

Actividad 4 - Diseño del prototipo

Presentado por: Miguel Ángel Herrera Vargas

Presentado a: Tatiana Cabrera

CORPORACIÓN UNIVERSITARIA IBEROAMERICANA

Proyecto de Software

2025

Contenido	
Contextualización de la necesidad	3
Planteamiento del problema	4
Objetivos	5
Objetivos específicos	5
Alcance	6
Metodología	7
Matriz de riesgo	8
Levantamiento de información	10
Stakeholders	11
Requerimientos funcionales	12
Presupuesto	15
Historia de usuario	16
Prototipo funcional	17
Link – repositorio backend	17
Link – repositorio frontend – avances	17
Modelos de comportamiento y estructura	20
Diagrama de casos de uso	20
Diagrama de secuencia	21
Diagrama de clases	21
Diagrama de componentes	22
Diagrama de despliegue	22
Modelo de base de datos	23
Prototipos de interfaz	24
Link prototipo baja fidelidad	24
Link prototipo alta fidelidad	24
Link maze	24
Patrones arquitectónicos	25
Justificacion	26
GitHub	27
Bibliografia	28

Contextualización de la necesidad

En muchas comunidades urbanas, las canchas sintéticas han dejado de ser simples espacios deportivos para convertirse en escenarios de encuentro social, recreación y construcción de lazos vecinales. Sin embargo, el uso de estos espacios suele estar marcado por la falta de organización y reglas claras, lo que genera tensiones entre los habitantes. La necesidad de una herramienta que permita administrar las reservas de manera justa no responde únicamente a un tema logístico, sino también a un aspecto social: garantizar que el deporte y la convivencia se desarrollosen en condiciones de equidad, inclusión y respeto. Esta situación abre la puerta a la creación de un prototipo tecnológico pensado desde las realidades comunitarias, que permita mejorar el acceso a un recurso compartido de gran valor.

Planteamiento del problema

La existencia de canchas sintéticas en los barrios no garantiza que estas se utilicen de forma adecuada o equitativa. Por el contrario, la ausencia de un sistema transparente para gestionar reservas provoca conflictos frecuentes, favoritismo hacia ciertos grupos y exclusión de sectores como mujeres, adultos mayores o jóvenes sin afiliación a clubes. Esto no solo limita el acceso al deporte, sino que afecta directamente la convivencia comunitaria. Aunque en contextos privados existen sistemas digitales de gestión, estas soluciones no responden a las condiciones de las comunidades, donde predominan la autogestión y la organización local. Ante esta situación surge la necesidad de diseñar una herramienta accesible, simple y participativa que permita organizar de manera justa el uso de las canchas, fortaleciendo la cohesión social y evitando conflictos.

Objetivos

El propósito general de este proyecto es diseñar un prototipo funcional que permita gestionar las reservas de canchas sintéticas en comunidades urbanas de manera equitativa y sostenible. La idea es que la tecnología no solo resuelva un problema de organización, sino que también aporte a la construcción de confianza entre los vecinos y al fortalecimiento del sentido de pertenencia hacia los espacios públicos.

Objetivos específicos

1. Analizar, junto con los actores locales, las necesidades y dinámicas que surgen en torno al uso de las canchas comunitarias.
2. Formular los requerimientos funcionales y no funcionales de un sistema de reservas, garantizando que sean coherentes con la realidad de los usuarios.
3. Diseñar un prototipo navegable, accesible y adaptable que sirva como base para la validación comunitaria y como insumo para futuros desarrollos tecnológicos.

Alcance

El proyecto se centrará en el diseño de un prototipo funcional navegable que permita gestionar las reservas de canchas sintéticas en un entorno comunitario específico. El alcance no contempla un producto final en producción, sino una versión inicial que evidencie los flujos principales de reserva, consulta y validación de horarios. Su desarrollo estará condicionado por las limitaciones de tiempo propias del semestre académico y los recursos tecnológicos disponibles, lo cual restringirá la implementación a un piloto único dentro de una comunidad determinada. A pesar de estas limitaciones, el prototipo debe cumplir con criterios de aceptación claros: ofrecer una interfaz simple y accesible que permita a los usuarios organizar las reservas de manera transparente, evitar conflictos de dobles reservas y validar su funcionamiento mediante retroalimentación directa de los actores comunitarios involucrados.

Metodología

Para el desarrollo del proyecto se empleará la metodología ágil Scrum, ya que esta ofrece un marco flexible y colaborativo que se ajusta a la necesidad de trabajar con entregas parciales y validaciones constantes con los usuarios finales. La estructura de Scrum permitirá dividir el trabajo en sprints cortos, con revisiones periódicas que aseguren que el prototipo responda a las expectativas de la comunidad. En este esquema, el docente asesor asumirá el rol de Scrum Master, el equipo de estudiantes actuará como el grupo de desarrollo, y un representante comunitario podrá cumplir el rol de Product Owner, garantizando que las decisiones técnicas estén alineadas con las necesidades sociales. Además, se gestionará el progreso y las historias de usuario en un tablero digital (Trello o Jira), lo que facilitará la transparencia y la organización de las tareas del equipo.

Matriz de riesgo

No	Riesgo identificado	Probabilidad	Impacto	Nivel de riesgo	Estrategia de mitigación
1	Baja participación comunitaria en las entrevistas y grupos focales.	Media	Alta	Alto	Generar confianza a través de líderes barriales, realizar convocatorias inclusivas y ofrecer horarios flexibles para la participación.
2	Resistencia al cambio por parte de ciertos grupos acostumbrados a un manejo informal de las canchas.	Alta	Media	Alto	Involucrar a estos actores desde las primeras fases del diseño, destacando los beneficios colectivos de la solución.
3	Limitaciones técnicas (falta de internet o acceso a dispositivos móviles).	Media	Media	Medio	Diseñar un prototipo ligero, multiplataforma y accesible desde celulares básicos, con posibilidad de uso offline parcial.
4	Retrasos en la validación comunitaria del prototipo.	Media	Alta	Alto	Definir un cronograma claro con espacios de retroalimentación y mantener comunicación constante con los participantes.
5	Conflictos internos en la comunidad que afecten la implementación del proyecto.	Baja	Alta	Medio	Garantizar un enfoque neutral, mediado por la Junta de Acción Comunal o líderes reconocidos para reducir tensiones.
6	Dificultades en la recolección y análisis de la	Media	Media	Medio	Usar diferentes técnicas de triangulación

	información cualitativa.				(entrevistas, grupos focales, observación) para complementar los hallazgos.
7	Limitaciones de tiempo académico para cumplir todas las fases del proyecto.	Alta	Alta	Critico	Priorización de actividades esenciales, ajuste del cronograma y asignación clara de responsabilidades dentro del equipo.
8	Escasa apropiación de la solución por parte de la comunidad después del diseño del prototipo.	Baja	Alta	Medio	Asegurar que la herramienta se construya con participación activa, validación continua y adaptaciones según la retroalimentación.

Levantamiento de información

El levantamiento de información se realizará a través de metodologías participativas que permitan recoger las voces y experiencias de los usuarios de la cancha comunitaria. Para ello, se aplicarán entrevistas semiestructuradas a líderes locales y usuarios frecuentes, complementadas con grupos focales que faciliten la identificación de problemáticas comunes y expectativas colectivas. Además, se emplearán herramientas propias del diseño centrado en el usuario, como mapas de experiencia y perfiles de usuario, que ayudarán a estructurar los hallazgos de forma visual y estratégica. Estas técnicas permitirán comprender de manera profunda cómo se percibe actualmente la gestión de reservas, qué tensiones surgen entre los distintos actores y cuáles son las oportunidades de mejora. De manera preliminar, se ha identificado que la comunidad percibe injusticias en la asignación de horarios, manifiesta la necesidad de mayor transparencia en la administración de los espacios y espera una herramienta tecnológica sencilla, rápida y accesible desde dispositivos móviles.

Stakeholders

Stakeholders	Rol del proyecto	interés	poder	clasificación
Comunidad local	Uso del sistema	alto	medio	Clave
Junta de acción comunal	administradores	Alto	Alto	Decisiones
Estudiantes	Desarrollo	Alto	Alto	Facilitador

Requerimientos funcionales

código	Requisito funcional
RQF001	Nombre: Registro de usuario Descripción: el sistema permitirá al usuario realizar la acción de registrarse Usuarios: usuario

código	Requisito funcional
RQF002	Nombre: registro del administrador Descripción: el sistema permitirá al usuario realizar la acción de registrarse por medio de CLI Usuarios administrador

código	Requisito funcional
RQF003	Nombre: inicio de sesión Descripción: el sistema permitirá al usuario realizar la acción de iniciar sesión Usuarios: usuario , administrador

código	Requisito funcional
RQF004	Nombre: Cerrar sesión Descripción: el sistema permitirá al usuario realizar la acción de cerrar sesión Usuarios: usuario , administrador

código	Requisito funcional
RQF005	Nombre: actualizar perfil Descripción: el sistema permitirá al usuario realizar la acción de ver sus datos personales Usuarios: usuario , administrador

código	Requisito funcional
RQF006	Nombre: actualizar perfil Descripción: el sistema permitirá al usuario realizar la acción de actualizar sus datos personales Usuarios: usuario , administrador

código	Requisito funcional
RQF006	Nombre: crear espacios deportivos

	<p>Descripción: el sistema permitirá al usuario realizar la acción de registrar los espacios deportivos</p> <p>Usuarios: administrador</p>
--	--

código	Requisito funcional
RQF006	<p>Nombre: listar espacios deportivos</p> <p>Descripción: el sistema permitirá al usuario realizar la acción de listar los espacios deportivos</p> <p>Usuarios: administrador</p>

código	Requisito funcional
RQF006	<p>Nombre: listar espacios deportivos</p> <p>Descripción: el sistema permitirá al usuario realizar la acción de listar los espacios deportivos habilitados</p> <p>Usuarios: administrador, usuario</p>

código	Requisito funcional
RQF006	<p>Nombre: obtener espacio deportivo</p> <p>Descripción: el sistema permitirá al usuario realizar la acción de obtener el espacio deportivo</p> <p>Usuarios: administrador</p>

código	Requisito funcional
RQF006	<p>Nombre: obtener espacio deportivo</p> <p>Descripción: el sistema permitirá al usuario realizar la acción de obtener el espacio deportivo habilitado</p> <p>Usuarios: administrador, usuario</p>

código	Requisito funcional
RQF006	<p>Nombre: crear reserva</p> <p>Descripción: el sistema permitirá al usuario realizar la acción de reservar el espacio deportivo</p> <p>Usuarios: usuario</p>

código	Requisito funcional
RQF006	<p>Nombre: actualizar reserva</p> <p>Descripción: el sistema permitirá al usuario realizar la acción de actualizar la reserva</p>

Usuarios: administrador

código	Requisito funcional
RQF006	<p>Nombre: eliminar reserva</p> <p>Descripción: el sistema permitirá al usuario realizar la acción de actualizar (soft delete) la reserva</p> <p>Usuarios: usuario</p>

código	Requisito funcional
RQF006	<p>Nombre: eliminar reserva</p> <p>Descripción: el sistema permitirá al usuario realizar la acción de actualizar (soft delete) la reserva</p> <p>Usuarios: usuario</p>

código	Requisito funcional
RQF006	<p>Nombre: eliminar reserva</p> <p>Descripción: el sistema permitirá al usuario realizar la acción de eliminar la reserva</p> <p>Usuarios: administrador</p>

Código	Requisito no funcional
RQNF001	<p>Nombre: Especificaciones mínimas de hardware y software</p> <p>Descripción: Tener acceso a una computadora o mas</p> <p>Mínimo de RAM 4gb</p> <p>Sistema operativo Windows 8 o superior, Ubuntu 20.04 o superior</p> <p>Disco duro 256gb</p> <p>Conectividad a internet wifi, ethernet</p> <p>Que tenga las herramientas y programas para el desarrollo de la aplicación</p>

Presupuesto

1. Recursos humanos

Rol	Integrante	Horas estimadas	Valor ref	Hora	Subtotal
Analista de requerimientos	1	60h	\$ 20.000		\$ 1.200.000
Diseñador ux/ui	1	50h	\$ 20.000		\$ 1.000.000
Desarrollador backend	1	80h	\$ 20.000		\$ 1.600.000
Desarrollador frontend	1	80h	\$ 20.000		\$ 1.600.000
Tester QA	1	40h	\$20.000		\$ 800.000
Total					\$ 6.200.000

2. Recursos tecnológicos

Recurso	Descripción	Costo estimado
Dominio web	Registro por 1 año	\$ 50.000
Hosting web	Plan básico (6 meses)	\$ 600.000
Licencias de diseño(Figma)	Suscripción mensual (6 meses)	\$ 300.000
Herramientas de gestión (Trello - jira)	Versión gratuita	\$ 0
Total		\$ 950.000

3. Actividades de levantamiento de información

Recurso	Descripción	Costo estimado
materiales	Impresiones, formularios	\$ 100.000
Logística	Reuniones comunitarias (refrigerios)	\$ 200.000
Total		\$ 300.000

Para un total estimado para el desarrollo del proyecto es de \$8.195.000

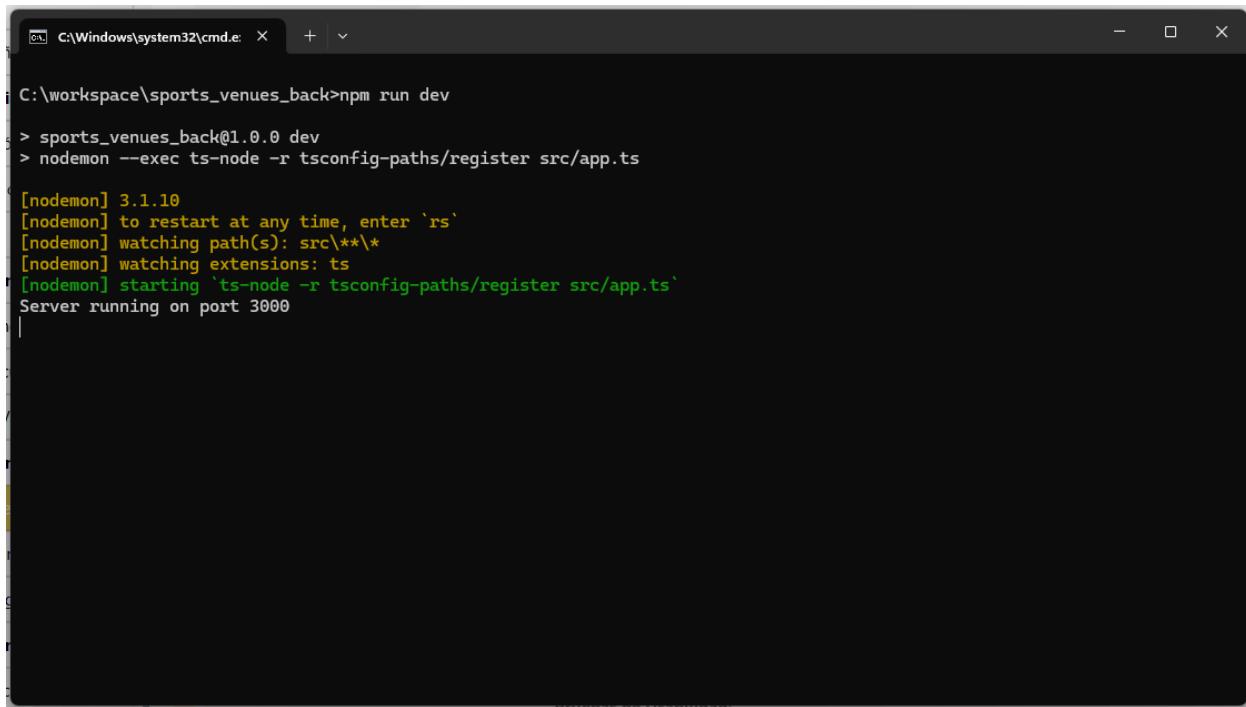
Historia de usuario

<https://trello.com/invite/b/68df1fa5ee4e9fd39fa5bfc9/ATTI51e3c75a2f3107edb1131739784e93b3D5ACD9A9/analisis-y-diseno>

Prototipo funcional

Link – repositorio backend

https://github.com/maherrera603/sports_venues_back.git

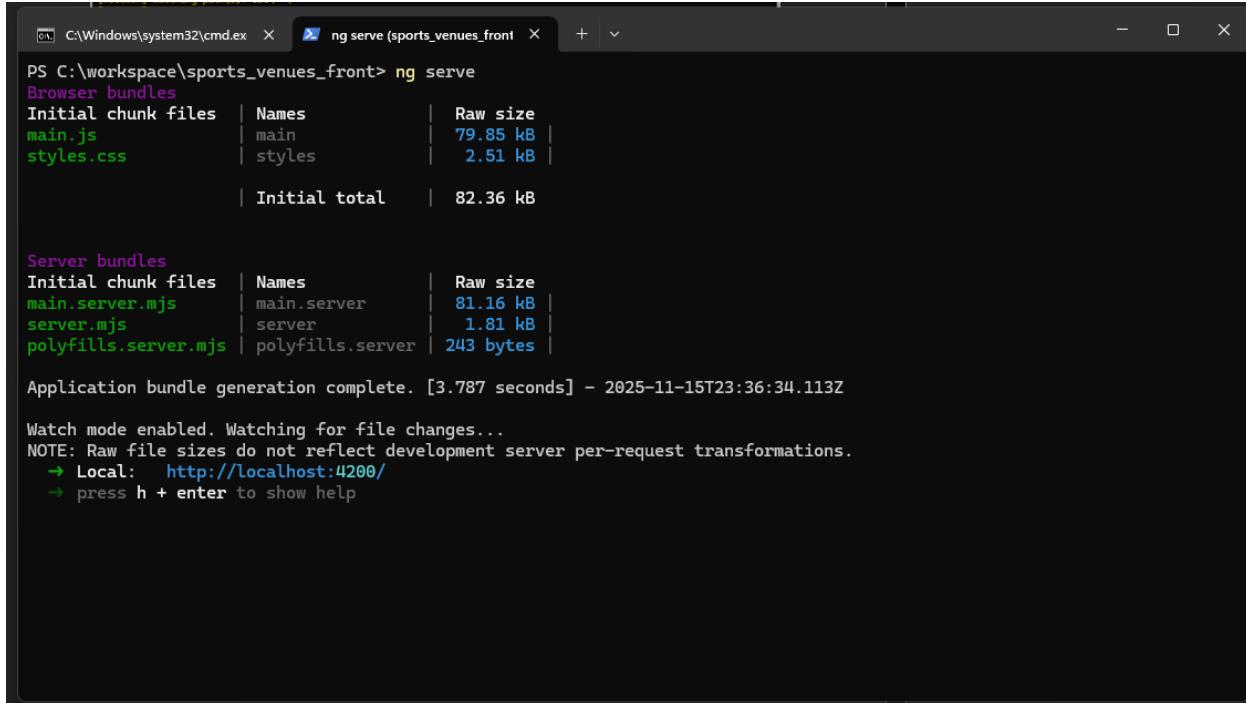


```
C:\workspace\sports_venues_back>npm run dev
> sports_venues_back@1.0.0 dev
> nodemon --exec ts-node -r tsconfig-paths/register src/app.ts

[nodemon] 3.1.10
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): src\**\*
[nodemon] watching extensions: ts
[nodemon] starting 'ts-node -r tsconfig-paths/register src/app.ts'
Server running on port 3000
```

Link – repositorio frontend – avances

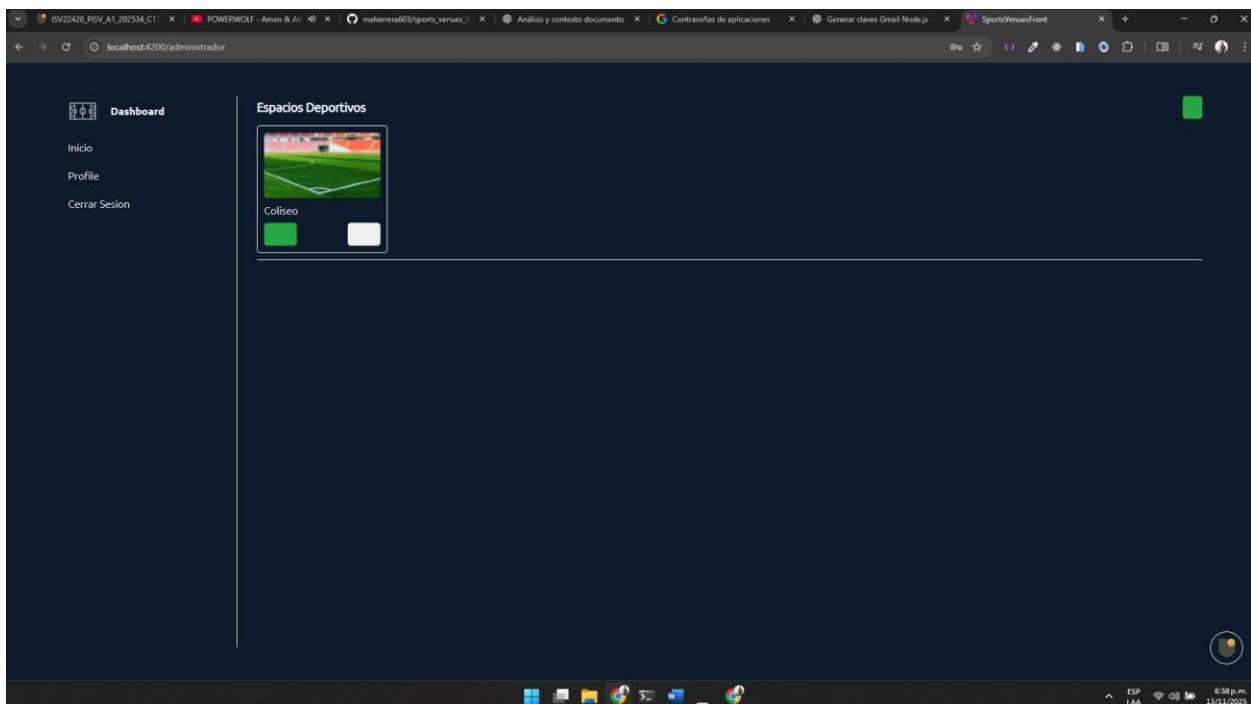
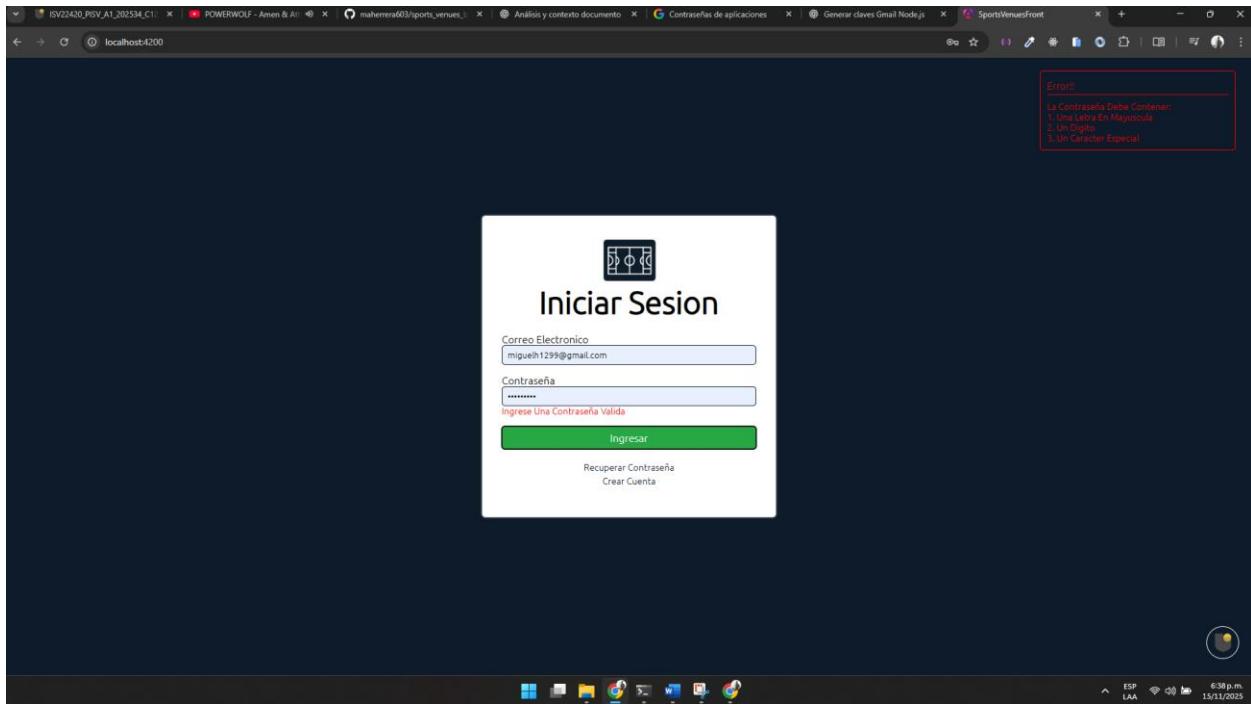
https://github.com/maherrera603/sports_venues_front

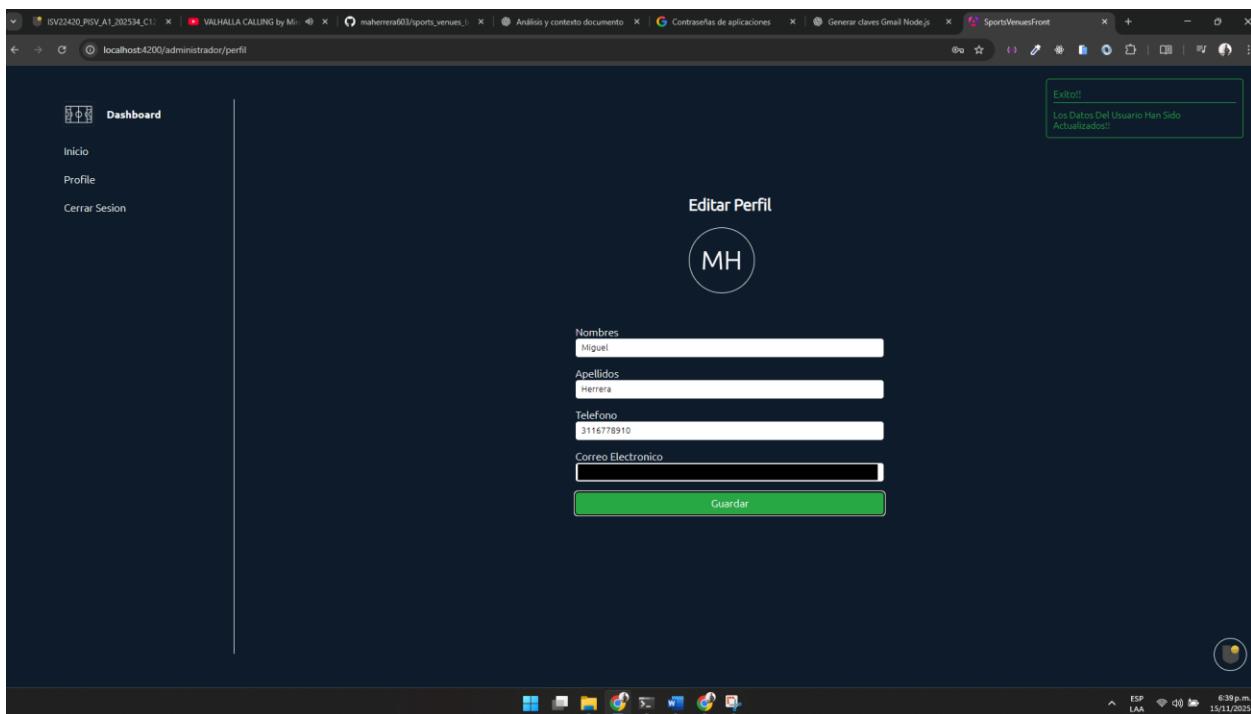


```
PS C:\workspace\sports_venues_front> ng serve
Browser bundles
Initial chunk files | Names | Raw size
main.js | main | 79.85 kB |
styles.css | styles | 2.51 kB |
| Initial total | 82.36 kB

Server bundles
Initial chunk files | Names | Raw size
main.server.mjs | main.server | 81.16 kB |
server.mjs | server | 1.81 kB |
polyfills.server.mjs | polyfills.server | 243 bytes |

Application bundle generation complete. [3.787 seconds] - 2025-11-15T23:36:34.113Z
Watch mode enabled. Watching for file changes...
NOTE: Raw file sizes do not reflect development server per-request transformations.
→ Local: http://localhost:4200/
→ press h + enter to show help
```





Modelos de comportamiento y estructura

Diagrama de casos de uso

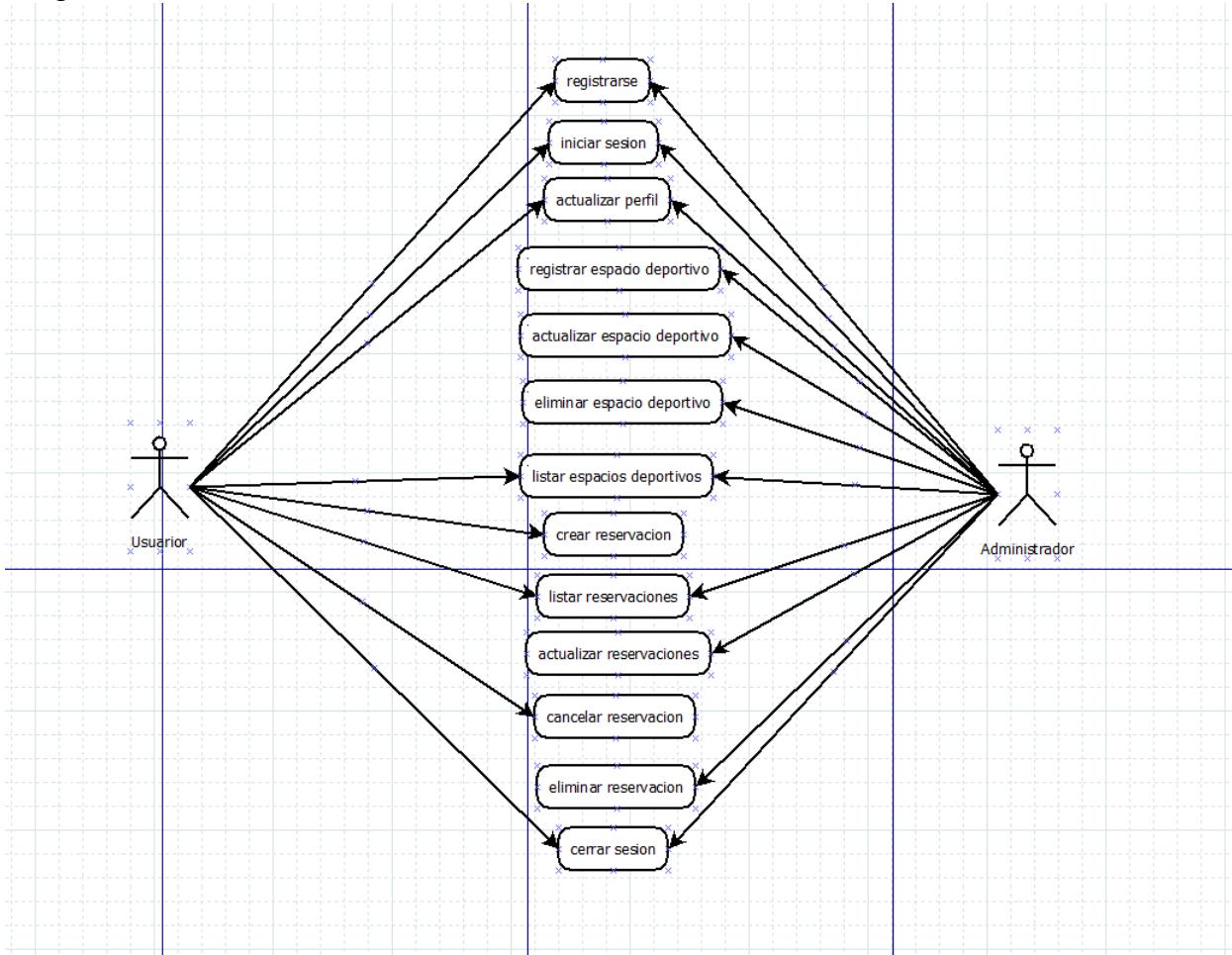


Diagrama de secuencia

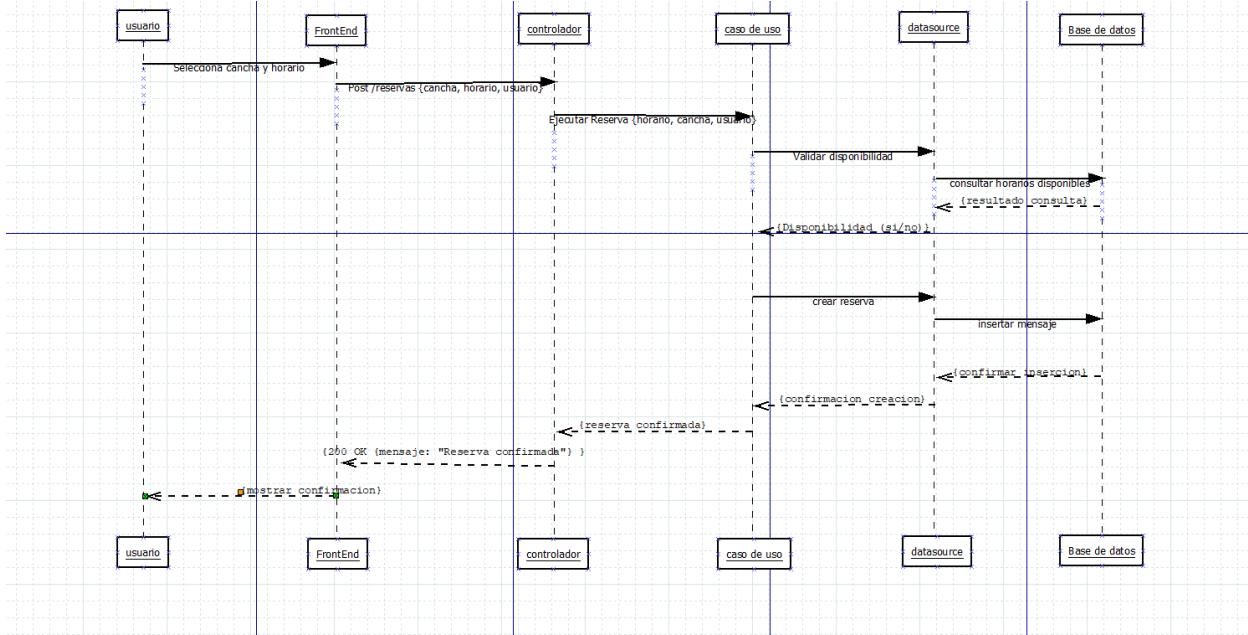


Diagrama de clases

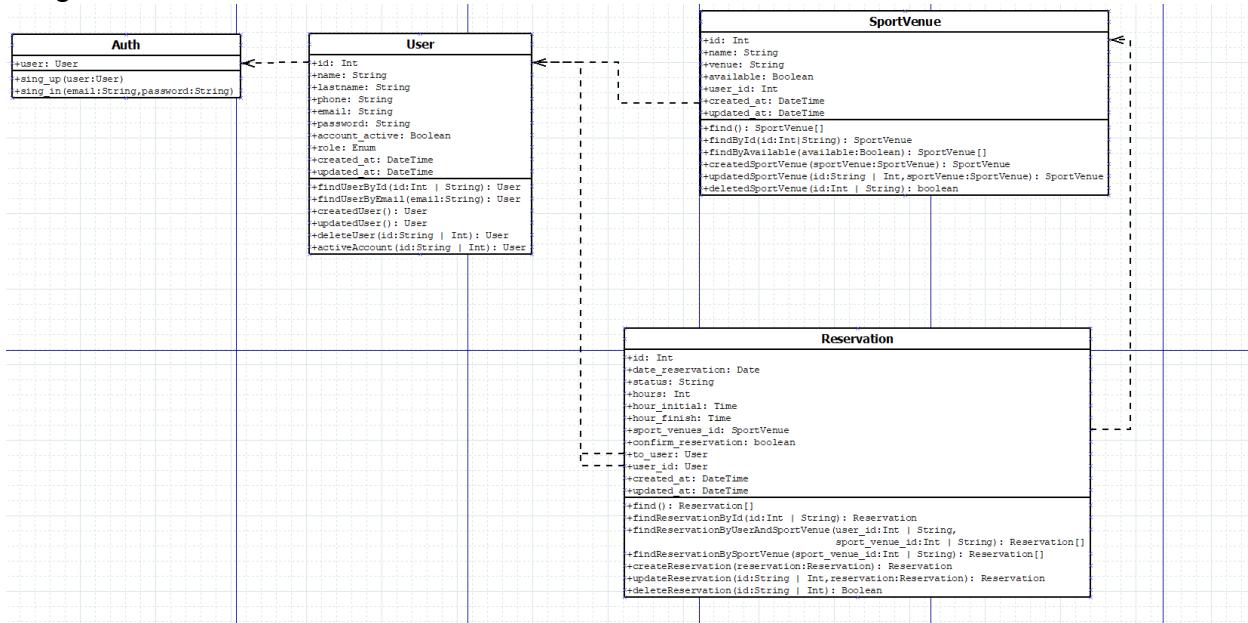


Diagrama de componentes

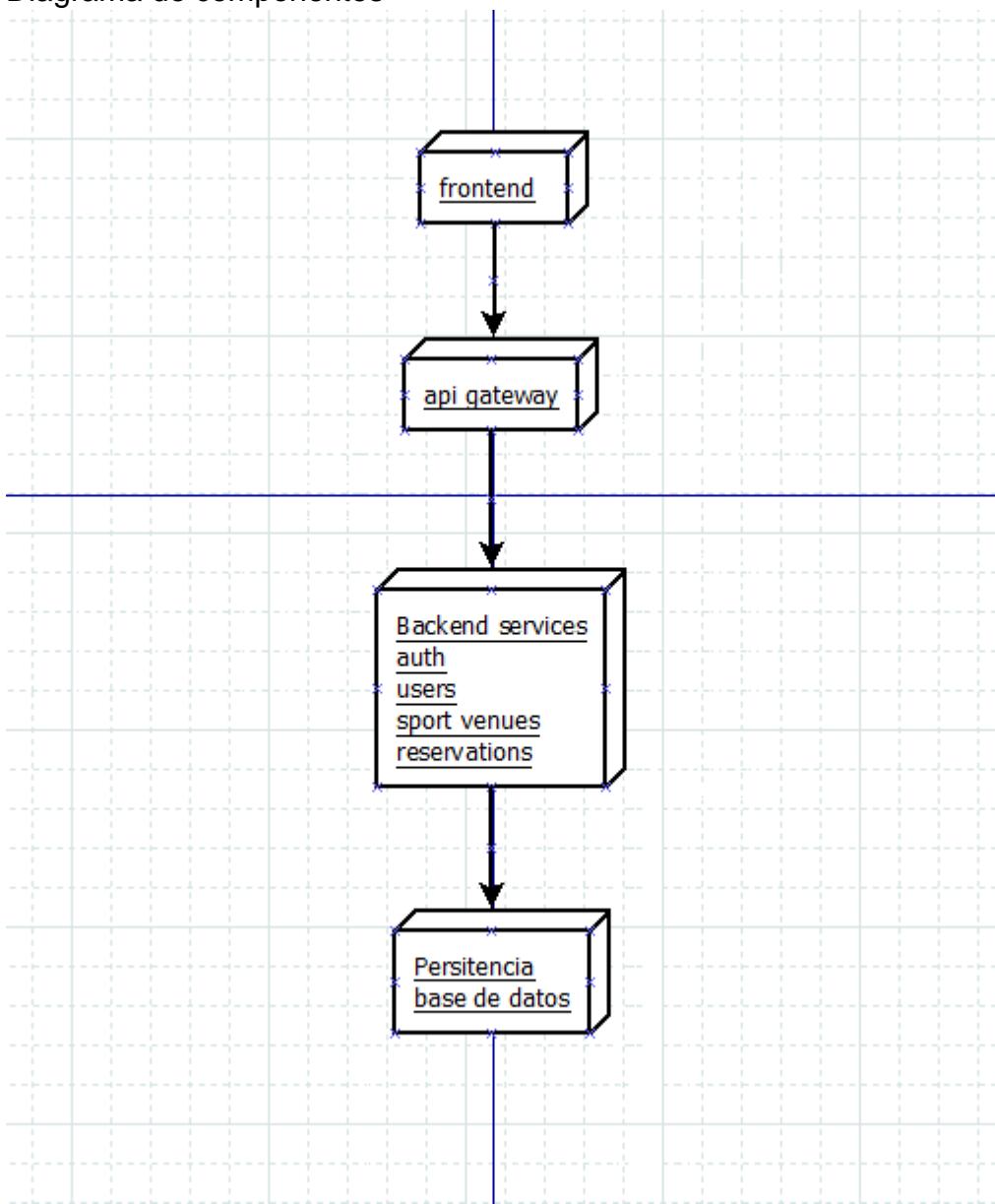
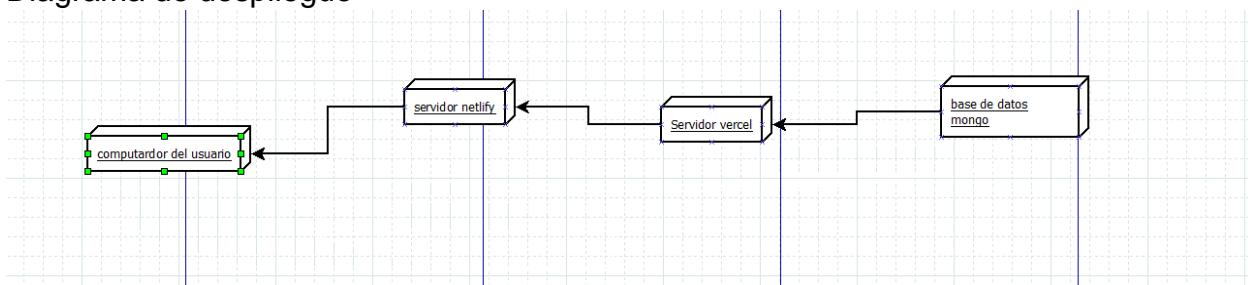
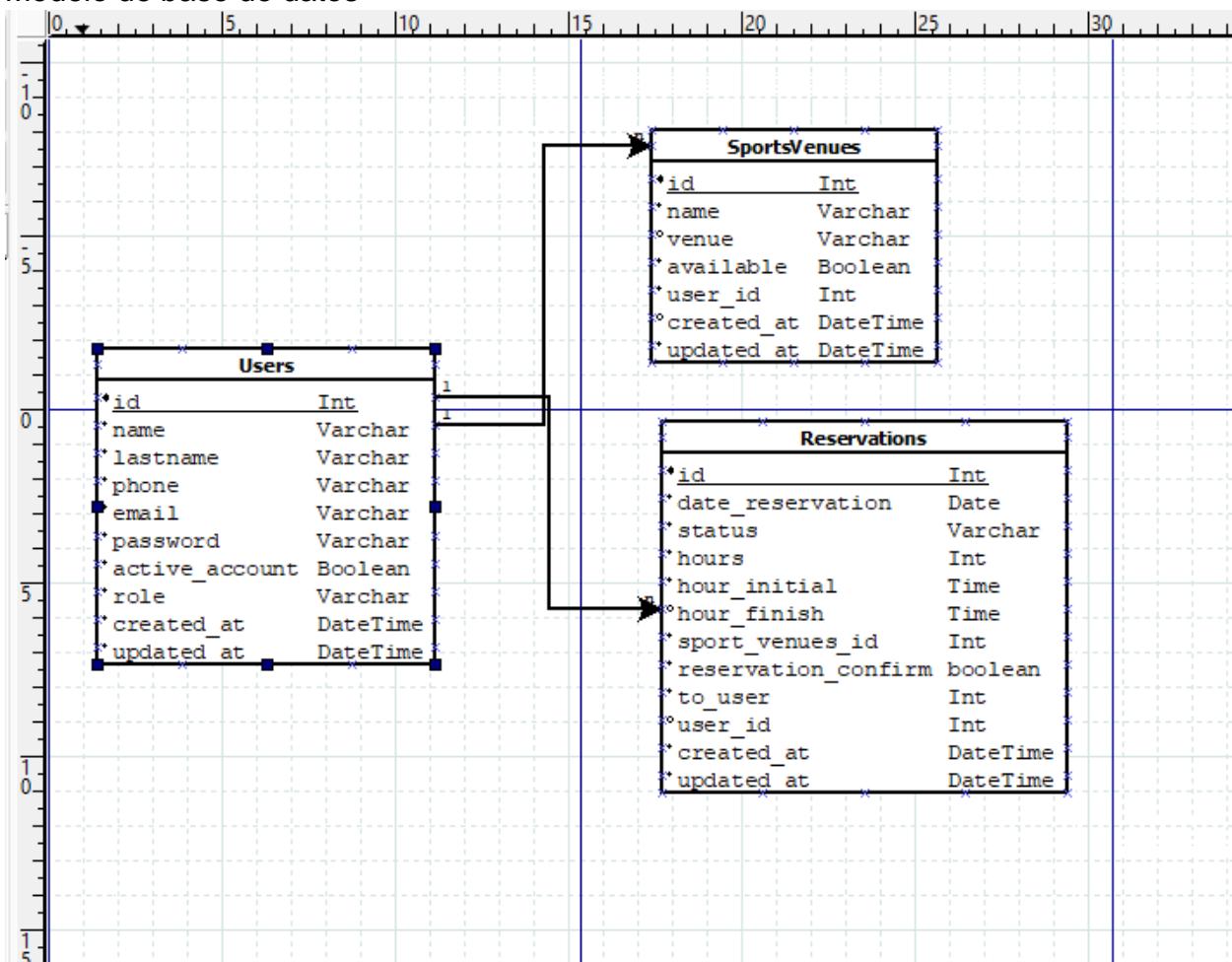


Diagrama de despliegue



Modelo de base de datos



Prototipos de interfaz

Link prototipo baja fidelidad

<https://www.figma.com/proto/BAbEMqFACx6loAk8Y0dp1V/Untitled?page-id=0%3A1&node-id=1-43&p=f&viewport=173%2C-181%2C0.98&t=1Ou7SSUiwfnAdixO-1&scaling=min-zoom&content-scaling=fixed&starting-point-node-id=1%3A43&showproto-sidebar=1>

Link prototipo alta fidelidad

<https://www.figma.com/proto/BAbEMqFACx6loAk8Y0dp1V/Untitled?page-id=1%3A2&node-id=25-1517&viewport=-550%2C307%2C0.98&t=1LSXE1Lrgm03kdOy-1&scaling=min-zoom&content-scaling=fixed&starting-point-node-id=25%3A1517>

Link maze

<https://t.maze.co/469439679>

Patrones arquitectónicos

Clean Architecture (Arquitectura Limpia)

El backend se diseñó bajo el patrón Clean Architecture, que separa el sistema en capas independientes, permitiendo que la lógica de negocio no dependa del framework, la base de datos ni servicios externos.

Capas implementadas:

Domain: entidades, reglas de negocio y modelos esenciales.

Use Cases / Application: casos de uso (crear reserva, validar disponibilidad, cancelar reserva, CRUD de espacios).

Interfaces / Adapters: controladores HTTP, DTOs, validaciones, repositorios como interfaces.

Infrastructure: implementación real de la base de datos, correo, cache, almacenamiento y API.

Justificación

La arquitectura se seleccionó en función de los siguientes criterios:

- Escalabilidad
La división en módulos (auth, reservas, espacios, usuarios) permite escalar horizontalmente cada parte cuando el sistema crezca.
- Robustez y mantenibilidad
 - Clean Architecture garantiza:
 - i. Código fácil de extender
 - ii. Dependencias bien controladas
 - iii. Reglas de negocio inalterables ante cambios tecnológicos
- Separación clara de responsabilidades
 - Cada capa cumple una función específica:
 - Domain → Reglas puras
 - Use Cases → Orquestación
 - Controllers → Entrada/salida HTTP
 - Infra → conexión con servicios externos
- Preparación para microservicios
Aunque el sistema es monolítico para reducir complejidad inicial, la estructura modular permite migrar cada servicio a microservicio sin reescribir la lógica.
- Facilita pruebas unitarias
Los casos de uso pueden probarse sin necesidad de base de datos real.

GitHub

<https://github.com/maherrera603/proyecto-software>

Bibliografia

Cabrera, A., & López, D. (2020). Diseño e implementación de un sistema de reservas para instalaciones deportivas comunitarias. Universidad Politécnica de Madrid.

Mata, A., & Sánchez, R. (2019). Plataformas web para la gestión de recursos comunitarios: modelos, requisitos y mejores prácticas. *Revista Iberoamericana de Sistemas*, 15(2), 45–60.

Villalobos, J., & Gómez, P. (2021). Arquitectura y desarrollo de aplicaciones web basadas en API REST para reservas y control de disponibilidad de recursos. *Revista Colombiana de Ingeniería de Software*, 13(1), 22–34.

Martin, R. C. (2017). *Clean Architecture: A Craftsman's Guide to Software Structure and Design*. Prentice Hall.

García, M., & Hernández, T. (2022). Modelado UML y despliegue de aplicaciones para sistemas de información comunitaria. *Revista Tecnología y Sociedad*, 10(3), 75–89.