<b>Identificador (ID):</b> HU004	<b>Usuario:</b> Vendedor / Comprador
<b>Nombre Historia:</b> Editar propiedad	
<b>Prioridad en Negocio:</b> Alta	<b>Riesgo en Desarrollo:</b> Bajo
<b>Iteración Asignada:</b> 3	
<b>Responsable:</b> Ronny Cajas	

<p><b>Descripción:</b></p> <p>El usuario podrá seleccionar una de sus propiedades previamente creadas con el formulario de la Historia HU003. Al momento de abrir esta pantalla, los campos se llenarán automáticamente con la información actual de la propiedad.</p> <p>El botón para confirmar la edición tendrá el mensaje “Guardar cambios”.</p> <p>Cuando el usuario accede a esta pantalla los campos deben precargar la información actual de la propiedad sin una petición al servidor de por medio.</p>
<p><b>Observaciones:</b></p> <p>Después de que el usuario presione el botón “Guardar cambios”, un componente de carga deberá ocupar la pantalla e impedir que el usuario interactúe con el formulario hasta recibir una respuesta del servidor.</p>

**TABLA V Historia de usuario #5**

<b>HISTORIA DE USUARIO</b>	
<b>Identificador (ID):</b> HU005	<b>Usuario:</b> Vendedor / Comprador
<b>Nombre Historia:</b> Eliminar propiedad	
<b>Prioridad en Negocio:</b> Media	<b>Riesgo en Desarrollo:</b> Media
<b>Iteración Asignada:</b> 3	
<b>Responsable:</b> Ronny Cajas	
<p><b>Descripción:</b></p> <p>El usuario:</p> <ul style="list-style-type: none"> <li>Accede al listado de publicaciones</li> <li>Presiona el botón con icono de flecha</li> <li>Accede a la vista detallada de la propiedad</li> <li>Presiona el botón con icono de bote de basura</li> <li>El sistema elimina la publicación</li> </ul>	
<b>Observaciones:</b>	

**TABLA VI Historia de usuario #6**

HISTORIA DE USUARIO	
<b>Identificador (ID):</b> HU006	<b>Usuario:</b> Vendedor / Comprador
<b>Nombre Historia:</b> Buscar propiedades	
<b>Prioridad en Negocio:</b> Alta	<b>Riesgo en Desarrollo:</b> Baja
<b>Iteración Asignada:</b> 4	
<b>Responsable:</b> Ronny Cajas	
<b>Descripción:</b> El usuario inicia sesión. El usuario presiona el botón con icono de lupa desde la <i>navbar</i> del sitio. El usuario manipula los filtros El usuario presiona el botón buscar Se muestra un componente de carga en progreso Los resultados se despliegan	
<b>Observaciones:</b>           	

### ANEXO III

El Manual de Usuario lo puede encontrar accediendo al siguiente enlace.

<https://youtu.be/ZHTNXhHqcDA>

Se abrirá un video en el que se explican las funcionalidades de la aplicación a detalle.

Repositorio

<https://github.com/stealth14/feria-para-inmuebles>