ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE FORMACIÓN DE TECNÓLOGOS

DESARROLLO DE UN SISTEMA WEB Y APLICACIÓN MÓVIL PARA LA RESERVA DE CANCHAS Y OFERTA DE CURSOS EN LA URBANIZACIÓN LOS RETOÑOS

DESARROLLO DE UN FRONTEND

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

PAUL ISAAC GUALA BRAVO

DIRECTOR: ING. BYRON LOARTE

DMQ, febrero 2023

CERTIFICACIONES

Yo, Paul Isaac Guala Bravo 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.

PAUL ISAAC GUALA BRAVO

paul.guala@epn.edu.ec

paulgualab@gmail.com

Certifico que el presente trabajo de integración curricular fue desarrollado por Paul Isaac Guala Bravo, bajo mi supervisión.

Ing. BYRON LOARTE, MSc.

DIRECTOR

byron.loarteb@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.

Paul Isaac Guala Bravo

DEDICATORIA

A mis padres y hermanos, por todo el amor, las rabietas, la paciencia y el tiempo brindado.

A mis familiares, por el apoyo moral, las tardes deportivas y las celebraciones vividas.

A mis amigos, por su buena compañía, los sabios consejos y los tragos compartidos.

Paul Isaac Guala Bravo

AGRADECIMIENTO

A mis padres, por su apoyo, comprensión, paciencia, tiempo y sacrificio, que me han brindado durante el largo recorrido de mi vida universitaria.

A mis hermanos, por los buenos momentos, las noches de videojuegos y las experiencias que compartimos.

A mis familiares y amigos que, durante este recorrido, me han regalado un sabio consejo, un momento ameno, un abrazo, una copa y su grata compañía.

A mi tutor, el Ing. Byron Loarte, quien ha sabido guiar con sus conocimientos y experiencia el enfoque en el desarrollo de mi proyecto, sin los cuales, no hubiese culminado con éxito este trabajo.

A la Escuela Politécnica Nacional, por el acceso a la educación de calidad y las herramientas técnicas que me han permitido formarme como un profesional.

Paul Isaac Guala Bravo

ÍNDICE DE CONTENIDO

1 INTRODUCCIÓN 1
1.1 Objetivo general 2
1.2 Objetivos específicos 2
1.3 Alcance 2
1.4 Marco Teórico 4
2 METODOLOGÍA 6
2.1 Metodología de Desarrollo 6
Roles 7
Artefactos 8
2.2 Diseño de interfaces 10
Herramienta de diseño 10
2.3 Diseño de la arquitectura 12
Patrón arquitectónico 12
2.4 Herramientas de desarrollo 13
Librerías 13
3 RESULTADOS 14
3.1 *Sprint* 0. Configuración del ambiente 14
Restricciones del proyecto 14
Organización del proyecto 15
Roles de usuario 16
3.2 *Sprint* 1: Resultados obtenidos en las interfaces de los usuarios administrador y morador. 17
Iniciar y cerrar sesión 17
Registrar morador 18
3.3 *Sprint* 2: Resultados obtenidos en las interfaces del usuario administrador 21
Gestionar moradores 21
Gestionar turnos 22
Gestionar cursos 24
Enviar notificaciones 25
3.4 *Sprint 3:* Resultados obtenidos en las interfaces del usuario morador 27
Modificar los datos del perfil morador 27
Reservar un turno 28

Registrar y anular curso vacacional 29
Enviar comentarios 32
Visualizar notificaciones 33
3.5 *Sprint* 4: Pruebas y despliegue del *frontend* 34
Ejecución y resultado de las pruebas unitarias 34
Ejecución y resultado de las pruebas de compatibilidad 35
Ejecución y resultado de las pruebas de aceptación 35
Despliegue del *frontend* 36
4 Conclusiones 37
5 Recomendaciones 38
6 REFERENCIAS BIBLIOGRÁFICAS 39
7 ANEXOS 42
ANEXO I 43
ANEXO II 44
ANEXO III 87
ANEXO IV 88

RESUMEN

La Urbanización “Los Retoños”, ubicada en la parroquia de Conocoto, en la ciudad de Quito, cuenta con un total de diez canchas deportivas, las cuales son utilizadas con frecuencia por los moradores, provocando ocasionalmente aglomeraciones por el uso de una determinada cancha. Además, la urbanización no cuenta con un sistema de registro de asistencia a las canchas, ya que el control y organización de otros documentos se lo realiza mayoritariamente mediante medios impresos y en pocas ocasiones mediante medios ofimáticos.

En la actualidad, la implementación de un sistema *web* para la gestión de la información y procesos internos de una empresa presenta múltiples beneficios que están relacionados con el manejo de la información así como con el acceso a la misma por parte de los usuarios finales. Considerando los aspectos mencionados anteriormente, el presente trabajo de Integración Curricular muestra el desarrollo de un *frontend* denominado *Sportfield*, que permite a los moradores, por una parte, la reserva de un turno para hacer uso de las canchas deportivas, y por otra, el registro en cursos vacacionales ofertados en la urbanización. Además, permite el envío de notificaciones y comentarios, y que toda la organización interna de la información de la urbanización sea gestionada por parte de un perfil administrador.

El actual proyecto de integración se encuentra estructurado de la siguiente forma: en primer lugar, se establece la problemática existente en la urbanización, lo que da paso a definir y desarrollar los objetivos, el alcance y el marco teórico; en segundo lugar, se describe la implementación de la metodología aplicada en el proyecto, se detallan las herramientas que se han utilizado en los procesos de diseño de interfaces, codificación, pruebas, despliegue y el patrón arquitectónico que se implementado; en tercer lugar, se desglosa cada una de las tareas que se han establecido en cada uno de los módulos de codificación del proyecto, así como los resultados que se han obtenido en cada *Sprint* y su respectiva prueba, para la verificación de su correcto funcionamiento. Finalmente, se presentan las conclusiones y recomendaciones que se han generado en todas las etapas del desarrollo del proyecto.

PALABRAS CLAVE: *Frontend,* urbanización*,* gestión de turnos, canchas deportivas, Firebase.

ABSTRACT

The urbanization "Los Retoños", located in the parish of Conocoto, in the city of Quito, has a total of ten sports fields, which are frequently used by the residents, causing occasional crowds for the use of a particular field. In addition, the urbanization does not have a system for registering attendance at the courts, since the control and organization of other documents is mostly done by printed means and rarely by office automation.

Currently, the implementation of a web system for the management of information and internal processes of a company presents multiple benefits that are related to the management of information and access to it by end users. Considering the aspects mentioned above, the present work of Curricular Integration shows the development of a frontend called Sportfield, which allows residents, on the one hand, to reserve a time to use the sports courts, and on the other hand, to register in vacation courses offered in the urbanization. In addition, it allows the sending of notifications, comments and all the internal organization of the information of the urbanization to be managed by an administrator profile.

The current integration project is structured as follows: first, the existing problems in the development are established, which gives way to define and develop the objectives, scope and theoretical framework; second, the implementation of the methodology applied in the project is described, the tools that have been used in the processes of interface design, coding, testing, deployment and the architectural pattern that has been used are detailed; thirdly, each of the tasks that have been established in each of the coding modules of the project are broken down, as well as the results that have been obtained in each Sprint and their respective test for the verification of its correct operation. Finally, the conclusions and recommendations that have been generated in all the stages of the project development are presented.

KEYWORDS: Frontend, urbanization, turns management, sport fields, Firebase.

1 INTRODUCCIÓN

La urbanización “Los Retoños” se encuentra ubicada en la parroquia de Conocoto, vía el Parque Metropolitano de la Armenia, la cual dispone de un total de 10 canchas deportivas (fútbol, basquetbol, voleibol y tenis) distribuidas en dos grupos de cinco canchas, ubicadas en sus dos parques deportivos. Además, estas canchas son utilizadas por todos los moradores para actividades de recreación [1].

Por otra parte, en la urbanización no existe un sistema que permita llevar el control de asistencia a las canchas deportivas, por lo cual, ocasionalmente, se producen aglomeraciones para el uso de estas, causando inconvenientes entre los moradores que quieren hacer uso de una determinada cancha al mismo tiempo, y provocando molestias e inconformidades. Además, el control y registro que utiliza la administración de la urbanización es mediante medios impresos, anotaciones y en pocas ocasiones hacen uso de medios ofimáticos, comenta el Sr. Fernando Mora, miembro de la directiva de la urbanización. Esto dificulta la agilidad para buscar un determinado registro de manera instantánea, ya que la gestión de la información que realizan consume espacio para el almacenamiento de archivos, además de presentar otras vulnerabilidades como duplicidad de información, posibilidad de robo, pérdida o daño de los datos, entre otros.

En la actualidad, la implementación de un sistema *web* que proporcione servicios relacionados con la gestión de la información proporciona múltiples beneficios tales como: disponibilidad e integridad de los datos, información en tiempo real, acceso a funciones del sistema mediante interfaces interactivas para los usuarios finales, reutilización de módulos para la escalabilidad del producto, soporte para la integración de otros servicios y aplicaciones de internet como bases de datos, API’s, *hosting* de aplicaciones, entre otros [2].

Considerando los aspectos antes mencionados sobre las ventajas de un sistema web, y con el objetivo de ofrecer una solución a la problemática que actualmente afecta la urbanización, el presente Proyecto de Integración Curricular considera el desarrollo de un *frontend* interactivo, intuitivo y funcional denominado *Sportfield*, el cual ofrece a los moradores un servicio para el agendamiento de turnos de canchas deportivas y el registro en cursos vacacionales ofertados en la urbanización. Además, los moradores tienen la posibilidad de recibir notificaciones sobre eventos de la urbanización, así como el envío de comentarios, quejas o sugerencias para que sean atendidas por el administrador. Por otra parte, el *frontend* dispone de una serie de módulos para que el usuario con el rol de administrador pueda gestionar a todos los moradores de la urbanización, aprobar o rechazar el agendamiento de turnos de canchas, así como la gestión de cursos vacacionales, la visualización de los comentarios, quejas o sugerencias de los moradores y el envío de notificaciones, logrando

con ello manejar adecuadamente la información gracias al uso de tecnologías modernas y escalables.

1.1 Objetivo general

Desarrollar un sistema web y aplicación móvil para la reserva de canchas y oferta de cursos en la urbanización Los Retoños.

1.2 Objetivos específicos

1. Determinar los requerimientos funcionales y no funcionales de la urbanización para el desarrollo del *frontend*.
2. Diseñar las interfaces de usuario según los requisitos recopilados previamente.
3. Programar el *frontend* y el consumo de métodos por parte del *backend.*
4. Realizar diversas pruebas al *frontend* para la aceptación del dueño del producto.
5. Realizar el despliegue a producción para el uso de la urbanización.

1.3 Alcance

Se considera el desarrollo de un *frontend*, también conocido como aplicación del lado del cliente, mediante el cual los moradores de la urbanización puedan interactuar de forma directa, tomando en cuenta que la información presentada sea relevante y en un formato de fácil comprensión, visualización y adaptable al tamaño de pantalla de cualquier dispositivo electrónico [3].

Algunas de las características más relevantes del *frontend* denominado *Sportfield* son la gestión de turnos para la reserva de las canchas deportivas y la oferta de cursos vacacionales. De esta forma, la interfaz del usuario administrador tiene la capacidad de reservar o finalizar turnos para la agenda de una cancha, ofertar cursos vacacionales, gestionar moradores que se encuentren registrados, visualizar comentarios emitidos por los moradores y enviar notificaciones. Adicionalmente, la interfaz del usuario morador puede registrar un turno para reservar una cancha deportiva, modificar la información personal, inscribirse en un curso vacacional ofertado, emitir un comentario a los directivos y visualizar las notificaciones enviadas por el administrador. Para lograr lo mencionado, se ha establecido un conjunto de parámetros y directrices que aseguren la calidad del producto final. En primer lugar, toda la información se gestiona por medio de un *backend* que se ha desarrollado previamente, a través del consumo de varios métodos que permitan interactuar con la información que presenta el *frontend*; en segundo lugar, se ha estableciendo una capa de modularidad para la

seguridad de los datos, una metodología de desarrollo como guía en los avances del proyecto gracias al uso de artefactos y un patrón de arquitectura para organizar el código fuente y garantizar buenas prácticas de programación; por último, se han establecido un conjunto de pruebas para verificar la calidad del *frontend* y una etapa de despliegue a producción. A continuación, se describen los perfiles que se han determinado, juntamente con sus respectivas funciones:

Perfiles que intervienen en el f*rontend:*

- Administrador y Morador

Usuario administrador

Se han consumido varios métodos con el fin de realizar las siguientes acciones:

- Presentar información del proyecto en una *landing page*.
- Iniciar y cerrar sesión.
- Gestionar usuarios.
- Gestionar turnos.
- Gestionar cursos.
- Enviar notificaciones.

Usuario morador

Se han consumido varios métodos con el fin de realizar las siguientes acciones:

- Presentar información del proyecto en una *landing page*.
- Iniciar y cerrar sesión.
- Actualizar los datos de registro del usuario morador
- Agendar un turno.
- Registrar y anular un curso vacacional.
- Enviar comentarios.

1.4 Marco Teórico

El crecimiento de las páginas *web* ha incrementado considerablemente desde la introducción de JavaScript en 1995 por parte de *Netscape*, como un lenguaje de *scripting* en el lado del cliente, permitiendo una mayor interactividad con los elementos dentro de las interfaces a los usuarios finales [4]. En años posteriores, en 2005, la propuesta de una comunicación asíncrona en la solicitud y respuesta de datos, en conjunto con JavaScript y XML, agregó una nueva capa adicional a la arquitectura tradicional de los sistemas *web*, permitiendo de esta manera la ejecución de las peticiones de datos sin la necesidad de recargar la estructura completa del documento HTML cuando se producía la respuesta por parte del servidor, es decir, sin la necesidad de renderizar de nuevo todo el contenido de la página web. Este nuevo enfoque se denominó AJAX (siglas de *Asynchronous* JavaScript + XML), y en consolidación con la nueva versión de HTML, HTML5 (que permitía una mayor interoperabilidad y e interactividad), emergió un nuevo concepto: las *Single Page Applications* (SPA, por sus siglas en inglés) [5]. Este nuevo tipo de aplicaciones *web* permiten renderizar una única página HTML en conjunto con los recursos de JavaScript y CSS necesarios, modificando únicamente las partes de la interfaz que hayan actualizado su estado y sin la necesidad de enviar o recibir la petición de una nueva página completa desde el servidor [6]. De esta forma, este nuevo tipo de aplicación permite un mayor énfasis en la lógica de programación como la validación de campos y el renderizado de los componentes y mejoras en la experiencia de usuario al reducir los tiempos de carga y al aumentar la interacción del usuario con la interfaz [7]. Como resultado, actualmente existen múltiples *Frameworks* y librerías modernas basadas en JavaScript para el desarrollo web con enfoque en SPA, como React, Angular o Vue, por mencionar las más populares.

React es una librería desarrollada en JavaScript, la cual es declarativa y basada en componentes para la creación de interfaces de usuario [8], originalmente creada por los ingenieros de Facebook en el año 2013 para resolver los desafíos implicados al desarrollar interfaces de usuario complejas con conjuntos de datos que cambian en el tiempo y actualmente es mantenida por Meta [9]. Una de las características más importantes de *React* es su proceso de renderizado de los elementos, ya que únicamente actualiza aquellos componentes que han modificado su estado en las etapas posteriores al primer *render*, evitando así la necesidad de volver a recargar todo el contenido de la interfaz. Trabaja con un conjunto de funciones en JavaScript denominados componentes que son los distintos elementos y secciones que forman parte de la interfaz y su funcionalidad en su totalidad. Además, para mostrar el contenido a mostrarse en la interfaz utiliza JSX (por su nombre en inglés *JavaScript Syntax Extension*), el cual permite estructurar el contenido a renderizarse utilizando una sintaxis equivalente al lenguaje de marcado HTML o XML. Por otra parte, facilita

el uso de funciones preestablecidas por la librería denominado *Hooks*, los cuales permiten configurar varios aspectos y procesos de un componente como la modificación del estado de una variable, la configuración de la condición para el renderizado de un elemento, rutas de navegación entre las distintas interfaces, entre otras.

CSS es un lenguaje de hojas de estilo en cascada enfocado en el diseño de interfaces [10], el cual utiliza un conjunto de propiedades y valores declarados para un tipo de selector HTML. Su expresión "en cascada" hace referencia a la prioridad (técnicamente definido como "especificidad") que el navegador da a los diferentes estilos para una aplicación [11]. CSS es el lenguaje base para varios *frameworks* y para la estilización de interfaces como *Bootstrap*, *Tailwind CSS* o *Bulma*, lo cual resalta y pone en manifiesto su robusta capacidad para desarrollar cualquier tipo de diseño sin la necesidad de requerir funcionalidades externas.

Tailwind CSS es un *framework* de CSS de primera utilidad para desarrollar rápidamente interfaces de usuario personalizadas [12]. Su funcionamiento se basa en escanear nombres de clase en todos los archivos HTML, componentes de JavaScript y cualquier otra plantilla para generar los correspondientes estilos y escribirlos en un archivo estático de CSS [13]. Por tanto, su implementación no implica aplicar estrictamente reglas de estilos ya predefinidas ya que es muy flexible y personalizable en la configuración de diseños.

Material UI es una librería de código abierto enfocada en el desarrollo con *React* creada por Material UI SAS y con implementación de Material Design de Google, el cual es parte de Material Core, un conjunto de herramientas especializadas en el diseño de interfaces y la implementación de componentes. Material UI consta de un amplio conjunto de componentes gráficos prediseñados con múltiples funciones que son de libre personalización y de fácil implementación en cualquier proyecto *web* [14].

Firebase es una plataforma *web* creada por Google que permite desarrollar aplicaciones móviles y *web* mediante el consumo de una amplia gama de servicios [15], como una base de datos en tiempo real, almacenamiento en la nube, autenticación de usuarios, *hosting* de aplicaciones, *performance*, entre otras. Además, ofrece tecnologías relacionadas con el crecimiento de un producto, como analítica de datos, mensajes en la nube, configuración remota, testeo de aplicaciones e indexación de aplicaciones, etc. [16].

Para el consumo de los servicios de Firebase es necesario, en primer lugar, crear y registrar un proyecto en la consola de Firebase, el cual dispone de una interfaz gráfica que permite activar y administrar todos los servicios que se requieran. A continuación, es importante la instalación del SDK de *Firebase* dentro del proyecto mediante un instalador de paquetes como npm (*node package manager*), e inicializar la aplicación en un archivo con la extensión del lenguaje de programación seleccionado [17].

2 METODOLOGÍA

El estudio de casos es un método empírico que investiga un evento dentro de su contexto real [18], el cual es empleando para la recolección y recopilación de información de una o varias entidades en particular, como personas, grupos u organizaciones [19]. En la ingeniería de *software*, el estudio de casos es una investigación basada en múltiples fuentes de evidencia para un determinado caso dentro de su contexto, sobre todo cuando los límites entre el evento y el contexto no pueden ser determinados con claridad [20].

Considerando lo anteriormente citado, se ha considerado la implementación de un estudio de casos en el desarrollo del *frontend* denominado *Sportfield* como parte del Proyecto de Integración Curricular actual, ya que los requerimientos que obtenidos se han establecido a través de encuestas y entrevistas a los administradores y moradores de la urbanización, así como investigaciones en otros sitios y aplicaciones *web* dedicadas a la gestión de turnos y oferta de cursos, permitiendo de esta manera, brindar a los administradores un medio tecnológico que les permita gestionar la oferta de turnos agendados y cursos vacacionales.

2.1 Metodología de Desarrollo

Como parte del área de desarrollo de *software*, una metodología es un conjunto de reglas y directrices que se utilizan en el desarrollo de un determinado producto *software*. Además, este conjunto incluye los valores y principios de la metodología, así como los roles y artefactos que se pueden utilizar en todas las etapas del proceso [21].

Los métodos ágiles en la ingeniería de *software* permiten la entrega de un producto de alta calidad y mantenibilidad, comprendiendo todas las etapas de ingeniería ,desde el diseño de la arquitectura y la gestión de proyectos hasta la mejora de procesos [22]. Además, permite la comunicación continua y efectiva entre todas las personas involucradas en el desarrollo (clientes y programadores), agilizando y dinamizando los tiempos de entrega ante los cambios que pueden suceder en el proceso.

Considerando los principales beneficios en la aplicación e implementación de una metodología ágil, el Proyecto de Integración Curricular actual se ha desarrollado mediante la implementación de *Scrum*, la cual es una de las metodologías ágiles que forman parte del desarrollo de *software* y que otorga una serie de ventajas para el proyecto tales como: planificación en las distintas etapas del proyecto, desarrollo y codificación mediante iteraciones y tareas, equipo de trabajo organizado y comprometido, entre otras. Es por esta razón, que *Scrum* faculta a las personas, grupos y organizaciones a generar valor mediante soluciones adaptativas a problemas simples o complejos. A continuación, se detalla la aplicación de esta metodología en el desarrollo del *frontend*.

Roles

El *Scrum Team* consta de tres roles fundamentales: un *Product Owner*, un *Scrum Master* y un *Development Team*. Adicionalmente, como parte de la metodología pueden existir más roles adicionales, pero *Scrum* requiere fundamentalmente de estos tres roles principales durante todo el proceso de desarrollo de un producto *software* [23]. A continuación, se presenta a las personas asignadas a cada uno de los roles.

Product Owner

Es la autoridad única y responsable en la decisión de qué características y funcionalidades se deben programar y en qué orden se lo deben hacer. Además, mantiene una comunicación con todos los miembros involucrados en la elaboración del producto [23]. En ese sentido, la persona asignada para dirigir este rol se presenta en la **TABLA I.**

Scrum Master

Es la persona que actúa como un líder, el cual ayuda a comprender y adoptar los principios, prácticas y valores de *Scrum* durante todas las etapas del desarrollo del producto [23]. Además, se encarga de resguardar al *Development Team* de impedimentos o interferencias que afecten la productividad del equipo. En ese sentido, la persona asignada para dirigir este rol se presenta en la **TABLA I.**

Development Team

Son las personas responsables del diseño, desarrollo y testeo del producto [23]. Además, tienen una organización propia para determinar la mejor forma de cumplir con las metas establecidas por el *Product Owner* y usualmente está conformado por un grupo de cinco a nueve personas. En ese sentido, la **TABLA I** indica la persona encargada destinada para cumplir con este rol**.**

TABLA I: Roles del proyecto *frontend*.

ROLES	NOMBRES
Product Owner	Sr. Fernando Mora.
Scrum Master	Ing. Byron Loarte, MSc.
Development Team	Sr. Paul Guala.

Artefactos

En *Scrum*, los artefactos representan un valor para el proyecto ya que están diseñados para aumentar la claridad de la información recolectada [24]. Por otra parte, cada artefacto asegura que la información recopilada y almacenada permita tener un mejor enfoque dentro de todo el equipo de trabajo. A continuación, se detallan los artefactos para almacenar la información recopilada por parte del dueño del producto.

Recopilación de requerimientos

Los requerimientos son una descripción de las necesidades y expectativas del cliente sobre las funcionalidades de un producto de *software* a ser implementadas y desarrolladas, además de detallar cómo el usuario final puede interactuar con el producto [25]. Este proceso se lleva a cabo mediante encuestas y entrevistas con el titular del producto, donde la información recopilada, analizada y organizada se refleja como los módulos a ser implementados en el desarrollo del *frontend*. De esta forma, la **TABLA II** expone la plantilla que se ha establecido para elaborar y receptar la Recopilación de requerimientos, concretamente el requerimiento RR10, mientras los demás requerimientos se encuentran en el **ANEXO II** del documento actual.

TABLA II: Requerimiento R001.

RECOPILACIÓN DE REQUERIMIENTOS		
TIPO DE SISTEMA	**ID - RR**	**ENUNCIADO DEL ÍTEM**
Frontend	**RR10**	El usuario con perfil morador necesita consumir varios métodos para: • Enviar comentarios al administrador

Historias de Usuario

Permiten describir las acciones y resultados que el cliente desea sin detalles técnicos de implementación [26]. Además, son utilizadas por el *Product Owner* para asegurar el desarrollo de una funcionalidad y su importancia. A continuación, se expone la plantilla que se ha establecido para elaborar cada una de las historias de usuario, como se indica en la **TABLA III,** concretamente, la historia de usuario HU03. Las demás historias se ubican en el **ANEXO II** del documento actual.

TABLA III: Registrar morador – Historias de usuario HU07

HISTORIA DE USUARIO	
Identificador: HU07	**Usuario:** Morador
Nombre historia: Registrar morador	
Prioridad en Negocio: Alta	**Riesgo en Desarrollo:** Medio
Iteración asignada: 1	
Responsable: Paul Guala	
Descripción: El morador puede registrarse en el *frontend*	
Observación: • Los datos solicitados para el registro son su nombre, apellido, número de lote, teléfono celular, correo electrónico y contraseña. • Un morador puede registrarse con un único número de lote y correo electrónico	

Product Backlog

Es una lista de prioridades sobre las funcionalidades que se desean implementar en el producto *software*. Además, provee una visión compartida y centralizada sobre qué desarrollar y en qué orden se lo debe hacer [26]. El *Product Owner*, juntamente con el *Scrum Master,* elaboran dicha, lista la cual será luego comunicada a todo el equipo de trabajo. En tal sentido, se expone la plantilla que se ha establecido para elaborar el *Product Backlog*, como muestra la **TABLA IV**, específicamente, las tareas número 8 y 12. La lista completa se encuentra en el **ANEXO II** del documento actual.

TABLA IV: *Product Backlog.*

ELABORACIÓN DEL *PRODUCT BACKLOG*				
ID – HU	**HISTORIA DE USUARIO**	**ITERACIÓN**	**ESTADO**	**PRIORIDAD**
HU08	Reservar turno	2	Finalizada	Alta
HU11	Visualizar notificaciones	3	Finalizada	Media

HU12	Editar información	4	Finalizada	Baja

Sprint Backlog

Un *Sprint* es una iteración o ciclo que define el trabajo tangible a ser entregado al usuario o cliente final, mientras que el *Sprint Backlog* es un listado de actividades que provee una estimación (usualmente en horas) del esfuerzo requerido para completar cada tarea [26]. En ese sentido, se expone la plantilla que se ha establecido para elaborar el *Sprint Backlog*, como muestra la **TABLA V,** concretamente, la iteración número 0, mientras que la lista completa de tareas por cada *Sprint* se localiza en el **ANEXO II** del documento actual.

TABLA V: *Sprint Backlog.*

ELABORACIÓN DEL *SPRINT BACKLOG*						
ID-SB	**NOMBRE**	**MÓDULO**	**ID-HU**	**HISTORIA DE USUARIO**	**TAREAS**	**TIEMPO ESTIMADO**
SB-00	Configuración del entorno	-	-	-	• Restricciones del proyecto • Organización del proyecto • Roles de usuario	10H

2.2 Diseño de interfaces

Las interfaces de usuario son un medio que permite la comunicación e interacción entre un sistema *software* y el usuario final. Por tanto, el diseño de interfaces incluye cualquier característica o función de un sistema que es visible para el usuario, analizando y comprendiendo sus necesidades y adaptando las imágenes, colores, formas y gráficos a su lenguaje y contexto [27].

Herramienta de diseño

Figma

Software de edición gráfica de vectores especializada en el diseño de prototipos de interfaces de usuario y en el diseño de experiencia de usuario para aplicaciones móviles, *web* y de escritorio [28]. Además, posee una interfaz intuitiva, explicativa y organizada sobre cada una

de las herramientas que dispone y a la vez ofrece plantillas preconfiguradas con tamaños de pantallas de diversos dispositivos móviles y de escritorio. También, ofrece un conjunto de transiciones, animaciones y efectos para la simulación de una adecuada experiencia de usuario al navegar entre las distintas interfaces de la aplicación.

En el proceso de diseño de interfaces se han establecido aproximadamente 10 interfaces para el usuario administrador y cliente, juntamente con una paleta de colores definida y una estética base. La **Fig. 1** muestra un ejemplo de la interfaz de la *Landing Page* del *frontend*, mientras que los diseños restantes se encuentran en el **ANEXO II** del presente documento.

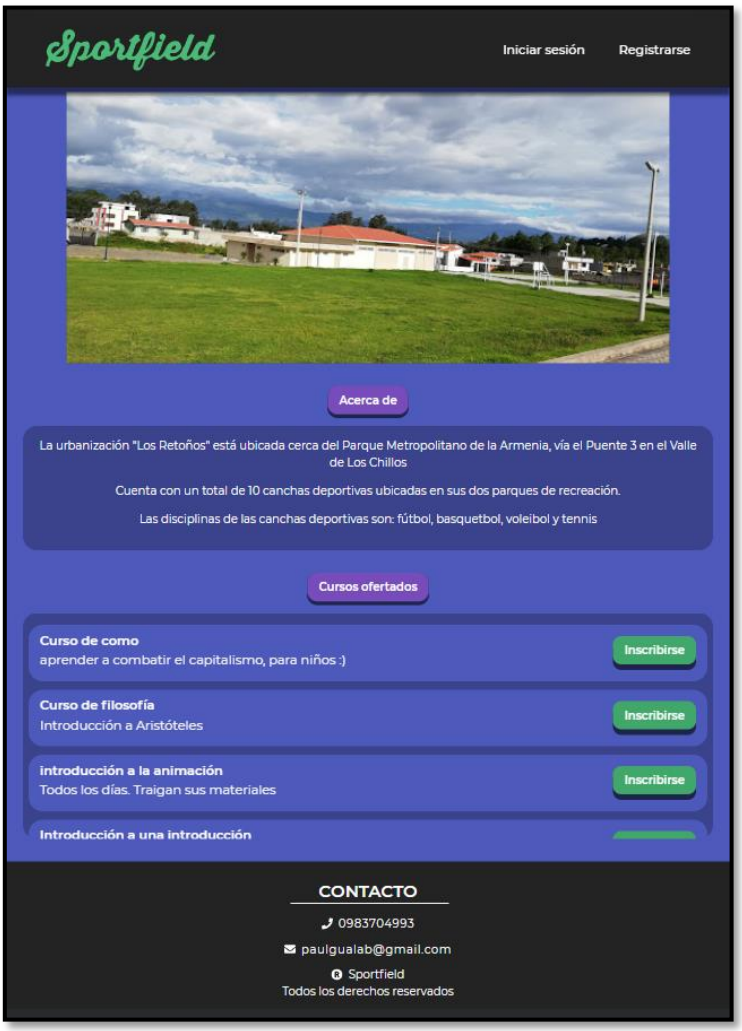

Fig. 1: *Landing Page* de *Sportfield.*

2.3 Diseño de la arquitectura

La arquitectura comprende los elementos que conforman el *software*, sus características y las relaciones entre ellos. Además, es una parte fundamental en el desarrollo sistema, ya que permite minimizar los recursos y esfuerzos para construir y mantener un *software* a futuro [29].

Patrón arquitectónico

En el diseño de *software*, el Modelo Vista Controlador es un patrón con alta aceptación y utilidad debido a su alta modularidad, flexibilidad y escalabilidad, tanto en sistemas aislados como en aplicaciones robustas. También, se basa en tres componentes principales:

- **Modelo:** almacena, encapsula y abstrae los datos e información, sin contener la lógica ni los métodos necesarios para el funcionamiento del sistema o aplicación.
- **Vista:** está encargada de presentar la información del Modelo al usuario e interpreta sus eventos, que son enviados al controlador.
- **Controlador:** contiene la lógica y funcionamiento del sistema ya que es el intermediario entre la capa del modelo y la vista.

La **Fig. 2** muestra la implementación del Modelo Vista Controlador como parte del patrón de arquitectura en el desarrollo del *frontend*, en conjunto con las herramientas y tecnologías seleccionadas para la codificación y el despliegue a producción del *frontend*.

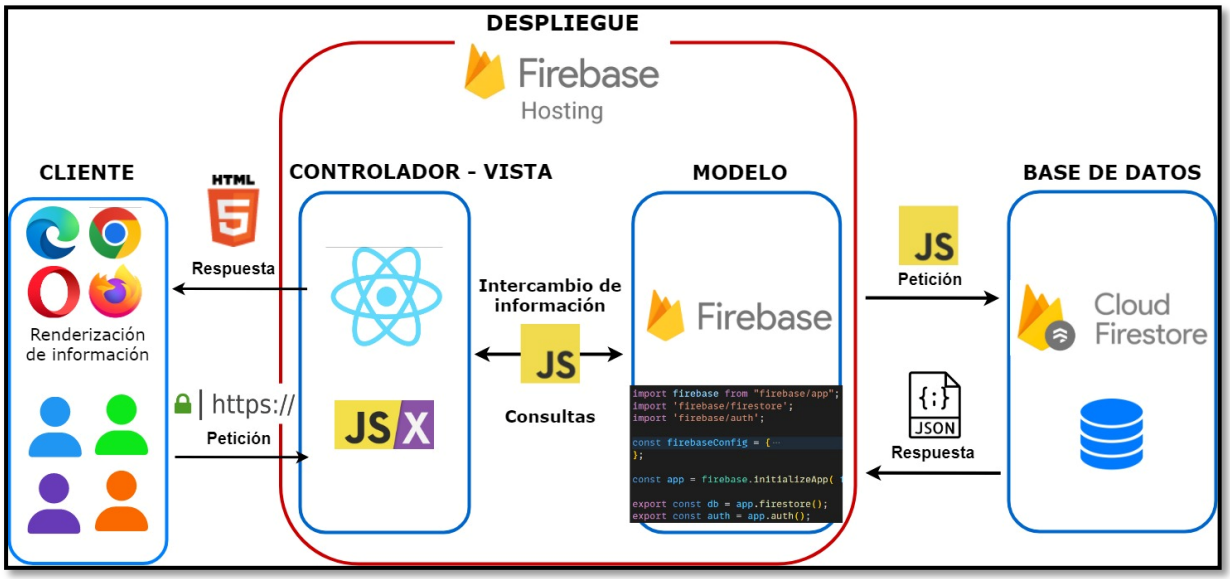

Fig. 2: Patrón arquitectónico de *Sportfield.*

2.4 Herramientas de desarrollo

Dentro del área de desarrollo de *software*, las herramientas conforman el conjunto de tecnologías enfocadas en satisfacer de manera efectiva los requerimientos del proyecto y las necesidades del equipo de desarrollo. En tal sentido, la **TABLA VI** lista el conjunto de herramientas de software designadas en el proceso de codificación del *frontend*, además de la respectiva justificación del porqué de su elección.

TABLA VI: Conjunto de herramientas designadas para la estada de codificación.

HERRAMIENTA	JUSTIFICACIÓN
React	Framework de JavaScript enfocada en el desarrollo *Frontend*, de fácil implementación, desarrollo de componentes y consumo de funciones [8].
Firebase	Plataforma en la nube que permite el consumo se servicios de almacenamiento, autenticación, hosting, entre otros, para el desarrollo de aplicaciones web [16]

Librerías

Son un conjunto de paquetes que permiten agregar funcionalidades y componentes adicionales y prediseñados al desarrollo de un producto *software*. Además, deben ser sencillas, poseer una alta compatibilidad y mantenibilidad. En ese sentido, el conjunto de librerías designadas para el desarrollo del proyecto, junto con la justificación del porqué de su implementación, se lista en la **TABLA VII**.

TABLA VII: Librerías seleccionadas para la programación del *frontend*.

HERRAMIENTA	JUSTIFICACIÓN
Tailwind CSS	Librería de CSS para el diseño de interfaces mediante clases pre establecidas y de libre personalización [12].
Material UI	Librería de componentes de React prediseñados, de fácil implementación y libre personalización [14].

3 RESULTADOS

A continuación, se ponen de manifiesto el conjunto de resultados que se han obtenido en el transcurso del desarrollo del *frontend Sportfield a* través de cada *Sprint* planificado, además de las pruebas que se han realizado y el despliegue a producción. El **ANEXO II** contiene el listado completo de los requerimientos y *Sprints* que se han establecido para el desarrollo del proyecto.

3.1 *Sprint* 0. Configuración del ambiente

Las actividades establecidas para el *Sprint* 0 son:

- Restricciones del proyecto.
- Organización del proyecto.
- Roles de usuario.

Restricciones del proyecto

Iniciar y cerrar sesión

En el *frontend*, únicamente las credenciales de acceso del usuario administrador están almacenadas en la base de datos, mientras los moradores pueden iniciar sesión con las credenciales que registren. Si la cuenta de un morador se encuentra en estado “inactivo”, el morador no puede iniciar sesión.

Registrar morador

En el *frontend*, los campos de correo electrónico y lote son únicos para el registro, es decir, no hay dos moradores registrados con un correo o lote similar.

Gestionar moradores

En el *frontend*, el usuario administrador solo puede cambiar el estado de la cuenta de un morador a “inactivo” o “activo”, mas no puede eliminar la cuenta de la base de datos.

Gestionar turnos

En el *frontend*, únicamente el usuario administrador puede finalizar los turnos agendados.

Gestionar cursos

En el *frontend*, la inscripción de un morador a un curso vacacional ofertado no puede ser anulada por el usuario administrador.

Enviar notificaciones

En el *frontend*, el usuario administrador únicamente puede enviar una notificación a todos los moradores registrados, más no puede seleccionar a qué morador desea enviar una notificación personal, ni tampoco puede eliminar o editar las notificaciones.

Modificar datos del perfil morador

En el *frontend*, el usuario morador debe ingresar su contraseña de registro para actualizar los campos de información.

Reservar un turno

En el *frontend*, los moradores únicamente pueden agendar un turno en las fechas disponibles en el calendario. Además, los moradores no pueden finalizar el estado de un turno agendado. Para ello, deben contactarse directamente con el administrador.

Registrar y anular curso vacacional

En el *frontend*, únicamente el usuario morador puede anular su inscripción a un curso vacacional. Además, no puede visualizar a los demás moradores registrados en ese curso.

Enviar comentarios

En el *frontend*, el usuario morador no puede volver a revisar los comentarios que envía al administrador.

Visualizar notificaciones

En el *frontend*, el usuario morador únicamente puede visualizar las notificaciones enviadas por el administrador, mas no puede responder a dichas notificaciones.

Organización del proyecto

Para el desarrollo y codificación de cada componente del *frontend*, la organización de las carpetas y archivos se ha realizado en el editor de código *Visual Studio Code*. En tal sentido, la **Fig. 3** muestra la organización del proyecto dentro del editor.

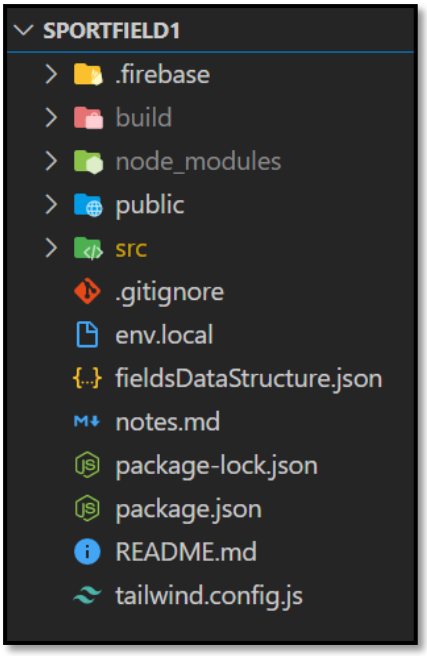

Fig. 3: Organización del *frontend Sportfield* en *Visual Studio Code*.

Roles de usuario

Son presentados en la **Fig. 4**, en conjunto con las funcionalidades que puede realizar cada uno.

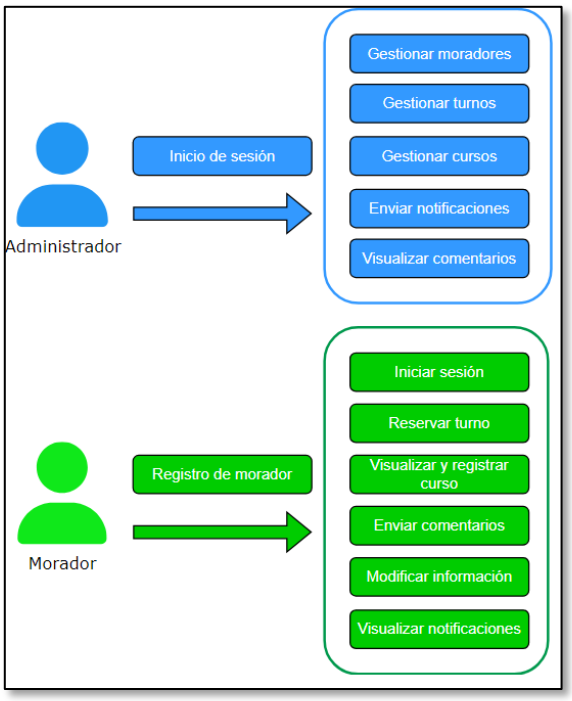

Fig. 4: Representación de los usuarios del proyecto y sus acciones

3.2 *Sprint* 1: Resultados obtenidos en las interfaces de los usuarios administrador y morador.

Dentro de actividades establecidas para el *Sprint* 1, están:

- Iniciar y cerrar sesión.
- Registrar morador.

Iniciar y cerrar sesión

Los usuarios con el rol administrador y morador pueden autenticarse ante el *frontend* mediante la interfaz de inicio de sesión, donde deben ingresar sus credenciales de correo electrónico y contraseña para otorgarles el acceso a sus respectivos módulos. Cabe recalcar que únicamente las credenciales del usuario administrador están previamente almacenadas en *Cloud Firestore*, mientras que, para el usuario morador, es necesario que se registre con las credenciales que elija. Además, esta funcionalidad se implementa mediante el consumo de métodos de *Firebase* y la información es presentada en el *frontend* como muestra la **Fig. 5**, mientras en la **Fig. 6** se presenta la ejecución de la prueba unitaria de la funcionalidad "iniciar sesión" y su resultado. Adicionalmente, la descripción completa de las funcionalidades y pruebas restantes se localizan en el **ANEXO II**.

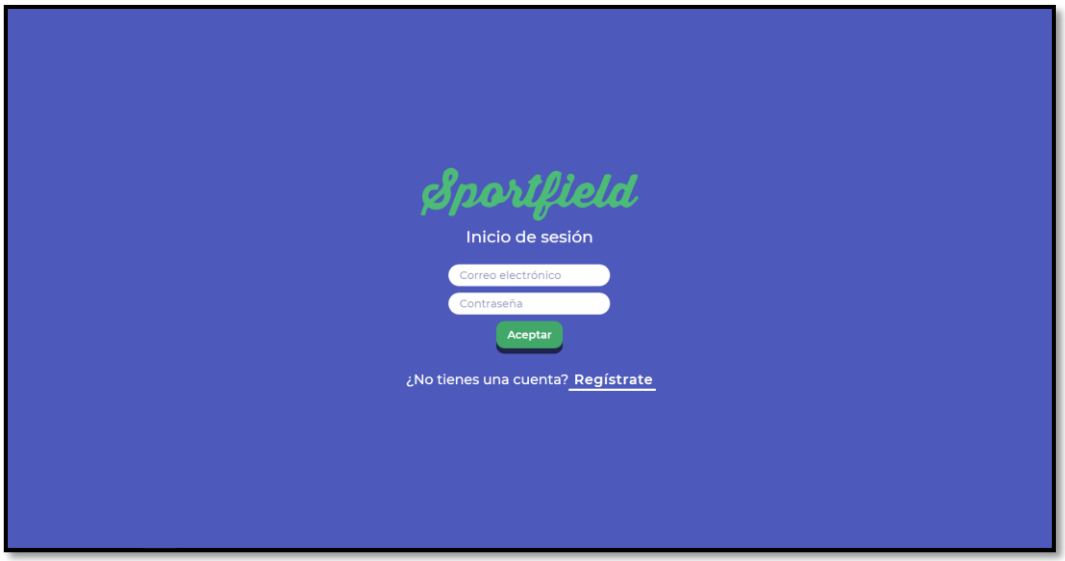

Fig. 5: Interfaz de inicio de sesión de los usuarios administrador y morador.

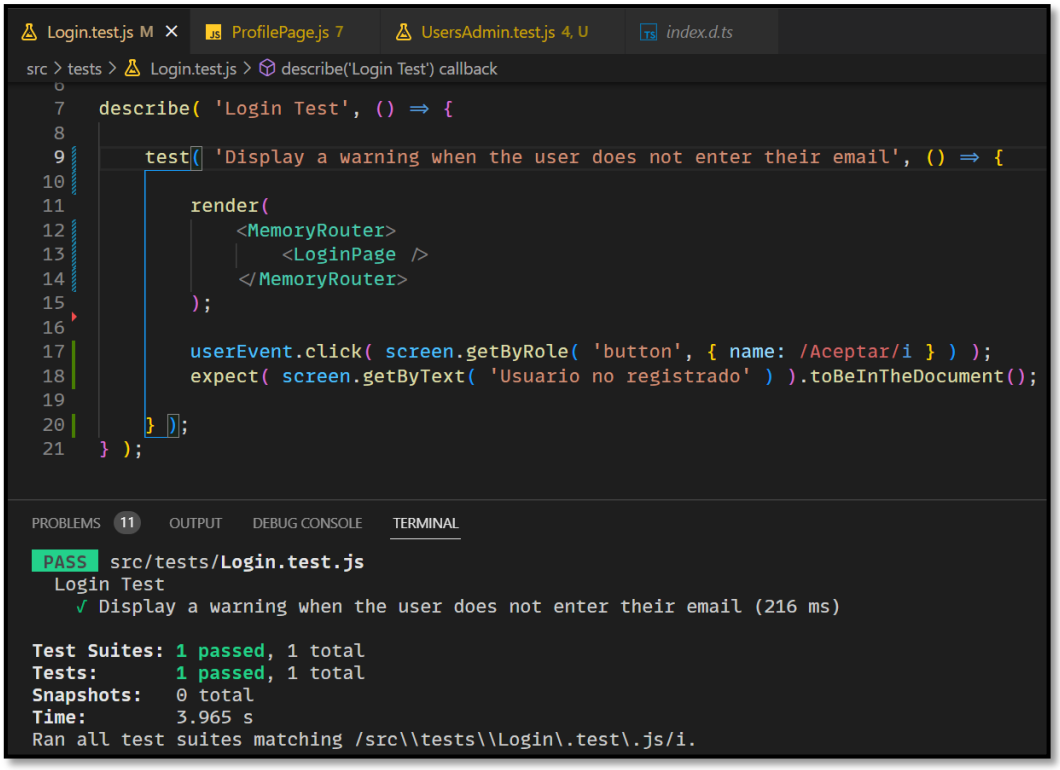

Fig. 6: Ejecución y resultados para el inicio de sesión.

Registrar morador

El registro de un morador en el *frontend* es posible tanto para el usuario administrador como para el usuario morador, llenando todos los campos solicitados para el registro como son: nombre, apellido, correo electrónico, teléfono, número de lote y una contraseña. Esta funcionalidad se implementa mediante el consumo de métodos de *Firebase*, y la información es presentada en el *frontend* como muestran las **Fig. 7** y **Fig. 8**, mientras que en las **Fig. 9** y **Fig. 10** se presenta la estructura de la prueba unitaria y el resultado obtenido tras su ejecución para la funcionalidad “registrar morador”, en conjunto para los usuarios administrador y morador. Adicionalmente, la descripción completa de las demás funcionalidades y pruebas se localizan en el **ANEXO II**.

Fig. 7: Interfaz de registro para un morador.

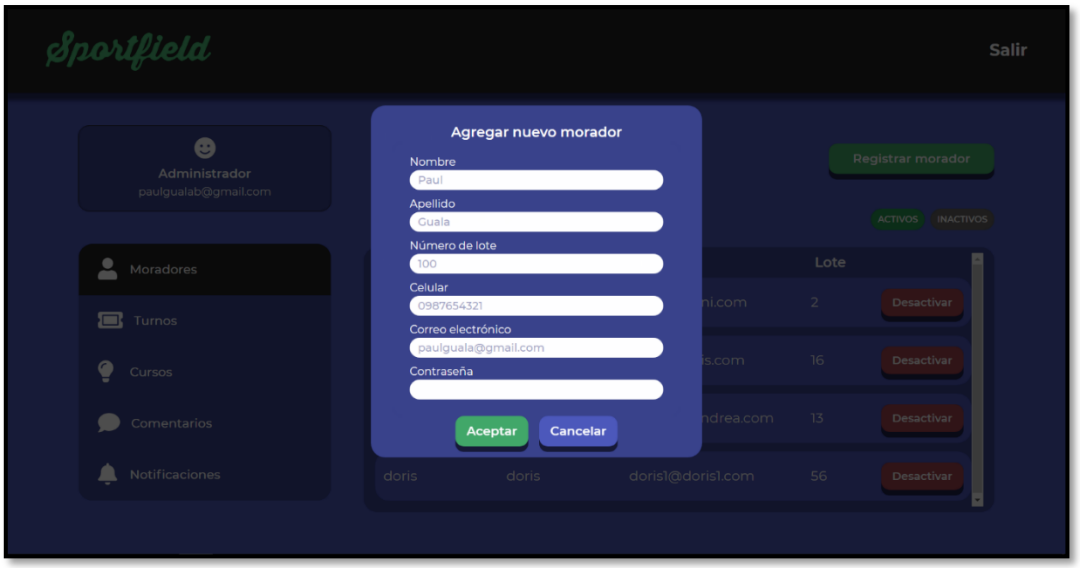

Fig. 8: Interfaz de registro del administrador.

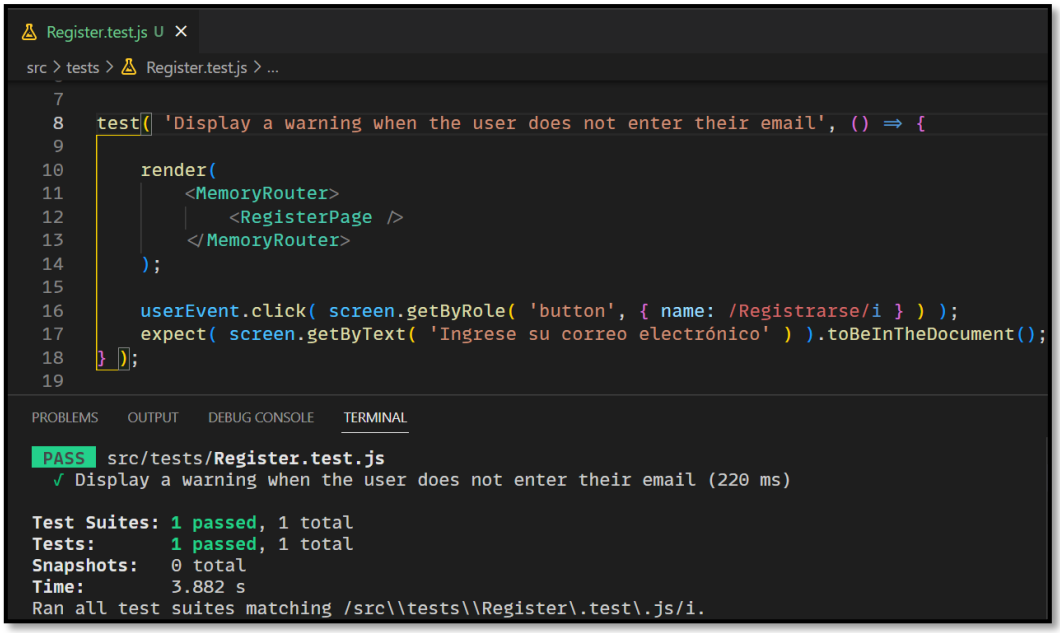

Fig. 9: Prueba para el registro del usuario morador.

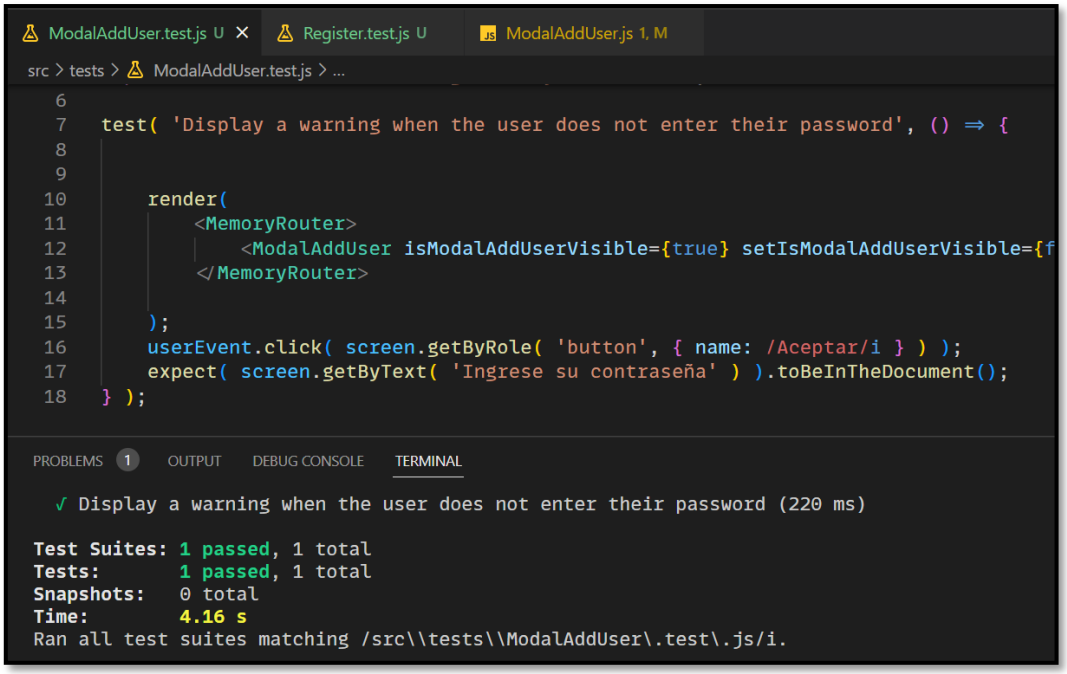

Fig. 10: Prueba para el registro del usuario administrador.

3.3 *Sprint* 2: Resultados obtenidos en las interfaces del usuario administrador

Las actividades establecidas para el *Sprint* 2 son:

- Gestionar moradores.
- Gestionar turnos.
- Gestionar cursos.
- Enviar notificaciones.

Gestionar moradores

El usuario administrador puede cambiar el estado de la cuenta de los moradores registrados entre “activo” e “inactivo”. Cuando el estado de la cuenta de un morador es “activo” puede autenticarse en el *frontend* e ingresar a su perfil con las credenciales respectivas (correo electrónico y contraseña). Por otra parte, cuando el estado de la cuenta de un morador es “inactivo” no puede autenticarse en el *frontend* y por tanto no puede ingresar a su perfil. Esta funcionalidad se implementa mediante el consumo de métodos de *Firebase* y la información es presentada en el *frontend* como muestra la **Fig. 11,** mientras en la **Fig. 12** se presenta la estructura de la prueba unitaria y el resultado obtenido tras su ejecución para la funcionalidad “desactivar morador”. Adicionalmente, la descripción completa de las funcionalidades y pruebas restantes se localizan en el **ANEXO II.**

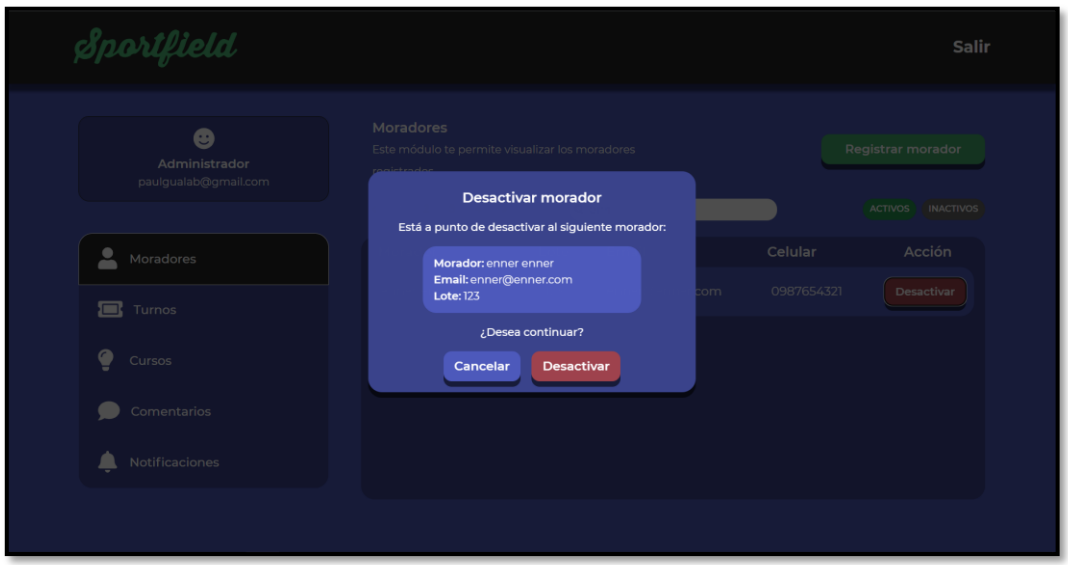

Fig. 11: Interfaz para desactivar la cuenta de un morador.

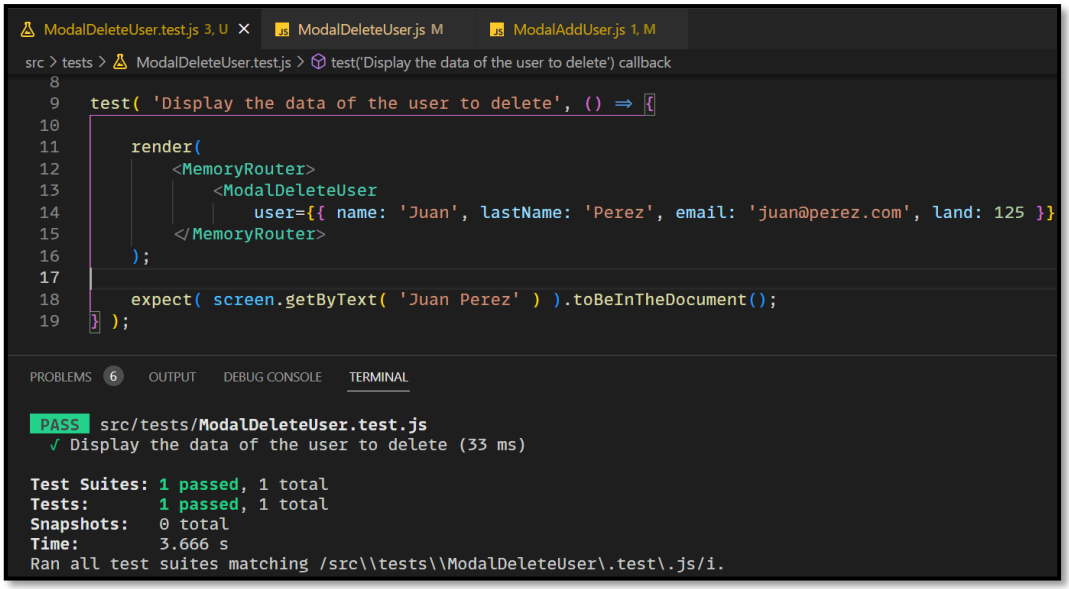

Fig. 12: Prueba para desactivar un morador.

Gestionar turnos

El usuario administrador puede reservar un turno para los moradores que se encuentren registrados o finalizar el estado de un turno ya reservado. Además, cuando el administrador finaliza el estado de un turno, este puede ser reservado de nuevo en el horario y cancha seleccionada si aún no ha pasado la fecha límite de agendamiento en el calendario. Esta funcionalidad se implementa mediante el consumo de métodos de *Firebase* y la información es presentada en el *frontend* como muestran las **Fig. 13** y **Fig. 14**, mientras en las **Fig. 15** y **Fig. 16** se presentan la estructura de la prueba unitaria y el resultado obtenido tras su ejecución para las funcionalidades “Reservar turno” y “Finalizar turno”, respectivamente. Por último, la descripción completa de las demás funcionalidades y pruebas se localizan en el **ANEXO II**.

Fig. 13: Interfaz del administrador para agendar un turno.

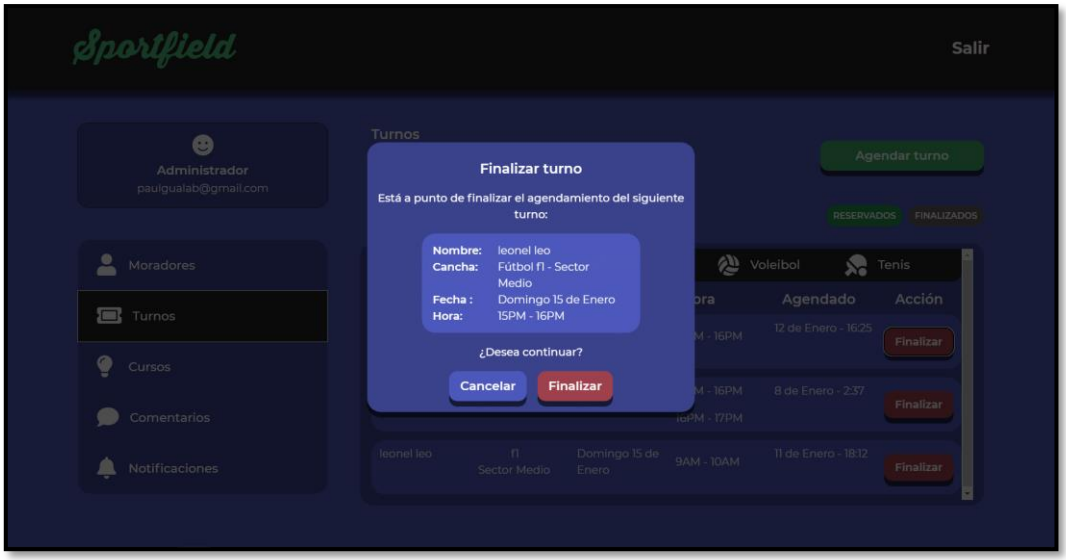

Fig. 14: Interfaz para finalizar un turno

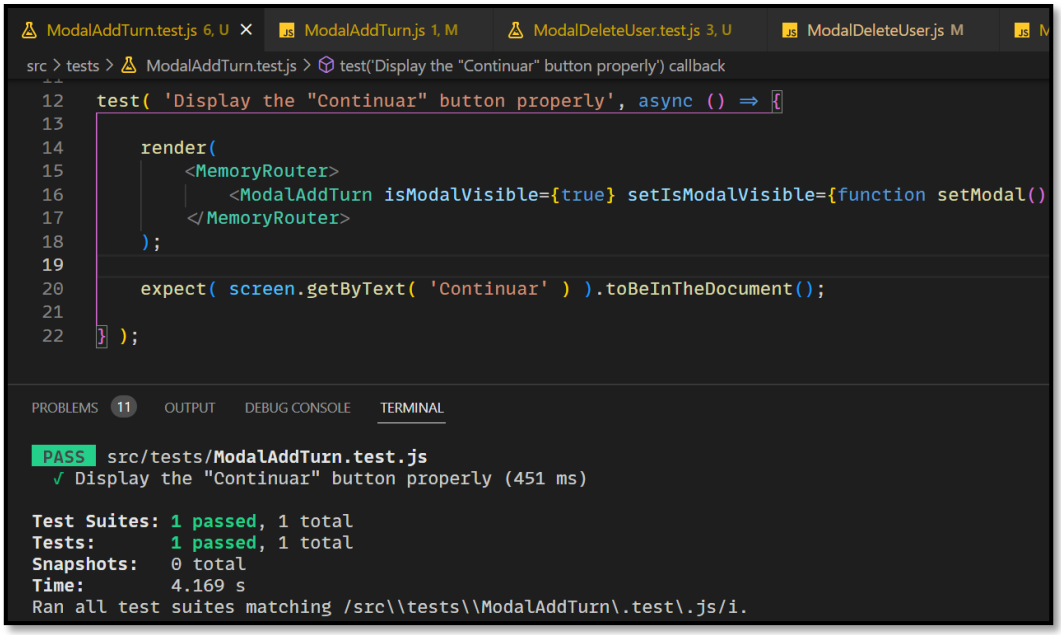

Fig. 15: Prueba para seleccionar al morador al que se le agendará un turno.

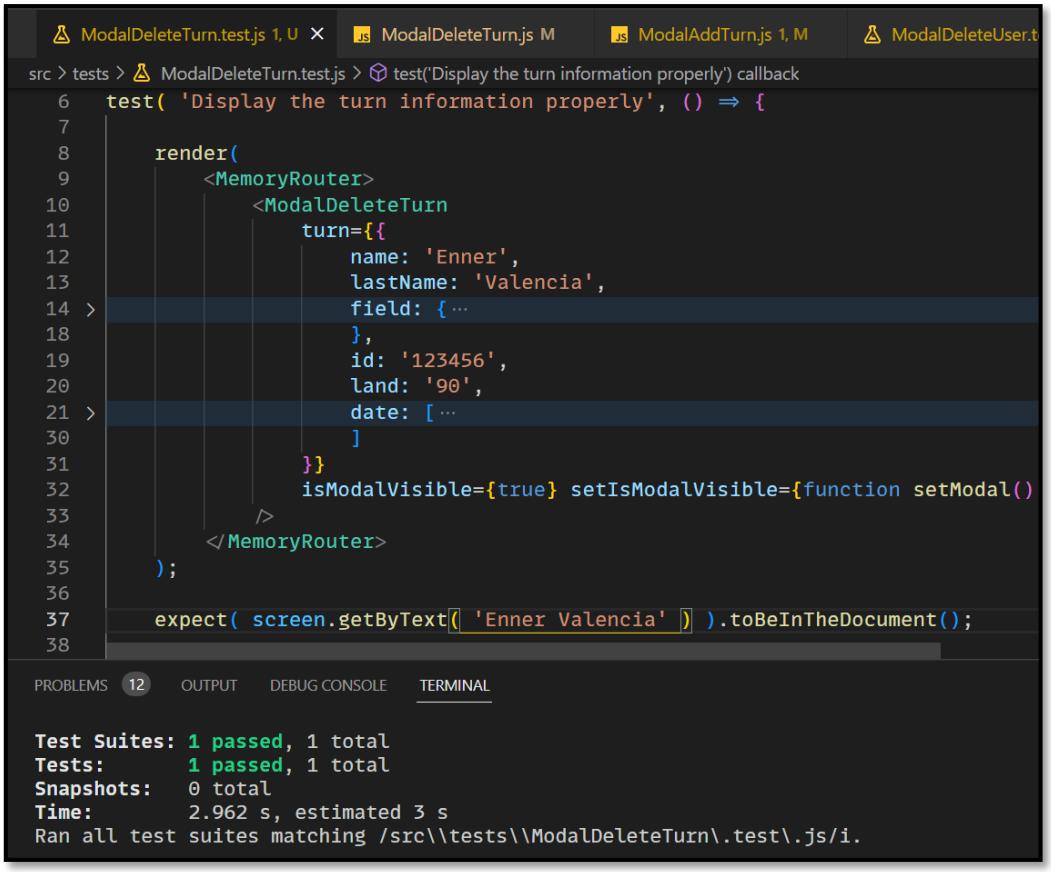

Fig. 16: Prueba para finalizar un turno.

Gestionar cursos

El usuario administrador puede ofertar un nuevo curso vacacional especificando su título, la fecha de inicio, la fecha de finalización, el horario, el costo y su descripción. Además, los nuevos cursos creados son ofertados en la *Landing page* del *frontend* y en el perfil de los moradores. Por otra parte, el administrador puede editar alguno de los campos establecidos para un curso ya ofertado. También, puede cambiar el estado de un curso vacacional a "finalizado", dando por terminado la inscripción en ese curso y su oferta en la *Landing page*. Adicionalmente, el administrador puede visualizar el número de moradores inscritos en un curso vacacional, así como sus nombres, correos electrónicos y teléfonos. Esta funcionalidad se implementa mediante el consumo de métodos de *Firebase* y la información es presentada en el *frontend* como muestra la **Fig. 17**, y en la **Fig. 18** se presenta la estructura de la prueba unitaria y el resultado obtenido tras su ejecución para la funcionalidad "Crear curso". Adicionalmente, la descripción completa de las demás funcionalidades y pruebas se localizan en el **ANEXO II.**

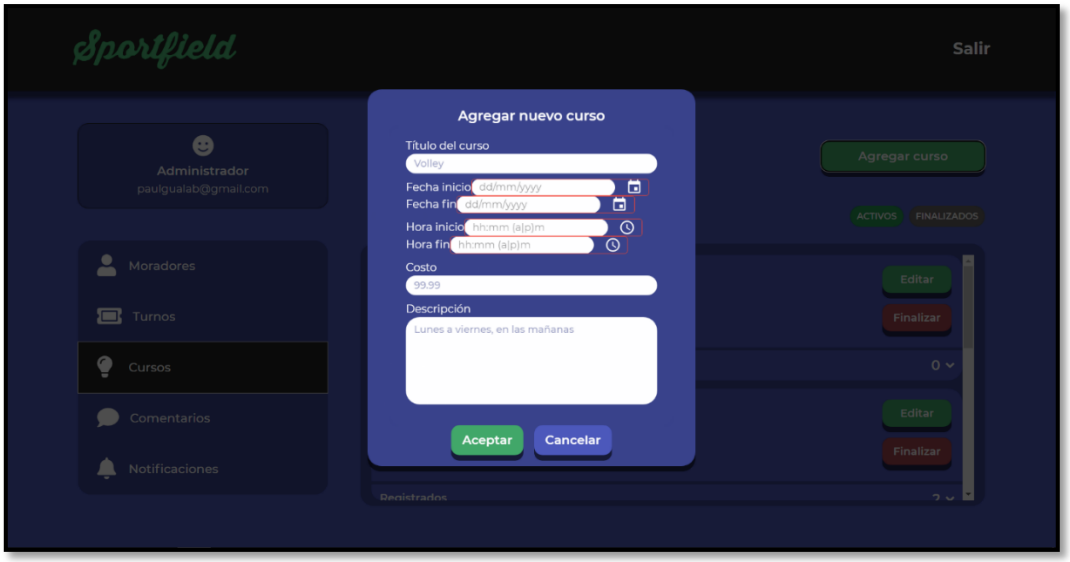

Fig. 17: Interfaz para la funcionalidad “crear un curso”.

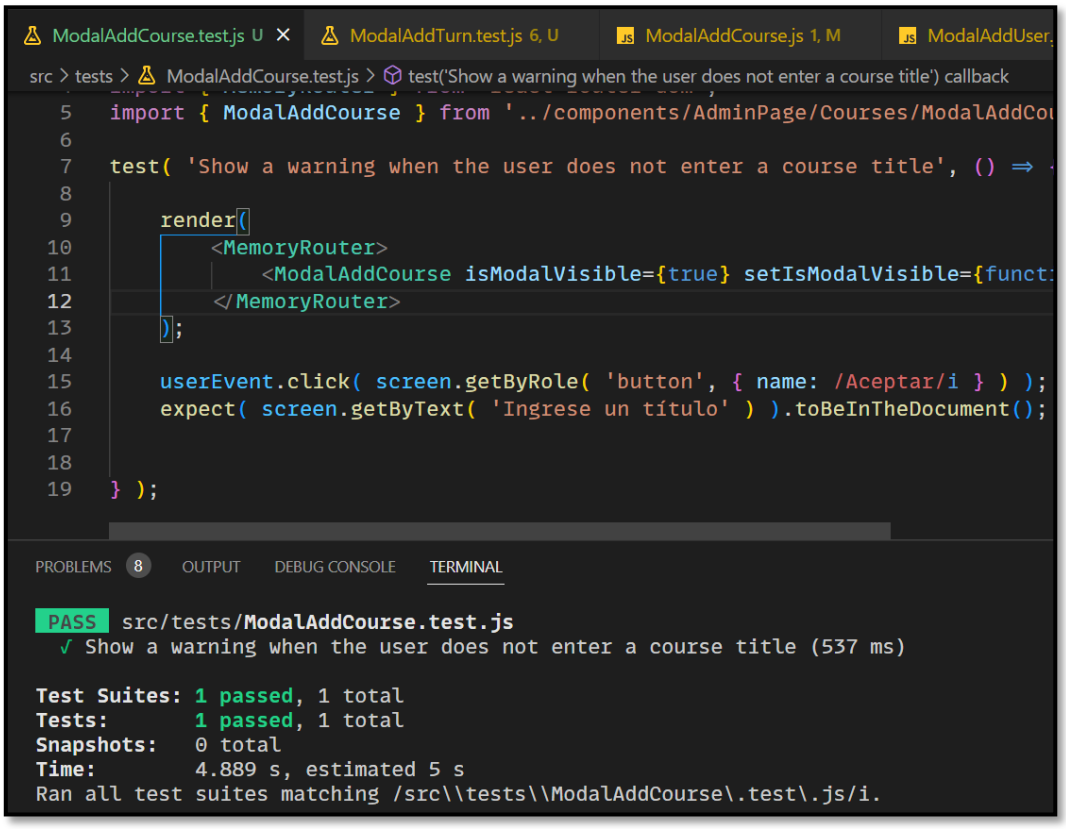

Fig. 18: Prueba de la funcionalidad “crear un curso”.

Enviar notificaciones

El usuario administrador puede enviar notificaciones a todos los moradores que se encuentren registrados, especificando un título y una descripción. Esta funcionalidad se implementa mediante el consumo de métodos de *Firebase* y la información es presentada en el *frontend*

como muestra la **Fig. 19**, mientras en la **Fig. 20** se presenta la estructura de la prueba unitaria y el resultado obtenido tras su ejecución para la funcionalidad “Enviar notificación”. Adicionalmente, la descripción completa de las funcionalidades y pruebas restantes se localizan en el **ANEXO II**.

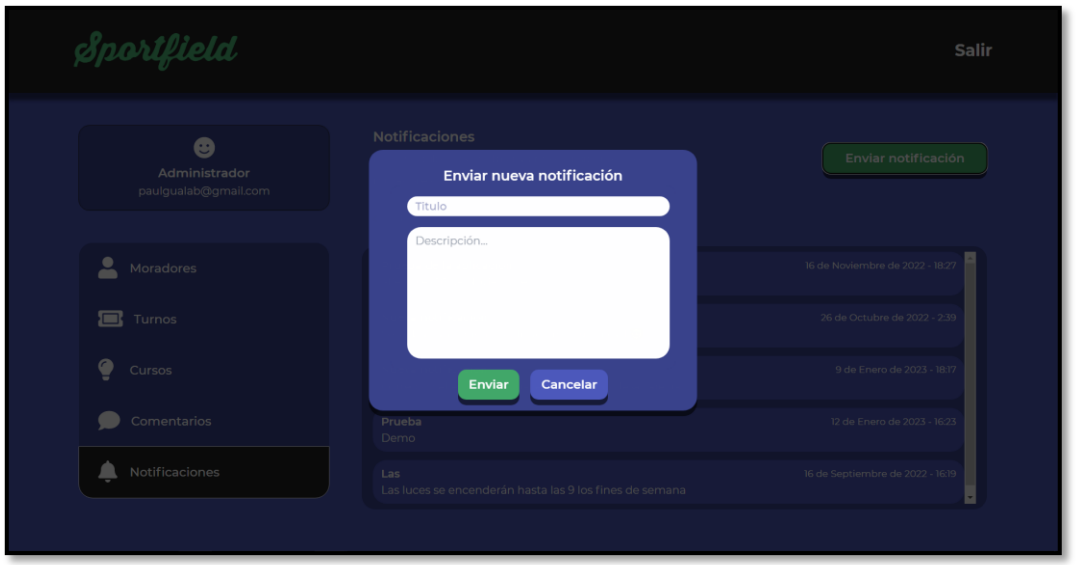

Fig. 19: Interfaz para enviar una notificación.

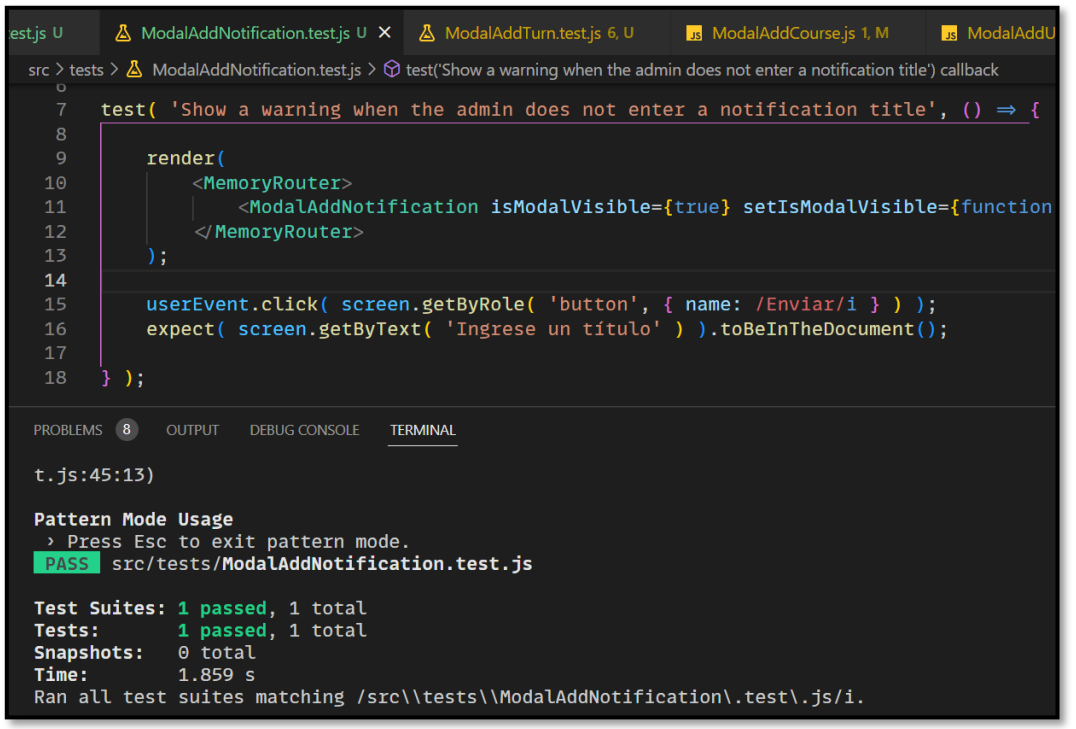

Fig. 20: Prueba para enviar notificaciones.

3.4 *Sprint 3:* Resultados obtenidos en las interfaces del usuario morador

Las actividades establecidas para el *Sprint* 3 son:

- Modificar datos del perfil morador.
- Reservar un turno.
- Registrar y anular curso vacacional.
- Enviar comentarios.
- Visualizar Notificaciones

Modificar los datos del perfil morador

El usuario morador puede modificar su información personal como: nombre, apellido, correo electrónico y teléfono. Además, para concluir el proceso y registrar la nueva información, el usuario morador debe ingresar su contraseña. Esta funcionalidad se implementa mediante el consumo de métodos de *Firebase* y la información es presentada en el *frontend* como muestra la **Fig. 21**, mientras en la **Fig. 22** se presenta la estructura de la prueba unitaria y el resultado obtenido tras su ejecución para la funcionalidad “Editar información”. Adicionalmente, la descripción completa de funcionalidades y pruebas restantes se localizan en el **ANEXO II**.

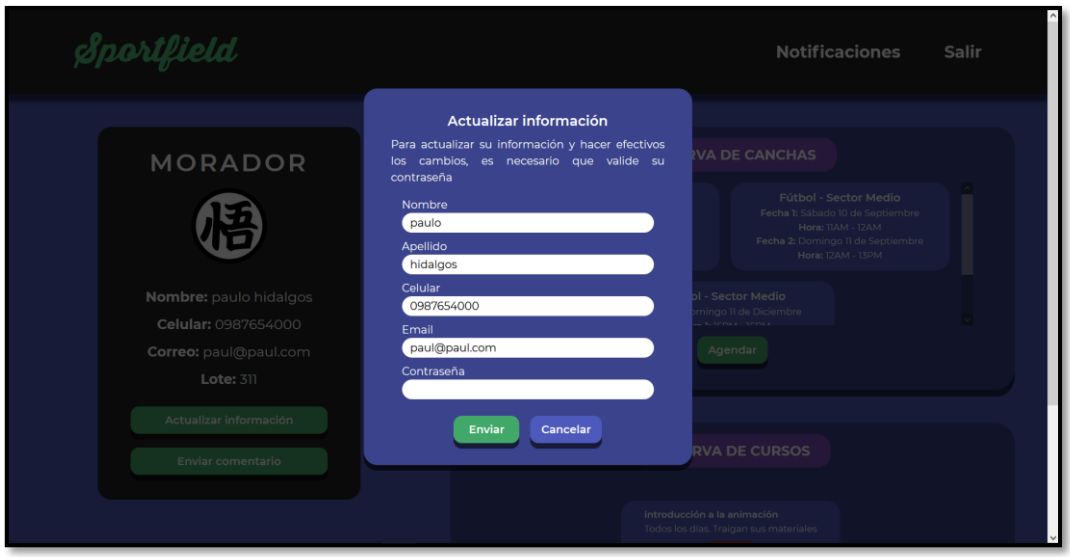

Fig. 21: Interfaz para editar la información personal del morador.

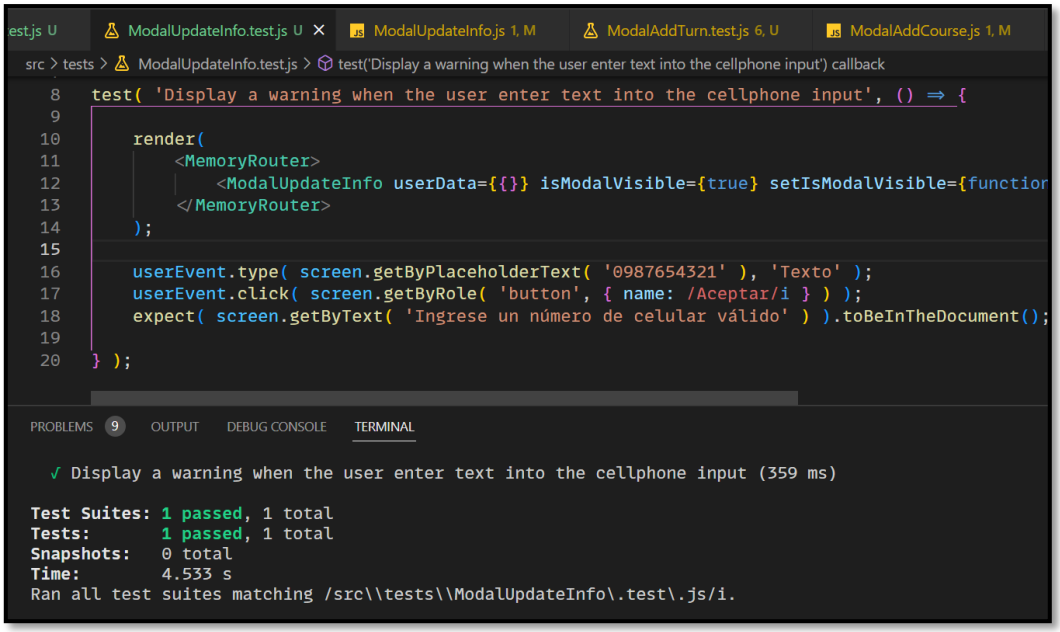

Fig. 22: Prueba para actualizar su información.

Reservar un turno

El usuario morador puede reservar un turno para el agendamiento de una cancha deportiva en una fecha y horario disponibles. Para ello, en primer lugar, debe escoger el sector de ubicación, a continuación, el tipo de cancha deportiva y, finalmente, una fecha y hora disponibles en el calendario semanal. Además, el usuario morador puede visualizar los horarios no disponibles en el calendario para una cancha seleccionada y observar los turnos que han reservado en la interfaz principal de su perfil. Esta funcionalidad se implementa mediante el consumo de métodos de *Firebase* y la información es presentada en el *frontend* como muestra la **Fig. 23**, mientras en la **Fig. 24** se presenta la estructura de la prueba unitaria y el resultado obtenido tras su ejecución para la funcionalidad “reservar turno”. Adicionalmente, la descripción completa de las funcionalidades y pruebas restantes se localizan en el **ANEXO II**.

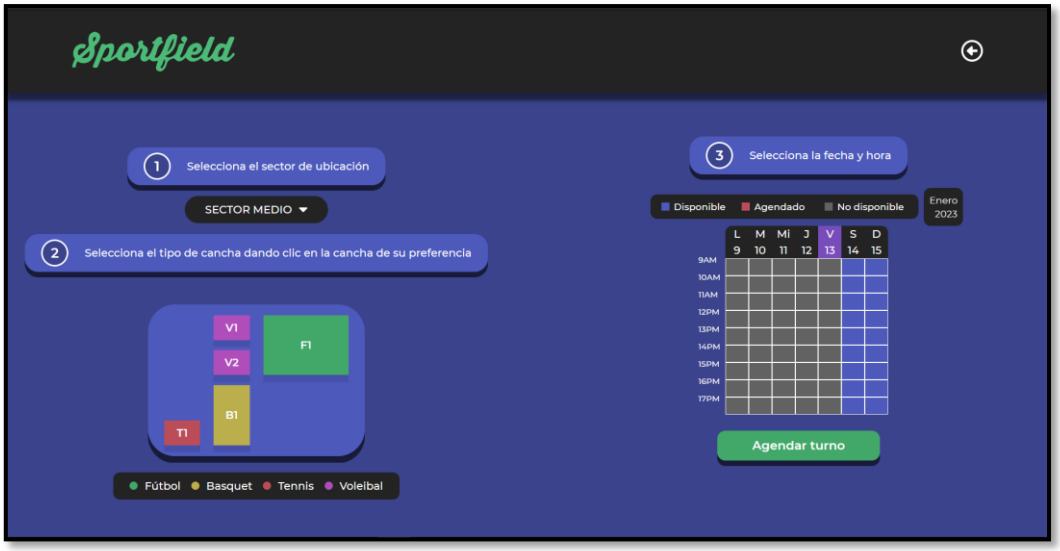

Fig. 23: Interfaz para reservar un turno.

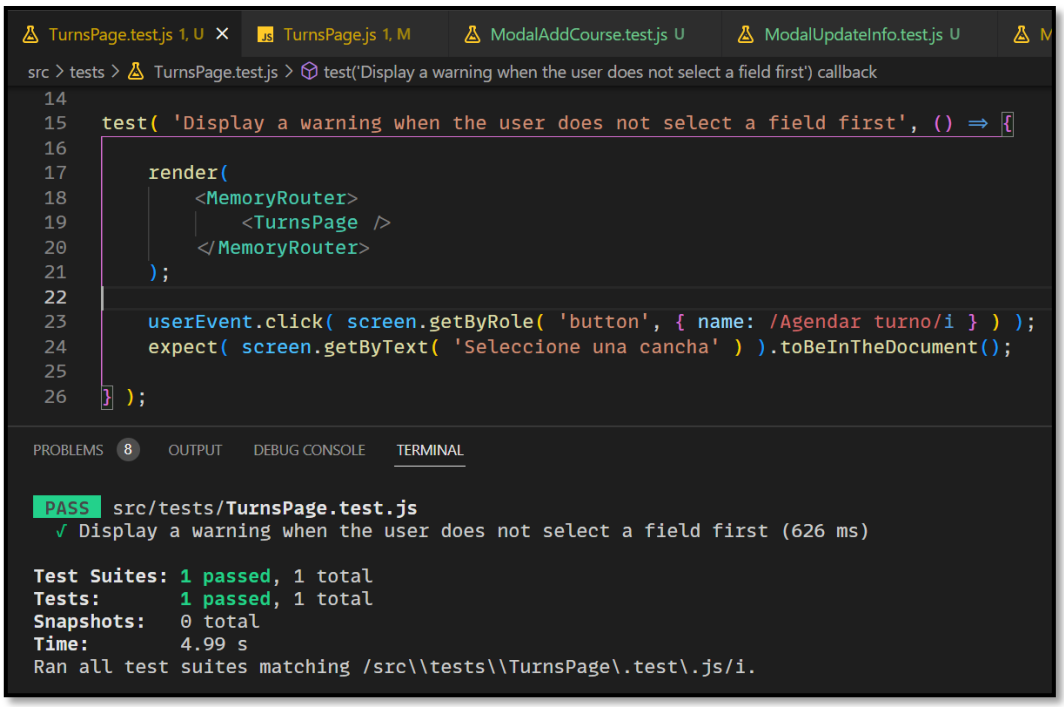

Fig. 24: Prueba para reservar un turno.

Registrar y anular curso vacacional

El usuario morador puede registrarse en un curso vacacional ofertado por el usuario administrador únicamente seleccionando el curso de su elección. Además, puede anular su inscripción en un curso en el cual se haya registrado desde su perfil, así como visualizar los todos los cursos que se encuentran ofertados. Esta funcionalidad se implementa mediante el consumo de métodos de *Firebase* y la información es presentada en el *frontend* como

muestran las **Fig. 25** y **Fig. 26**, mientras que en las **Fig. 27** y **Fig. 28** se presentan la estructura de la prueba unitaria y el resultado obtenido tras su ejecución para las funcionalidades “Registrar curso” y “Anular curso”, respectivamente. Por último, la descripción completa de las funcionalidades y pruebas restantes se localizan en el **ANEXO II**.

Fig. 25: Interfaz para registrarse en un curso.

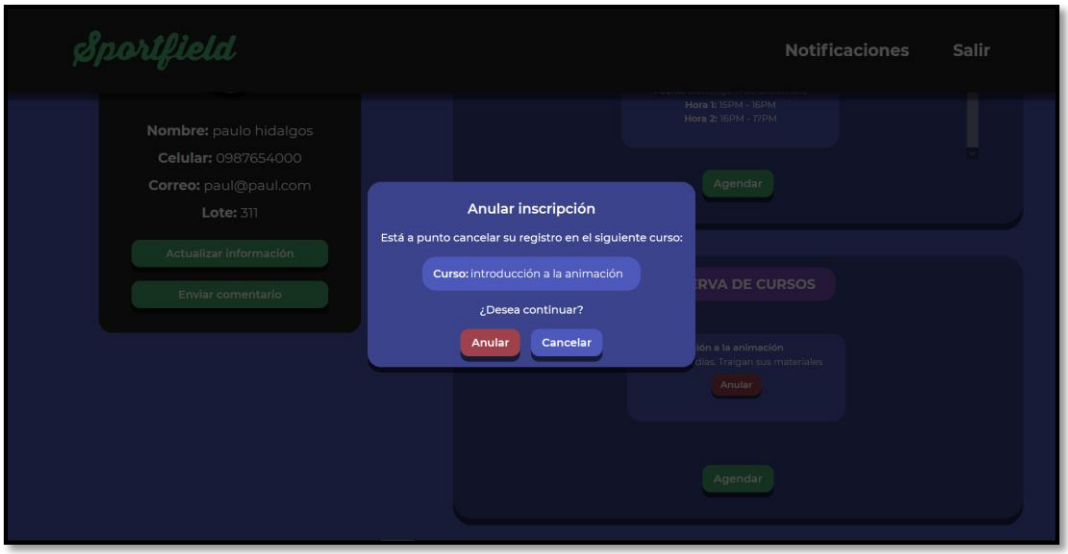

Fig. 26: Interfaz para anular el registro en un curso.

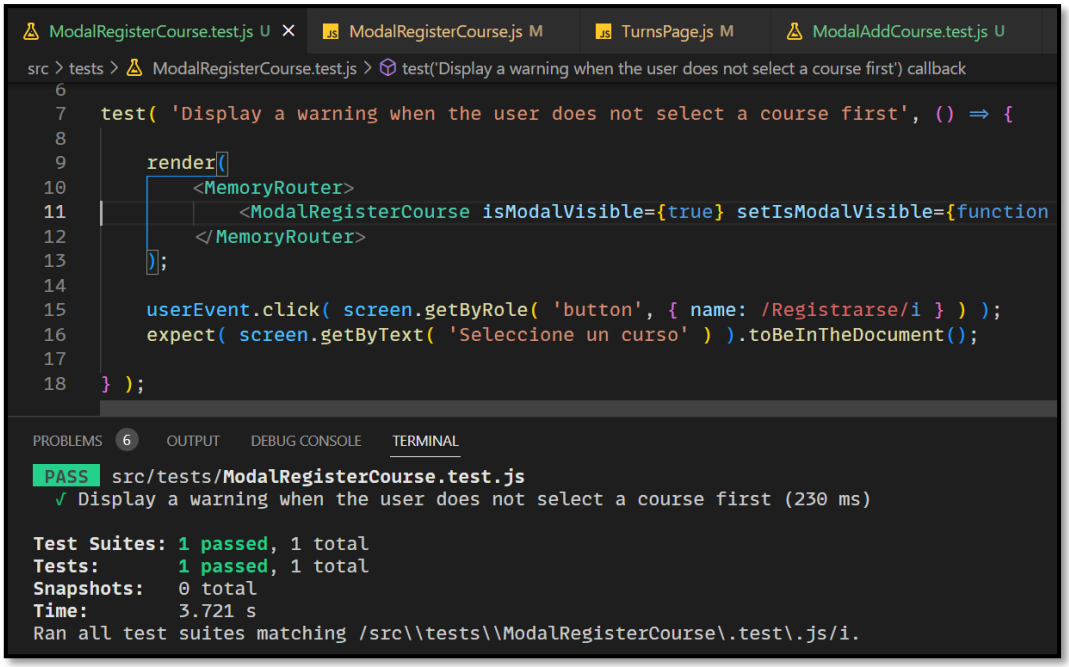

Fig. 27: Prueba para registrarse en un curso.

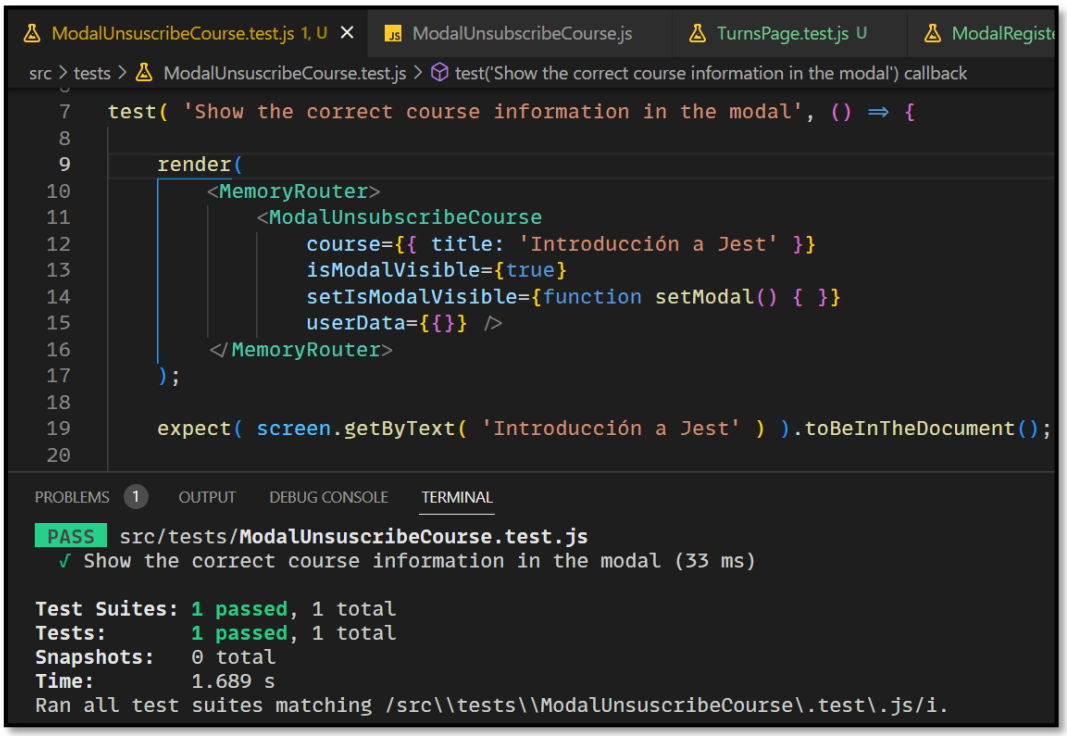

Fig. 28: Prueba para anular la inscripción a un curso.

Enviar comentarios

El usuario morador puede enviar comentarios al usuario administrador, especificando un título y una descripción. Por último, esta funcionalidad se implementa mediante el consumo de métodos de *Firebase* y la información es presentada en el *frontend* como muestra la **Fig. 29**, mientras en la **Fig. 30** se presenta la estructura de la prueba unitaria y el resultado obtenido tras su ejecución para la funcionalidad “Enviar comentarios”. Adicionalmente, la descripción completa de las funcionalidades y pruebas restantes se localizan en el **ANEXO II**.

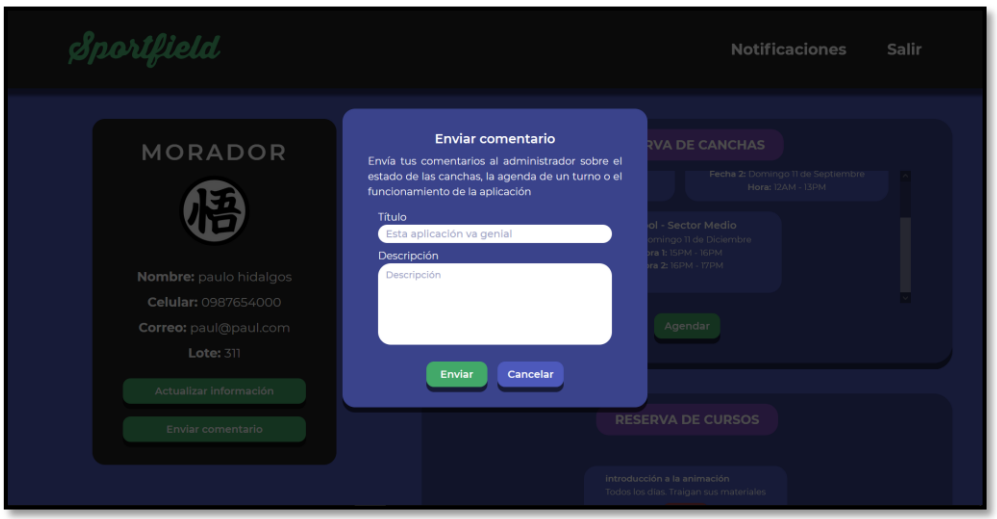

Fig. 29: Interfaz para enviar comentarios al administrador.

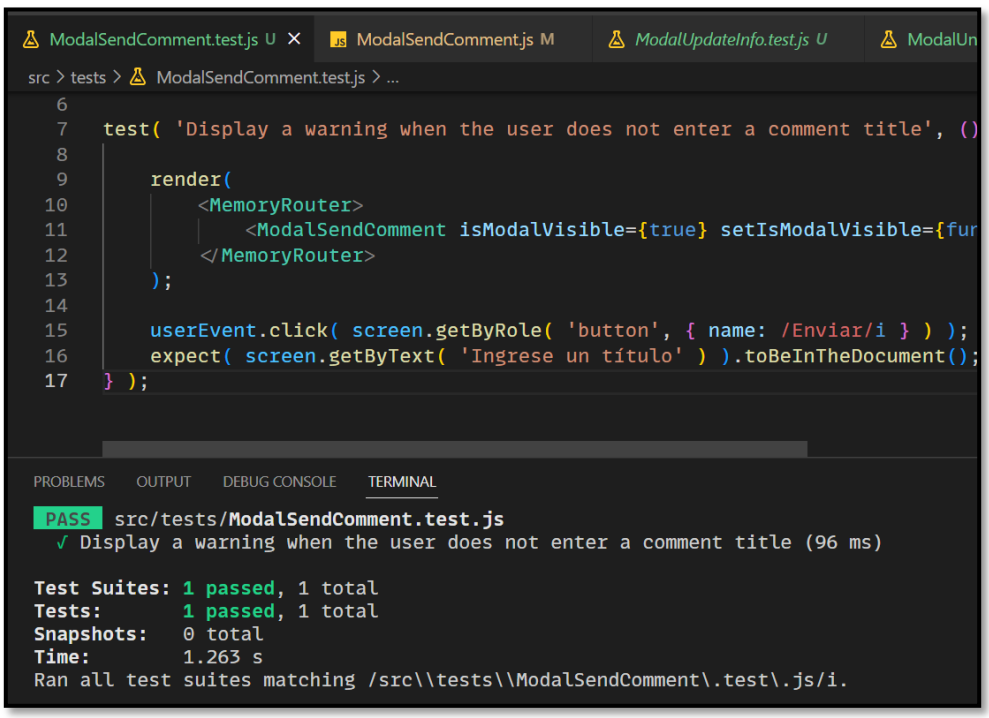

Fig. 30: Prueba para enviar comentarios al administrador.

Visualizar notificaciones

El usuario morador puede visualizar las notificaciones enviadas por el usuario administrador. Esta funcionalidad se implementa mediante el consumo de métodos de *Firebase* y la información es presentada en el *frontend* como muestra la **Fig. 31,** mientras que en la **Fig. 32** se presenta la estructura de su respectiva prueba y el resultado obtenido tras su ejecución. Adicionalmente, la descripción completa de las funcionalidades y pruebas restantes se localizan en el **ANEXO II**.

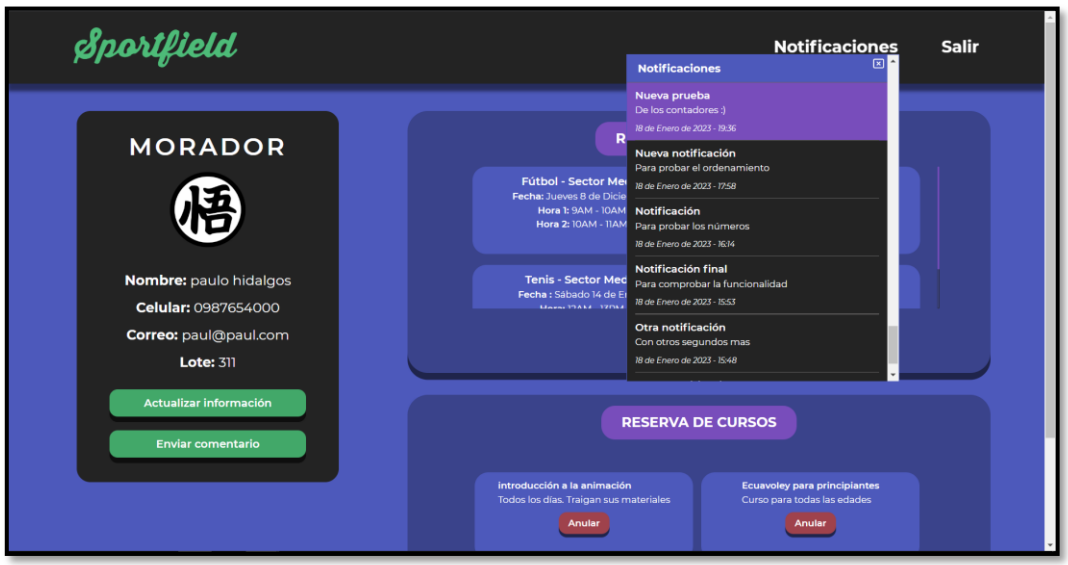

Fig. 31: Interfaz de visualización de notificaciones.

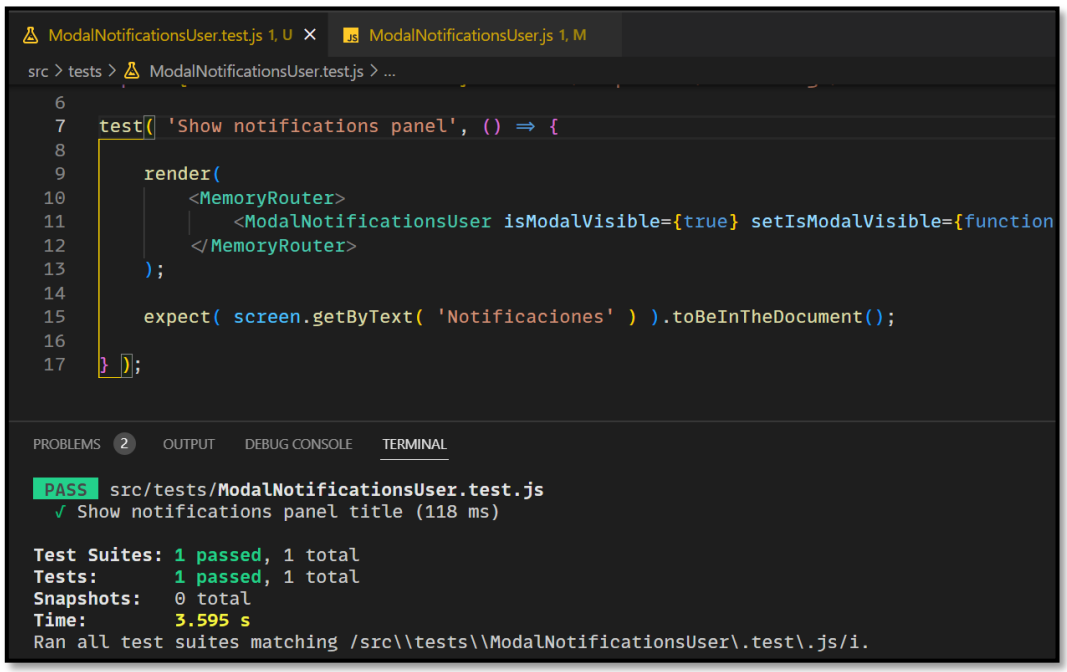

Fig. 32: Prueba para visualizar notificaciones.

3.5 *Sprint* 4: Pruebas y despliegue del *frontend*

Culminada de forma satisfactoria la etapa de desarrollo, esta sección corresponde realizar las tareas programadas para el Sprint 4, que son:

- Ejecución y resultado de las pruebas unitarias.
- Ejecución y resultado de las pruebas de compatibilidad.
- Ejecución y resultado de las pruebas de aceptación.
- Despliegue del *frontend*.

Ejecución y resultado de las pruebas unitarias

Este conjunto de pruebas permiten evaluar el comportamiento de un componente o funcionalidad en específico, es decir, sin la necesidad de integrar todo el complejo adicional **[30]**. En tal sentido, la **Fig. 33**, muestra el resultado del testeo de la prueba sobre la interfaz para finalizar un turno del módulo “Gestionar moradores” del usuario administrador. Adicionalmente, los detalles de la estructura y ejecución de las pruebas restantes se localizan en el **ANEXO II**.

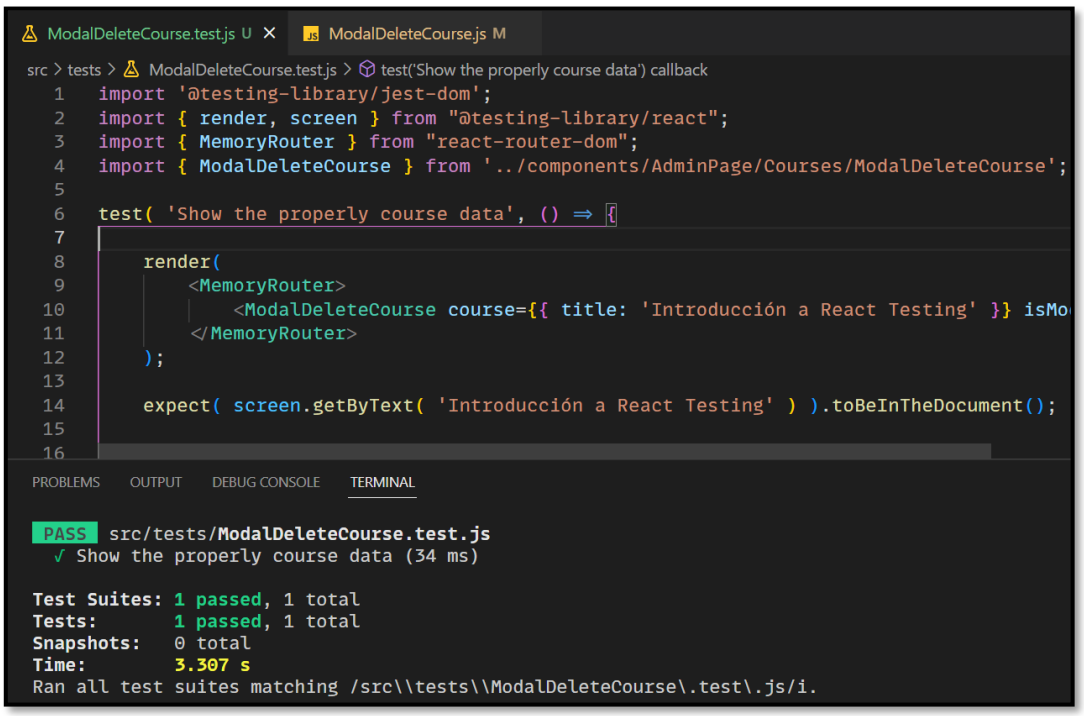

Fig. 33: Prueba para finalizar un curso.

Ejecución y resultado de las pruebas de compatibilidad

Permiten comprobar si las diferentes interfaces de usuario y la información que se presenta es similar cuando se renderiza en diferentes pantallas de dispositivos electrónicos y en diferentes navegadores [31]. De esta forma, se han estructurado y ejecutado un conjunto de pruebas de compatibilidad, seleccionando tres de los navegadores web más populares en la actualidad para su realización, como muestra la **TABLA VIII.** Adicionalmente, los detalles de la ejecución y resultados obtenidos en estas pruebas se localizan en el **ANEXO II**.

TABLA VIII: Navegadores, sus versiones y resultados obtenidos tras la ejecución de las pruebas.

NOMBRE	VERSIÓN	OBSERVACIÓN
Mozilla Firefox	108.0.2	Renderización eficaz
Microsoft Edge	109.0.1518.52	Renderización eficaz
Google Chrome	111.0.5532.2	Renderización eficaz

Una vez se ha culminado con el proceso de elaboración y ejecución las pruebas, y, considerando los diferentes resultados que se han obtenido en los navegadores seleccionados, es determinante concluir que el *frontend Sportfield* es *responsive* para múltiples dispositivos electrónicos y no presenta fallas en la renderización de la información.

Ejecución y resultado de las pruebas de aceptación

Las pruebas de aceptación permiten determinar la validez en la ejecución de un conjunto de requerimientos o funcionalidades pertenecientes a un determinado *software* [32]. Por este motivo, la **TABLA IX** presenta la estructura de la ejecución de la prueba de aceptación número 5: “Enviar notificaciones”. Por último, la descripción completa con la ejecución de las pruebas restantes se localiza en el **ANEXO II.**

TABLA IX: Ejecución de la prueba “Enviar notificaciones”.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA05	**Identificador de historia de usuario:** HU05
Nombre: Enviar notificaciones	
Descripción: El administrador puede enviar notificaciones a los moradores sobre información de las canchas deportivas o los cursos ofertados	

Pasos de ejecución:

Para enviar una notificación, el usuario administrador debe:

- Iniciar sesión
- En el menú principal, seleccionar la opción “Notificaciones”
- Cliquear el botón “Enviar notificación”
- Llenar los campos solicitados para enviar una notificación
- Dar clic en el botón “Enviar”

Una vez se ha culminado con el proceso de elaboración y ejecución de las pruebas de aceptación, y, concedida del *Product Owner* su autorización, se determina que el *frontend Sportfield* satisface todo el conjunto de requerimientos establecidos al comienzo del proyecto.

Despliegue del *frontend*

Finalizada la etapa de pruebas, a continuación, corresponde el despliegue del *frontend* a producción. En tal sentido, la herramienta seleccionada para el despliegue ha sido *Firebase Hosting*, la cual proporciona un servicio gratuito y eficaz para alojar la el *frontend Sportfield*. Finalmente, la **Fig. 34** muestra el resultado del *frontend* desplegado a producción a través de un dominio concedido por *Firebase*, al cual se tiene acceso mediante la siguiente dirección web:

https://sportfield-489a8.web.app/

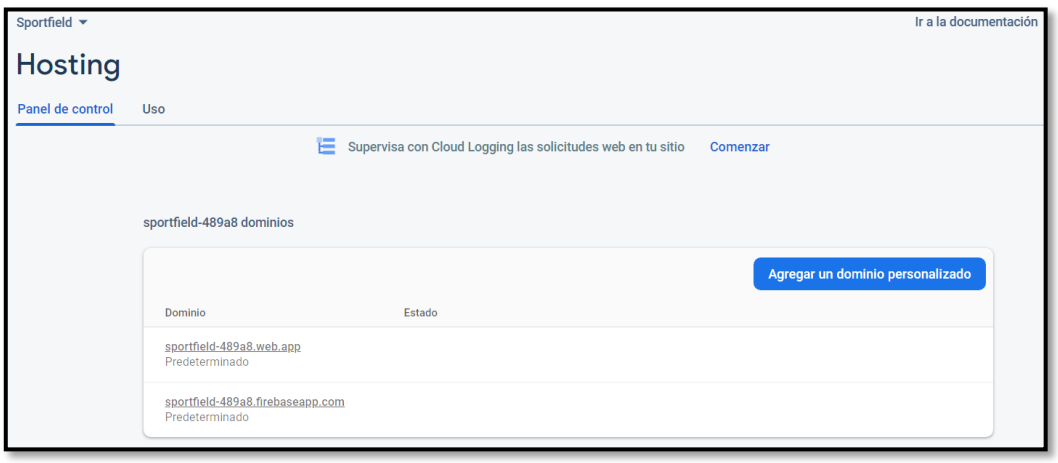

Fig. 34: Despliegue de *Sportfield* en *Firebase Hosting.*

4 CONCLUSIONES

Como parte final del desarrollo del *frontend Sportfield*, se listan las conclusiones generadas:

- El *frontend* denominado *Sportfield* cumple con todos los requerimientos y objetivos que se han establecido, permitiendo así que los administradores de la urbanización “Los Retoños” puedan gestionar a los moradores registrados, gestionar los turnos de reservas de canchas deportivas y gestionar la oferta de cursos vacacionales; además de permitir a los moradores la reserva de un turno para una cancha deportiva y el registro en un curso vacacional que se haya ofertado.
- La aplicación de la metodología ágil *Scrum* ha permitido dinamizar el proceso de desarrollo del *frontend* para culminar cada fase del proyecto en el tiempo establecido.
- Definir de forma correcta la lista de requerimientos ha permitido seleccionar las herramientas adecuadas para la codificación del programa, así como cada uno de los roles y artefactos en la metodología *Scrum*.
- El diseño de las interfaces de usuario ha permitido agilizar el proceso de codificación de cada componente del *frontend*, así como establecer una paleta de colores definida para la etapa de diseño.
- La aplicación y guía del patrón arquitectónico MVC ha permitido una implementación correcta de los métodos que se han consumido para la conexión con la plataforma *Firebase*.
- Las herramientas y librerías de codificación han presentado una alta compatibilidad e integración entre ellas, lo que ha permitido un desarrollo completo y dinámico de cada componente que forma parte del *frontend*.
- Los resultados que se han obtenido en cada una de las pruebas de compatibilidad, aceptación y unitarias han presentado resultados favorables en cada ejecución, por lo cual no se han presentado complicaciones considerables en el proceso de despliegue del *frontend* a producción.

5 RECOMENDACIONES

Como parte final del desarrollo del *frontend Sportfield*, se listan las recomendaciones generadas:

- Se recomienda la implementación de un módulo de chat comunitario, mediante el cual los moradores puedan comunicarse con los demás residentes de la urbanización en caso de una urgencia o asunto importante.
- Se recomienda la capacitación a los administradores como a los moradores sobre el funcionamiento del *frontend Sportfield*, para que puedan hacer un uso completo y adecuado de todas las funcionalidades que se han desarrollado.
- Se recomienda la implementación de un botón de alarma comunitaria o botón de pánico dentro del *frontend* que envíe una señal de alerta a los correos o celulares de los moradores en caso de una emergencia.
- Se recomienda la implementación de un módulo que permita a los administradores de la urbanización responder a las preguntas y comentarios enviados por los moradores.

6 REFERENCIAS BIBLIOGRÁFICAS

[1] CONSEJO METROPOLITANO DE QUITO, “ORDENANZA No. 3756.” Feb. 17, 2011. Accessed: Jan. 24, 2023. [Online]. Available: https://www7.quito.gob.ec/mdmq_ordenanzas/ordenanzas/ORDENANZAS%20MUNICIPALES%202011/ORDM-0058%20%20%20%20LOS%20RETO%C3%91OS%20DE%20LOS%20SERVIDORES%20DEL%20MAG%20-URBANIZACI%C3%93N.pdf

[2] E. Cavanaugh, “Web services: Benefits, challenges, and a unique, visual development solution,” *white paper, Feb*, vol. 10, 2006.

[3] “What Is a Front-End Developer?” https://frontendmasters.com/books/front-end-handbook/2018/what-is-a-FD.html (accessed Dec. 02, 2022).

[4] R. Kulesza, M. F. de Sousa, M. L. M. de Araújo, C. P. de Araújo, and A. M. Filho, “Evolution of Web Systems Architectures: A Roadmap,” in *Special Topics in Multimedia, IoT and Web Technologies*, V. Roesler, E. Barrére, and R. Willrich, Eds. Cham: Springer International Publishing, 2020, pp. 3–21. doi: 10.1007/978-3-030-35102-1_1.

[5] B. G. L. Cajamarca, “Desarrollo de un backend para la gestión del sistema penitenciario del Ecuador,” *ConcienciaDigital*, vol. 5, no. 3.2, Art. no. 3.2, Sep. 2022, doi: 10.33262/concienciadigital.v5i3.2.2319.

[6] B. G. L. Cajamarca and I. F. M. Soliz, “Desarrollo de una aplicación web y móvil en tiempo real, una evolución de las aplicaciones actuales,” *Ciencia Digital*, vol. 3, no. 1, Art. no. 1, Feb. 2019, doi: 10.33262/cienciadigital.v3i1.282.

[7] M. S. Mikowski and J. C. Powell, *Single page web applications: JavaScript end-to-end*. Shelter Island, NY: Manning, 2014.

[8] “What Is React? | SpringerLink.” https://link.springer.com/chapter/10.1007/978-1-4842-1245-5_1 (accessed Dec. 04, 2022).

[9] “Tutorial: Intro to React – React.” https://reactjs.org/tutorial/tutorial.html (accessed Dec. 04, 2022).

[10] “CSS - MDN Web Docs Glossary: Definitions of Web-related terms | MDN.” https://developer.mozilla.org/en-US/docs/Glossary/CSS (accessed Dec. 04, 2022).

[11] “Cascade, specificity, and inheritance - Learn web development | MDN.” https://developer.mozilla.org/en-US/docs/Learn/CSS/Building_blocks/Cascade_and_inheritance (accessed Dec. 04, 2022).

[12] “Introduction to Tailwind CSS,” *GeeksforGeeks*, Jun. 24, 2020. https://www.geeksforgeeks.org/introduction-to-tailwind-css/ (accessed Dec. 04, 2022).

[13] “Installation - Tailwind CSS.” https://tailwindcss.com/docs/installation (accessed Dec. 04, 2022).

[14] “Overview - Material UI.” https://mui.com/material-ui/getting-started/overview/ (accessed Dec. 04, 2022).

[15] “Introduction.” https://frontend-masters-firebase.web.app/1-intro/ (accessed Dec. 02, 2022).

[16] L. Moroney, “An Introduction to Firebase,” in *The Definitive Guide to Firebase: Build Android Apps on Google’s Mobile Platform*, L. Moroney, Ed. Berkeley, CA: Apress, 2017, pp. 1–24. doi: 10.1007/978-1-4842-2943-9_1.

[17] “Agregue Firebase a su proyecto de JavaScript.” https://firebase.google.com/docs/web/setup?hl=es-419 (accessed Dec. 02, 2022).

[18] R. K. Yin, *Case Study Research: Design and Methods*. Los Angeles, 2013.

[19] I. Benbasat, D. K. Goldstein, and M. Mead, “The Case Research Strategy in Studies of Information Systems,” *MIS Quarterly*, vol. 11, no. 3, pp. 369–386, 1987, doi: 10.2307/248684.

[20] P. Runeson, M. Höst, A. Rainer, and B. Regnell, *Case Study Research in Software Engineering: Guidelines and Examples*, 1st ed. Wiley, 2012. doi: 10.1002/9781118181034.

[21] M. L. Despa, “Comparative study on software development methodologies,” *Database Systems Journal*, vol. 5, no. 3, pp. 37–56, 2014.

[22] A. Stellman and J. Greene, *Learning Agile*, 2nd ed. United States of America: O’Reilly Media, 2015. Accessed: Dec. 04, 2022. [Online]. Available: https://www.oreilly.com/library/view/learning-agile/9781449363819/

[23] K. Rubin, *Essential Scrum: A Practical Guide to the Most Popular Agile Process*. Upper Saddle River, NJ, 2012.

[24] “Scrum Guide | Scrum Guides.” https://scrumguides.org/scrum-guide.html#scrum-definition (accessed Dec. 04, 2022).

[25] R. S. Pressman, *Ingeniería de Software*. 2010.

[26] J. Shore and S. Warden, *The Art of Agile Development*. Beijing Boston Farnham Sebastopol, CA Tokyo, 2021.

[27] R. J. K. Jacob, “User interface,” in *Encyclopedia of Computer Science*, GBR: John Wiley and Sons Ltd., 2003, pp. 1821–1826.

[28] “Free Prototyping Tool: Build Interactive Prototype Designs,” *Figma*. https://www.figma.com/prototyping/ (accessed Dec. 04, 2022).

[29] R. Martin, *Arquitectura limpia: Guía de un artesano para la estructura y el diseño del software*. London, England, 2017.

[30] M. Olan, “Unit testing: Test early, test often,” *Journal of Computing Sciences in Colleges - JCSC*, vol. 19, Jan. 2003.

[31] I.-C. Yoon, A. Sussman, A. Memon, and A. Porter, “Effective and scalable software compatibility testing,” in *Proceedings of the 2008 international symposium on Software testing and analysis*, New York, NY, USA, Jul. 2008, pp. 63–74. doi: 10.1145/1390630.1390640.

[32] R. W. Miller and C. T. Collins, “Software Developer RoleModel Software, Inc. 342 Raleigh Street Holly Springs, NC 27540 USA +1 919 557 6352”.

7 ANEXOS

A continuación, se presenta cada uno de los Anexos que se ha utilizado para el desarrollo del *frontend*, los cuales se encuentran detallados de la siguiente manera:

- **ANEXO I.** Resultado del programa anti-plagio *Turnitin*.
- **ANEXO II.** Manual de Usuario.
- **ANEXO III.** Manual de Instalación.
- **ANEXO IV.** Credenciales de acceso y despliegue.

ANEXO I

A continuación, se presenta el certificado que el Director de Tesis ha emitido y en donde se evidencia el resultado que se ha obtenido en la herramienta antiplagio Turnitin.

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

CERTIFICADO DE ORIGINALIDAD

Quito, D.M. 14 de febrero de 2023

De mi consideración:

Yo, Loarte Cajamarca Byron Gustavo, en calidad de Director del Trabajo de Integración Curricular titulado Desarrollo de un frontend asociado al DESARROLLO DE UN SISTEMA WEB Y APLICACIÓN MÓVIL PARA LA RESERVA DE CANCHAS Y OFERTA DE CURSOS EN LA URBANIZACIÓN LOS RETOÑOS elaborado por el estudiante Paul Isaac Guala Bravo 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 secciones: Descripción del componente desarrollado, Metodología, Resultados, Conclusiones y Recomendaciones, producto del Trabajo de Integración Curricular indicado.

El documento escrito tiene un índice de similitud del 10%.

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,

Loarte Cajamarca Byron Gustavo
Profesor Ocasional a Tiempo Completo
Escuela de Formación de Tecnólogos

ANEXO II

Recopilación de requerimientos

En la **TABLA X** se muestra los requerimientos que han sido recopilados al inicio del proyecto en donde se evidencia lo solicitado por el *Product Owner*.

TABLA X: Recopilación de requerimientos de usuario

RECOPILACIÓN DE REQUERIMIENTOS		
TIPO DEL SISTEMA	**ID - RR**	**ENUNCIADO DEL ÍTEM**
Frontend	**RR01**	Los usuario administrador y morador necesitan consumir varios métodos para: • Iniciar sesión • Cerrar sesión
	RR02	El usuario con perfil administrador necesita consumir varios métodos para • Gestionar moradores.
	RR03	El usuario con perfil administrador necesita consumir varios métodos para: • Gestionar turnos
	RR04	El usuario con perfil administrador necesita consumir un método para: • Gestionar cursos
	RR05	El usuario con perfil administrador necesita consumir varios métodos para • Enviar notificaciones a los moradores
	RR06	El usuario con perfil administrador necesita consumir varios métodos para: • Visualizar los comentarios enviados por los moradores
	RR07	El usuario con perfil morador necesita consumir varios métodos para: • Registrarse en el *frontend*
	RR08	El usuario con perfil morador necesita consumir varios métodos para:

		• Reservar un turno
	RR09	El usuario con perfil morador necesita consumir varios métodos para: • Registrar un curso • Visualizar los cursos ofertados por el administrador
	RR11	El usuario con perfil morador necesita consumir varios métodos para: • Visualizar las notificaciones enviadas por el administrador
	RR12	El usuario con perfil morador necesita consumir varios métodos para: • Editar su información personal

Historias de Usuario

Una vez que se ha realizado y completado el proceso de recopilación de requerimientos, a continuación, se implementan las historias de usuario del *frontrend*. Así, se listan 11 de todo el conjunto de historias de usuario establecidas en base a la recopilación de requerimientos, las cuales se presentan desde la **TABLA XI** que corresponde a la historia de usuario 1, hasta la **TABLA XXI** que corresponde a la historia de usuario 12.

TABLA XI: Inicio y cierre de sesión.

HISTORIA DE USUARIO	
Identificador: HU01	**Usuario:** Administrador / Morador
Nombre historia: Iniciar y cerrar sesión	
Prioridad en Negocio: Alta	**Riesgo en Desarrollo:** Medio
Iteración asignada: 1	
Responsable: Paul Guala	
Descripción: Los usuarios Administrador y Morador pueden iniciar sesión con sus credenciales y cerrar sesión	

Observación: Solo el administrador estará registrado previamente en la base de datos con credenciales de administrador

TABLA XII: Gestionar moradores.

HISTORIA DE USUARIO	
Identificador: HU02	**Usuario:** Administrador
Nombre historia: Gestionar moradores	
Prioridad en Negocio: Alta	**Riesgo en Desarrollo:** Medio
Iteración asignada: 2	
Responsable: Paul Guala	
Descripción: El administrador puede registrar, activar y desactivar moradores	
Observación: Ninguna	

Tabla XIII: Gestionar turnos

HISTORIA DE USUARIO	
Identificador: HU03	**Usuario:** Administrador
Nombre historia: Gestionar turno	
Prioridad en Negocio: Alta	**Riesgo en Desarrollo:** Medio
Iteración asignada: 2	
Responsable: Paul Guala	
Descripción: El administrador puede reservar y finalizar turnos	
Observación: Ninguna	

TABLA XIV: Gestionar cursos.

HISTORIA DE USUARIO	
Identificador: HU04	**Usuario:** Administrador
Nombre historia: Gestionar cursos	
Prioridad en Negocio: Alta	**Riesgo en Desarrollo:** Medio
Iteración asignada: 2	
Responsable: Paul Guala	
Descripción: El administrador puede ofertar, editar y finalizar cursos	
Observación: • El administrador podrá observar el número de moradores inscritos en un curso • Los cursos se ofertarán en el *landing page*	

Tabla XV: Enviar notificaciones.

HISTORIA DE USUARIO	
Identificador: HU05	**Usuario:** Administrador
Nombre historia: Enviar notificaciones	
Prioridad en Negocio: Media	**Riesgo en Desarrollo:** Bajo
Iteración asignada: 3	
Responsable: Paul Guala	
Descripción: El administrador puede enviar notificaciones a los moradores sobre información de las canchas deportivas o los cursos ofertados	
Observación: Las notificaciones se enviarán a todos los moradores	

TABLA XVI: Visualizar comentarios.

HISTORIA DE USUARIO	
Identificador: HU06	**Usuario:** Administrador
Nombre historia: Visualizar comentarios	
Prioridad en Negocio: Baja	**Riesgo en Desarrollo:** Bajo
Iteración asignada: 3	
Responsable: Paul Guala	
Descripción: El administrador puede visualizar los comentarios enviados por los moradores	
Observación: Ninguna	

TABLA XVII: Reservar turno.

HISTORIA DE USUARIO	
Identificador: HU08	**Usuario:** Morador
Nombre historia: Reservar turno	
Prioridad en Negocio: Alta	**Riesgo en Desarrollo:** Medio
Iteración asignada: 2	
Responsable: Paul Guala	
Descripción: El morador puede reservar un turno con la cancha y fecha seleccionada	
Observación: • El morador solo puede reservar una cancha por turno • El morador solo puede reservar un máximo de dos horas por cancha en cada turno	

TABLA XVIII: Visualizar y registrar curso.

HISTORIA DE USUARIO	
Identificador: HU09	**Usuario:** Morador
Nombre historia: Visualizar y registrar cursos	
Prioridad en Negocio: Alta	**Riesgo en Desarrollo:** Medio
Iteración asignada: 2	
Responsable: Paul Guala	
Descripción: El morador puede visualizar los cursos ofertados y registrarse en un curso	
Observación: • Los datos utilizados para la inscripción son los registrados por el morador	

TABLA XIX: Enviar comentarios.

HISTORIA DE USUARIO	
Identificador: HU10	**Usuario:** Morador
Nombre historia: Enviar comentarios	
Prioridad en Negocio: Media	**Riesgo en Desarrollo:** Bajo
Iteración asignada: 3	
Responsable: Paul Guala	
Descripción: El morador puede enviar comentarios y sugerencias al administrador	
Observación: • El morador puede escribir un máximo de 250 palabras	

TABLA XX: Visualizar notificaciones.

HISTORIA DE USUARIO	
Identificador: HU11	**Usuario:** Morador
Nombre historia: Visualizar notificaciones	
Prioridad en Negocio: Media	**Riesgo en Desarrollo:** Bajo
Iteración asignada: 3	
Responsable: Paul Guala	
Descripción: El morador puede visualizar las notificaciones enviadas por el administrador	
Observación: Ninguna	

TABLA XXI: Editar información.

HISTORIA DE USUARIO	
Identificador: HU12	**Usuario:** Morador
Nombre historia: Editar información	
Prioridad en Negocio: Baja	**Riesgo en Desarrollo:** Medio
Iteración asignada: 4	
Responsable: Paul Guala	
Descripción: El morador puede modificar su información personal de registro	
Observación: • El morador debe ingresar su contraseña para editar su información	

Product Backlog

Este apartado presenta la definición completa del *Product Backlog*. La **TABLA XXII** muestra cuál es la prioridad que tiene cada historia de usuario en base a su importancia para el dueño del producto y su complejidad en el desarrollo.

TABLA XXII: *Product Backlog.*

ELABORACIÓN DEL *PRODUCT BACKLOG*				
ID – HU	**HISTORIA DE USUARIO**	**ITERACIÓN**	**ESTADO**	**PRIORIDAD**
HU01	Iniciar y cerrar sesión	1	Finalizada	Alta
HU02	Gestionar moradores	2	Finalizada	Alta
HU03	Gestionar turnos	2	Finalizada	Alta
HU04	Gestionar cursos	2	Finalizada	Alta
HU05	Enviar notificaciones	3	Finalizada	Media
HU06	Visualizar comentarios	3	Finalizada	Baja
HU07	Registrar morador	1	Finalizada	Alta
HU09	Visualizar y registrar cursos	2	Finalizada	Alta
HU10	Enviar comentarios	3	Finalizada	Media
HU11	Visualizar notificaciones	3	Finalizada	Media

Sprint Backlog

La **TABLA XXIII** presenta los *Sprints* que se han desarrollado por parte del *frontend*, describiendo las actividades planteadas y el tiempo para cumplirlas respectivamente, siendo esto establecido por el dueño del producto.

TABLA XXIII: *Sprint Backlog.*

ELABORACIÓN DEL *SPRINT BACKLOG*						
ID-SB	**NOMBRE**	**MÓDULO**	**ID-HU**	**HISTORIA DE USUARIO**	**TAREAS**	**TIEMPO ESTIMADO**
SB-00	Configuración del entorno	-	-	-	• Restricciones del proyecto • Organización del proyecto • Roles de usuario	10H
SB-01	Rol Administrador	Módulo – Inicio y cierre de sesión	HU001	Iniciar y cerrar sesión	• Diseño de prototipos. • Definir los campos obligatorios para iniciar sesión • Diseño de interfaz para el inicio de sesión • Validación de los campos de inicio de sesión • Consumir varios métodos para iniciar sesión • Consumir varios métodos para cerrar sesión • Pruebas con el usuario administrador	80H
		Módulo – Moradores	HU002	Gestionar moradores	• Diseño de prototipos. • Definir los campos para registrar un morador	

					• Diseño de la interfaz para registrar un morador • Consumir varios métodos para registrar un morador • Pruebas con el usuario administrador	
					• Diseño de prototipos. • Definir los campos para visualizar un morador activo • Definir los campos para visualizar un morador inactivo • Diseño de la interfaz para visualizar un morador activo • Diseño de la interfaz para visualizar un morador inactivo • Consumir varios métodos para visualizar un morador activo • Consumir varios métodos para visualizar un morador inactivo • Pruebas con el usuario administrador	
					• Diseño de prototipos. • Definir los campos para activar un morador • Diseño de interfaz para activar un morador • Consumir varios métodos para activar un morador • Pruebas con el usuario administrador	
					• Diseño de prototipos. • Definir los campos para desactivar un morador • Diseño de interfaz para desactivar un morador • Consumir varios métodos para desactivar un morador • Pruebas con el usuario administrador	

		Módulo – Turnos	HU03	Gestionar turnos	• Diseño de prototipos • Definir los campos para reservar un turno • Diseño de interfaz para reservar un turno • Consumir varios métodos para reservar un turno • Pruebas con el usuario administrador	
					• Diseño de prototipos • Definir los campos para visualizar un turno reservado • Definir los campos para visualizar un turno finalizado • Diseño de interfaz para visualizar un turno reservado • Diseño de interfaz para visualizar un turno finalizado • Consumir varios métodos para visualizar un turno reservado • Consumir varios métodos para visualizar un turno finalizado • Pruebas con el usuario administrador	
					• Diseño de prototipos • Definir los campos para finalizar un turno • Diseño de interfaz para finalizar un turno • Consumir varios métodos para finalizar un turno • Pruebas con el usuario administrador	

		Modulo – Cursos	HU04	Gestionar cursos	• Diseño de prototipos. • Definir los campos para registrar un curso • Diseño de la interfaz para registrar un curso • Consumir varios métodos para registrar un curso • Pruebas con el usuario administrador	
					• Diseño de prototipos • Definir los campos para visualizar un curso activo • Definir los campos para visualizar un curso finalizado • Diseño de interfaz para visualizar un curso activo • Diseño de interfaz para visualizar un curso finalizado • Consumir varios métodos para visualizar un curso activo • Consumir varios métodos para visualizar un curso finalizado • Pruebas con el usuario administrador	
					• Diseño de prototipos • Definir los campos para finalizar un curso • Diseño de interfaz para finalizar un curso • Consumir varios métodos para finalizar un curso • Pruebas con el usuario administrador	

		Modulo – Notificaciones	HU05	Enviar notificaciones	• Diseño de prototipos. • Definir los campos para enviar una notificación • Diseño de la interfaz para enviar una notificación • Consumir varios métodos para enviar una notificación • Pruebas con el usuario administrador	
					• Diseño de prototipos • Definir los campos para visualizar una notificación enviada • Diseño de interfaz para visualizar una notificación enviada • Consumir varios métodos para visualizar una notificación enviada • Pruebas con el usuario administrador	
		Módulo – Comentarios	HU06	Visualizar comentarios	• Diseño de prototipos • Definir los campos para visualizar un comentario recibido • Diseño de interfaz para visualizar un comentario recibido • Consumir varios métodos para visualizar un comentario recibido • Pruebas con el usuario administrador	
SB-02	Rol Morador	Módulo – Inicio y cierre de sesión	HU01	Iniciar y cerrar sesión	• Diseño de prototipos. • Definir los campos obligatorios para iniciar sesión • Diseño de interfaz para el inicio de sesión • Validación de los campos de inicio de sesión • Consumir varios métodos para iniciar sesión • Consumir varios métodos para cerrar sesión • Pruebas con el usuario morador	80H

		Módulo – Registro	HU07	Registrar morador	• Diseño de prototipos. • Definir los campos para registrar un morador • Diseño de la interfaz para registrar un morador • Consumir varios métodos para registrar un morador • Pruebas con el usuario morador	
		Módulo - Turnos	HU08	Reservar turno	• Diseño de prototipos • Definir los campos para reservar un turno • Diseño de interfaz para reservar un turno • Consumir varios métodos para reservar un turno • Pruebas con el usuario morador	
					• Diseño de prototipos • Definir los campos para visualizar un turno reservado • Diseño de interfaz para visualizar un turno reservado • Consumir varios métodos para visualizar un turno reservado • Pruebas con el usuario morador	
					• Diseño de prototipos • Definir los campos para finalizar un turno • Diseño de interfaz para finalizar un turno • Consumir varios métodos para finalizar un turno • Pruebas con el usuario morador	

		Módulo – Cursos	HU09	Visualizar y registrar cursos	• Diseño de prototipos • Definir los campos para visualizar un curso activo • Diseño de interfaz para visualizar un curso activo • Consumir varios métodos para visualizar un curso activo • Pruebas con el usuario morador	
					• Diseño de prototipos. • Definir los campos para registrarse en un curso • Diseño de la interfaz para registrarse en un curso • Consumir varios métodos para registrarse en un curso • Pruebas con el usuario administrador	
					• Diseño de prototipos • Definir los campos para anular un curso • Diseño de interfaz para anular un curso • Consumir varios métodos para anular un curso • Pruebas con el usuario morador	
		Modulo – Comentarios	HU10	Enviar comentarios	• Diseño de prototipos. • Definir los campos para enviar un comentario • Diseño de la interfaz para enviar un comentario • Consumir varios métodos para enviar un comentario • Pruebas con el usuario morador	

		Modulo – Notificaciones	HU11	Visualizar notificaciones	• Diseño de prototipos • Definir los campos para visualizar una notificación recibida • Diseño de interfaz para visualizar una notificación recibida • Consumir varios métodos para visualizar una notificación recibida • Pruebas con el usuario morador	
		Modulo – Información personal	HU12	Editar información	• Diseño de prototipos. • Definir los campos de información a editar • Diseño de la interfaz para editar información • Consumir varios métodos para editar información • Pruebas con el usuario morador	
					• Diseño de prototipos • Definir los campos para visualizar la información editada • Diseño de interfaz para visualizar la información editada • Consumir varios métodos para la información editada • Pruebas con el usuario morador	
SB-03	Pruebas del *frontend*		• Pruebas unitarias • Pruebas de compatibilidad • Pruebas de aceptación			20H
SB-04	Despliegue del *frontend*		• Despliegue del *frontend* en *Firebase Hosting*			10H

Documentación	• Trabajo de Integración Curricular • Anexos	40h
TOTAL		**240**

Diseño de interfaces

A continuación, se visualizan los diferentes *mockups* de cada uno de los módulos del *frontend*, que representa cada uno de los diseños de interfaz, los cuales van desde la **Fig. 35** hasta la **Fig. 47**.

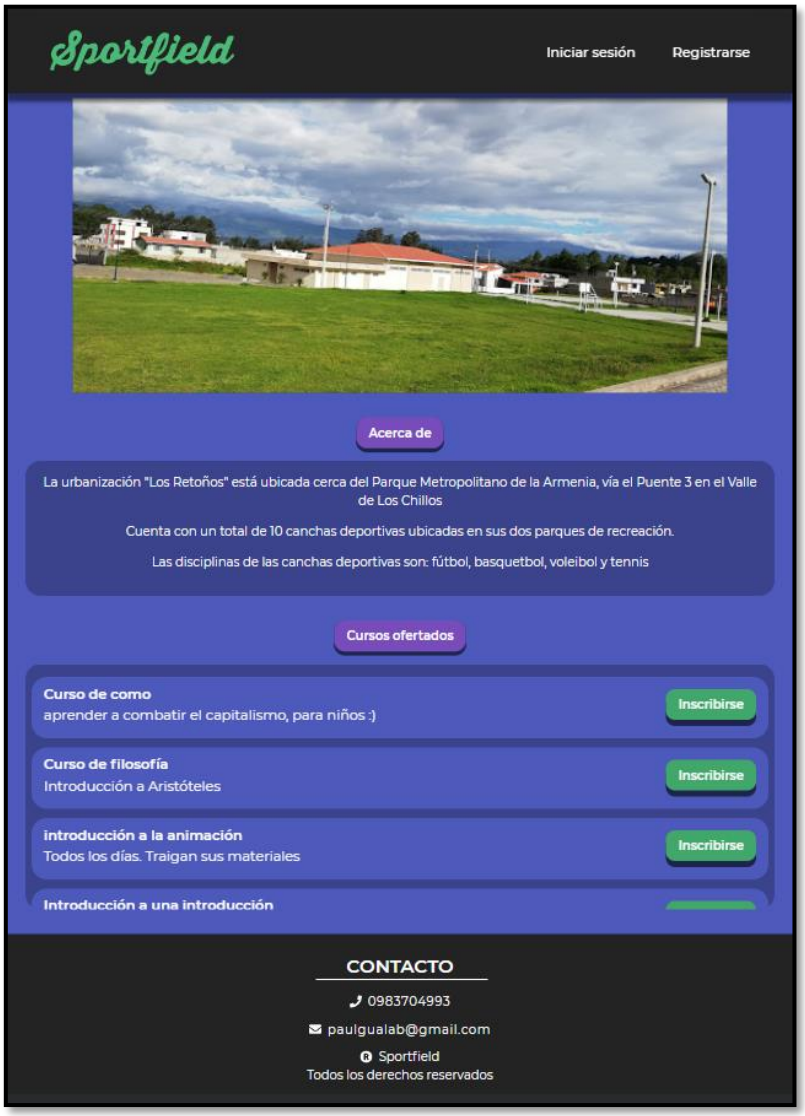

Fig. 35: Interfaz del *Landing Page* de *Sportfield*.

Fig. 36: Interfaz de inicio se sesión.

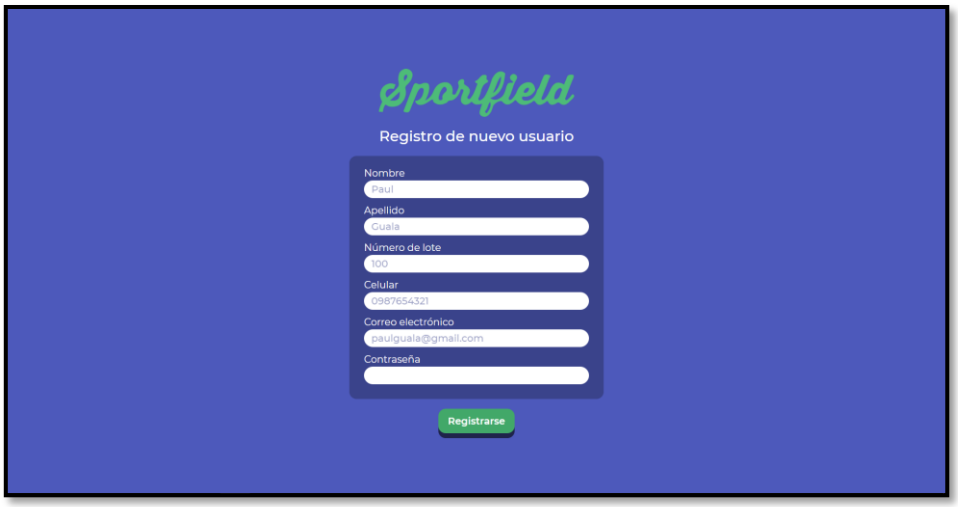

Fig. 37: Interfaz de registro de nuevo morador.

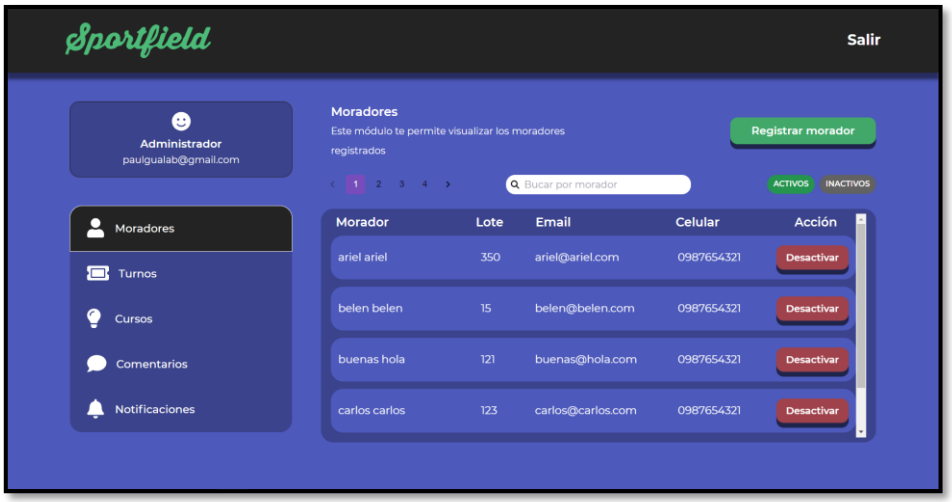

Fig. 38: Interfaz principal del usuario administrador.

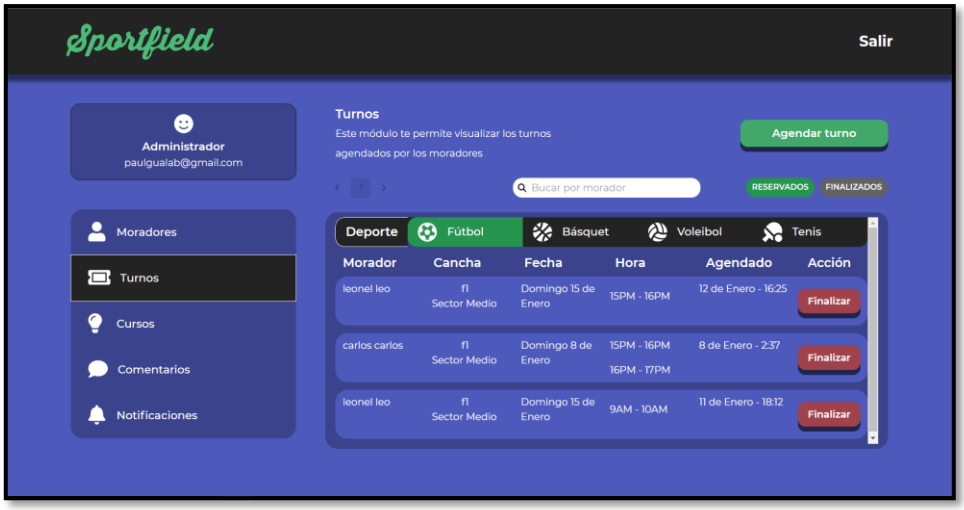

Fig. 39: Interfaz para la gestión de turnos.

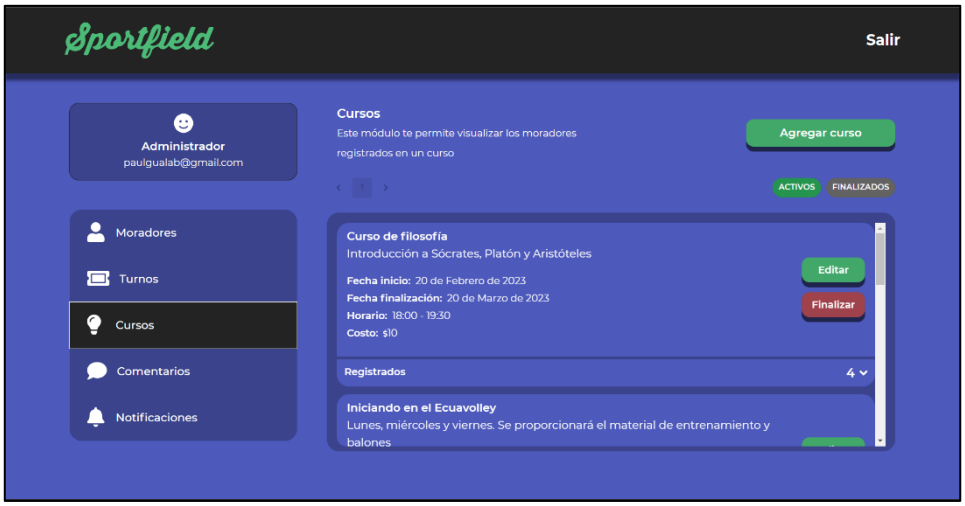

Fig. 40: Interfaz para la gestión de cursos.

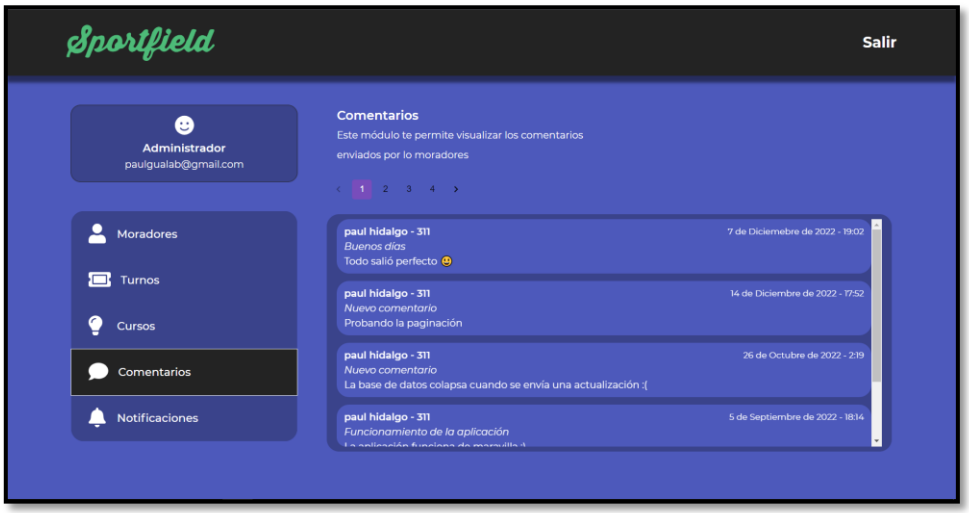

Fig. 41: Interfaz para la visualización de comentarios enviados por los moradores.

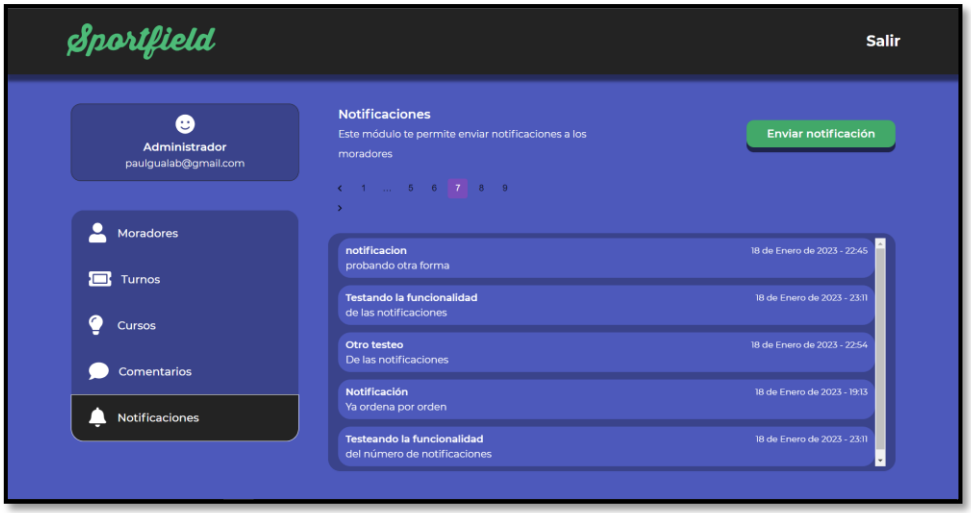

Fig. 42: Interfaz para la gestión de notificaciones.

Fig. 43: Interfaz principal del usuario morador.

Fig. 44: Interfaz para la agenda de un turno.

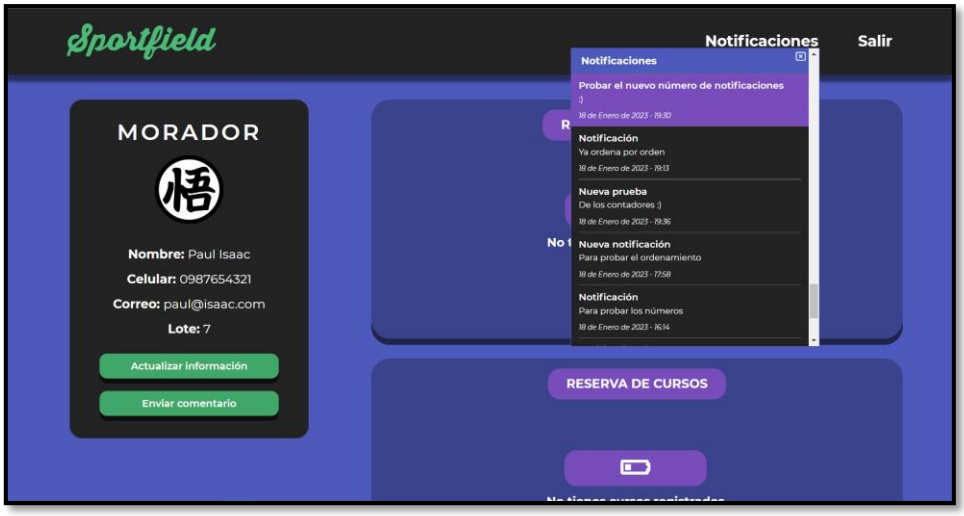

Fig. 45: Interfaz de visualización de notificaciones.

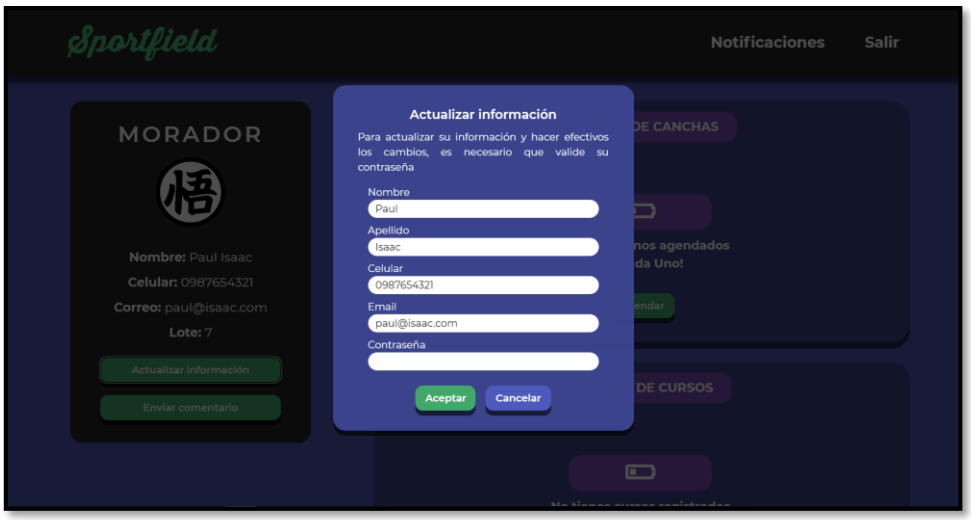

Fig. 46: Interfaz del usuario morador para editar su información.

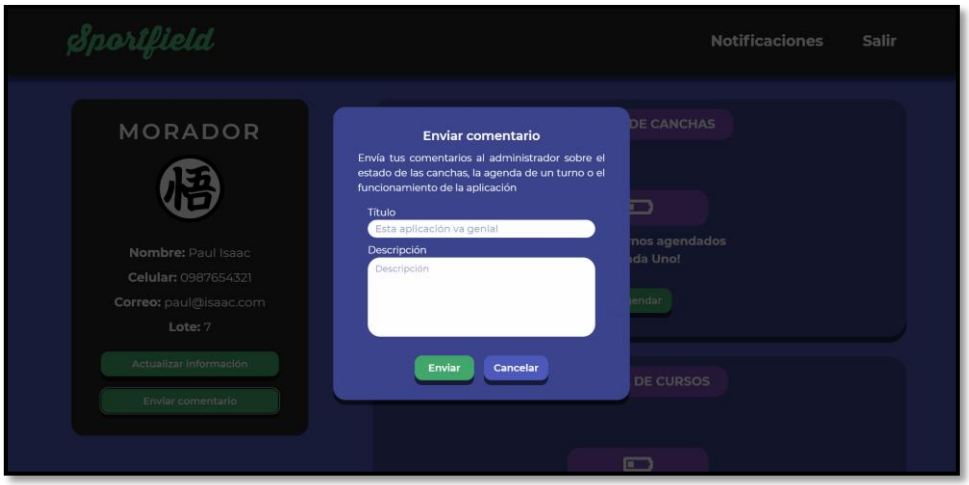

Fig. 47: Interfaz del usuario morador para enviar un comentario.

Pruebas

Al completarse la etapa de desarrollo se aplican las pruebas pertinentes para asegurar que el funcionamiento del *frontend* sea el deseado en base a lo estipulado por la recopilación de requerimientos y el dueño del producto.

Pruebas unitarias

Su finalidad es testear el comportamiento y los resultados de una funcionalidad en concreto. De este modo, desde la **Fig. 48** hasta la **Fig. 51** se presenta la estructura de las pruebas unitarias restantes que han sido elaboradas por cada requerimiento establecido, así como el resultado de la ejecución cada prueba.

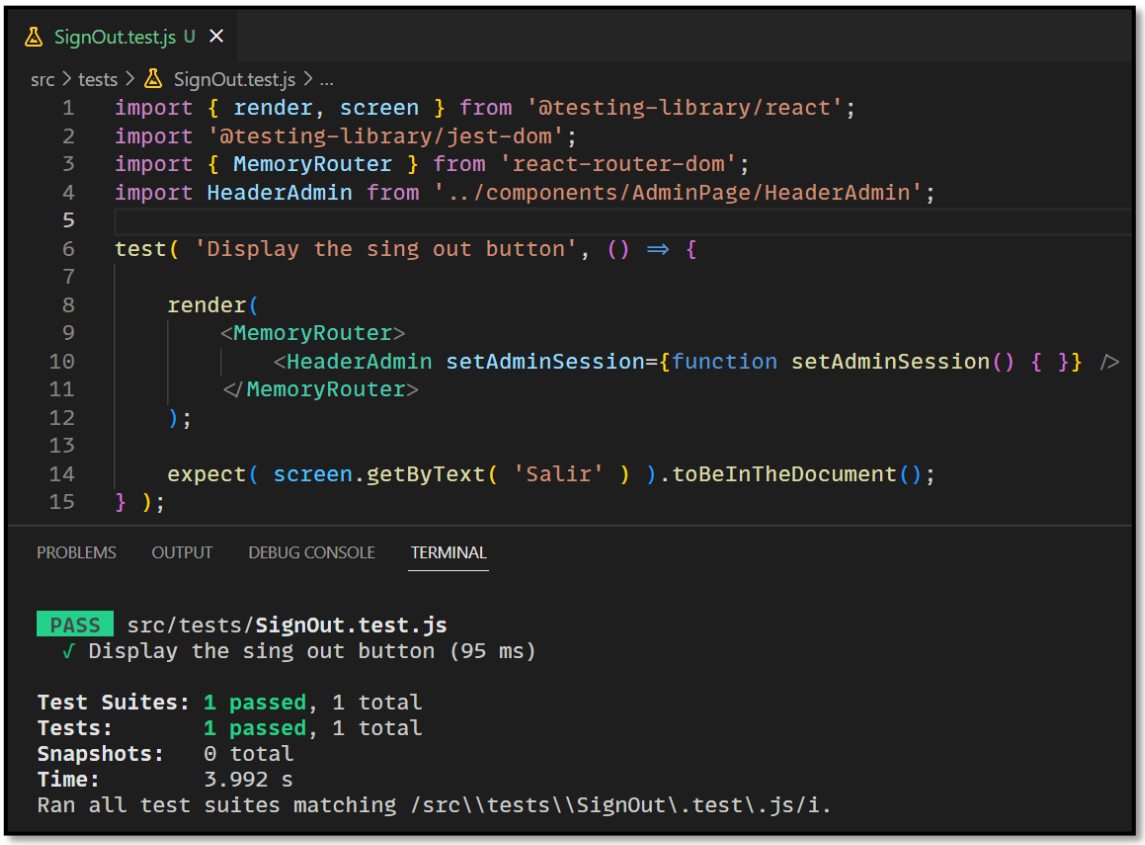

Fig. 48: Prueba de renderización de la función para cerrar sesión.

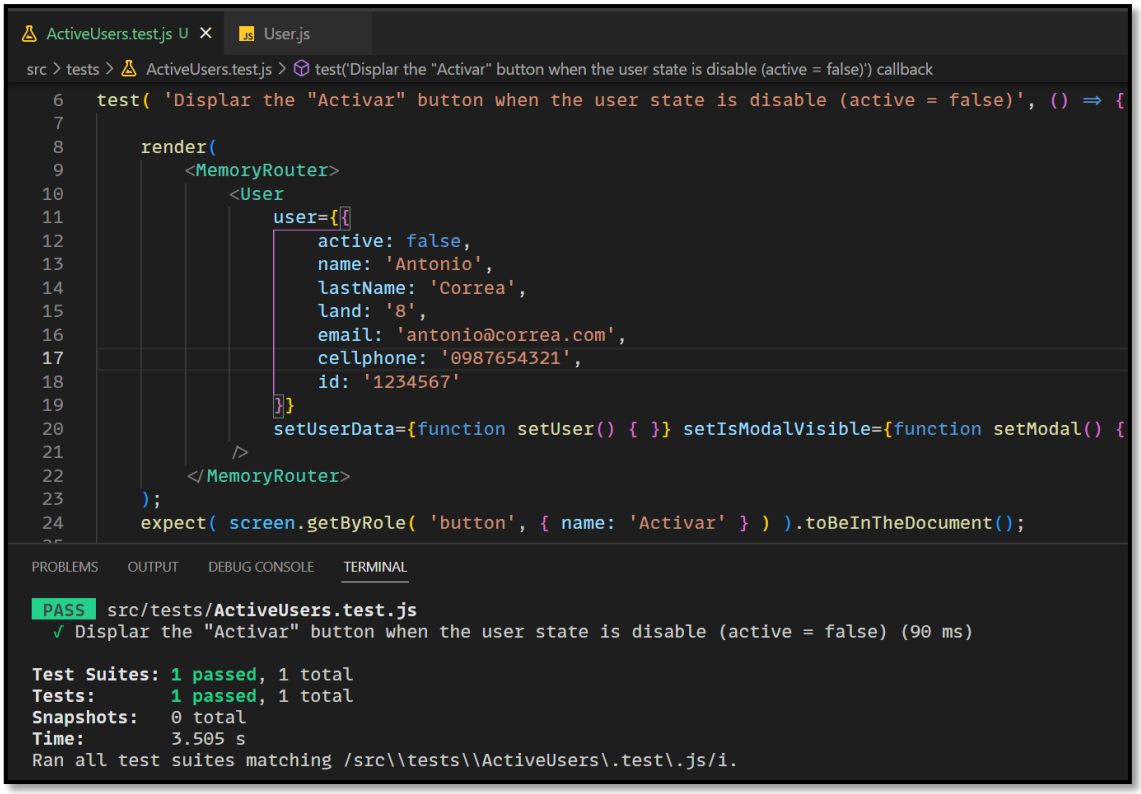

Fig. 49: Prueba de renderización de la función para activar un usuario.

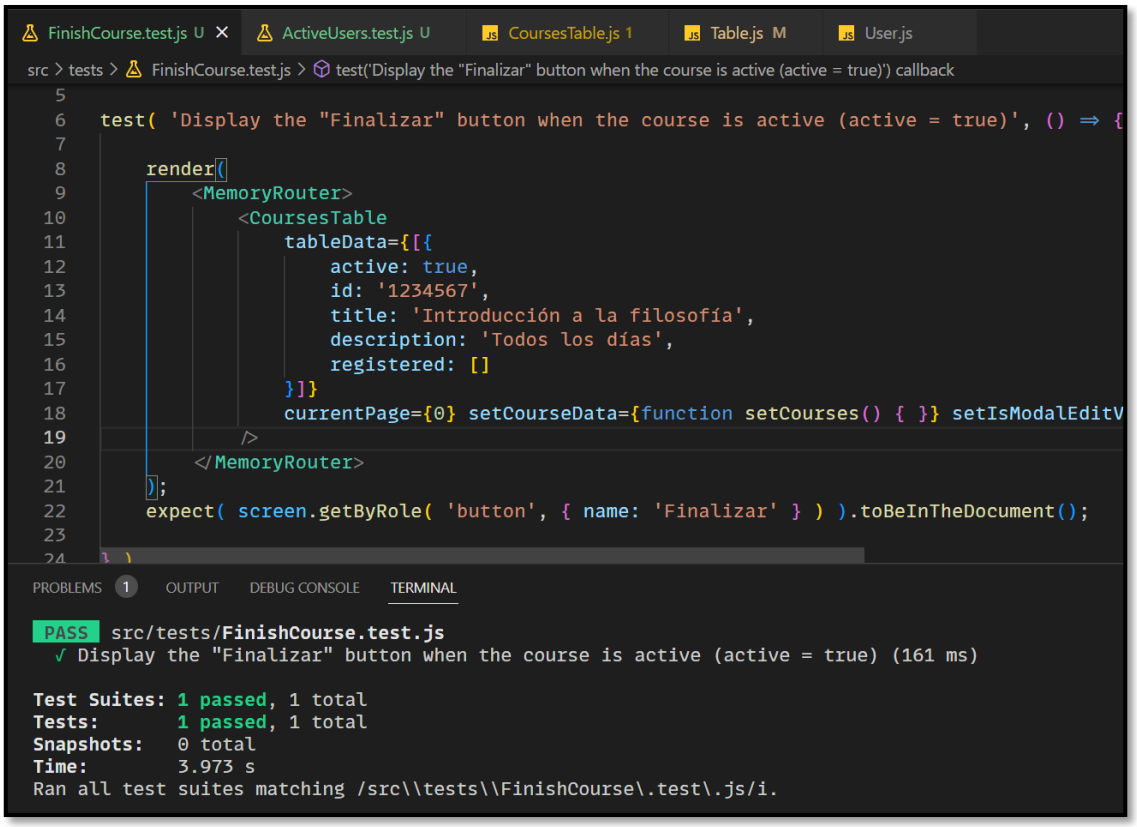

Fig. 50: Prueba de renderización de la función para finalizar un curso.

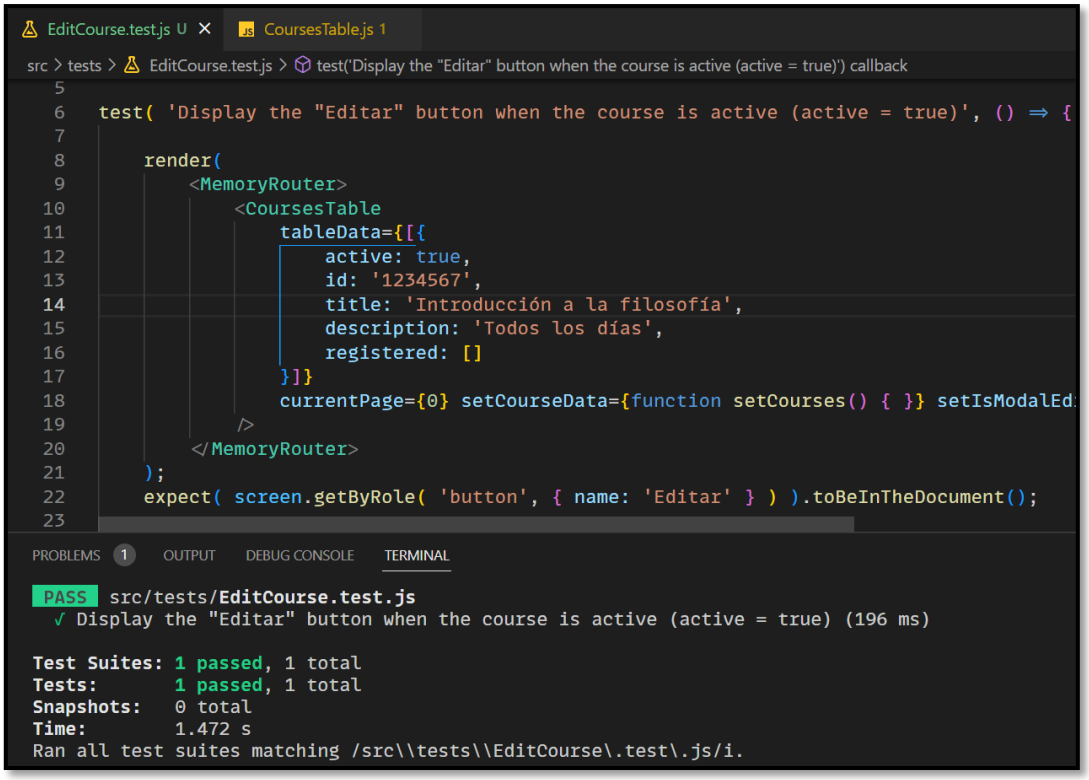

Fig. 51: Prueba de renderización de la función para editar un curso.

Pruebas de compatibilidad

En el *frontend*, estas pruebas se centran en comprobar las diversas funcionalidades en los principales navegadores con el objetivo de visualizar los posibles problemas al momento de presentar el contenido al usuario.

De este modo, desde la **Fig. 52** hasta la **Fig. 69** se visualizan los resultados que se han logrado obtener en los diferentes navegadores.

Fig. 52: *Landing page* de *Sportfield* en el navegador *Firefox.*

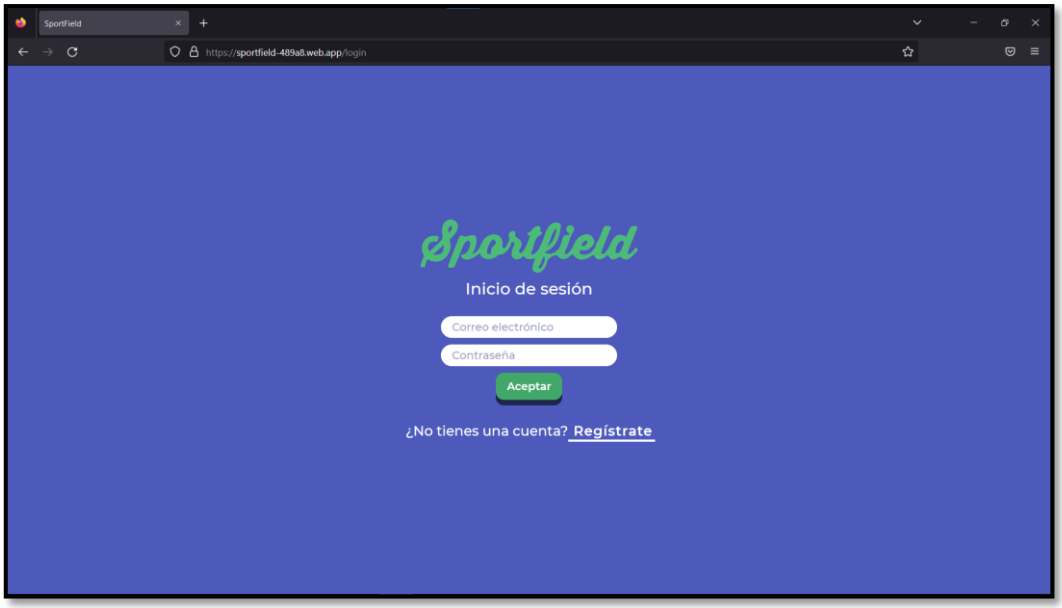

Fig. 53: Página de inicio de sesión de *Sportfield* en el navegador *Firefox.*

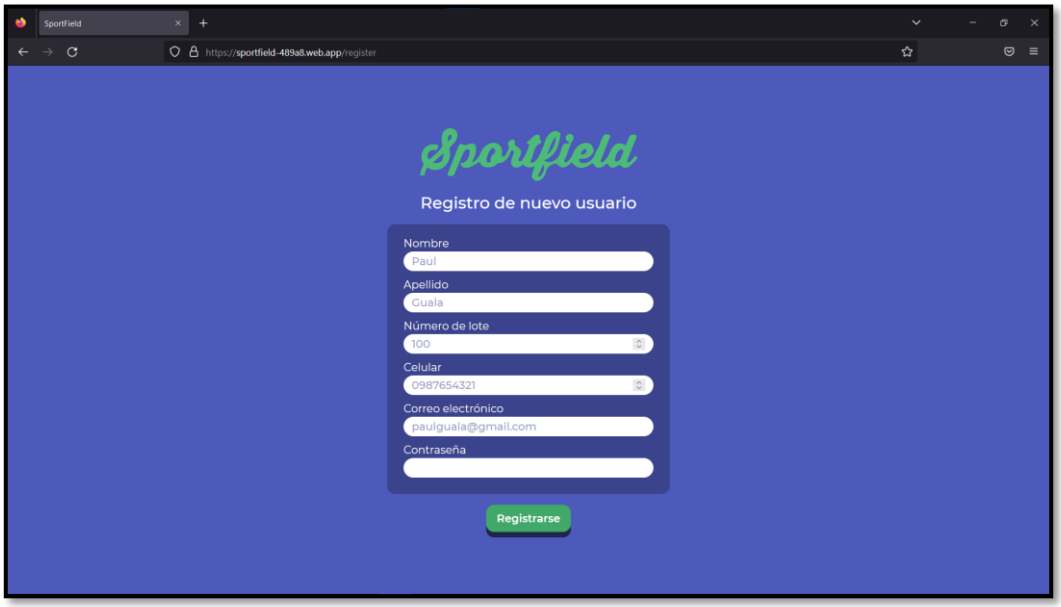

Fig. 54: Página de registro de *Sportfield* en el navegador *Firefox.*

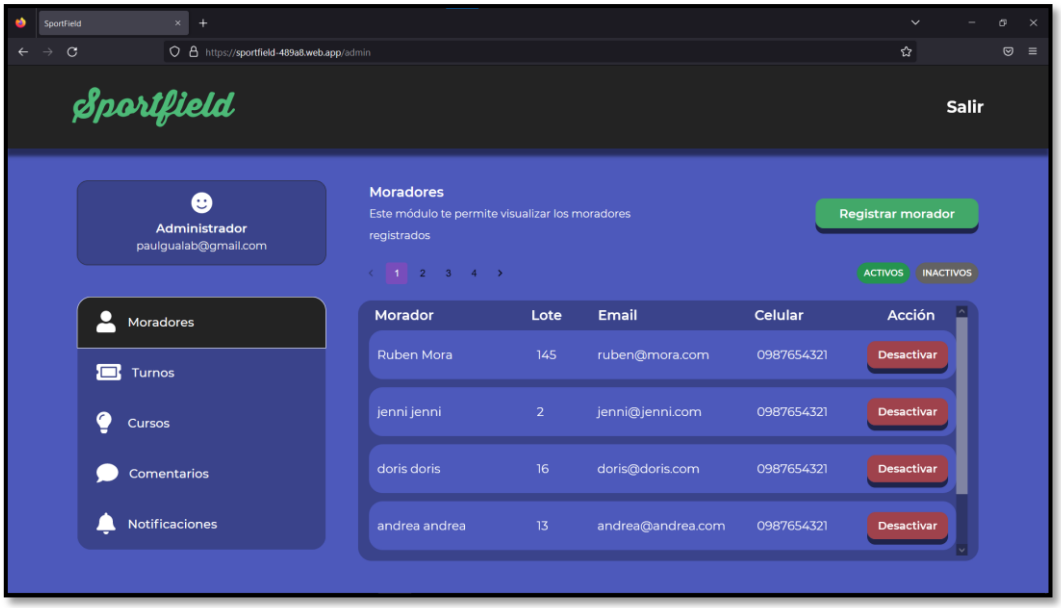

Fig. 55: Página principal del usuario administrador en el navegador *Firefox.*

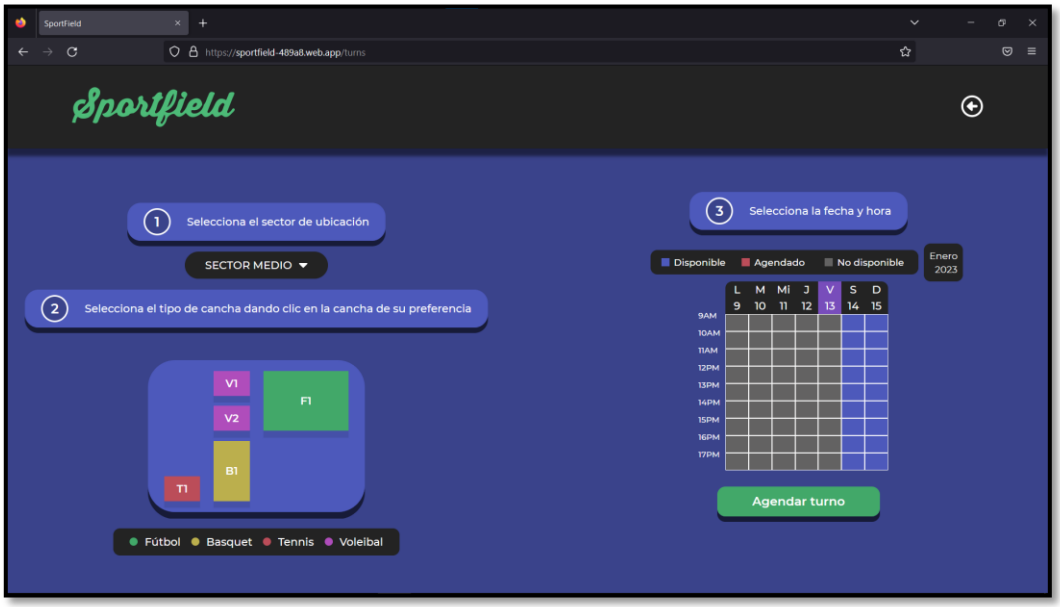

Fig. 56: Página para la reserva de un turno en el navegador *Firefox.*

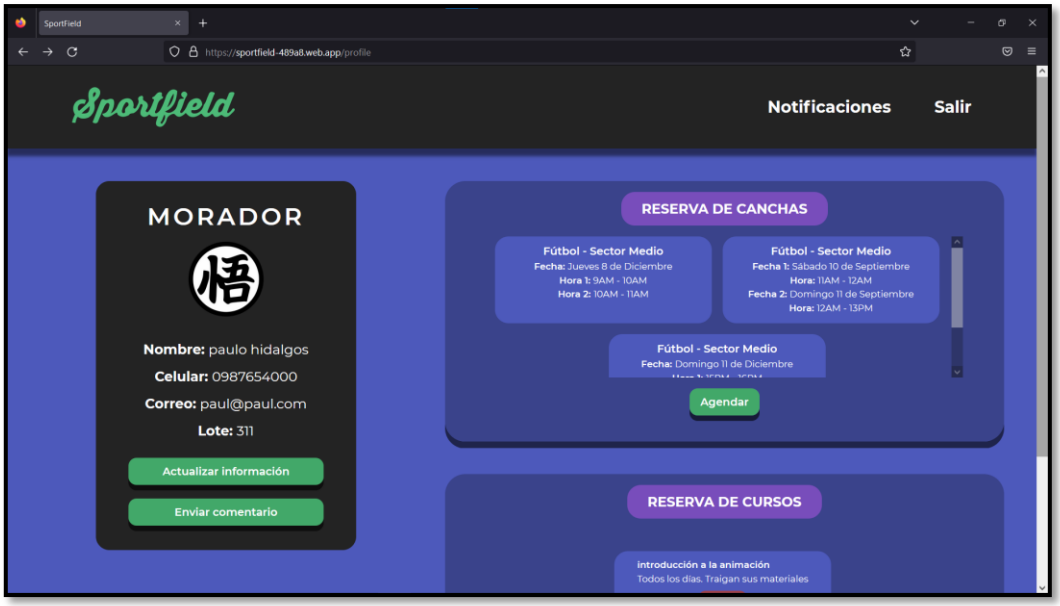

Fig. 57: Página principal del usuario morador en el navegador *Firefox.*

Fig. 58: *Landing page* de *Sportfield* en el navegador *Chrome.*

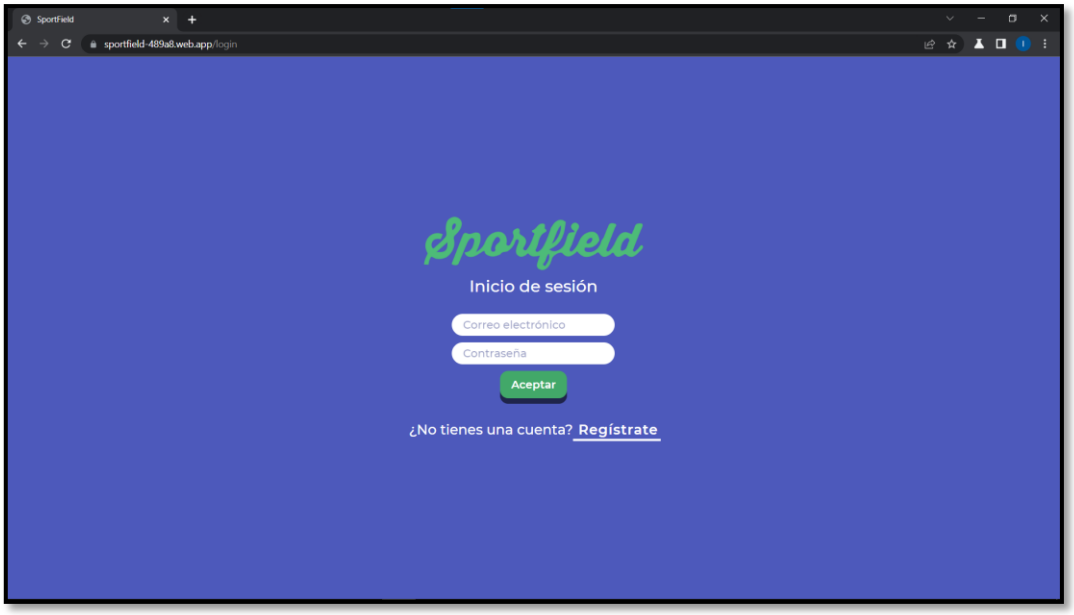

Fig. 59: Página de inicio de sesión de *Sportfield* en el navegador *Chrome.*

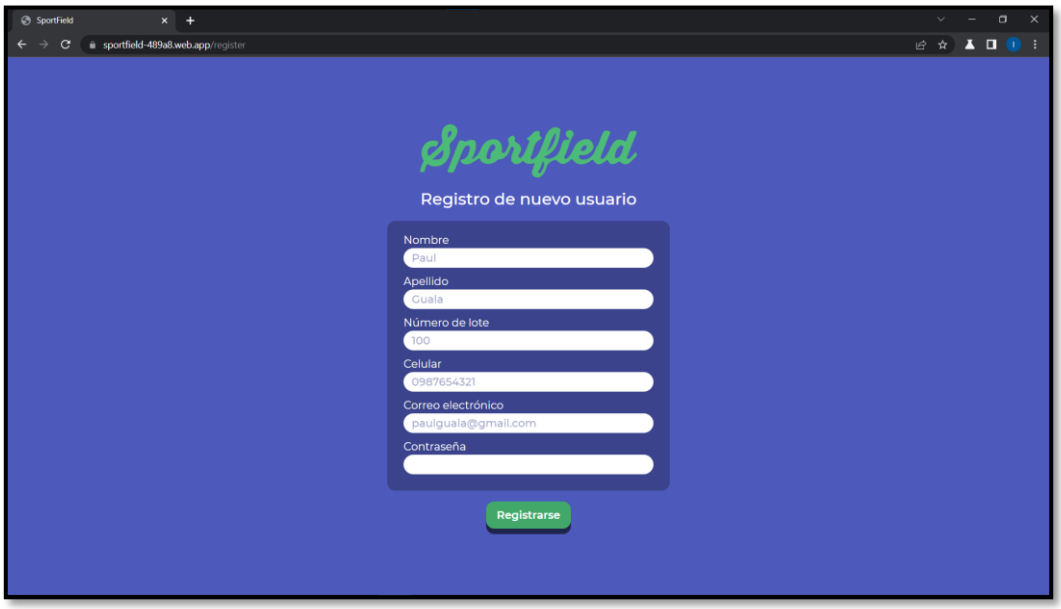

Fig. 60: Página de registro de *Sportfield* en el navegador *Chrome.*

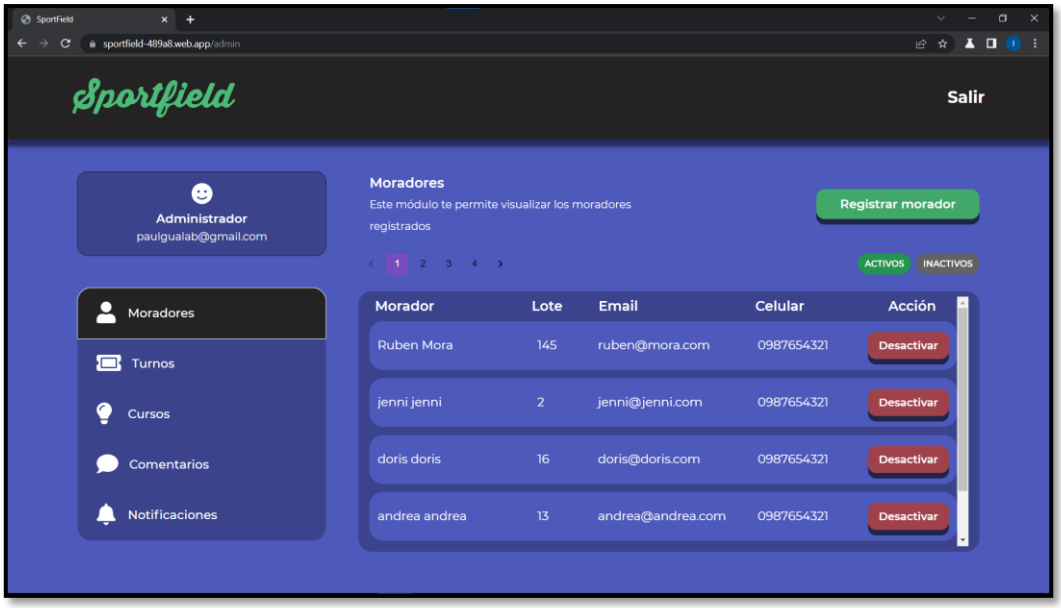

Fig. 61: Página principal del usuario administrador en el navegador *Chrome.*

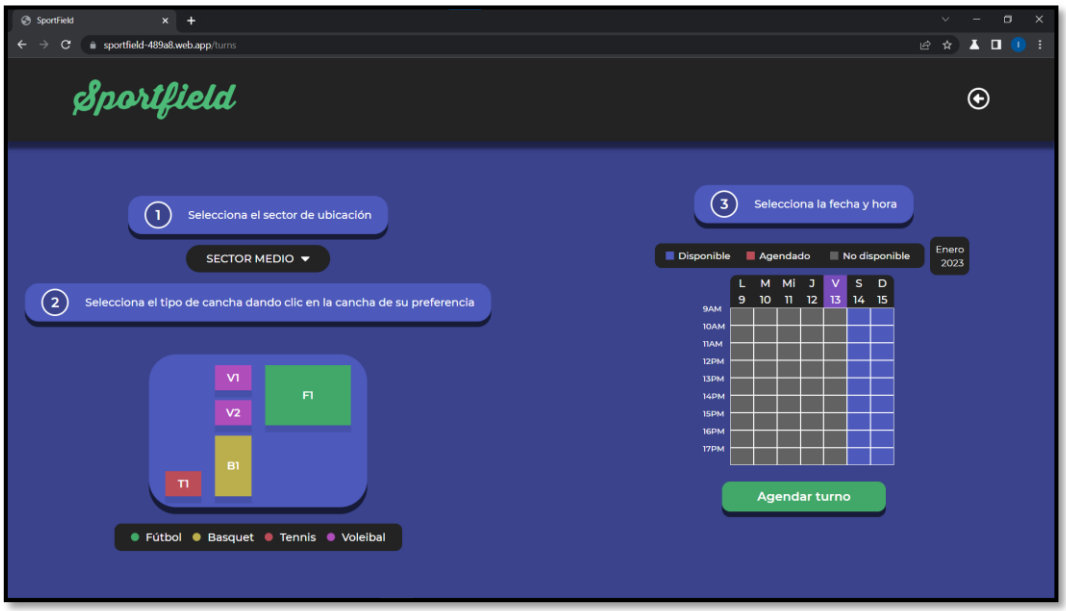

Fig. 62: Página para la reserva de un turno en el navegador *Chrome.*

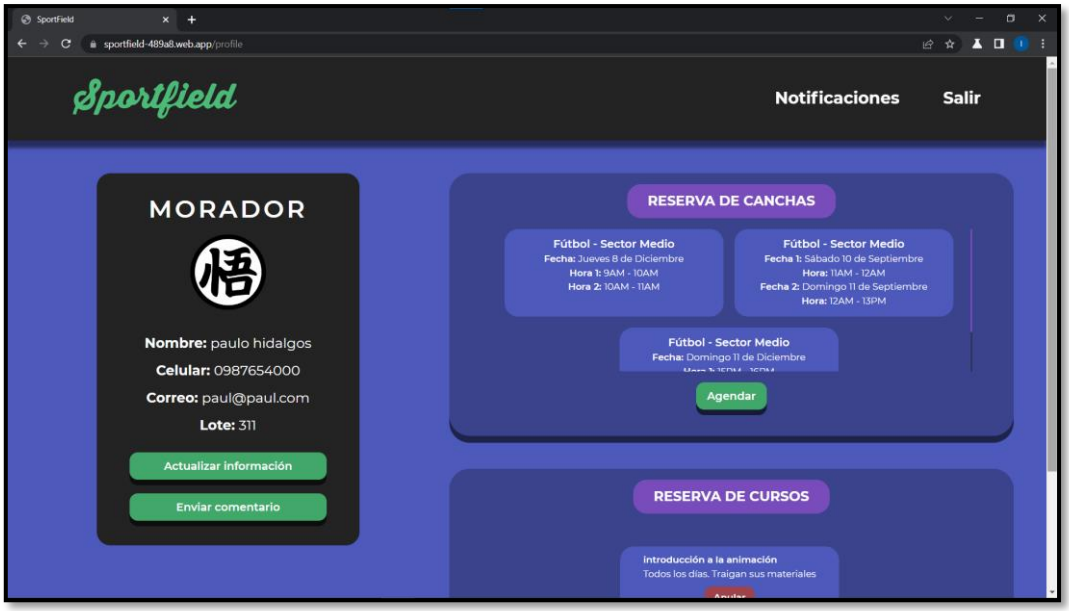

Fig. 63: Página principal del usuario morador en el navegador *Chrome.*

Fig. 64: *Landing page* de *Sportfield* en el navegador *Microsoft Edge.*

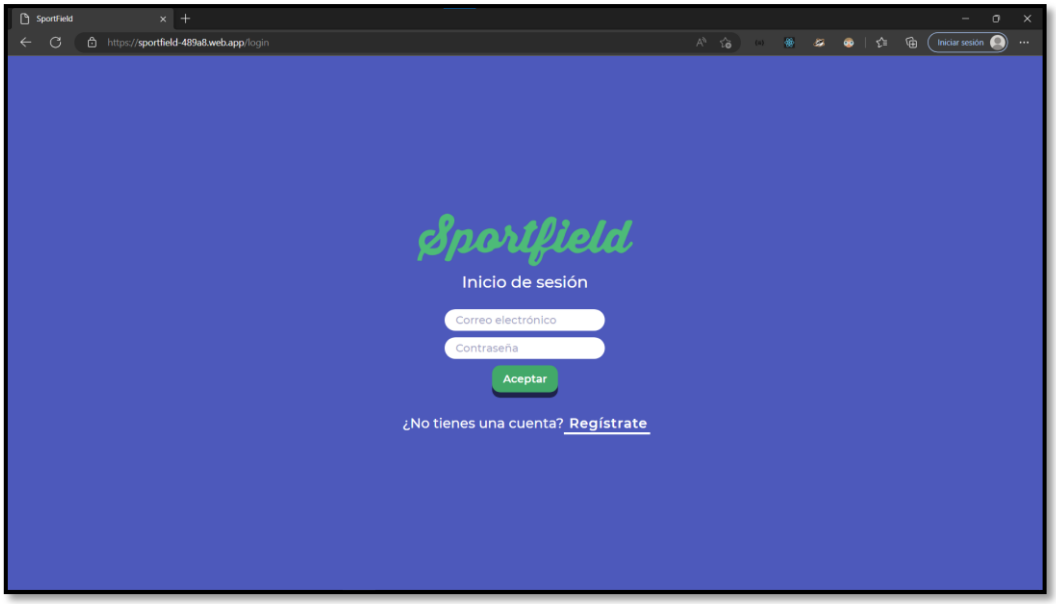

Fig. 65: Página de inicio de sesión de *Sportfield* en el navegador *Microsoft Edge.*

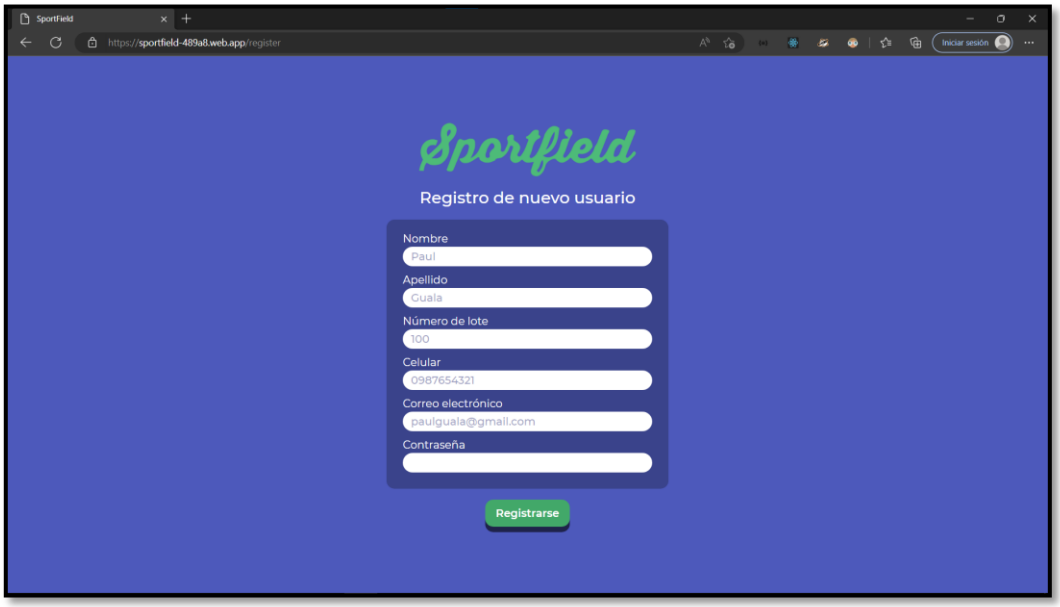

Fig. 66: Página de registro de *Sportfield* en el navegador *Microsoft Edge.*

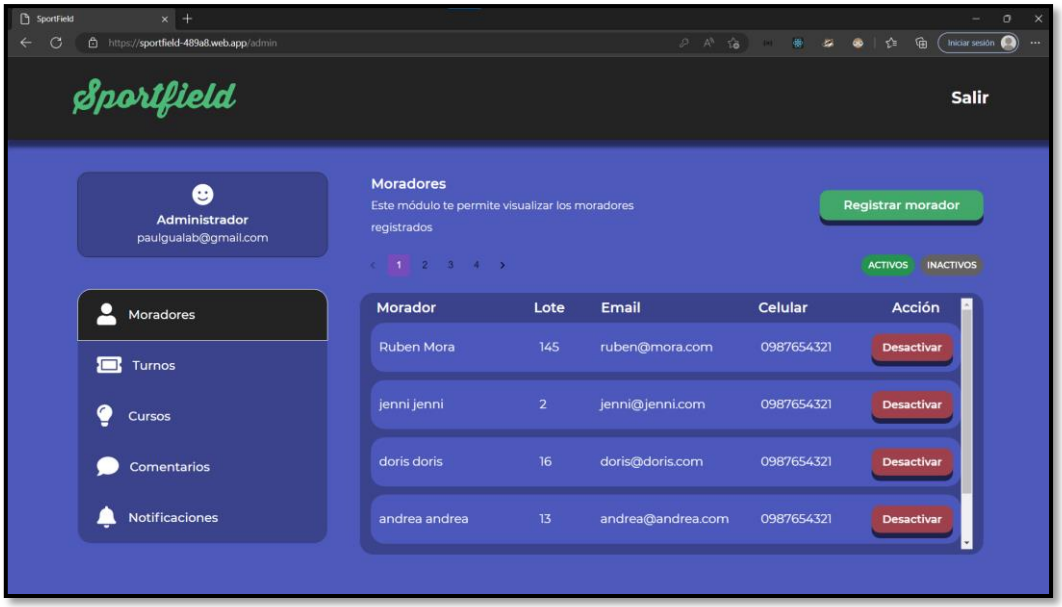

Fig. 67: Página principal del usuario administrador en el navegador *Microsoft Edge.*

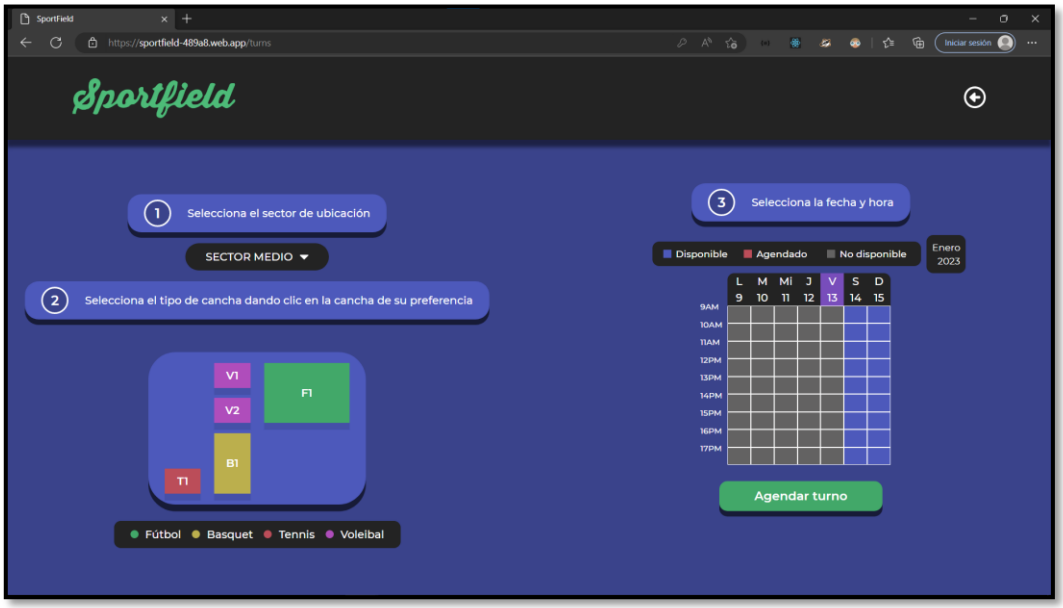

Fig. 68: Página para la reserva de un turno en el navegador *Microsoft Edge.*

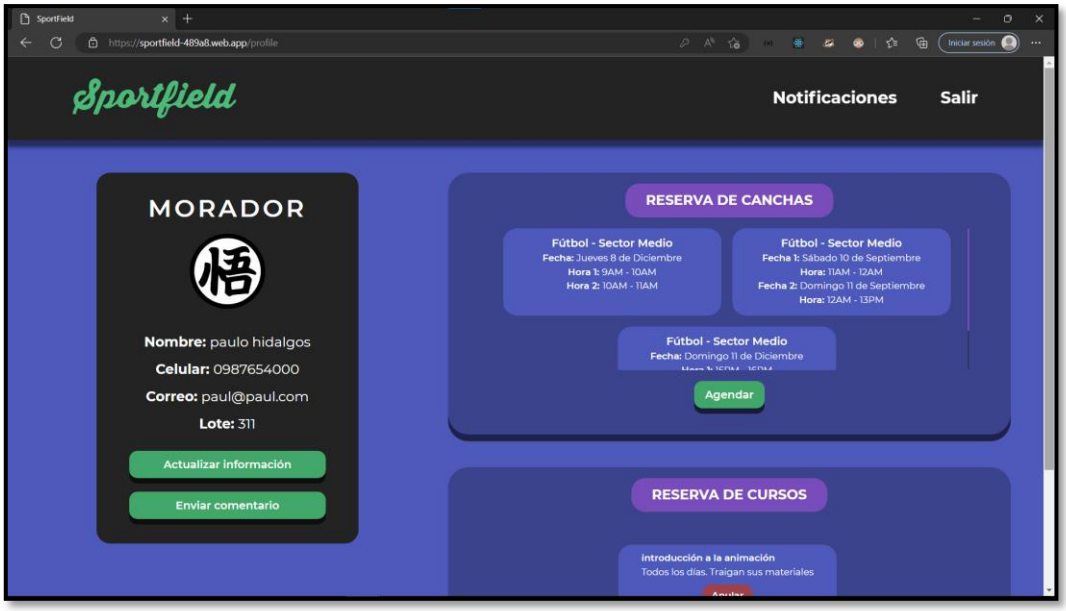

Fig. 69: Página principal del usuario morador en el navegador *Microsoft Edge.*

Pruebas de aceptación

A continuación, se procede a mostrar las 12 pruebas de aceptación, las cuales van desde la **TABLA XXIV** hasta la **TABLA XXXIV.** Es importante mencionar que todas las pruebas describen el proceso que corresponde a las diferentes tareas asignadas de los usuarios del *frontend* para su correcta funcionalidad y verificación de cada una de ellas.

TABLA XXIV: Prueba de aceptación Iniciar y cerrar sesión.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA01	**Identificador de historia de Usuario:** HU01
Nombre: Iniciar sesión y cerrar sesión	
Descripción: Los usuarios Administrador y Morador pueden iniciar sesión con sus credenciales y cerrar sesión	
Pasos de ejecución: Para iniciar sesión, los usuarios administrador y morador deben: • Colocar la URL del *frontend* en el navegador • Dar clic en la opción “Iniciar sesión”, ubicada en la parte superior derecha de la interfaz • Ingresar el correo electrónico y contraseña respectivos, dependiendo del usuario	

- Dar clic en “Aceptar”.

Para cerrar sesión:

- Haber Iniciado sesión en el *frontend.*
- Dar clic en la opción “Salir”, ubicada en la parte superior derecha de la página

Resultado deseado:

El *frontend Sportfield* permite iniciar sesión y cerrar sesión a los usuarios administrador y morador.

Evaluación de la prueba:

Se evidencia el resultado esperado.Conformidad del cliente 100%.

TABLA XXV: Prueba de aceptación gestionar moradores.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA01	**Identificador de historia de Usuario:** HU02
Nombre: Gestionar moradores	
Descripción: El administrador puede registrar, activar y desactivar moradores	

Pasos de ejecución:

Para registrar un morador, el usuario administrador debe:

- Iniciar sesión
- Dar clic en el botón “Registrar morador”
- Ingresar los campos de registro del morador
- Dar clic en el botón “Aceptar”.

Para activar un morador, el usuario administrador debe:

- Iniciar sesión
- Seleccionar la opción “Inactivos” ubicada en la parte superior derecha de la tabla
- Dar clic en el botón “Activar” de la fila deseada

Para desactivar un morador, el usuario administrador debe:

- Iniciar sesión
- Dar clic en el botón “Desactivar” de la fila deseada en la tabla de moradores
- Dar clic en el botón “Desactivar”

Resultado deseado:

El *frontend Sportfield* permite registrar, activar y desactivar moradores al usuario administrador.

Evaluación de la prueba:
Se evidencia el resultado esperado.Conformidad del cliente 100%.

TABLA XXVI: Prueba de aceptación gestionar turnos.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA03	**Identificador de historia de Usuario:** HU03
Nombre: Gestionar turnos	
Descripción: El administrador puede reservar y finalizar turnos	

Pasos de ejecución:

Para reservar un turno, el usuario administrador debe:

- Iniciar sesión
- Seleccionar la opción “Turnos” en el menú ubicado en la parte izquierda de la interfaz
- Dar clic en el botón “Agendar turno”
- Seleccionar el morador al que se le desea agendar un turno.
- Dar clic en el botón “Aceptar”
- Seleccionar el sector de ubicación en el menú ubicado en el paso número uno de la interfaz
- Seleccionar el tipo de cancha en el paso número dos de la interfaz

• Seleccionar la fecha y hora en la tabla ubicada en el paso número tres de la interfaz • Dar clic en el botón “Agendar turno” • Dar clic en el botón “Aceptar” Para finalizar un turno, el usuario administrador debe: • Iniciar sesión • Seleccionar la opción “Turnos” en el menú ubicado en la parte izquierda de la interfaz • Dar clic en el botón “Finalizar” de la fila deseada en la tabla de turnos • Dar clic en el botón “Finalizar”
Resultado deseado: El *frontend Sportfield* permite reservar y finalizar turnos al usuario administrador.
Evaluación de la prueba: Se evidencia el resultado esperado.Conformidad del cliente 100%.

TABLA XXVII: Prueba de aceptación gestionar cursos.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA04	**Identificador de historia de Usuario:** HU04
Nombre: Gestionar cursos	
Descripción: El administrador puede ofertar, editar y finalizar cursos	
Pasos de ejecución: Para ofertar un curso, el usuario administrador debe: • Colocar la URL del *frontend* en el navegador • Seleccionar la opción “Cursos” en el menú ubicado en la parte izquierda de la interfaz • Dar clic en el botón “Agregar curso”	

• Llenar todos los campos solicitados para agregar un curso • Dar clic en el botón “Aceptar” Para editar un curso, el usuario administrador debe: • Iniciar sesión • Seleccionar la opción “Cursos” en el menú ubicado en la parte izquierda de la interfaz • Dar clic en el botón “Editar” de la fila deseada en la tabla de cursos • Dar clic en el botón “Finalizar” Para finalizar un curso, el usuario administrador debe: • Iniciar sesión • Seleccionar la opción “Cursos” en el menú ubicado en la parte izquierda de la interfaz • Dar clic en el botón “Finalizar” de la fila deseada en la tabla de cursos • Dar clic en el botón “Finalizar”
Resultado deseado: El *frontend Sportfield* permite ofertar, editar y finalizar cursos al usuario administrador.
Evaluación de la prueba: Se evidencia el resultado esperado.Conformidad del cliente 100%.

TABLA XXVIII: Prueba de aceptación visualizar comentarios.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA06	**Identificador de historia de Usuario:** HU06
Nombre: Visualizar comentarios	
Descripción: El administrador puede visualizar los comentarios enviados por los moradores	
Pasos de ejecución:	

Para visualizar comentarios, el usuario administrador debe: • Iniciar sesión • Seleccionar la opción “Comentarios” en el menú ubicado en la parte izquierda de la interfaz
Resultado deseado: El *frontend Sportfield* permite visualizar los comentarios enviados por los moradores al usuario administrador.
Evaluación de la prueba: Se evidencia el resultado esperado.Conformidad del cliente 100%.

TABLA XXIX: Prueba de aceptación registrar morador.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA07	**Identificador de historia de Usuario:** HU07
Nombre: Registrar morador	
Descripción: El morador puede registrarse en el *frontend*	
Pasos de ejecución: Para registrarse en el *frontend*, el usuario morador debe: • Colocar la URL del *frontend* en el navegador • Dar clic en la opción “Registrarse”, ubicada en la parte superior derecha de la interfaz • Llenar todos los campos de registro • Dar clic en el botón “Registrarse”	
Resultado deseado: El *frontend Sportfield* permite registrarse al usuario morador.	
Evaluación de la prueba: Se evidencia el resultado esperado.Conformidad del cliente 100%.	

TABLA XXX: Prueba de aceptación reservar turno.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA08	**Identificador de historia de Usuario:** HU08
Nombre: Reservar turno	
Descripción: El morador puede reservar un turno con la cancha y fecha seleccionada	
Pasos de ejecución: Para reservar un turno, el usuario morador debe: • Iniciar sesión • Dar clic en el botón “Agendar” ubicado en la tabla de “Reserva de canchas” • Seleccionar el sector de ubicación en el menú ubicado en el paso número uno de la interfaz • Seleccionar el tipo de cancha en el paso número dos de la interfaz • Seleccionar la fecha y hora en la tabla ubicada en el paso número tres de la interfaz • Dar clic en el botón “Agendar turno” • Dar clic en el botón “Aceptar”	
Resultado deseado: El *frontend Sportfield* permite reservar turnos al usuario morador.	
Evaluación de la prueba: Se evidencia el resultado esperado.Conformidad del cliente 100%.	

TABLA XXXI: Prueba de aceptación visualizar y registrar cursos.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA09	**Identificador de historia de Usuario:** HU09
Nombre: Visualizar y registrar cursos	
Descripción: El administrador puede visualizar y registrarse en un curso ofertado	

Pasos de ejecución: Para visualizar cursos, el usuario morador debe: • Colocar la URL del *frontend* en el navegador • Dirigirse a la tabla “Cursos ofertados” ubicada en la sección inferior de la interfaz Para registrar un curso, el usuario morador debe: • Iniciar sesión • Dar clic en el botón “Agendar” ubicado en la tabla de “Reserva de cursos” • Seleccionar el curso en que se desea registrarse • Dar clic en el botón “Registrarse”
Resultado deseado: El *frontend Sportfield* permite visualizar y registrar cursos al usuario morador.
Evaluación de la prueba: Se evidencia el resultado esperado.Conformidad del cliente 100%.

TABLA XXXII: Prueba de aceptación enviar comentarios.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA10	**Identificador de historia de Usuario:** HU10
Nombre: Enviar comentarios	
Descripción: El morador puede enviar comentarios y sugerencias al administrador	
Pasos de ejecución: Para enviar un comentario, el usuario morador debe: • Iniciar sesión • Dar clic en el botón “Enviar comentario”, ubicado en la parte inferior del panel de datos del morador • Llenar todos los campos de envío	

• Dar clic en el botón “Enviar”
Resultado deseado: El *frontend Sportfield* permite enviar comentarios al usuario morador.
Evaluación de la prueba: Se evidencia el resultado esperado.Conformidad del cliente 100%.

TABLA XXXIII: Prueba de aceptación visualizar notificaciones.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA11	**Identificador de historia de Usuario:** HU11
Nombre: Visualizar notificaciones	
Descripción: El morador puede visualizar las notificaciones enviadas por el administrador	
Pasos de ejecución: Para visualizar notificaciones, el usuario morador debe: • Iniciar sesión • Dar clic en el botón “Notificaciones”, ubicado en la parte superior derecha de la interfaz	
Resultado deseado: El *frontend Sportfield* permite visualizar notificaciones al usuario morador.	
Evaluación de la prueba: Se evidencia el resultado esperado.Conformidad del cliente 100%.	

TABLA XXXIV: Prueba de aceptación editar información.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA12	**Identificador de historia de Usuario:** HU12
Nombre: Editar información	

Descripción: El morador puede modificar su información personal de registro
Pasos de ejecución: Para editar su información personal, el usuario morador debe: • Iniciar sesión • Dar clic en el botón “Editar información”, ubicado en la parte inferior del panel de datos del morador • Modificar los campos que requiera editar • Ingresar su contraseña de registro e inicio de sesión • Dar clic en el botón “Aceptar”
Resultado deseado: El *frontend Sportfield* permite editar la información personal al usuario morador.
Evaluación de la prueba: Se evidencia el resultado esperado.Conformidad del cliente 100%.

ANEXO III

A continuación, para visualizar el Manual de Usuario del *frontend* se debe digitar la siguiente URL:

https://www.youtube.com/watch?v=5NagvW0Xcpc&t=401s

En donde se explica de forma clara y sencilla las diversas funcionalidades del *frontend*, así como cada uno de los perfiles que forman parte de este componente.

ANEXO IV

A continuación, se presenta el encalce de acceso al *frontend*, además del repositorio de GitHub donde se encuentra alojado todo el código del proyecto.

Enlace de acceso al *frontend*.

Para ingresar al *frontend* ya en producción, se ingresa mediante la URL:

SportField (sportfield-489a8.web.app)

Repositorio del *frontend*

El proyecto se encuentra en un repositorio de *GitHub,* que se accede mediante la siguiente URL:

https://github.com/paulisaac116/Sportfield1.git