PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

FACULTAD DE HÁBITAT, INFRAESTRUCTURA Y CREATIVIDAD

CARRERA EN SISTEMAS DE INFORMACIÓN



# Desarrollo de un prototipo de página web para la reserva de canchas deportivas

AUTORA:

AMEL MENESES

DIRECTORA:

Dra.  SUSANA MASAPANTA CARRIÓN

QUITO DM , JUNIO  2025

## Tabla de Contenidos


*1. CAPÍTULO I: INTRODUCCIÓN ..... 6*

**1.1. Justificación ..... 6**

**1.2. Planteamiento del problema ..... 6**

**1.3. Antecedentes ..... 7**

**1.4. Objetivos ..... 8**

1.4.1. Objetivo general ..... 8

1.4.2. Objetivos específicos ..... 8

**1.5. Alcance ..... 8**

*2. CAPÍTULO II: FUNDAMENTACIÓN TEÓRICA ..... 9*

**2.1. Marco teórico ..... 9**

2.1.1. Principios de diseño centrado en el usuario (UX/UI) ..... 9

2.1.2. Herramientas para el desarrollo de prototipos interactivos ..... 10

2.1.3. Impacto de la gestión deportiva ..... 17

*3. CAPÍTULO III: DESARROLLO Y ANALISIS ..... 20*

**3.1. Diseño del prototipo ..... 20**

3.1.1. Diseño Conceptual ..... 20

3.1.2. Metodología de desarrollo ..... 20

3.1.3. Análisis de requerimientos ..... 22

3.1.4. Diagramas de caso de uso ..... 24

3.1.4.1. Caso de uso general ....24
3.1.4.2. Caso de uso F0 – Iniciar sesión....25
3.1.4.3. Caso de uso F1 – Registro de usuario....25
3.1.4.4. Diagrama de Caso de Uso F2 – Reservas de Canchas....26
3.1.4.5. Caso de uso F3 –Pago simulado ....27
3.1.4.6. Caso de uso F4 –Gestión de empresas ....28
3.1.4.7. Caso de uso F5 - Gestión de usuarios ....29
3.1.5. Diagramas de flujo y especificaciones técnicas ....30
3.1.5.1. Diagrama de flujo Principal del sistema ....30
3.1.5.2. Diagrama de Flujo a detalle de inició de sesión ....32
3.1.5.3. Diagrama de Flujo a detalle de registro de sesión ....33
3.1.5.4. Diagrama de Flujo Integrado –Reservas ....34
3.1.5.5. Diagrama de Flujo a detalle de gestión de usuarios ....35
3.1.5.6. Diagrama de Flujo de gestión de empresas ....35
3.1.6. Diagrama conceptual de base de datos ....36
3.1.7. Diseño de la Interfaz gráfica ....37
**3.2. Desarrollo del prototipo funcional ....38**
**3.3. Pruebas funcionales del sistema ....45**
3.3.1. Prueba FT-01: Registro de un usuario exitoso ....45
Precondición: No debe existir una cuenta con el correo indicado. ....45
3.3.2. Prueba FT-02: Activación de cuenta....46
***4. . CAPÍTULO IV: CONCLUSIONES Y RECOMENDACIONES....51***
**4.1. Conclusiones ....51**
**4.2. Recomendaciones ....51**

*Bibliografías* ..... *52*

## Indice de figuras y tablas

*Figura 1: Tablero Kanban en herramienta Jira* ..... *22*

*Figura 2: Caso de uso general* ..... *24*

*Figura 3: Caso de uso Iniciar sesión* ..... *25*

*Figura 4: Caso de uso registro de usuario* ..... *25*

*Figura 5: Caso de Uso Reservas de Canchas* ..... *26*

*Figura 6: Caso de uso Pago simulado* ..... *27*

*Figura 7: Caso de uso gestión empresas* ..... *28*

*Figura 8: Caso de uso gestión de usuarios* ..... *29*

*Figura 9: Flujo principal del sistema* ..... *30*

*Figura 10: Flujo inicio de sesión* ..... *32*

*Figura 12: Flujo Reservas de canchas* ..... *34*

*Figura 14: Flujo gestión Empresas* ..... *35*

*Figura 15: Diagrama conceptual de base de datos* ..... *36*

*Figura 15: Lista de tareas y sus estados en Jira* ..... *39*

*Figura 16: Board con el flujo de las tareas dependiendo del desarrollo* ..... *40*

*Figura 17: Interfaz loggin del sistema* ..... *41*

*Figura 18: Interfaz de registro del sistema* ..... *42*

*Figura 19: Interfaz de canchas* ..... *42*

*Figura 20: Interfaz de disponibilidad* ..... *43*

*Figura 21: Interfaz Horarios de reserva* ..... *44*

*Figura 22: Interfaz simulación del pago* ..... *44*

*Figura 23: Prueba de funcionalidad Registrarse* ..... *45*

*Figura 24: Prueba funcional de activación por correo.* ..... *46*

*Figura 25: Prueba funcional local storage* ..... *47*

*Figura 26: Login Validación de errores* ..... *48*

*Figura 27: Pruebas listado de canchas* ..... *49*

# 1. CAPÍTULO I: INTRODUCCIÓN

## 1.1. Justificación

Los deportes son esenciales para el bienestar integral de los seres humanos, no obstante, el acceso a espacios deportivos organizados y disponibles sigue siendo un desafío en muchas comunidades debido a la falta de sistemas de reservas centralizados. En este momento, los métodos tradicionales como las llamadas telefónicas o las reuniones en persona son ineficaces, crean barreras de comunicación y no ofrecen una experiencia de usuario satisfactoria.

El problema planteado anteriormente se busca solucionar mediante una solución digital contemporánea, utilizando el prototipo de reservas de canchas. Este prototipo tiene como propósito concentrar la información de la disponibilidad de las canchas deportivas en Quito e integrar los pasos para realizarlas.

Como se ha evidenciado, un prototipo es una primera aproximación y por eso mismo, este proyecto ofrece nuevas posibilidades a plenas versiones que se podrían realizar en el futuro adaptándose a los requerimientos de la comunidad deportiva en temas de escalabilidad y personalización.

## 1.2. Planteamiento del problema

La administración y la reserva de canchas deportivas aún no han alcanzado un nivel óptimo de eficiencia en el proceso relacionado con la gestión, debido a que en el proceso de reserva se siguen empleando métodos tradicionales dentro de la industria como llamadas telefónicas y citas presenciales, originando problemas como desorganización, dificultad para obtener disponibilidad de espacios, errores en las reservas y escasa publicidad sobre las instalaciones.

Por ello, los usuarios sufren en gran medida al momento de elegir canchas que coincidan con sus horarios, mientras que los que administran el espacio pierden la oportunidad de maximizar la ocupación de sus reservas. Por otro lado, la no automatización limita en gran medida el orden y la gestión sobre el proceso.

### 1.3. Antecedentes

En Quito, hay múltiples plataformas digitales como EasyCancha, Reva y Alpeloteo que simplifican la reserva de canchas deportivas, particularmente para el desarrollo del fútbol, el tenis o pádel. EasyCancha (2025) permite a los usuarios reservar canchas en diferentes ubicaciones, aunque su cobertura en Quito está limitada a ciertos deportes y clubes. Según Reva (2025) , ofrece un sistema que permite a los usuarios gestionar la reserva de canchas y clases, pero se encuentra en una fase de expansión y no proporciona una integración completa además de que se encuentra orientado más hacia empresas que hacía clientes finales, al ser una plataforma internacional no se acopla bien a las necesidades locales. Alpeloteo (2025) también cubre varias canchas en Quito, pero su enfoque esta más centrado en realizar reservas para fútbol y no para los otros deportes además de ser únicamente una aplicación móvil.

Playbooker, por su parte, proporciona una solución más completa y adaptada a las necesidades del mercado de Quito. Mientras las otras opciones muestran disponibilidad de canchas de ciertos deportes, Playbooker enfocado en los deportes más realizados en la ciudad de Quito te permite tener una experiencia mucho más fluida en relación con las disponibilidades en tiempo real y su proceso de reserva optimizando la experiencia del usuario.

## 1.4. Objetivos

### 1.4.1. Objetivo general

Desarrollar un prototipo funcional de página web que permita la reserva de canchas deportivas de manera centralizada, rápida y eficiente, brindando una buena experiencia a los usuarios y la gestión por parte de los administradores.

### 1.4.2. Objetivos específicos

- Diseñar una interfaz centrada en el usuario (UX/UI) que facilite la navegación y el proceso de reserva.
- Implementar funcionalidades clave en el prototipo, como selección de canchas, calendario de disponibilidad y la simulación del pago en línea.
- Realizar las pruebas funcionales y no funcionales del prototipo.

## 1.5. Alcance

El proyecto "Desarrollo de un prototipo de página web para la reserva de canchas deportivas" se enfoca en la creación de la gestión de reservas deportivas. El alcance incluye el diseño de una interfaz de usuario moderna con un enfoque en la experiencia del usuario e implementa las pantallas principales de la plataforma, como la página de inicio, selección de deporte, calendario de disponibilidad y el formulario de reserva. El prototipo permite la gestión de reservas, incluyendo la selección de deporte, visualización de canchas disponibles, selección de fecha y horario, y simula el proceso de pago. El sistema se desarrolla con React para el frontend, Vite y TypeScript para mejorar la eficiencia, y Axios para la comunicación con el backend, gestionado con Node.js, Express y SQLite, con esto concluye el presente trabajo de integración curricular.

# 2. CAPÍTULO II: FUNDAMENTACIÓN TEÓRICA

## 2.1. Marco teórico

### 2.1.1. Principios de diseño centrado en el usuario (UX/UI)

Este principio descrito representa un enfoque que nombra la creación de productos digitales considerando las exigencias generales de los clientes, el cual, en lugar de trabajar solo con la parte técnica o estética, esta metodología pone al usuario en el corazón del proceso de diseño, es decir; se necesita medir el comportamiento de los usuarios para construir soluciones efectivas. Según Gaetán et al., (2017), para lograr productos que sean funcionales y satisfactorios, la sistematización de diseño debería de surgir a partir de la comprensión profunda de las personas y sus objetivos cotidianos.

Por lo cual, la empatía en el (DCU) se establece como uno de los principios más relevantes, debido a que permite al diseñador colocarse en el lugar del usuario final como es el caso, del desarrollo de una plataforma de reservas deportivas, siendo esencial conocer la disponibilidad, la experiencia tecnológica y el equipamiento que poseen los usuarios es fundamental para crear la experiencia ideal. Según Macías & Loor (2023)

En esa línea, esta metodología también incluye el manejo y uso, donde por medio de una interfaz accesible permite que personas con discapacidades visuales, auditivas y motoras interactúen con el sistema sin barreras, aquello es clave para el diseño incluya un contraste adecuado de colores y rotulación, navegación a través del teclado, y texto alternativo a las imágenes para desempeñarse de manera eficiente. Trujillo (2023) menciona que la inclusión digital trasciende la ética, también constituye una mejorada a la experiencia de todos los usuarios con distintas condiciones.

Por ello, la consistencia y la retroalimentación continua constituye el otro principio que señala el hecho que el usuario siempre debe recibir algún tipo de respuesta, ya sea explícita o advertida, contribuyendo a que se mantengan un patrón de visual con gráficos, botones, textos, y navegadores que aporten al usuario aprender de forma fácil. De acuerdo con Hurtado (2024), la consistencia en los aspectos gráficos de un interfaz social reduce el aprendizaje, incrementa la ejecución de tareas y minimiza los errores.

### 2.1.2. Herramientas para el desarrollo de prototipos interactivos

La construcción de prototipos interactivos implica el uso de herramientas modernas que permiten la creación de sistemas totalmente funcionales, elegantes y atractivos, las mismas que abarcan todo el diseño de la interfaz así como los aspectos de ingeniería del producto, lo que hace posible que desarrolladores y diseñadores minimicen el vacío entre el concepto inicial y la ejecución final. Con herramientas especializadas, procesos como la validación de lógica, navegación y conectividad de bases de datos, que son críticos para la funcionalidad de sistemas de reservas de canchas deportivas, son más eficientes (Kuz y Ariste, 2021).

Por lo mencionado, el uso de Figma para el diseño visual y React para el desarrollo de la interfaz, junto con Node.js y Express para el backend, proporciona una arquitectura extremadamente responsiva, mantenible y escalable, donde tecnologías adicionales como TypeScript para reforzar el código, SQLite para la gestión de bases de datos ligeras y GitHub para el control de versiones, establecen un entorno de desarrollo moderno. Estas herramientas no solo aumentan la productividad, sino que también mejoran la experiencia del usuario final al integrar todos los niveles de desarrollo de manera fluida (Hernández y Contreras, 2022).

#### Figma: Herramienta de diseño para crear prototipos

Figma es un recurso de elaboración de interfaces basada en la nube que ha revolucionado la colaboración en el diseño de UI/UX al permitir a múltiples usuarios editar documentos simultáneamente desde cualquier navegador web, esta característica la convierte en una opción ideal para equipos distribuidos o proyectos académicos debido a su interfaz fácil de usar, Figma ha experimentado una adopción explosiva en entornos profesionales y educativos. Como señaló Salazar (2023), las herramientas de diseño colaborativo como Figma agudizan la interacción entre diseñadores y desarrolladores, optimizando los flujos de trabajo en el diseño digital.

Figma otorga diversos beneficios, como es la capacidad de crear prototipos funcionales que simulan el comportamiento real de un sistema web o móvil, esta función permite validar la experiencia de un usuario sin necesidad de codificación preliminar, lo que ahorra tiempo y recursos valiosos, además, el uso de componentes reutilizables asegura consistencia y uniformidad en todos los visuales. Como señalaron Bernhard & Botha (2022), la creación de prototipos en etapas tempranas ayuda a identificar defectos de la presentación en la interfaz antes de incurrir en costosos costos de desarrollo.

Por esa razón, destaca por ofrecer herramientas en el trabajo específico de la especificación técnica del diseño, donde los desarrolladores pueden acceder a las propiedades del elemento visual, sus márgenes, color, tipografía e incluso píxeles, sin ningún archivo adjunto adicional, aquello elimina una serie de errores mientras agiliza el flujo de trabajo de diseño-desarrollo. Robalino (2024), argumenta que la fusión fluida del diseño y la programación aumenta la calidad del producto final mientras reduce los errores de refinamiento.

### React: Framework de JavaScript para construir la interfaz de usuario interactiva

React se constituye como una biblioteca asociada a JavaScript que fue construida por Facebook, permitiendo el desarrollo de diversas interfaces que se asocian los usuarios de forma eficaz a través del uso de componentes reutilizables, su estructura modular facilita la organización del código y proporciona una experiencia fluida para el usuario, donde cada componente puede desarrollarse de forma independiente, reduciendo la complejidad del sistema. Según Díaz (2022), React favorece la escalabilidad del desarrollo web debido a su arquitectura compuesta por componentes reutilizables y jerárquicos.

Una ventaja notable de React es el uso del DOM Virtual, que mejora el rendimiento al eliminar la necesidad de actualizaciones completas de la interfaz, solo las partes que cambian se actualizan, mejorando la capacidad de respuesta, aquello es de vital importancia para aplicaciones interactivas, como sistemas de reservas. Como describe Quintana (2022), el DOM Virtual reduce significativamente el tiempo de carga y mejora la experiencia del usuario final en aplicaciones dinámicas.

En el marco del prototipo para la reserva de campos deportivos, React optimiza la implementación de un conjunto de interfaces para los clientes, desde la selección de fechas, el calendario y los formularios de confirmación, debido a su gran capacidad para actualizar dinámicamente las vistas de acuerdo con las interacciones del usuario es crítica para las experiencias responsivas modernas. Como argumentan Herrera (2023), las interfaces reactivas mejoran las percepciones del usuario sobre el rendimiento del sistema, lo que a su vez aumenta la satisfacción general.

### Vite: Herramienta de desarrollo rápida para optimizar la construcción del frontend con React

Es una herramienta moderna para el desarrollo web que ofrece una experiencia más fluida y rápida durante el desarrollo en comparación con herramientas más antiguas, como Webpack, aquella es conocida por sus recargas en caliente y la capacidad de iniciar el servidor casi instantáneamente, lo que mejora mucho la productividad, lo cual ha llevado a que Vite sea adoptado cada vez más en proyectos que necesitan velocidad y eficiencia. Como señala Bishmark (2022), Vite agiliza el entorno de desarrollo al reducir drásticamente el tiempo de compilación y espera.

La implementación de módulos ES nativos es una de las contribuciones más sobresalientes de Vite, consiguiente cargar exclusivamente los archivos en uso a la hora de utilizarlos, eliminando la necesidad de 'bundle' o agrupar todo el proyecto de una vez, lo cual resulta en que las modificaciones en tiempo real sean más rápidas y eficaces. Domínguez (2023) señala que el contenedor modular de Vite aumenta la eficiencia del desarrollo y el desempeño en velo de la programación en relación con el propicio trabajo sobre grandes estructuras de código.

Vite también es conocido por su capacidad para crear compilaciones de producción optimizadas utilizando Rollup, lo que asegura que la salida final esté bien estructurada, sea ligera y tenga tiempos de carga eficientes, este paso de compilación se asegura de que todos los archivos irrelevantes sean omitidos y el trabajo innecesario se automatice. Según (2022) Carreño, tener herramientas que optimizan los entornos de desarrollo y producción proporciona claramente una ventaja competitiva para cualquier proyecto web.

TypeScript: Lenguaje que extiende JavaScript añadiendo tipado estático para mejorar la robustez y escalabilidad del código.

Este lenguaje de código es esencial para añadir tipos estáticos a JavaScript, permitiendo que los errores sean detectados durante el desarrollo en lugar de durante la ejecución, esto facilita la fiabilidad del código y ayuda a mantener proyectos complejos. Sin embargo, TypeScript se vuelve indispensable en entornos colaborativos donde muchos desarrolladores trabajan en la misma base de código. Soriano (2025) ilustró cómo las características de tipado estático de Typescript contribuyen a la calidad del software al permitir verificaciones profundas durante la escritura o edición de código.

TypeScript es más conocido por su integración perfecta con editores de código como Visual Studio Code, donde es esencial para la navegación de archivos, la autocompletación de fragmentos y las sugerencias contextuales, siendo crucial enfatizar que no solo se acelera significativamente el ritmo de desarrollo, sino que se reduce drásticamente el volumen de errores humanos debido a equivocaciones tipográficas y de referencia. De acuerdo con Llerena (2021), la incorporación inteligente de TypeScript dentro de un entorno de desarrollo particular mejora la productividad en general y la comprensión del código por parte de un programador.

TypeScript permite que se personalicen características adicionales con interfaces y tipos establecidos por los clientes, lo que ayuda a entender la estructura de datos dentro de una aplicación, aquello es bastante útil al tratar con respuestas de API y estructuras complejas como objetos de reservas, fechas y usuarios. Como señaló Segarra (2022), utilizar interfaces en TypeScript proporciona una documentación implícita del código, minimizando los comentarios necesarios.

### Node.js y Express: Tecnologías para el desarrollo del backend, gestionando las reservas y la comunicación con la base de datos.

Node.js permite construir aplicaciones web rápidas en forma escalonada debido a su arquitectura basada en eventos no bloqueantes, aquello es ventajoso en sistemas de reservas porque pueden manejar múltiples solicitudes a la vez sin que el rendimiento disminuya. Según Esquiviel y Caluña (2024), la naturaleza asíncrona de Node.js maximiza el rendimiento de aplicaciones en tiempo real y su naturaleza eficiente.

Express es un framework que se ejecuta sobre Node.js y es uno de los más usados por su desacoplamiento y flexibilidad en la construcción de APIs, por lo cual, es sencillo realizar el procesamiento de rutas, solicitudes HTTP, manejo de bases de datos, entre otros. Rodríguez et al. (2023) indica que Express es orientado a la velocidad y a la customización de la arquitectura a la aplicación, lo que permite hacer cambios rápidos y ajustes en la estructura y lógica de la aplicación web.

Dentro de la lógica del backend del sistema de reservas, Node.js y Express procesan las funciones para registrar nuevas reservas, consultar horarios disponibles y validar usuarios. Gracias a su modularidad y compatibilidad con múltiples bases de datos, el equipo puede agregar diversas funcionalidades donde sea necesario. Bangare (2022) argumenta que la modularidad de Node.js, al entregarle a un sistema múltiples piezas con funciones definidas, hacen que extensibilidad y el mantenimiento, sea muy simplificado.

### SQLite: Base de datos.

Se define como un gestor de bases de datos justificado en el modelo relacional (RDBMS) ligero y portátil que no necesita un servidor separado para funcionar, esto lo convierte en una

opción adecuada para aplicaciones pequeñas y prototipos. Como menciona Torres (2022), la simplicidad y robustez de SQlite han fomentado su uso generalizado incluso en aplicaciones móviles y de escritorio.

Una de las características destacadas de SQLite es que los datos se mantienen en un solo archivo local, facilitando la gestión de los datos, además, esto permite que el sistema se implemente de manera rápida y eficiente, contribuyendo al desarrollo de software a focalizarse en la lógica aplicativa en vez de ocuparse de la infraestructura de la base de datos de configuraciones de bases de datos complejas. Según Barthy (2023), El conjunto de características minimalistas de SQLite alivia la carga asociada al trabajo durante las fases de desarrollo e inicial de un proyecto.

SQLite mantiene las particularidades de SQL, las cuales incluyen la realización de transacciones, manejo de clave primaria, integridad referencial de clave externa, así como la ejecución de queries complejas que simulan comportamientos sofisticados de una base de datos. Esto, en cualquier caso, resulta imprescindible para la administración de datos estructurados y no compromete la calidad del funcionamiento del sistema. Como señala Bauther (2022), SQLite proporciona una solución confiable y de bajo mantenimiento para sistemas con necesidades moderadas.

### Git hub: controlador de versiones

Es un servicio que se basa en la nube que aporta al acceso remorto, el control de versiones y la gestión colaborativa de múltiples desarrolladores del código fuente de un proyecto, basado en una plataforma que controla de forma eficiente su ejecución, el mismo que es ampliamente utilizado que rastrea los cambios en los archivos de un proyecto determinado. Paredes y Arcos

(2019) mencionan que GitHub permite una mejor colaboración y trazabilidad durante el desarrollo de software.

En este marco se destacan los atributos de los repositorios públicos y privados, las ramas para la asignación de tareas, así como las solicitudes de extracción para la revisión del código antes de la fusión, estas funcionalidades permiten que un equipo de trabajo colabore de manera simultánea sin tener que interrumpirse unos a otros. GitHub fomenta la colaboración en equipo y disminuye los conflictos de fusión con puertas de acceso controladas al desarrollo (Beckman y Horton, 2021)

De igual forma, GitHub permite manejar la documentación en archivos README, gestionar los proyectos mediante issues y automatizar las tareas con GitHub Actions, estos aspectos optimizan la organización y el orden, además de agilizar el trabajo a través de la simplificación de entornos de trabajo. Los sistemas de control de versiones aumentan la calidad del código al integrar la revisión, las pruebas y la depuración del sistema (Maya, 2024).

### 2.1.3. Impacto de la gestión deportiva

La gestión deportiva abarca planificación, organización, administración y control de actividades vinculadas al deporte en sus distintivos niveles: recreativo, formativo o profesional, generando un impacto en instituciones y comunidades, lo cual le permitido crecer en las últimas décadas, especialmente al reconocerse su desarrollo integral. No se centra solo a lo administrativo, sino que incluye acciones para optimizar el desempeño deportivo, incrementar la participación ciudadana y asegurar la sostenibilidad de proyectos en el largo plazo (Yufei y Baghaei , 2024).

En ese sentido, en un desarrollo eficiente de los eventos y programas relacionado al deporte, se observa uno de los efectos más visibles de una gestión deportiva eficiente, donde los

recursos humanos, logísticos y financieros, cuando son administrados correctamente, permiten la planificación de competiciones, talleres, entrenamientos y eventos que satisfacen las demandas de los usuarios. Por otro lado, se impulsa el trabajo interdisciplinario, mejorando sustancialmente dentro de los procesos internos de las organizaciones deportivas (Sierra y Sánchez, 2024).

La preocupación relacionada con el deporte tiene efectos colaterales profundos en lo social, dependiendo de la forma que se programa, la cual si es bien orientada se pueden usar como elementos de inclusión para grupos vulnerables, jóvenes en riesgo, o para individuos con discapacidades, mediante estos programas, el deporte se constituye como un mecanismo importante para promover la cohesión social, construir una identidad común, y facilitar la el impulso de valores éticos como el respeto, y el esfuerzo. Esto posiciona al deporte como un campo de acción social que sobrepasa la competencia (Philipps, 2024).

Desde una perspectiva económica, una gestión deportiva adecuada impulsa el crecimiento de industrias asociadas como el turismo, el comercio y la publicidad, donde la organización de eventos deportivos atrae visitantes, fomenta el gasto local y crea empleos tanto temporales como permanentes. De esta manera, las actividades deportivas se convierten en un catalizador para el desarrollo económico sostenible, especialmente en aquellas comunidades que apoyan el deporte como política pública (Yufei y Baghaei , 2024).

Además, afectan la educación y los caminos profesionales, motivando la educación continua de los gestores deportivos, entrenadores y técnicos deportivos con la integración de la tecnología, modelos de evaluación del desempeño de la institución y planificación estratégica, las organizaciones deportivas pueden operar a un nivel superior de calidad institucional. Este rigor

profesional mejora la percepción del deporte como un campo interdisciplinario crucial y esencial para el desarrollo de las sociedades (Niebles & Sanabria , 2023).

# 3. CAPÍTULO III: DESARROLLO Y ANALISIS

## 3.1. Diseño del prototipo

### 3.1.1. Diseño Conceptual

En este prototipo el usuario tendrá una experiencia guiada desde el primer contacto con la landing page, donde se puede explorar secciones informativas del prototipo como su misión y visión. Si el usuario decide usar la plataforma, debe completar el formulario de registro y confirmar su cuenta por correo electrónico; a partir de ese momento, puede iniciar sesión con sus credenciales para acceder las reservas y su historial. (Sommerville, 2011).

Una vez autenticado el usuario accede a una lista de 4 deportes (básquet, pádel, futbol y tenis); al elegir uno, se muestran las canchas asociadas, después se muestra un calendario que resalta los dias con disponiblilidad cada uno con horarios disponibles del dia seleccionado y realiza la simulacióndel pago. El usuario tambien accede a un historial en donde revisa sus reservas pasadas y actuales.

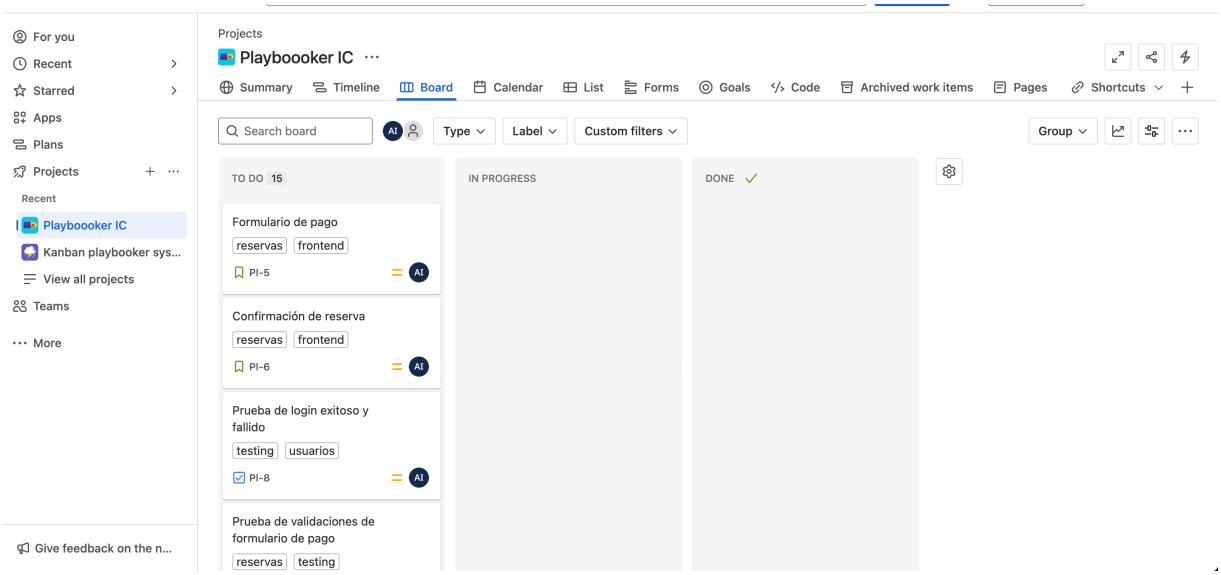
Para garantizar la coherencia y la seguridad, el sistema impone restricciones claras: el usuario no está autorizado a editar o cancelar reservas, estas acciones quedan exclusivamente al alcance del administrador del sistema de poder cancelar las reservas tampoco gestiona información de otros usuarios o de las empresas. El usuario tampoco accede a las pantallas de administración, lo que preserva la integridad de los datos y respeta el principio de mínimo privilegio en la gestión de roles (Fowler, 2003)

### 3.1.2. Metodología de desarrollo

La metodología de desarrollo escogida para el proyecto ha sido Kanban ya que al tratarse de un prototipo en el que se busca validar rápidamente flujos críticos como el registro, login, reserva y simulación de pago mediante pruebas funcionales locales, no queremos quedar atrapados en ciclos de planificación rígidos ni con demasiadas tareas a medias, esto reduce significativamente el cambio de contexto, pues el esfuerzo del desarrollador se concentra en completar cada funcionalidad antes de pasar a la siguiente, lo que mejora la calidad de la implementación y acelera el tiempo de entrega de cada módulo (Anderson, 2010)

Para el Proyecto, Kanban facilita la identificación rápida de cuellos de botella y la mejora continua del proceso es por eso que implemento de la siguiente forma, cuenta con 3 columnas **principales que son el "**TO DO"**, "**IN PROGRESS" **y "**DONE" esto permite que de forma sencilla se pueda identificar el de trabajo del prototipo. En la primera columna se encontraran las tareas listas para ser desarrolladas, como las funciones a implementar y las pruebas a rralizarse. En la siguiente columna tendremos las tareas que actualmente se encuentran en desarrollo y en la última columna nos mostrará las tareas ya terminadas. De esta manera nos permite tener una estructura adecuada para la gestión ágil del prototipo.

*Figura 1: Tablero Kanban en herramienta Jira*




*Fuente: de Autoria propia*

### 3.1.3. Análisis de requerimientos

***Requisitos Funcionales***

- **Registro de Usuario**: El sistema permite que cualquier persona complete un formulario con nombre, apellidos, correo electrónico y contraseña creará una nueva cuenta de usuario.

- **Inicio de Sesión**: El sistema muestra un formulario donde el usuario ingresa con su correo y contraseña registrado, Si las credenciales son válidas, el usuario es redirigido a su panel de control. Si no, se muestra un mensaje de error indicando “Usuario o contraseña incorrectos”.

- **Visualización de Deportes y Canchas**: Una vez autenticado, el usuario debe poder navegar por una lista de deportes disponibles (fútbol, pádel, tenis, básquet), Al seleccionar un deporte, el sistema mostrará todas las canchas activas asociadas a ese deporte.

- **Consulta de Disponibilidad**: Cuando el usuario elige una cancha concreta, debe desplegarse un calendario que indique los días disponibles para esa realizar la reserva.

- **Creación de Reservas:** El usuario puede seleccionar una fecha y franja horaria disponible y proceder a realizar la reserva.

- **Simulación de Pago**: En la simulación de pago una vez que el usuario hace clic en Reservar el sistema muestra un formulario para poder realizar el pago, este cuenta con validación interna, si da clic en Pagar se confirma el pago.

- **Gestión de Usuario**s: El administrador tiene un apartado de usuarios en donde se muestra un listado de todos los usuarios registrados, mostrando nombre, correo, rol y estado.

- **Gestión de Empresas**: El administrador tendrá un listado de todas las empresas registradas, mostrando nombre, correo de contacto y estatus.

- **Visualización de Reservas**: El usuario tendrá su historial de reservas de las canchas que ha reservado en el sistema web. El administrador tendrá una lista de todas las reservas de las canchas existentes y podrá cancelar las reservas.

***Requisitos No funcionales***

- Notificaciones de errores al ingresar datos o dejar campos vacíos.

- Buen uso y comprensión para los usuarios con diferentes niveles de conocimientos y habilidades tecnológicas.

- Las contraseñas de usuarios deben almacenarse cifradas (bcrypt) en la base de datos.
- El acceso a rutas protegidas (panel de administrador) se debe realizar únicamente con tokens JWT válidos y roles autorizados.
- El código debe organizarse en módulos separados (rutas, controladores, modelos, middlewares) para facilitar la extensión futura.
- Los calendarios de disponibilidad deben mostrar dias ocupados y libres con colores contrastantes (verde para libre, rojo para ocupado).

### 3.1.4. Diagramas de caso de uso

Los diagramas de casos de uso son importantes ya que nos ayudan a entender como los usuarios o roles existentes interactuan con la aplicación que acciones puenden realizar y cuales no.

Estos diagramas definen la función de la página como el flujo de reservas, horarios de disponibilidad, la gestión de usuarios.

#### 3.1.4.1. Caso de uso general

*Figura 2: Caso de uso general*

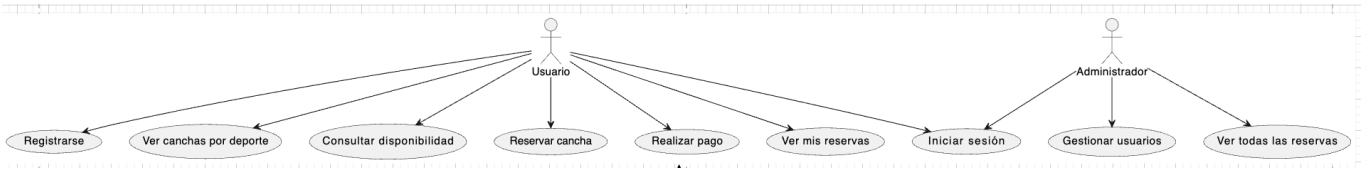



*Diagrama de caso de uso general de todo el prototipo. Fuente: Autoría propia*

### 3.1.4.2. Caso de uso F0 – Iniciar sesión

*Figura 3: Caso de uso Iniciar sesión*

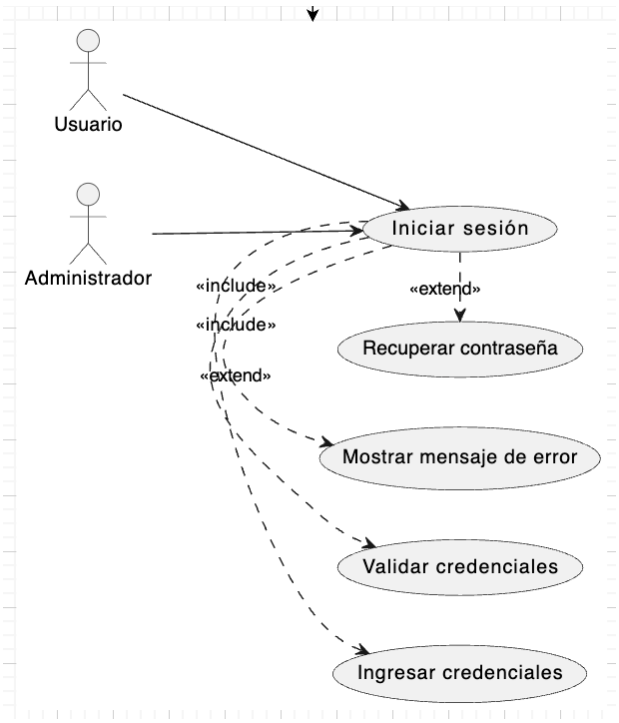



*Diagrama de caso de uso de inicio de sesión del prototipo. Fuente: Autoría propia*

**Actores principales:**

Usuario, Administrador

**Descripción:**

Roles que acceden al sistema introduciendo credenciales para autenticarse y acceder a sus funciones y rutas respectivas.

### 3.1.4.3. Caso de uso F1 – Registro de usuario

*Figura 4: Caso de uso registro de usuario*

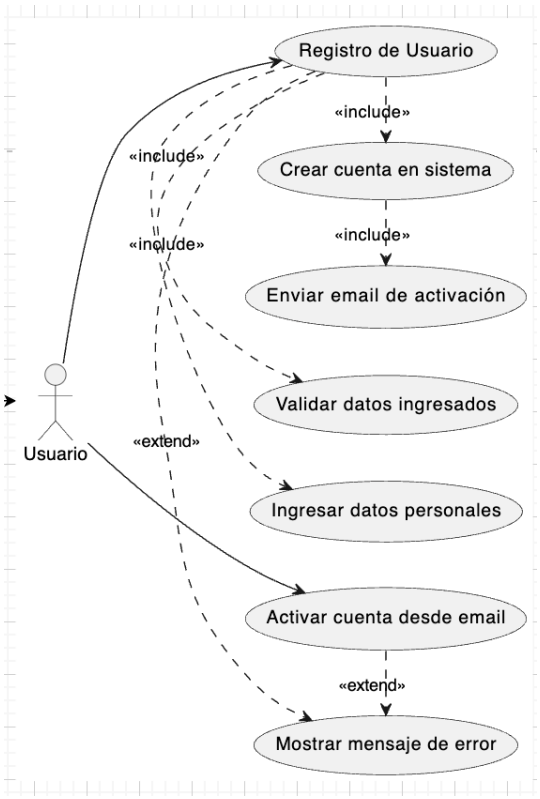



*Diagrama de caso de uso registro de usuario en el prototipo. Fuente: Autoría propia*

**Actor principal:**

Usuario

**Descripción:**

Persona que ingresa sus datos para crear una cuenta, validar información y confirmar su registro en la plataforma.

#### 3.1.4.4. Diagrama de Caso de Uso F2 – Reservas de Canchas

*Figura 5: Caso de Uso Reservas de Canchas*

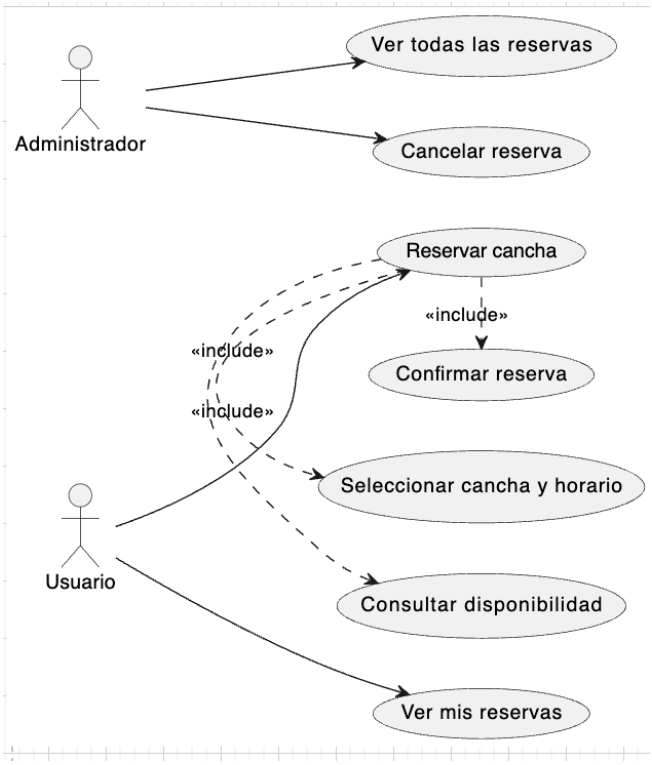



*Diagrama de caso de uso Reserva de canchas. Fuente: Autoría propia*

**Actor principal:**

Administrador, Usuario

**Descripción:**

El Administrador ve todas las reservas del sistema ademas de ser el único que puede cancelar las reservas siempre y cuando el usuario se contacte directamente a solicitar la cancelación, el Usuario solo ve sus propias reservas y realizar reservas en la cancha de su elección.

#### 3.1.4.5. Caso de uso F3 –Pago simulado

*Figura 6: Caso de uso Pago simulado*

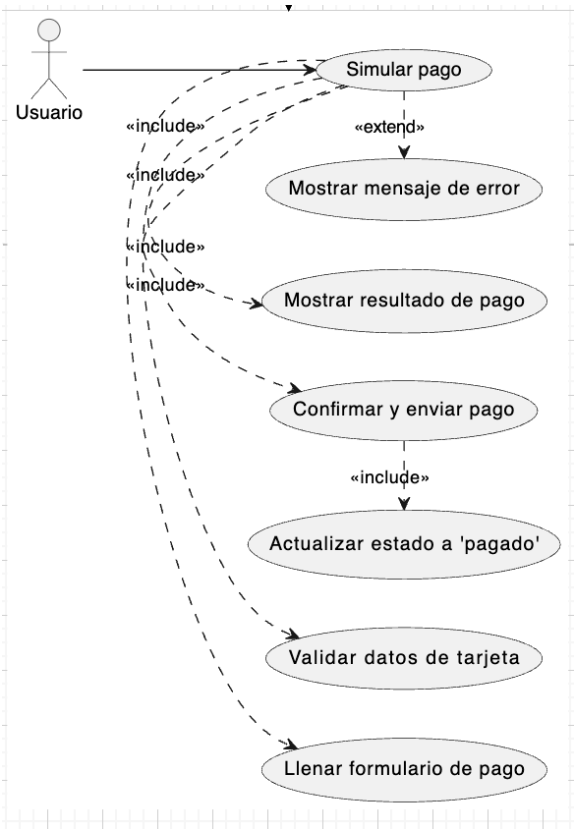



*Diagrama de caso de uso de pago simulado. Fuente: Autoría propia*

**Actor principal:**

Usuario

**Descripción:**

Permite que le usuario pueda reservar un horario en la cancha escogida mediante un formulario de pago simulado, sin conexión real con pasarelas de pago al ser un prototipo.Este formulario realiza validaciones básicas de los datos ingresados, si todo es correcto la reserva se marcara con estado pagado.

#### 3.1.4.6. Caso de uso F4 –Gestión de empresas

*Figura 7: Caso de uso gestión empresas*

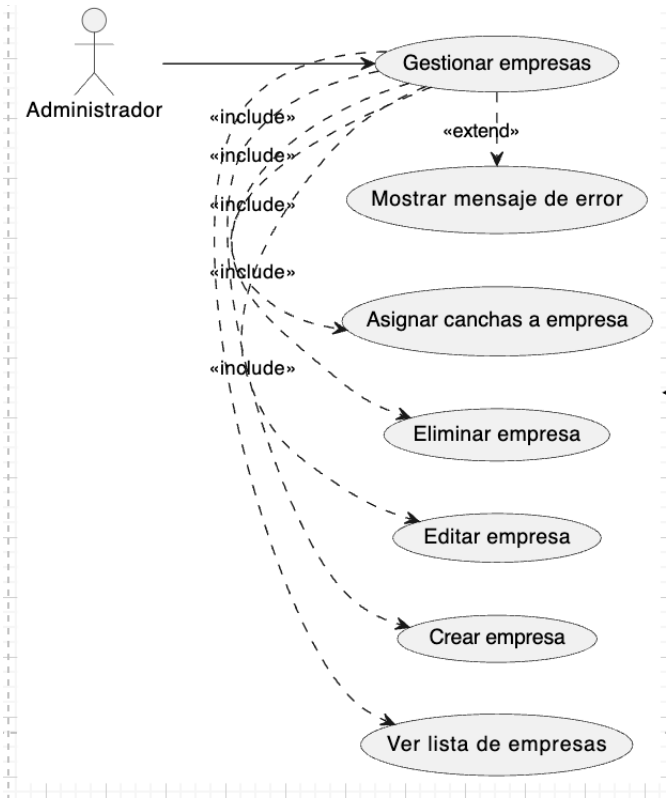



*Diagrama de caso de uso de gestión de empresas. Fuente: Autoría propia*

**Actor principal:**

Administrador

**Descripción:**

El Administrador crea, edita, elimina empresas, revisa horarios de funcionamiento.

#### 3.1.4.7. Caso de uso F5 - Gestión de usuarios

*Figura 8: Caso de uso gestión de usuarios*

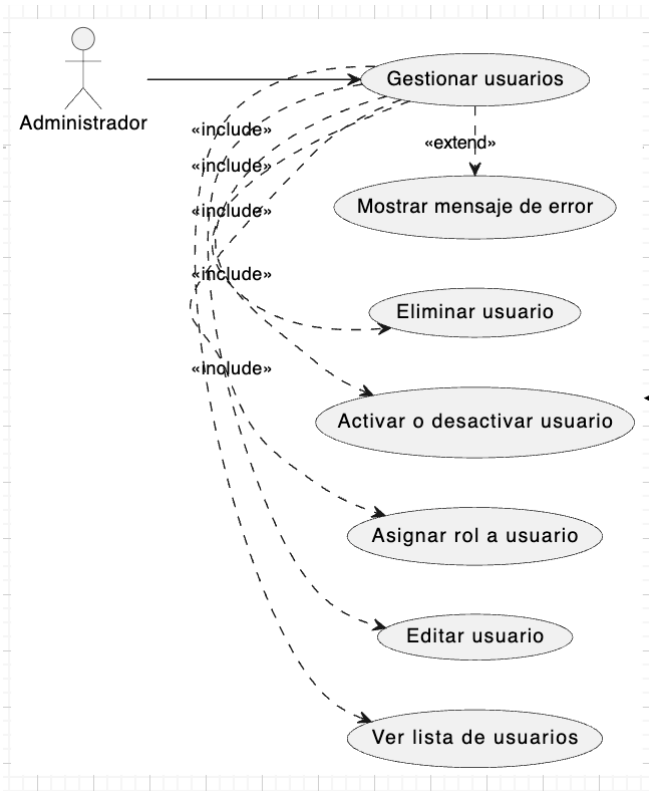



*Diagrama de caso de uso de gestión de usuarios. Fuente: Autoría propia*

**Actor principal:**

Administrador

**Descripción:**

Usuario que gestiona (ver, crear, editar,eliminar o desactivar ) a las cuentas de otros usuarios.

### 3.1.5. Diagramas de flujo y especificaciones técnicas

#### 3.1.5.1. Diagrama de flujo Principal del sistema

*Figura 9: Flujo principal del sistema*

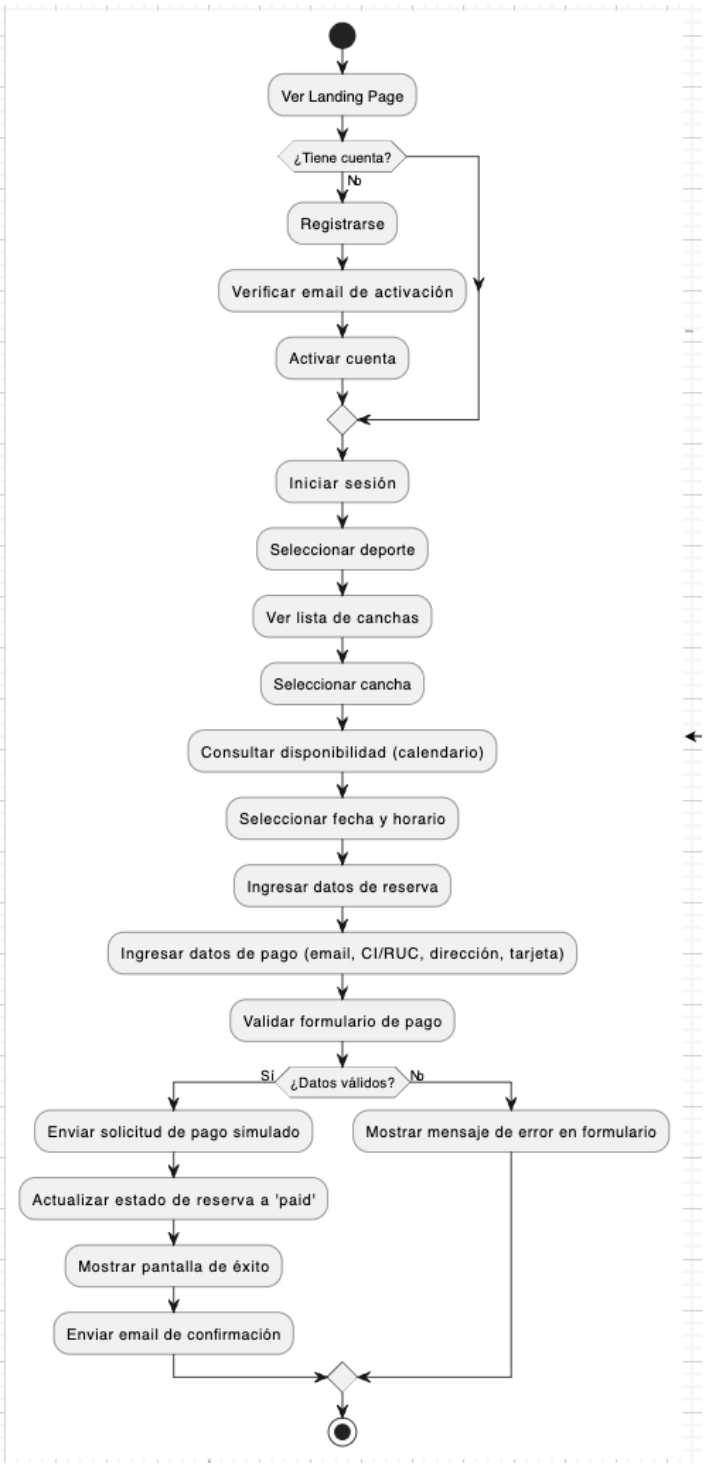



*Diagrama de flujo del sistema. Fuente: Autoría propia*

### 3.1.5.2. Diagrama de Flujo a detalle de inició de sesión

*Figura 10: Flujo inicio de sesión*

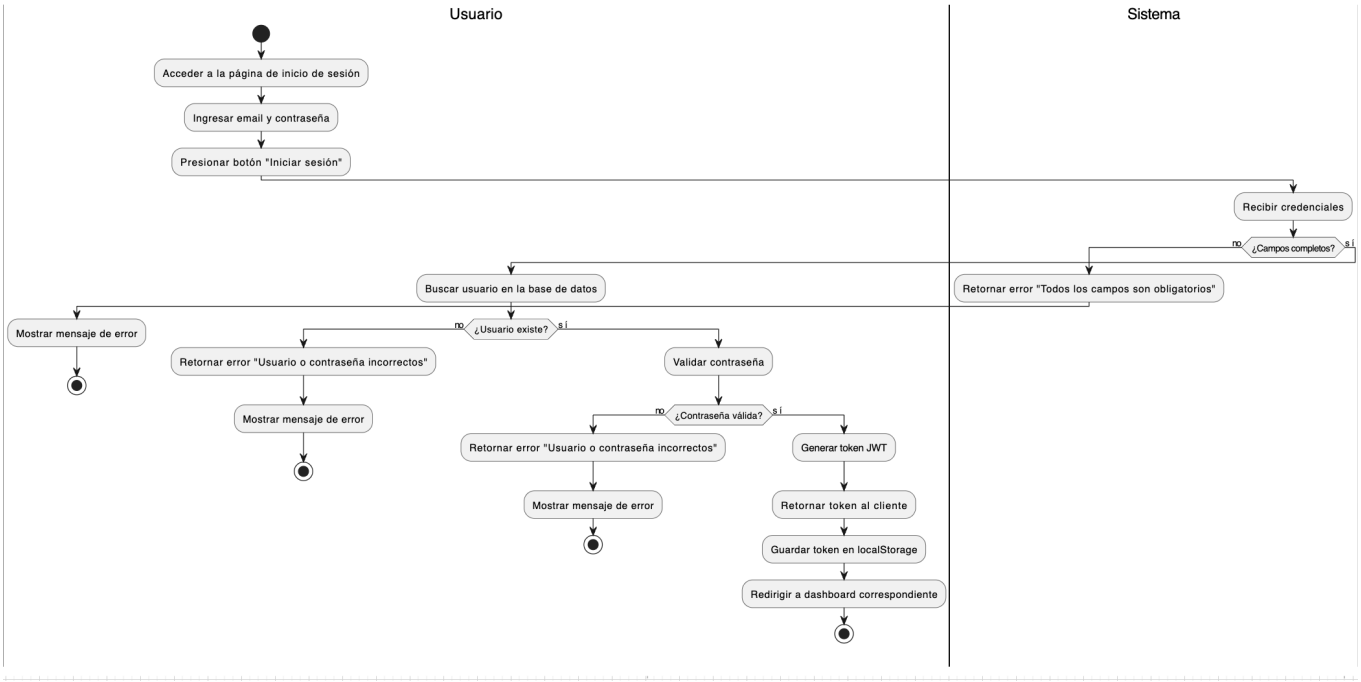



*Diagrama de flujo de inicio de sesión. Fuente: Autoría propia*

1. Usuario abre la página de login.
2. Usuario ingresa email y contraseña.
3. Usuario pulsa “Iniciar sesión”.
4. Sistema verifica que ambos campos estén completos:

   4.1 Si falta alguno → muestra “Todos los campos son obligatorios” y termina.

5. Sistema busca el email en la base de datos:

   5.1 Si no existe → muestra “Usuario o contraseña incorrectos” y termina.

6. Si el email existe, sistema valida la contraseña:

   6.1 Si no coincide → muestra “Usuario o contraseña incorrectos” y termina.

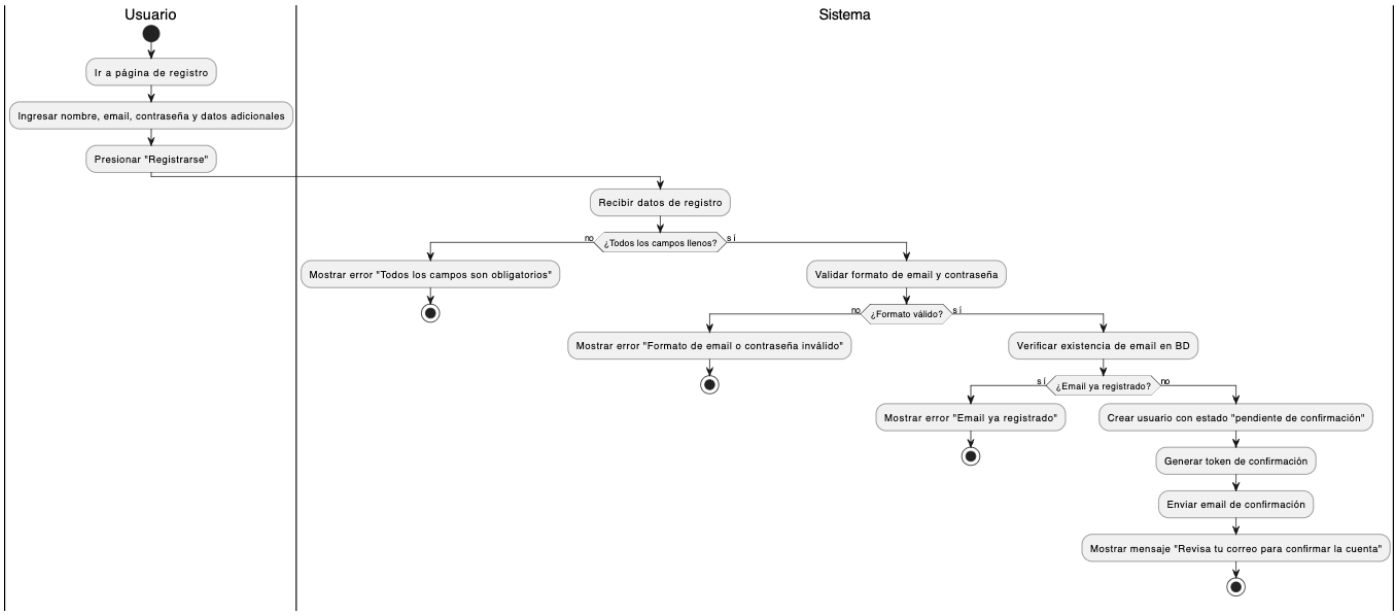
6.2 Si coincide → genera token de sesión y lo almacena en cookie.

7. Sistema redirige al usuario a su dashboard según su rol.

8. Fin del flujo (usuario autenticado y con acceso a funcionalidades).

#### 3.1.5.3. Diagrama de Flujo a detalle de registro de sesión

Figura 11: Flujo Registro de sesión




1. Usuario ingresa  a la página de registro.

2. Usuario ingresa  sus datos nombre, email, contraseña y datos adicionales, luego da clic en “Registrarse”.

3. Sistema recibe los datos y verifica que todos los campos estén completos:

   3.1 Si falta alguno → muestra “Todos los campos son obligatorios” y detiene el flujo.

4. Si todos los campos están llenos, el sistema valida el formato de mail y contraseña :

   4.1 Si el formato es inválido → muestra “Formato de mail o contraseña inválido”

5. Si el formato es válido, el sistema comprueba en la base de datos si el mail ya está registrado:

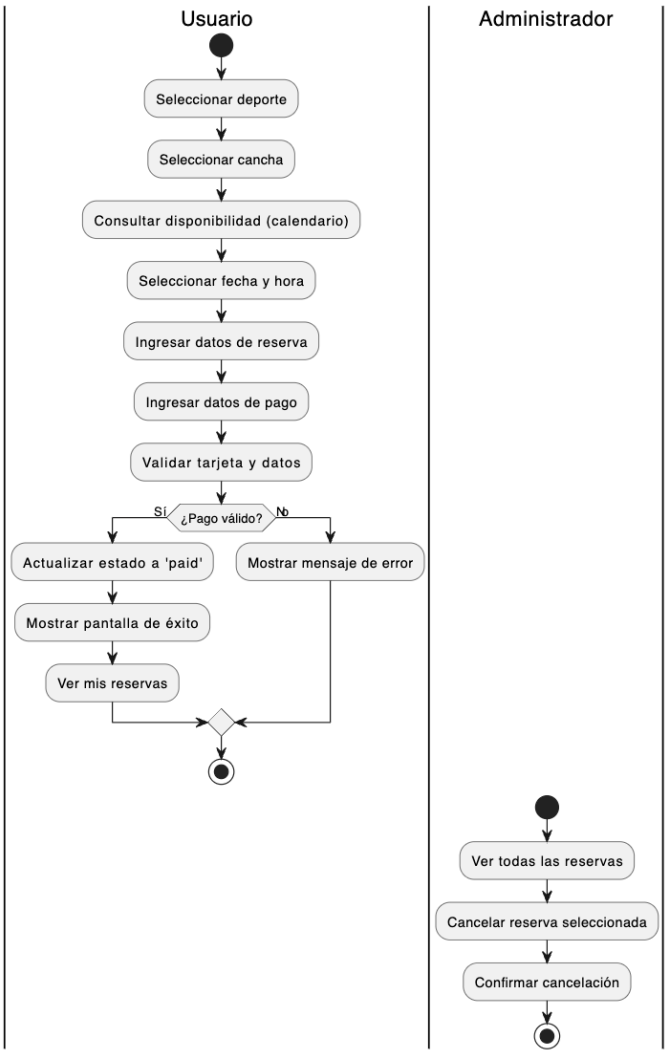
   5.1 Si el mail existe → muestra “Email ya registrado” y detiene el flujo.

   5.2 Si el mail no existe → crea el usuario con estado “pendiente de confirmación”.

6. Se genera un token de confirmación y se envía el mail de confirmación al usuario.
7. Sistema muestra al usuario el mensaje “Revisa tu correo para confirmar la cuenta”.
8. Fin del flujo (usuario creado y pendiente de confirmación).

#### 3.1.5.4. Diagrama de Flujo Integrado –Reservas

*Figura 12: Flujo Reservas de canchas*



1. Usuario

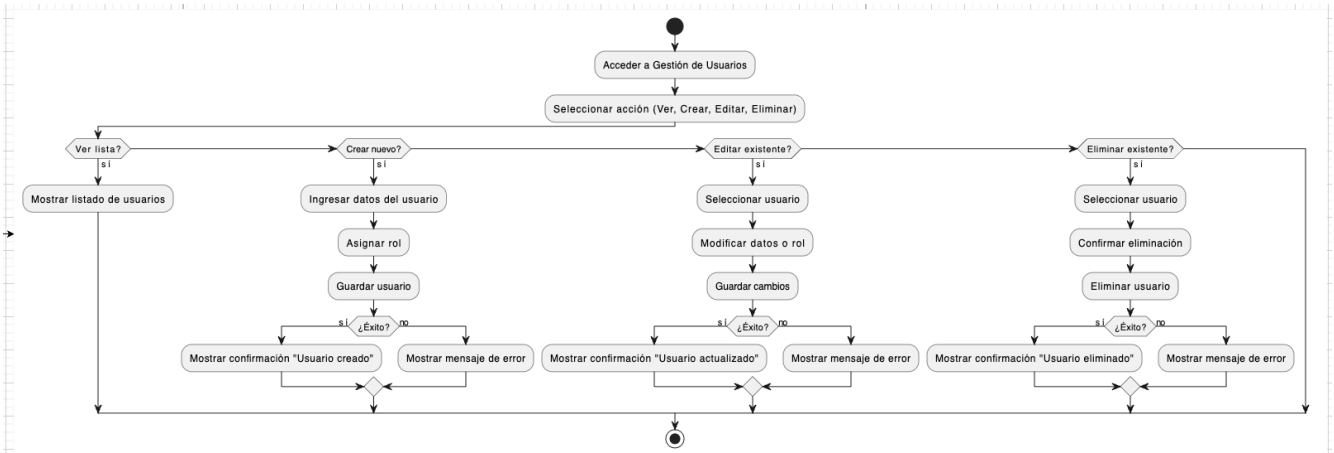
- Selecciona deporte y cancha, ingresa detalles de la reserva y simula el pago.
- Si el pago es exitoso, confirma la reserva y envía correo; si no, muestra error.
- Consulta el listado de sus propias reservas.

1. Administrador

- Ve todas las reservas

#### 3.1.5.5. Diagrama de Flujo a detalle de gestión de usuarios

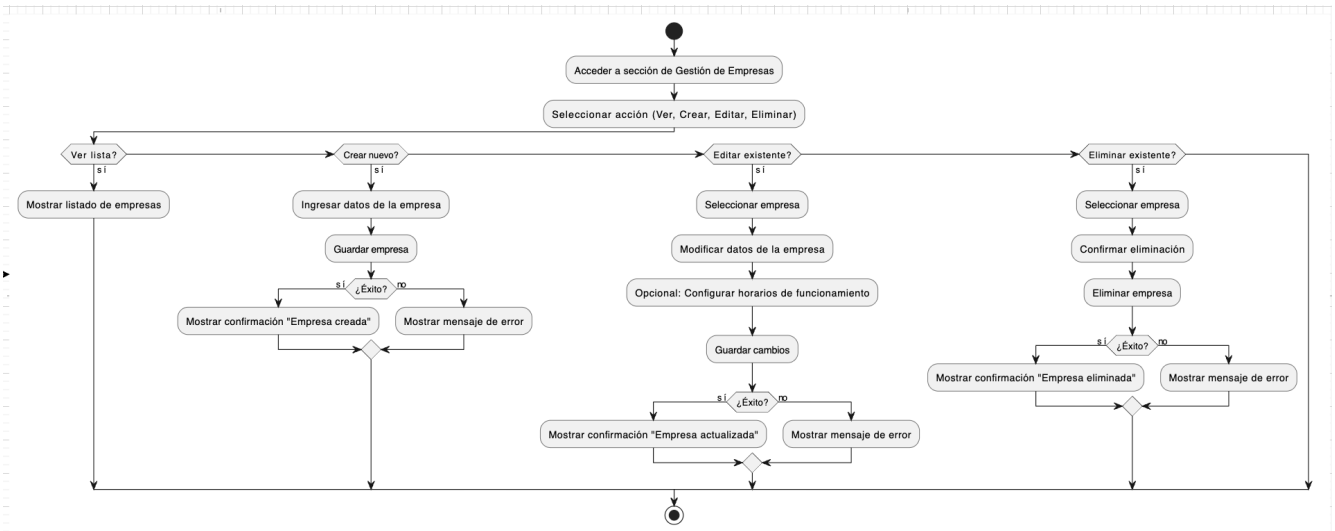
Figura 13: Flujo gestión usuarios




El administrador ingresa al Módulo de usuarios y ve la lista de usuarios, Puede crear. Editar y desactivar usuarios.

#### 3.1.5.6. Diagrama de Flujo de gestión de empresas

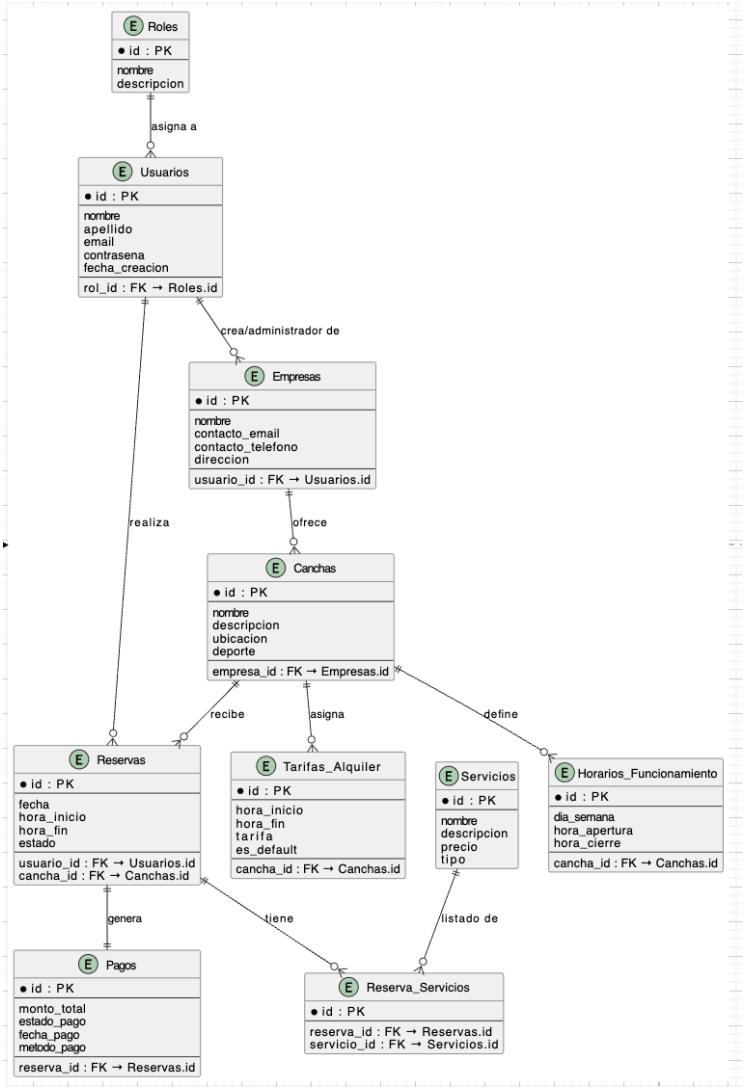
*Figura 14: Flujo gestión Empresas*




El administrador ingresa al módulo de gestión de empresas y se muestra la lista de las empresas existentes en el sistema con su respectivo estatus  el admin puedes crear, editar, desactivar empresas.

### 3.1.6. Diagrama conceptual de base de datos

*Figura 15: Diagrama conceptual de base de datos*




### 3.1.7. Diseño de la Interfaz gráfica

El diseño de la interfaz de usuario del proyecto tiene como objetivo mostrar los detalles clave de manera directa para el usuario. Se aplica un patrón de color consistente como es un gradiente de azul-naranja atenuado para elementos como botones y principalmente blanco para el donde del sistema, cada pantalla mantiene un diseño sobrio y con una arquitectura de información. Los información esta organizada de tal manera de que permite la legibilidad sin interrupciones. Los formularios constan de verificaciones y manejo de errores para que el usuario sepa muy bien

cuando comete un error en el uso del sistema.. Además, se toma en cuenta que al usuario no se sobrecargue con demasiada información.

### 3.2. Desarrollo del prototipo funcional

Para crear el prototipo de reservas, utilizamos un conjunto de herramientas que permitio revisiones rápidas y a preservar una buena interacción de usuario. El lado del servidor se construyó utilizando Node.js, una configuración que maneja los trabajos ocultos de manera efectiva debido a su diseño ya que agrega una capa básica para describir las rutas y las respuestas en el servidor. (Node.js, n.d)

Para la base de datos, utilizamos SQLite, la cual nos ayuda a manejar los datos como si fueran objetos en Javascript además permite el uso sin necesidad de usar un servidor adicional. (Sequelize, n.d.)

En el frontend, elegimos React, que permite la creación de componentes reutilizables ya que la interfaz se actualiza rápidamente cuando cambia los datos asi evitar malas experiencias hacia los usuarios y el uso sin pausas notables. (React, n.d)

La arquitectura Usada para el prototipo se encuentra en capas separadas, la interfaz de usuario, la lógica del negocio y el acceso a datos. Por el lado del cliente se renderiza los componentes reutulizables como formularios, calendarios, el panel de usuario y administrador.

El servidor se encarga de cada petición atraves de rutas, controladores y un middleware para poder realizar las validaciones necesarias. Se Mapean las entidades hacia la base de datos para asi garantizar la transaccionabilidad, además cuenta con la autenticación basada en tokens JWT para poder proteger las rutas sensibles, un CRON el cual despues de 15 minutos de no haber realizado el pago simulado esta expira.

Teniendo un sitema modular este permite flexibilidad y escalabilidad para la integración de nuevos módulos sin necesidad de afectar el resto del sistema.

El diseño de la interfaz de usuario es una de las partes más importantes en el desarrollo del sistema, ya se busca crear una experiencia visual agradable y sencilla de usar para el usuario, mucho más sabiendo que ese es el diferenciador frente a otros sistemas los cuales no cuentan con una experiencia enrequecedora hacia el usuario. Después de realizar los  análisis de requisitos, se procedio a listar las tareas en Jira herramienta que se una para manejar la metodología de Kanban.

El tablero se encuentra compuesto de 3 columnas y 5 estados internos.

**Por Hacer:** Maneja el Backlog y las tareas ya definidas y listas para comenzar a desarrollarlas

**En progreso:** Maneja las tareas que se encuentran en desarrollo

**Terminadas:** Manejan que se hayan realizado las pruebas de funcionamiento y se haya aprobado y se encuentren listas para un desplieque.

Cada una de las tarjeta que tenemos en nuesta lista avanza por los estados mencionados para asegurar que exista la trazabilidad y asi poder evitar los cuellos de botella. Se trabajan 3 tareas por colunma tomando en cuenta que es solo una persona que que se encarga de todo el proyecto.

*Figura 15: Lista de tareas y sus estados en Jira*

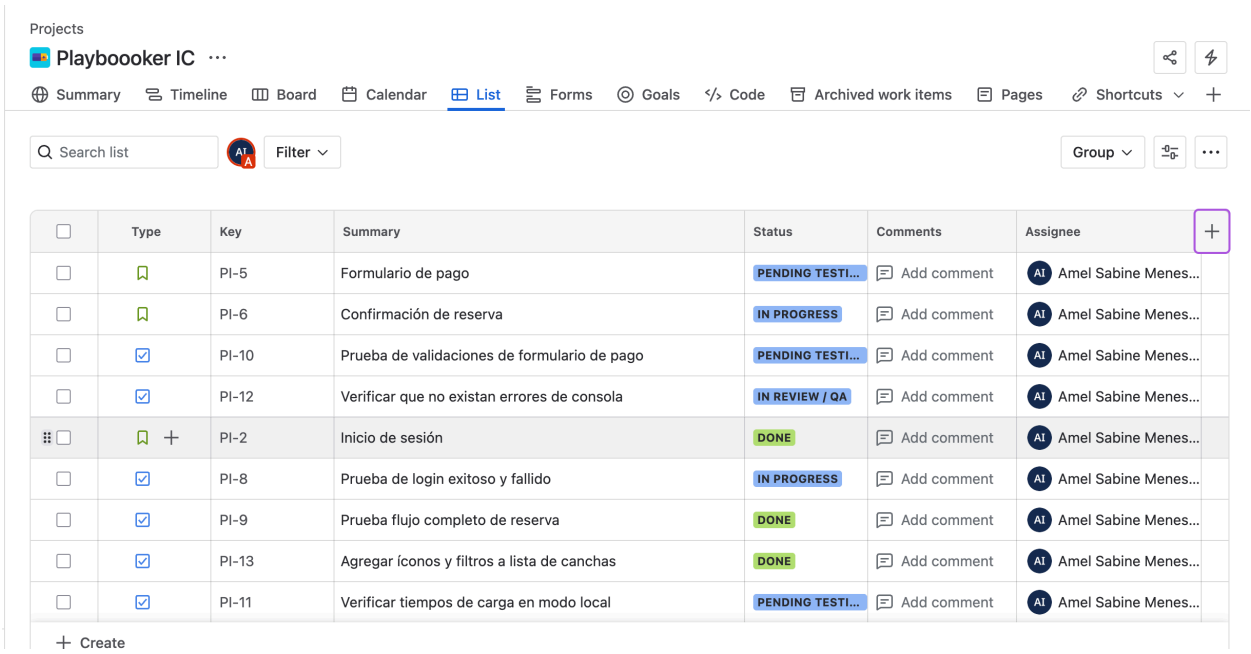
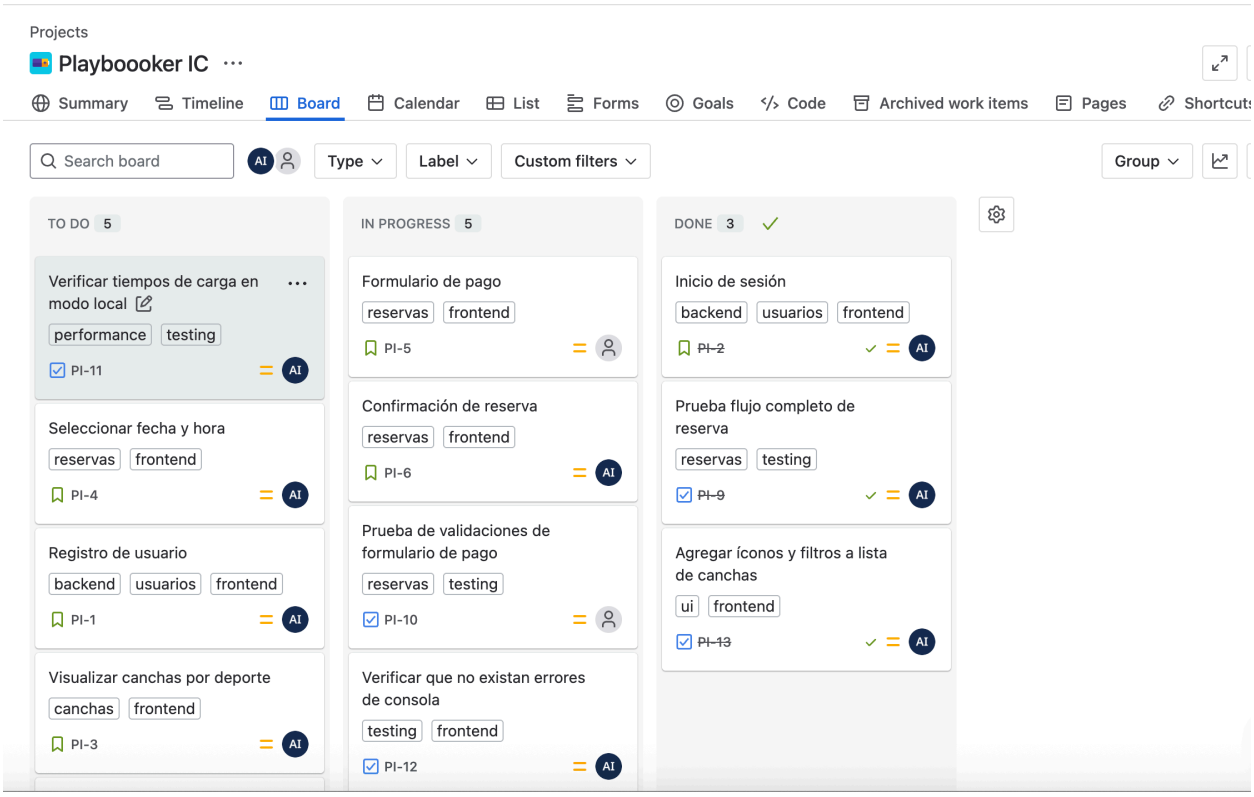


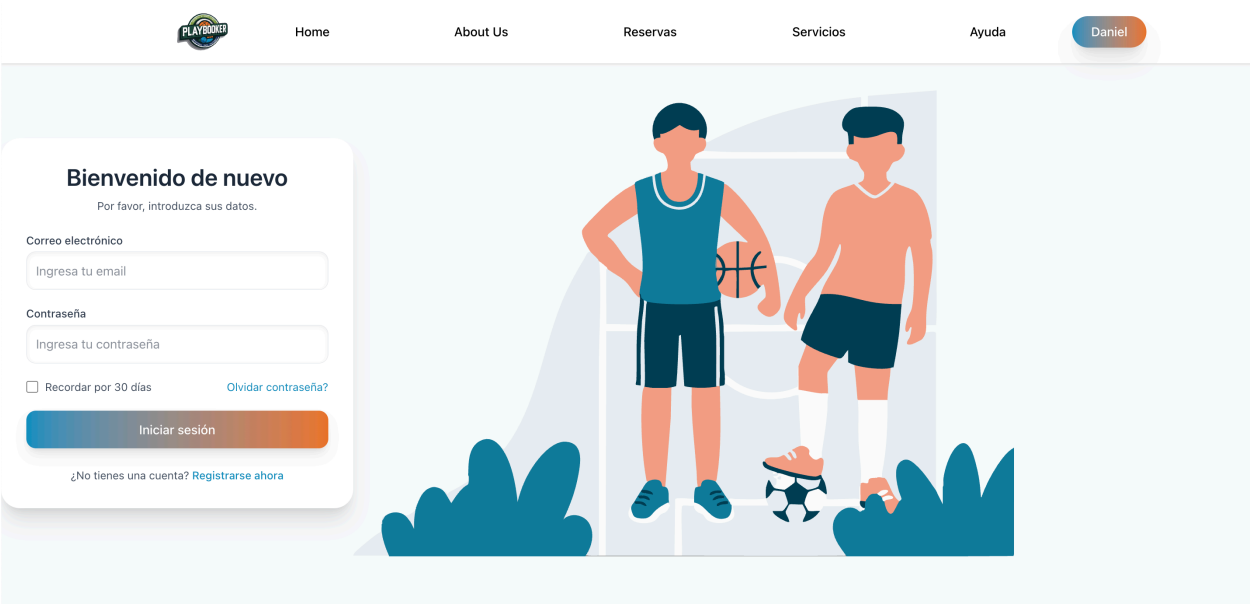

*Figura 16: Board con el flujo de las tareas dependiendo del desarrollo*



De esta manera Kanban nos permitió manejar un buen flujo en el desarrollo del sistema permitiendo asi que sus funcionalidades queden validadas y listas para en un futuro realizar un despliegue.

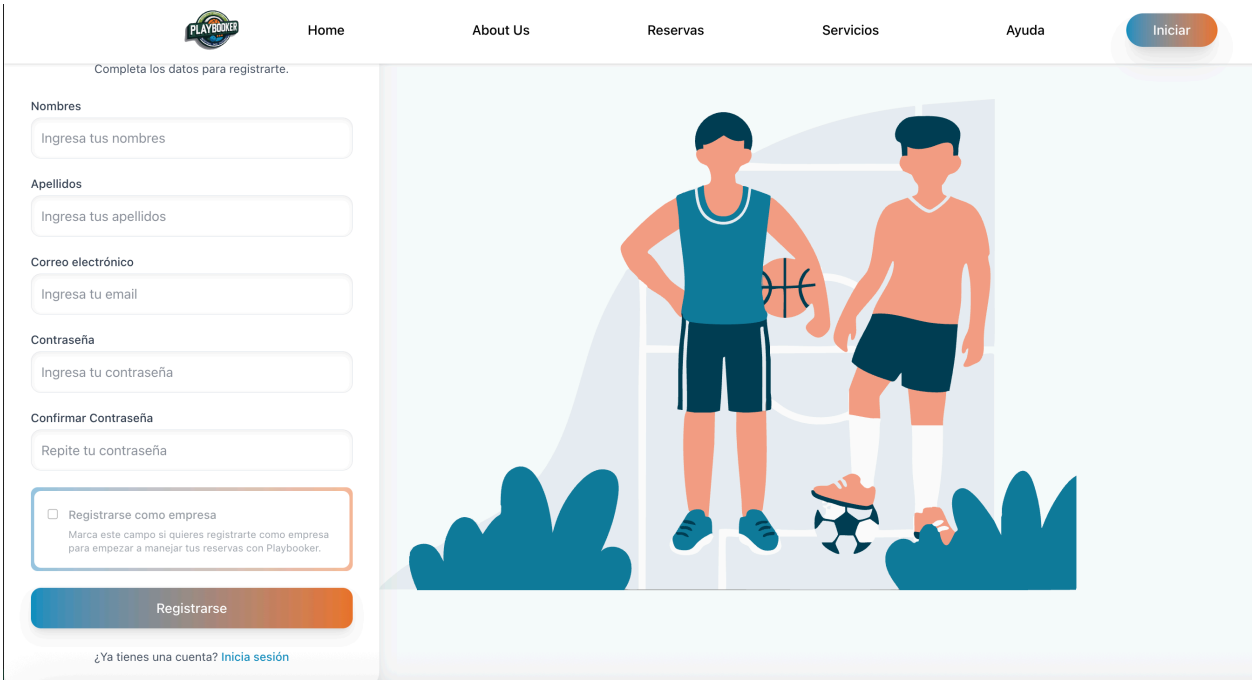
Teniendo un proceso de desarrollo bien definido y registrado en nuestro tablero Kanban, las vistas iban tomando forma, Las secciones más importantes se ven asi.

*Figura 17: Interfaz loggin del sistema*



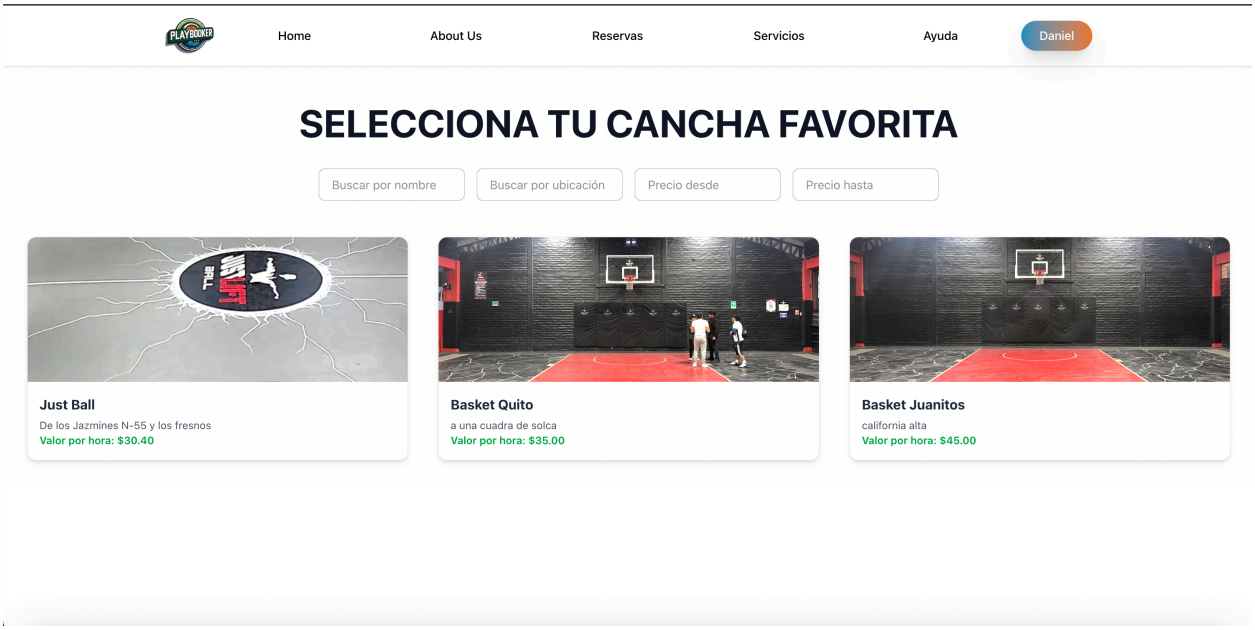

Tenemos un login con formulario validado, cuenta con un navbar, como podemos ver se respeta el diseño de toda la página, guarda una arquitectura de información limpia para que el usuario no se sienta abrumado o cansado de tanto texto en la pantalla.

*Figura 18: Interfaz de registro del sistema*



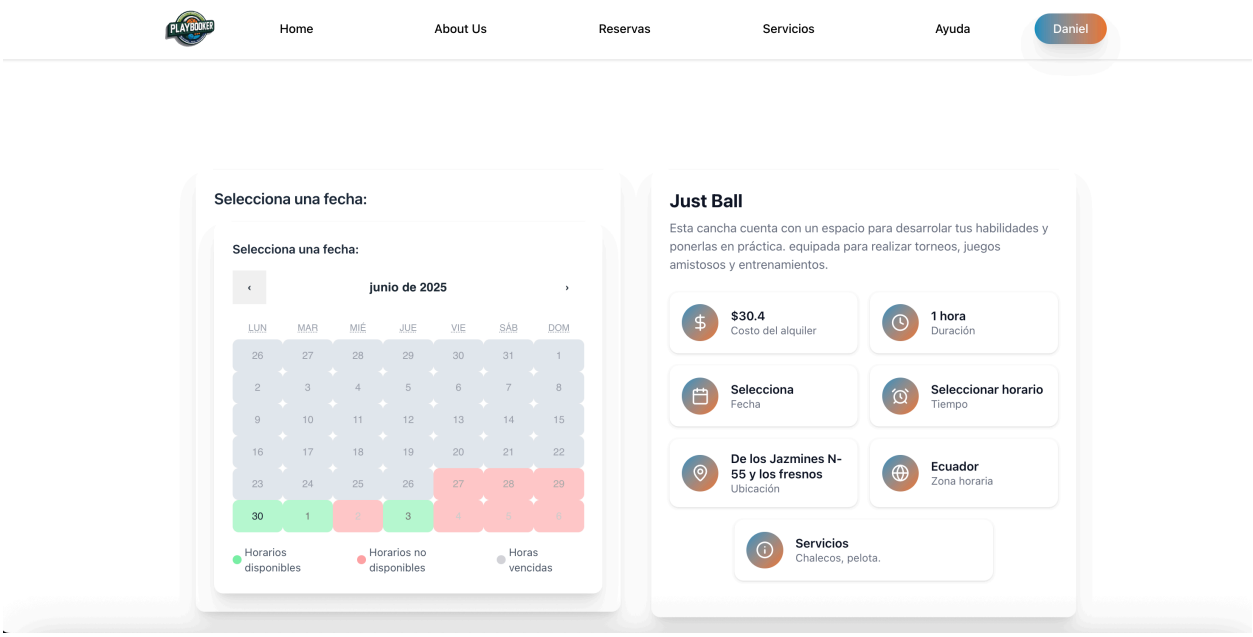

El registro cuenta con un formulario totalmente validado vemos como guarda coherencia en el diseño y la arquitectura, manteniendo el navbar y los colores.

*Figura 19: Interfaz de canchas*



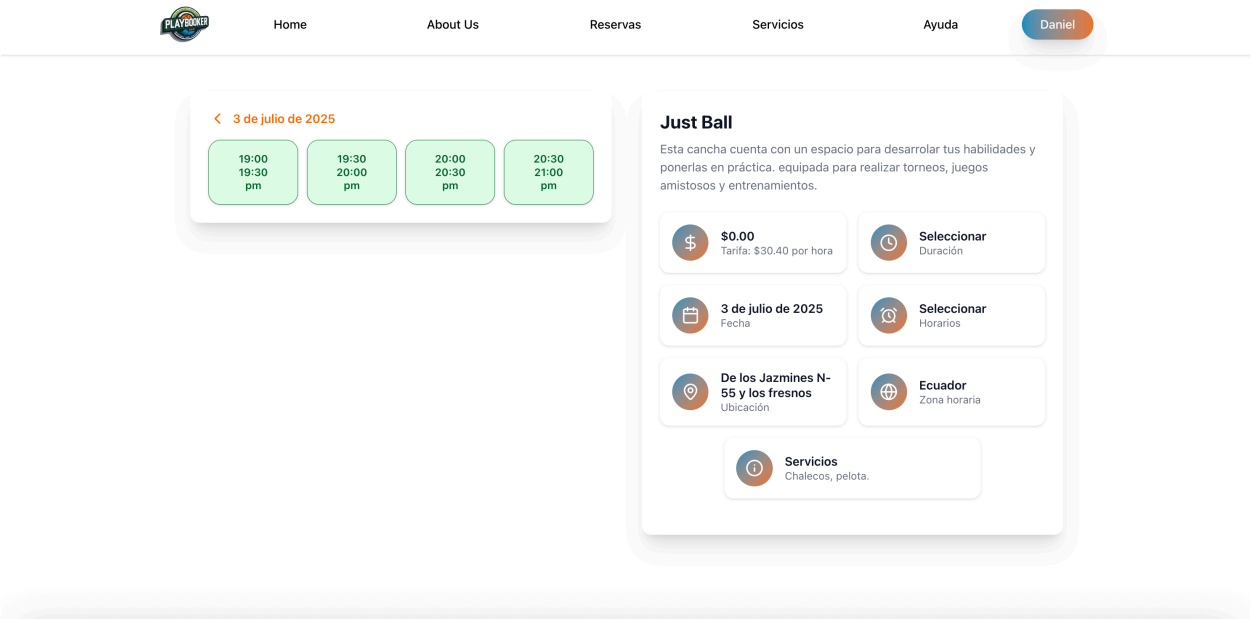
Podemos ver que esta interfaz cuenta con filtros de búsqueda para mejorar la experiencia del usuario, además de mostrar todas las canchas que se encuentran en el sistema  con sus precios por hora y su ubicación.

*Figura 20: Interfaz de disponibilidad*



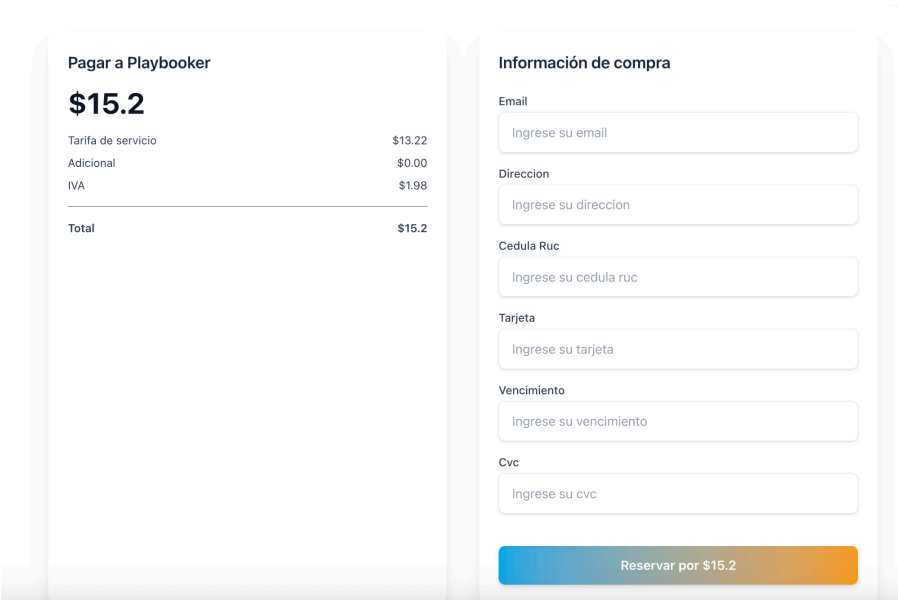

Tenemos la interfaz en donde se muestran los días disponibles de reserva de la cancha seleccionada, además podemos ver que al lado derecho tenemos el detalle de la cancha, los servicios que incluye y demás información importante para la reserva.

*Figura 21: Interfaz Horarios de reserva*




Aquí el usuario puede seleccionar el horario que desea realizar su reserva, hay que tomar en cuenta que están el múltiplos de 30 minutos debido a que existen usuarios que reservan 1h30 minutos y se tomó en cuenta para realizar la franja horaria en múltiplos de 30 minutos.

*Figura 22: Interfaz simulación del pago*



Podemos ver que tenemos el formulario de simulación del Pago, este formulario contiene validaciones básicas , hay que tomar en cuenta que al ser un prototipo no fue posible conectarlo con un botón de pagos real debido a que es necesario adquirir el servicio y ser una empresa en funcionamiento.

## 3.3. Pruebas funcionales del sistema

### 3.3.1. Prueba FT-01: Registro de un usuario exitoso

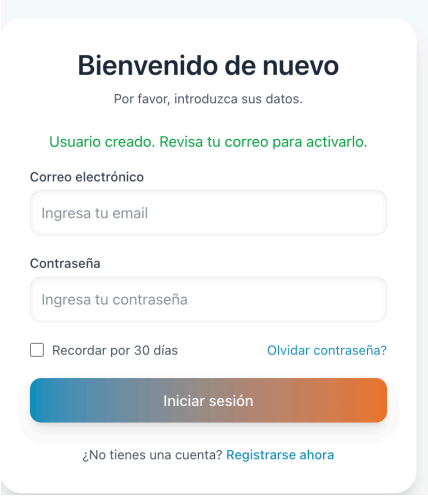
**Precondición:** No debe existir una cuenta con el correo indicado.

**Pasos:**

1. Navegar a la pantalla de registro.
2. Completar campos: nombre, apellidos, correo y contraseña válidos.
3. Clic en “Registrarse”.

**Resultado esperado:** Se muestra mensaje indicando que revise su correo para activar la cuenta.

***Figura 23:*** *Prueba de funcionalidad Registrarse*



### 3.3.2. Prueba FT-02: Activación de cuenta.

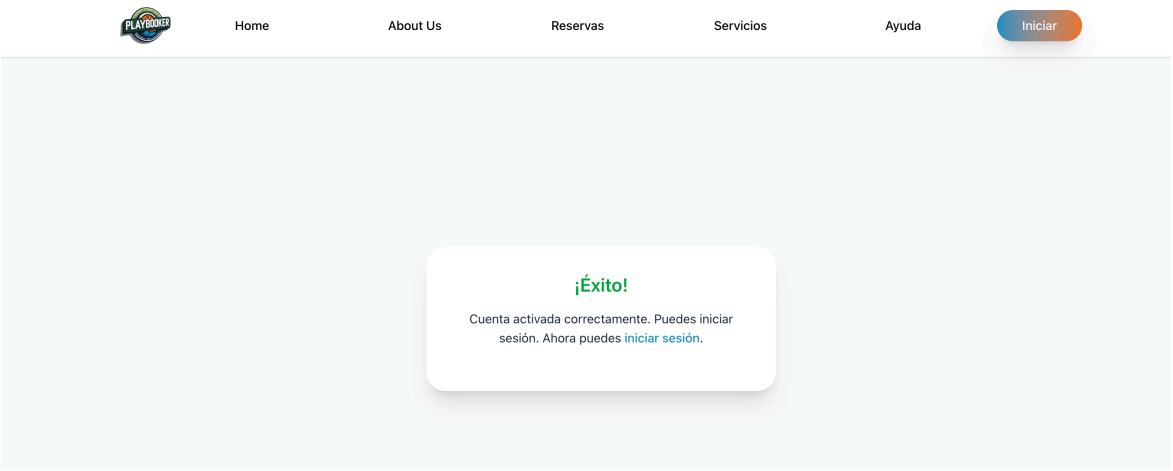
**Precondición:** Cuenta recién registrada con token de activación válido enviado al correo.

**Pasos:**

1. Hacer clic en el enlace de activación recibido por email.
2. Verificar redirección a la pantalla de login.

**Resultado esperado:** La cuenta pasa a “activo” y se muestra mensaje de confirmación.

***Figura 24:*** *Prueba funcional de activación por correo.*




**Prueba FT-03: Inicio de sesión exitoso**

**Precondición:** Cuenta activa existente.

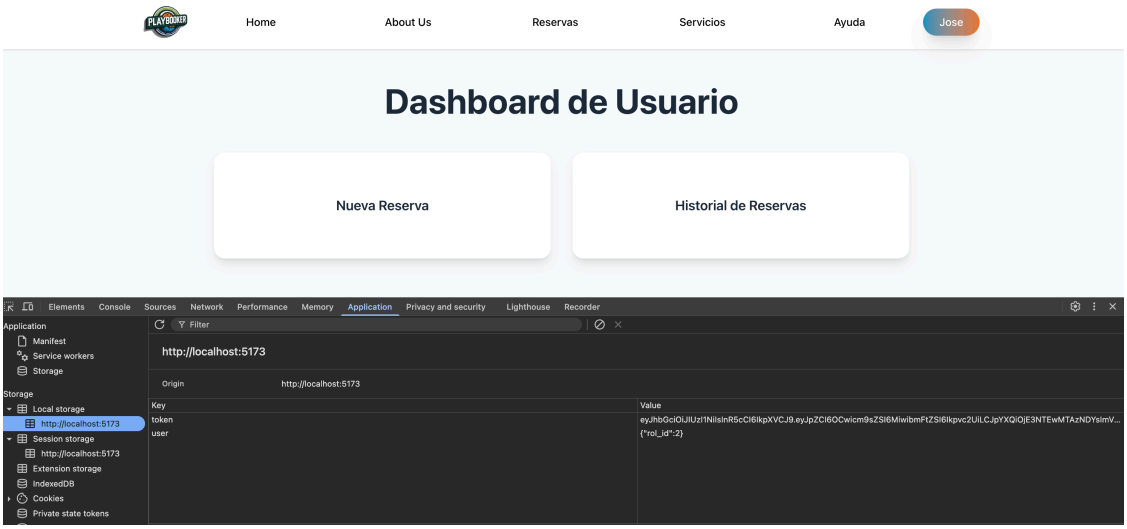
**Pasos:**

1. Navegar a la pantalla de login.
2. Introducir correo y contraseña correctos.

3. Pulsar “Iniciar sesión”.

   **Resultado esperado:** Redirección al dashboard de usuario y el token JWT queda guardado en localStorage.

*Figura 25: Prueba funcional local storage*




---

**Prueba FT-04: Inicio de sesión con credenciales incorrectas**

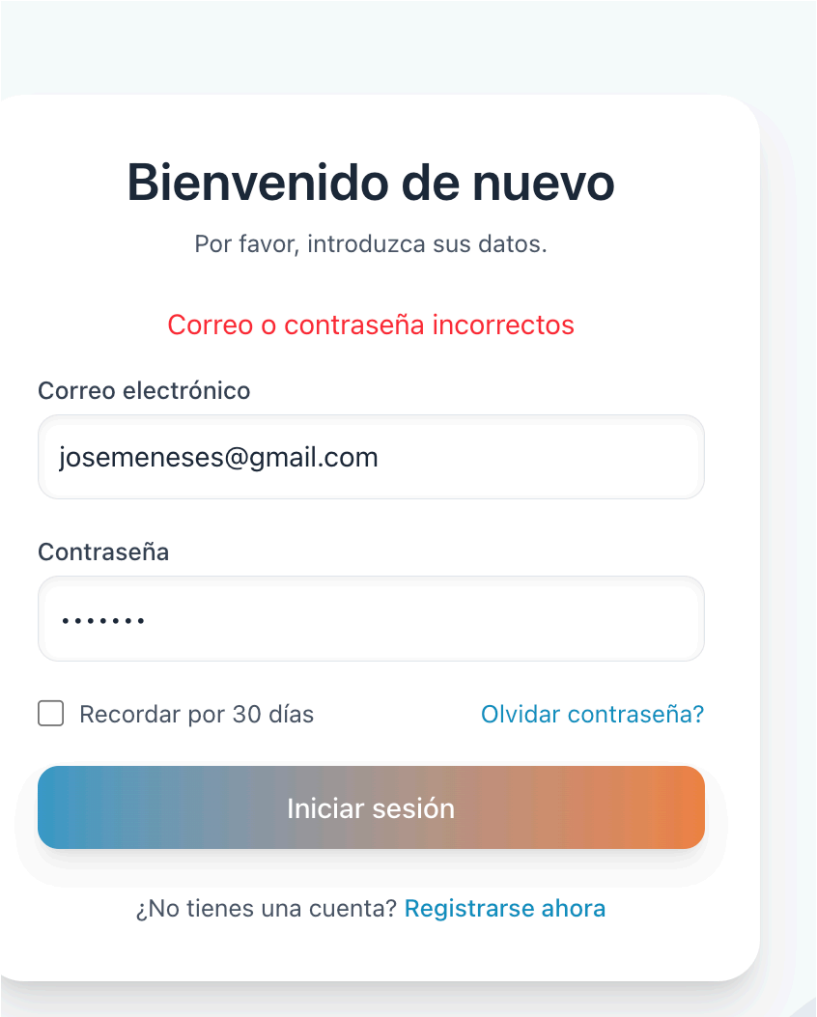
**Precondición:** Cuenta activa existente.

**Pasos:**

1. Acceder al formulario de login.
2. Introducir correo válido y contraseña incorrecta.
3. Pulsar “Iniciar sesión”.

**Resultado esperado:** Se muestra el mensaje “Usuario o contraseña incorrectos” y el usuario permanece en la misma pantalla.

*Figura 26: Login Validación de errores*



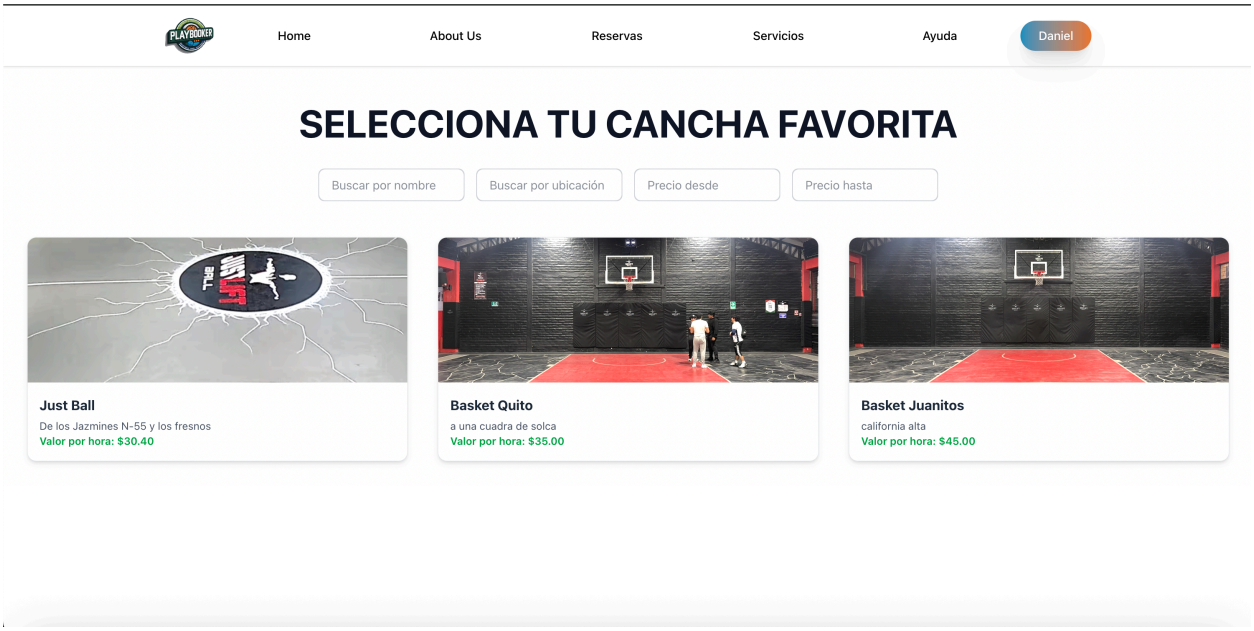

**Prueba FT-05: Listado de canchas por deporte**

**Precondición:** Usuario autenticado.

**Pasos:**

1. En el dashboard, pulsar el botón de un deporte (por ejemplo, “Basket”). **Resultado esperado:** Aparece un grid con todas las canchas disponibles de ese deporte, mostrando imagen, nombre y el precio.

*Figura 27: Pruebas listado de canchas*



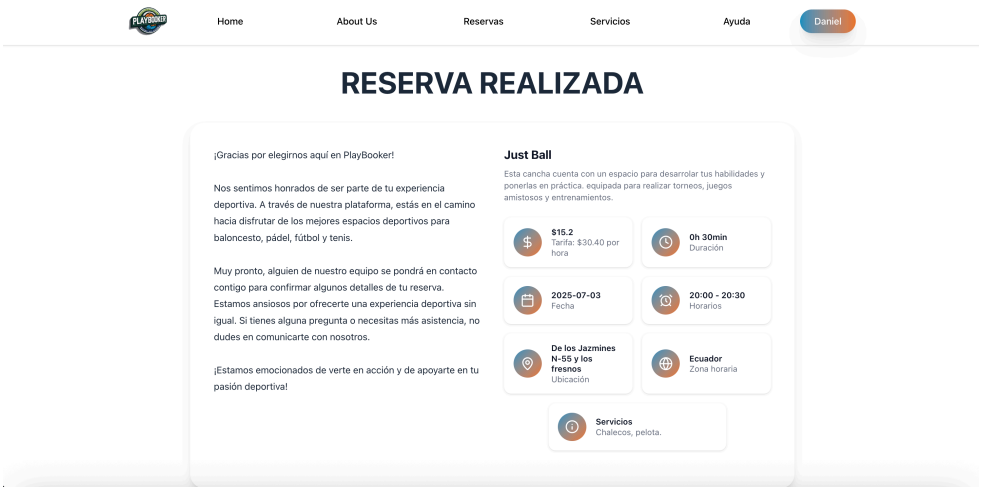

**Prueba FT-07: Realizar reserva y pago simulado (flujo exitoso)**

**Precondición:** Usuario autenticado; hay al menos una franja libre en el calendario.

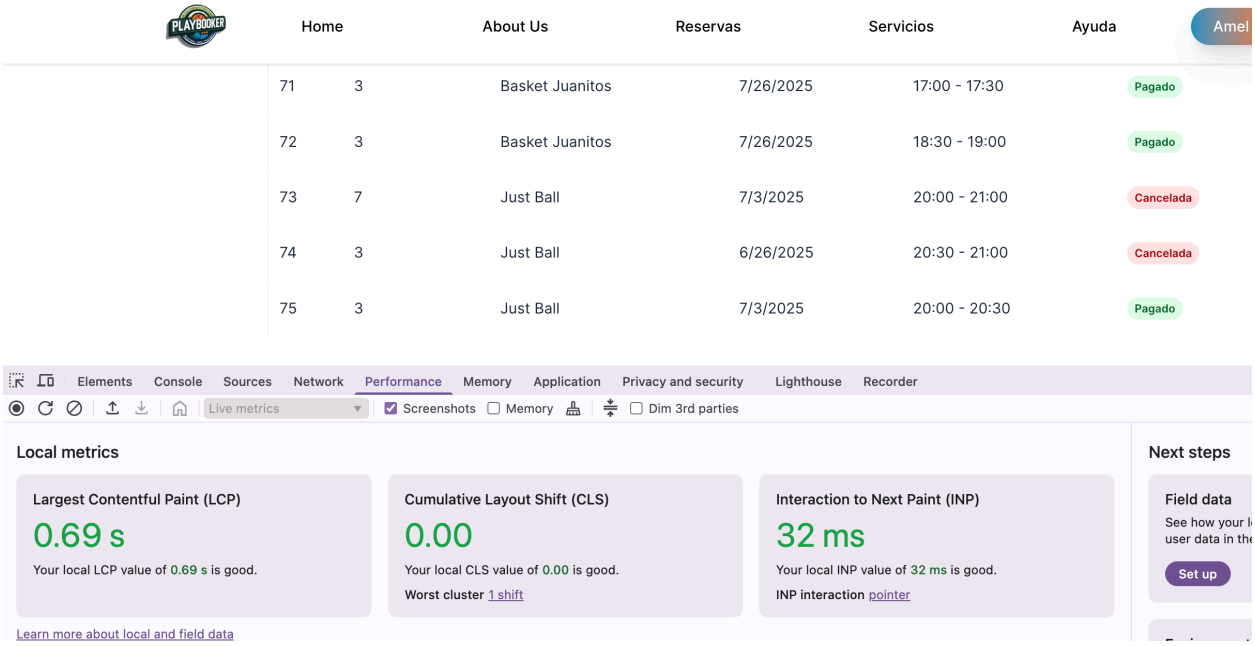
**Pasos:**

1. Seleccionar fecha y hora disponibles.
2. Rellenar el formulario de pago con datos válidos.
3. Pulsar “Confirmar”.

**Resultado esperado:** La reserva se crea con estado “paid”, se muestra pantalla de éxito al usuario.




# Prueba FT-07: Cargar 75  Reservas a un usuario y ver cuales son sus metricas



# 4. CAPÍTULO IV: CONCLUSIONES Y RECOMENDACIONES

## 4.1. Conclusiones

1. Se concluye que se ha cumplido con los objetivos general y específicos de este proyecto de integración curricular.
2. Podemos concluir que usando el framework React con tailwind CSS permitió que las vistas se rendericen mucho más rápido, también permitió mantener un orden, control y personalización sobresobre los estilos que se manejan en el sistema.
3. Al usar los frameworks React más Vite permite considerablemente la disminución de los tiempos de respuesta de la página web debido a que utilizan módulos ES nativos que permiten que la respuesta sea mucho más rápida y priorizando los elementos más principales de la página web.
4. Desarrollar un botón de pagos en un prototipo local no es una buena idea ya que existen muchas limitaciones y no existe un botón de pagos que te permita realizar pruebas sin que seas una empresa existente en el mercado.
5. Crear repositorios separados para el frontend y el backend en Github fue de gran ayuda debido a que al realizar microservicios nos aseguramos de que sea mucho más sencillo identificar errores y corregirlos además de que te permite seguir escalado a futuro.
6. Realizar un diseño previo al desarrollo nos ayuda a poder estar claros que tareas tomarán más tiempo y cuáles pueden ser los posibles errores al momento de desarrollar.

## 4.2. Recomendaciones

1. Para una implementación y uso del sistema se recomienda migrar a MySQL ya que eso permitiría un mejor balanceo de carga en la base de datos, además de permitir un alto

tráfico de datos, es una solución mucho más flexible para datos crecientes y se lo podría gestionar en la nube sin ningún problema.

2. Implementar testings automatizados como cypress ya que te ayuda asegurar regresiones en escenarios como autenticaciones, reservas y pagos.
3. Crear un botón de pagos que sirva para probar o usarlo en entornos locales sin necesidad de ser una empresa existente en el mercado.
4. Continuar con el desarrollo para implementar el módulo de facturación.
5. Implementar un endpoint que envie la ubicación de la cancha reservada al whatsapp del usuario.

# Bibliografía


Reva. (2025). *Reva*. Obtenido de Reva: https://reva.la

peloteo, A. (2025). *Alpeloteo*. Obtenido de Alpeloteo: https://alpeloteo.com/

EasyCancha. (2025). *EasyCancha*. Obtenido de EasyCancha: https://www.easycancha.com/es-EC

Gaetán, M., Saldaño, G., & Pires, V. (2017). Un enfoque integrador para diseñar y evaluar interfaces de usuario web. *Sedici*, 653-657.

Macías, E., & Loor, E. (2023). Propuesta de diseño de una metodología de desarrollo de software basado en el enfoque UI/UX. *Código Científico Revista de Investigación*, 387-405. doi:https://doi.org/10.55813/gaea/ccri/v4/nE1/83

Trujillo, M. (2023). Los métodos más característicos del diseño centrado en el usuario -DCU-, adaptados para el desarrollo de productos materiales. *Revista de Diseño Web*, 2010-2025. doi:http://dx.doi.org/10.18566/iconofact.v12.n19.a09

Hurtado, L. (2024). El diseño de experiencia de usuario (UX/UI) como apoyo en el aprendizaje de la expresión oral en los niños de 4 a 6 años. *Revista de Investigación Científica*, 147-158.

Salazar, O. (2023). Herramientas para el desarrollo rápido de aplicaciones web. *Scientia et technica*, 254-258.

Bernhard, B., & Botha, C. (2022). Herramienta de creación de prototipos. *Revista de Diseño Web*, 45-58.

Robalino, J. (2024). Herramientas de inteligencia artificial para diseñar prototipos. *Revista De Producción, Ciencias E Investigación*, 114-125. doi:https://doi.org/10.29018/issn.2588-1000vol8iss53.2024pp152-165

Díaz, M. (2022). Diseño de la biblioteca de JavaScript para construir interfaces de usuario. *Diseño e Investigación*, 45-48.

Quintana, D. (2022). Ract JS: La nueva tendencia en aplicaciones Web, enfocadas en el control dinámico de los datos. *Investigación Web*, 38-49.

Herrera, L. (2023). Propuesta de una herramienta de emprendimiento bajo el modelo React. *Revista de Desarrollo Web*, 47-58.

Bishmark, J. (2022). Vite: Revolucionando el desarrollo frontend de la web moderna. *Revista de creación de diseño web*, 35-48.

Domínguez, A. (2023). Vite: Herramienta Rápida y Eficiente para Desarrollar en JavaScript. *Diseño Web*, 19-28.

Carreño, N. (2022). Análisis, desarrollo y procesos tecnológicos. Una perspectiva de la aplicación Web. *Scielo*, 39-48.

Soriano, S. (2025). Desarrollo de una herramienta para la creación dinámica de interfaces CRUD con Node. js y React. *Revista Latina Multidisciplinar*, 36-48.

Llerena, L. (2021). Frameworks basados en typescript para el desarrollo de aplicaciones web interactivas. *Scielo*, 47-58. doi:https://doi.org/10.46377/dilemas.v8i3.2644

Segarra, D. (2022). Sitio web en lenguaje de programación de html y typescript para la asociación de no videntes de la zona sur del Cañar. *Revista Científica Multidisciplinar*, 54-68. doi:https://doi.org/10.37811/cl_rcm.v6i2.1884

Esquiviel, G., & Caluña, L. (2024). Frameworks del lado del Servidor: Caso de Estudio Node JS, Django y Laravel. *593 Digital Publisher CEIT*, 35-48. doi:doi.org/10.33386/593dp.2025.1.2729

Rodríguez, Z., Pino, L., & Gonzales, J. (2023). Arquitectura basada en Microservicios y DevOps para una ingeniería de software continua. *Redalyc*, 34-36. doi:http://dx.doi.org/10.15381/idata.v23i2.17278

Bangare, J. (2022). Uso de Node.Js para crear un servidor de base de datos backend escalable y de alta velocidad. *Scielo*, 34-48.

Torres, S. (2022). Componente de revisión de estándar de arquitectura de datos para el gestor de bases de datos SQLite. *Revista Innovación y Software*, 20-32.

Barthy, H. (2023). Sqlite Database - Análisis desde la perspectiva del desarrollo de software. *Redalyc*, 38-47.

Bauther, B. (2022). Investigación y aplicación de la tecnología de bases de datos integradas SQLite. *Scielo*, 58-78.

Paredes, L., & Arcos, G. (2019). Estudio del impacto del uso del sistema de control de versiones GitHub como herramienta de monitoreo y evaluación académica de trabajos colaborativos en instituciones de educación superior. *Revista Tecnológica ESPOL*, 124-148.

Maya, E. (2024). Documentación sobre el control de versiones. Enfoque del desarrollo Web. *Scielo*, 65-69.

Philipps, P. (2024). Gestión integral y desarrollo deportivo. Un enfoque desde la aplicación web. *Dialnet*, 35-65.

Niebles, W., & Sanabria , J. (2023). Análisis sistémico del contexto mundial de gestión deportiva: aportes significativos de la revista Retos. *Revista Retos*, 17-25.

Sommerville. (2011). *Software Engineering (9th ed.).* Boston, MA: Addison-Wesley.

Fowler. (2003). *Patterns of Enterprise Application Architecture.* Boston: Addison-Wesley.

Anderson, J. (2010). *Kanban: Successful Evolutionary Change for Your Technology Business.* Seattle, WA: Blue Hole Press.

Node.js, F. (n.d). *Node.js Foundation*. Obtenido de Overview of Blocking vs Non-Blocking: https://nodejs.org/en/learn/asynchronous-work/overview-of-blocking-vs-non-blocking

Sequelize. (n.d.). *Sequelize*. Obtenido de Feature-rich ORM for modern TypeScript & JavaScript: https://sequelize.org/

React. (n.d). *React*. Obtenido de React – A JavaScript library for building user interfaces: https://react.dev/

# Anexos

## A. Landing Page



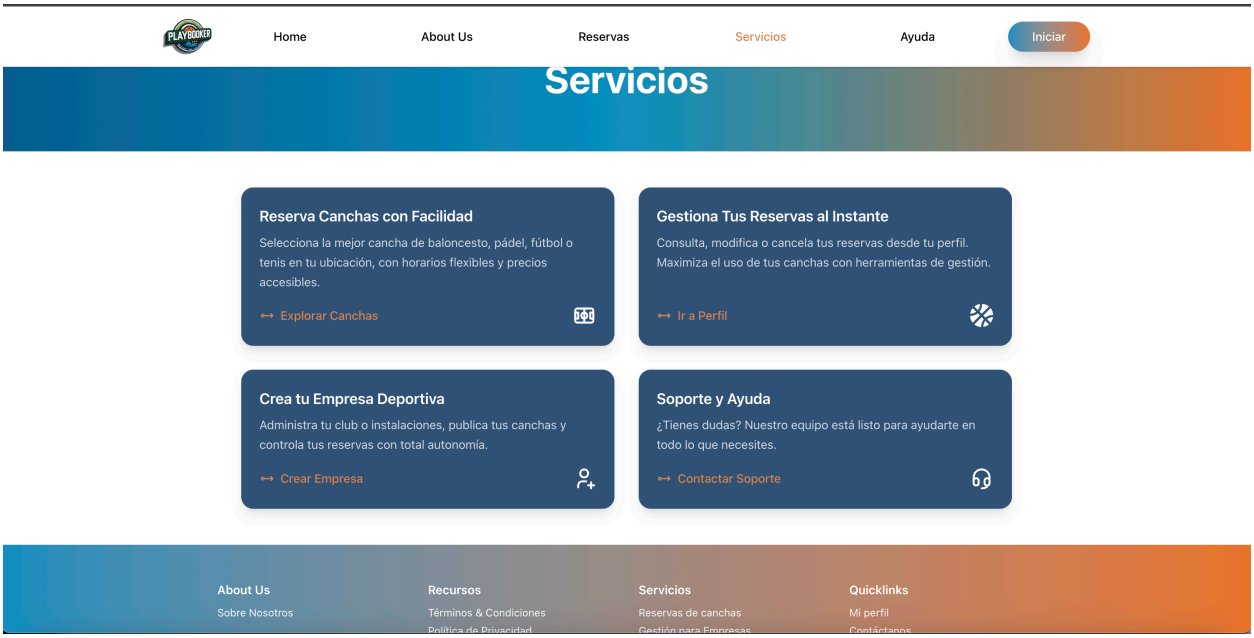




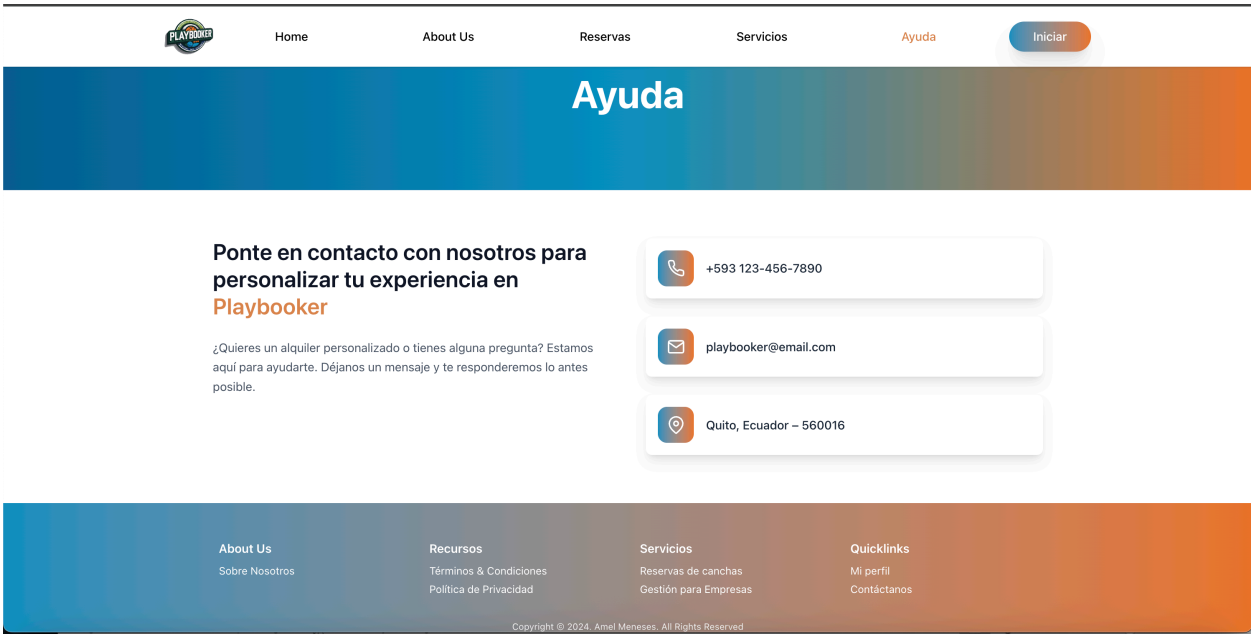
## B. About as



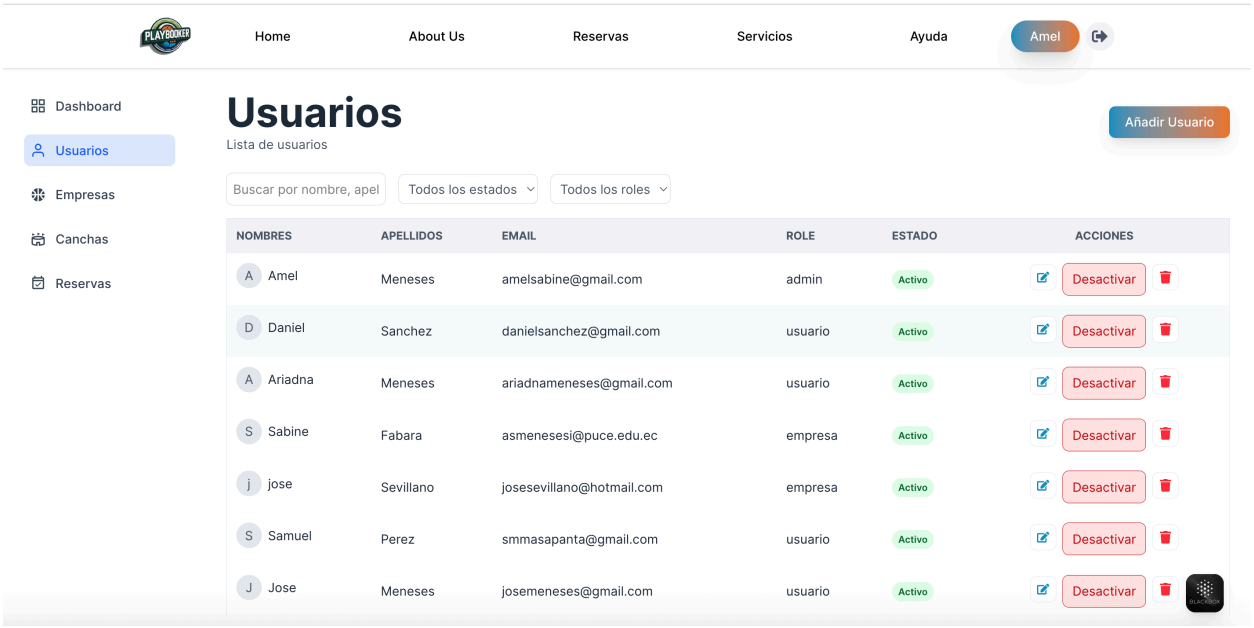
## C. Servicios




## D. Ayuda



E. Admin dashboard Usuarios




*F. Interfaz de reserva exitosa*



## *G. Interfaz de Historial de reservas*

