PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

Unidad Académica de Formación Técnica y Tecnológica – PUCE TEC

SISTEMA WEB INTERACTIVO PARA EL GIMNASIO FORTFIT

Proyecto de titulación previo a la obtención del título de: Tecnólogo Superior en Desarrollo de Software

Autor: Joab Josue Zambrano Zambrano

Ricardo Rodrigo Carrión Villarreal

Tutor: José Luis Galarraga Cañizares

Quito, Ecuador

2025

Dedicatoria

Dedico este trabajo a Dios, quien me ha dado la vida y permitirme haber llegado hasta este punto de mi formación personal y académica. A mi familia, quienes siempre me han mostrado su amor incondicional y me han enseñado el valor del esfuerzo y la perseverancia. A Gracia, por motivarme en los momentos más difíciles y apoyarme en cada paso, aunque no siempre haya sido sencillo. A mis amigos que han estado en las buenas y las malas para escuchar mis quejas y hacerme reír. A mis compañeros de clase porque sin ellos este camino no hubiera sido igual de llevadero.

Josue Zambrano

Dedicatoria

A mi familia, por su apoyo constante y comprensión durante este proceso. A mis compañeros y amigos, por compartir conocimientos, experiencias y momentos de aprendizaje, A mis profesores, por su orientación y dedicación en mi formación académica, A todos ellos, mi sincero agradecimiento.

Ricardo Carrión

Tabla de contenidos

Lista de tablas

Lista de figuras

Declaración y Autorización

Yo, **Joab Josue Zambrano Zambrano** con C.I. 1725344772 autor(a) del trabajo de titulación intitulado: **"SISTEMA WEB INTERACTIVO PARA EL GIMNASIO FORTFIT"**, previa a la obtención del título de **Tecnólogo Superior en Desarrollo de Software** en la Unidad Académica de Formación Técnica y Tecnológica PUCE TEC:

1.- Declaro tener pleno conocimiento de la obligación que tiene la Pontificia Universidad Católica del Ecuador, de conformidad con el artículo 144 de la Ley Orgánica de Educación Superior, de entregar a la SENESCYT en formato digital una copia del referido trabajo de graduación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2.- Autorizo a la Pontificia Universidad Católica del Ecuador a difundir a través de sitio web de la Biblioteca de la PUCE el referido trabajo de titulación, respetando las políticas de propiedad intelectual de Universidad.

Quito, 15 de agosto de 2025

Joab Josue Zambrano Zambrano

C.I. 1725344772

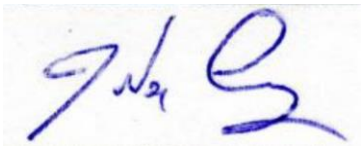
Declaración y Autorización

Yo, Ricardo Rodrigo Carrión Villarreal con C.I. 0706713997 autor(a) del trabajo de titulación intitulado: **"SISTEMA WEB INTERACTIVO PARA EL GIMNASIO FORTFIT"**, previa a la obtención del título de **Tecnólogo Superior en Desarrollo de Software** en la Unidad Académica de Formación Técnica y Tecnológica PUCE TEC:

1.- Declaro tener pleno conocimiento de la obligación que tiene la Pontificia Universidad Católica del Ecuador, de conformidad con el artículo 144 de la Ley Orgánica de Educación Superior, de entregar a la SENESCYT en formato digital una copia del referido trabajo de graduación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2.- Autorizo a la Pontificia Universidad Católica del Ecuador a difundir a través de sitio web de la Biblioteca de la PUCE el referido trabajo de titulación, respetando las políticas de propiedad intelectual de Universidad.

Quito, 15 de agosto de 2025



Ricardo Rodrigo Carrión Villarreal

C.I. 0706713997

Agradecimientos

Agradezco a Dios por bendecirme con personas en mi vida que me inspiran a ser mejor, por darme fuerzas para superar cualquier obstáculo que se me ha presentado y ayudarme a no decaer jamás.

A mis padres, porque sin su apoyo y amor no estaría donde estoy el día de hoy, por hacerme creer siempre que puedo lograr lo que sea que me proponga, y por amarme aun con mis defectos.

A Gracia, por ser la compañera ideal estando siempre a mi lado, por creer en mí y siempre tener una palabra de aliento en los momentos malos.

A mis amigos Avi, Juanfer, Juanjo y Willy, por siempre estar disponibles para conversar y reírnos de nuestras desgracias, por todos los momentos juntos y por muchos más.

A mis compañeros de clase, por todas esas sesiones de estudio y trabajos realizados, por el apoyo y perseverancia al alcanzar este logro.

Josue Zambrano

Agradecimientos

Quiero agradecer especialmente a mi madre, que siempre ha estado ahí apoyándome en todo momento y ha hecho posible que llegue hasta aquí.

A mis abuelos, por creer en mí desde el principio, el esfuerzo, las palabras de aliento y consejo. Sin su apoyo, nada de esto habría sido posible.

Gracias por ser mi mayor motivación y por nunca dejar de confiar en mí.

Ricardo Carrión

Introducción

Contexto y problemática

En la era digital actual, la transformación tecnológica ha revolucionado prácticamente todos los sectores de la economía, y la industria del fitness no es la excepción. Las empresas del sector deportivo han experimentado una aceleración grande en la adopción de tecnologías digitales, especialmente después de la pandemia de COVID-19, lo que ha llevado a redefinir las expectativas de los usuarios respecto a los servicios que reciben (Health & Fitness Association, 2020). Se estima que en Estados Unidos los usuarios de programas de fitness digitales pasaron de 63 millones en 2018 a 145 millones en 2023, un incremento del 130% en cinco años (Lopez, 2024).

El gimnasio FortFit, ubicado en Quito, Ecuador, es un establecimiento moderno que cuenta con instalaciones de primer nivel, equipos de última generación y un equipo de profesionales altamente capacitados. Sin embargo, como muchas empresas tradicionales del sector fitness, enfrenta el desafío de mantenerse competitivo en un mercado cada vez más digitalizado. La organización actualmente opera con un sistema administrativo que gestiona eficientemente aspectos como membresías, control de ingreso y facturación, pero carece de herramientas digitales orientadas a mejorar la experiencia directa del cliente.

La problemática central identificada radica en que la mayoría de los procesos no administrativos se manejan de manera manual, lo que incluye la reserva de citas con especialistas en nutrición y fisioterapia, la capacitación sobre ejercicios y rutinas por parte de los entrenadores, y el acceso a información relevante sobre los servicios del gimnasio. Esta situación genera ineficiencias operativas y limita la capacidad de FortFit para ofrecer una experiencia de usuario moderna y personalizada que cumpla con las expectativas de los clientes digitales actuales.

Además, FortFit presenta una presencia digital limitada, contando únicamente con perfiles en redes sociales, pero sin una plataforma web propia donde los clientes potenciales puedan acceder a información detallada sobre servicios, horarios, precios e instalaciones sin necesidad de contacto directo. Esta carencia representa una desventaja competitiva significativa en el mercado actual, donde la presencia digital se ha convertido en un factor determinante para la atracción y retención de clientes (M, 2023).

Justificación del proyecto

La justificación para el desarrollo de un sistema web interactivo para FortFit se fundamenta en múltiples dimensiones que abarcan aspectos tecnológicos, operativos y estratégicos. Desde la perspectiva tecnológica, la implementación de soluciones digitales en el sector fitness ha demostrado mejoras significativas en la satisfacción del cliente y la eficiencia operativa (Lopez, 2024). La automatización de procesos como la reserva de citas, el acceso a rutinas personalizadas y la consulta de información del gimnasio no solo reduce la carga de trabajo del personal, sino que también proporciona a los usuarios un mayor control sobre su experiencia de entrenamiento (Munkholm, 2025).

Desde el punto de vista económico, la digitalización de servicios representa una inversión estratégica que puede generar retornos significativos a través de la optimización de recursos humanos y la mejora en la retención de clientes (Obregon, 2024). Un sistema web interactivo permite al gimnasio operar de manera más eficiente, reduciendo el tiempo que el personal debe dedicar a tareas administrativas repetitivas y permitiendo que se concentren en actividades de mayor valor añadido, como el entrenamiento personalizado y la atención al cliente (Munkholm, 2025).

La justificación social del proyecto radica en la democratización del acceso a la información y servicios del gimnasio. Las plataformas digitales en el sector fitness han

contribuido significativamente a reducir las barreras de acceso y a proporcionar experiencias más inclusivas y personalizadas. El sistema propuesto permitirá a los usuarios acceder a información relevante, gestionar sus citas y rutinas, y utilizar recursos educativos desde cualquier dispositivo con conexión a internet, lo que representa una mejora sustancial en la accesibilidad de los servicios.

Finalmente, desde una perspectiva competitiva, el desarrollo de este sistema posicionará a FortFit como un gimnasio innovador y orientado al futuro, diferenciándolo de competidores que aún operan con métodos tradicionales y fortaleciendo su propuesta de valor en el mercado local.

Objetivos

Objetivo General

Desarrollar una aplicación web interactiva para el gimnasio FortFit que integre herramientas tecnológicas modernas con el fin de optimizar la experiencia del cliente, automatizar procesos clave y ofrecer una plataforma digital comprensiva que mejore el acceso a recursos y servicios del gimnasio.

Objetivos Específicos

1. Analizar las necesidades del gimnasio y de los usuarios mediante un diagnóstico de los problemas actuales en la gestión de servicios y la definición precisa de los requerimientos funcionales y técnicos de la aplicación.
2. Diseñar la arquitectura técnica y funcional de la aplicación a través de la creación de modelos de datos optimizados para Supabase y la elaboración de wireframes y mockups que garanticen una interfaz de usuario intuitiva, accesible y centrada en la experiencia del usuario.

3. Implementar un sistema full-stack robusto utilizando Next.js como framework principal, configurando la autenticación segura de usuarios mediante Supabase Auth y desarrollando API endpoints eficientes utilizando las API Routes de Next.js para garantizar una interacción fluida con la base de datos.

4. Desarrollar interfaces de usuario responsivas y optimizadas para dispositivos móviles, utilizando componentes dinámicos de Next.js e implementando funcionalidades interactivas como mapas del gimnasio, sistemas de reservas, biblioteca multimedia y gestión de rutinas personalizadas.

5. Ejecutar un proceso integral de pruebas y aseguramiento de calidad del sistema mediante el diseño y ejecución de pruebas unitarias, de integración y de usuario para validar el funcionamiento correcto de todas las funcionalidades, así como la implementación de medidas de seguridad robustas para proteger los datos sensibles de los usuarios.

Alcance y limitaciones

Alcance del proyecto

El presente proyecto abarca el desarrollo completo de una aplicación web interactiva dividida en dos secciones principales: una sección pública accesible a cualquier visitante y una sección privada exclusiva para miembros registrados del gimnasio.

La sección pública incluye el desarrollo de páginas informativas que presentan los servicios ofrecidos, planes de membresía con precios actualizados, horarios de operación para ambas sedes (América y El Inca), información sobre el personal especializado, y un sistema de contacto directo. Esta sección está optimizada para

motores de búsqueda (SEO) y diseñada con un enfoque responsivo que garantiza una experiencia óptima en dispositivos móviles y de escritorio.

La sección privada comprende el desarrollo de un sistema personalizado para usuarios autenticados que incluye: un sistema completo de gestión de citas con especialistas en nutrición y fisioterapia, herramientas para la creación de rutinas de entrenamiento personalizadas, un mapa interactivo de las instalaciones del gimnasio que permite la exploración virtual de los tres niveles del establecimiento, y una biblioteca de ejercicios con material instructivo categorizados por tipo de ejercicio y grupo muscular.

El proyecto incluye la implementación de un sistema de autenticación seguro con diferentes niveles de acceso (cliente, entrenador, especialista, administrador), la integración con la base de datos existente del sistema administrativo del gimnasio mediante la importación de datos, y el desarrollo de una aplicación web progresiva (PWA) que permite la instalación en los dispositivos y funcionalidad offline limitada.

Limitaciones del proyecto

Las limitaciones identificadas para este proyecto se clasifican en temporales, tecnológicas y de alcance funcional.

Limitaciones temporales. El desarrollo del proyecto está condicionado por el calendario académico, con un período de ejecución que se extiende desde febrero hasta agosto de 2025, lo que representa un marco temporal de seis meses para completar todas las fases del desarrollo. Esta limitación temporal influye directamente en la profundidad de algunas funcionalidades avanzadas y en la cantidad de iteraciones de prueba y refinamiento que se pueden realizar.

Limitaciones tecnológicas. El proyecto está circunscrito al uso de tecnologías web estándar y no incluye el desarrollo de aplicaciones móviles nativas para iOS o Android. Aunque la aplicación es completamente responsiva y funciona como PWA, carece de funcionalidades específicas de aplicaciones nativas como notificaciones push avanzadas o integración profunda con sensores del dispositivo. Además, las funcionalidades de procesamiento de video en tiempo real o análisis de movimiento mediante inteligencia artificial quedan fuera del alcance debido a la complejidad técnica y las limitaciones de tiempo.

Limitaciones de integración. Aunque el sistema se integra con el sistema administrativo existente para sincronizar información de usuarios y membresías, no incluye la integración completa con sistemas de pagos en línea o facturación electrónica automatizada, manteniéndose estos procesos en el sistema administrativo actual.

Limitaciones de contenido. La biblioteca multimedia inicial estará poblada con contenido de demostración y educativos básico. La creación de contenido multimedia profesional extenso queda como una actividad posterior al lanzamiento del sistema, requiriendo la colaboración continua del personal especializado del gimnasio.

Capítulo I: Levantamiento de Requisitos y Diseño del Sistema

Análisis de la situación actual

Contexto organizacional

FortFit es un gimnasio establecido en la ciudad de Quito, Ecuador, que opera con dos sedes estratégicamente ubicadas: una en el sector de la América y otra en El Inca. La organización se ha posicionado como un centro de acondicionamiento físico integral que ofrece servicios de musculación, entrenamiento cardiovascular, clases grupales, nutrición deportiva y fisioterapia. Con una trayectoria consolidada en el

mercado local, FortFit cuenta con instalaciones modernas distribuidas en tres niveles, equipamiento de última generación y un equipo de profesionales especializados que incluye entrenadores certificados, nutricionistas y fisioterapeutas.

La estructura organizacional del gimnasio se compone de aproximadamente 15 empleados distribuidos entre personal administrativo, entrenadores, especialistas en nutrición y fisioterapia, y personal de mantenimiento. Esta configuración permite atender a una base de clientes activos que, según los registros del sistema administrativo actual, supera los 600 miembros entre ambas sedes.

Infraestructura tecnológica actual

El análisis de la infraestructura tecnológica existente en FortFit revela una adopción parcial de soluciones digitales, concentrada principalmente en el área administrativa. El gimnasio cuenta con un sistema de gestión administrativa que cumple funciones específicas de control operativo, incluyendo:

- Sistema de control de acceso: Implementado mediante lectores biométricos que registran el ingreso y salida de los miembros.
- Gestión de membresías: Software administrativo que mantiene el registro de clientes, tipos de membresía, fechas de vencimiento y pagos.
- Facturación: Sistema integrado que permite la emisión de facturas electrónicas según la normativa ecuatoriana vigente.

Sin embargo, esta digitalización no se extiende a los procesos orientados directamente al servicio del cliente. Esta carencia representa una brecha tecnológica notable dado el creciente uso de plataformas conectadas en el sector fitness. Por ejemplo, un estudio indica que el 70% de los usuarios que comenzaron a ejercitarse en

línea durante la pandemia planea mantener o aumentar ese hábito incluso después (uso de aplicaciones y equipos conectados) (Falardeau et al., 2021). La ausencia de canales digitales propios (como apps o portales web) para rutina y citas contrasta con esta tendencia de mercado.

Procesos operativos actuales

El análisis detallado de los procesos operativos revela múltiples áreas donde la gestión manual predomina, generando ineficiencias y limitando la calidad del servicio:

Gestión de citas con especialistas. El proceso actual para agendar citas con nutricionistas o fisioterapeutas requiere que el cliente se acerque físicamente a la recepción o contacte mediante WhatsApp al especialista con el que desea ser atendido. El especialista consulta manualmente una agenda física o un archivo Excel para verificar disponibilidad, proceso que consume un promedio de 5-10 minutos por cita y es propenso a errores de doble agendamiento o pérdida de información.

Diseño y seguimiento de rutinas. Los entrenadores elaboran rutinas personalizadas, las cuales comunican o entregan a cada cliente cuando esta es solicitada. Esta metodología presenta varios inconvenientes: pérdida frecuente de las hojas de rutina por parte de los clientes, imposibilidad de actualizar las rutinas de manera dinámica, y falta de un sistema de seguimiento del progreso.

Capacitación sobre uso de equipos. La instrucción sobre el uso correcto de máquinas y técnicas de ejercicio depende exclusivamente de la disponibilidad inmediata de los entrenadores en el piso del gimnasio. Esto genera situaciones donde los usuarios nuevos o aquellos que desean probar equipos diferentes deben esperar a que un entrenador esté disponible, lo que puede resultar en uso inadecuado del equipo o, en el peor de los casos, lesiones.

Comunicación con clientes. La comunicación se limita a llamadas telefónicas, mensajes de WhatsApp al número corporativo, o interacciones a través de las redes sociales del gimnasio. Esta fragmentación de canales dificulta el seguimiento sistemático de las consultas y genera respuestas inconsistentes o tardías.

Presencia digital y posicionamiento en línea

La evaluación de la presencia digital de FortFit revela una estrategia limitada que no capitaliza el potencial del marketing digital en el sector fitness. Actualmente, el gimnasio mantiene:

Perfiles en redes sociales. Cuentas activas en Instagram (@fortfit.gym) y Facebook (FortFit.gym) con publicaciones regulares pero sin una estrategia de contenido estructurada.

Ausencia de sitio web. No existe una plataforma web oficial donde los clientes potenciales puedan obtener información detallada sobre servicios, precios, horarios o ubicación.

SEO local inexistente. La falta de presencia web implica que FortFit no aparece en búsquedas locales relevantes como "gimnasio Quito" o "gimnasio sector América", perdiendo oportunidades significativas de captación de clientes.

Esta situación es crítica: en mercados comparables se observa que la mayoría de las inscripciones suceden online. Por ejemplo, en México se reportó que más de 8 de cada 10 inscripciones a gimnasios se realizan por internet (*Más de 8 de Cada 10 Inscripciones A Gimnasios Se Realizan Por Internet*, 2024). La ausencia de sitio web y estrategias SEO locales priva a FortFit de captar esos potenciales clientes. En contraste, las mejores prácticas del sector indican que un sitio web profesional y canales digitales

sólidos son fundamentales para mejorar la imagen de servicio y atraer usuarios. La falta de una plataforma web oficial y de contenido en línea reduce significativamente la visibilidad de la marca frente a consumidores que investigan en línea antes de elegir un gimnasio (*Más de 8 de Cada 10 Inscripciones A Gimnasios Se Realizan Por Internet*, 2024).

Análisis de la competencia local

Un análisis comparativo con los principales competidores en el mercado de Quito revela que FortFit se encuentra en desventaja tecnológica. Gimnasios como Phisique (Phisique, 2024) o SmartFit (SmartFit, s.f.) han implementado soluciones digitales que incluyen:

- Aplicaciones móviles propietarias para reserva de clases
- Sistemas de seguimiento de progreso integrados
- Plataformas web con información completa y actualizada
- Integración con wearables y dispositivos de fitness
- Sistemas de gamificación para mejorar la retención de clientes

En resumen, FortFit está detrás en la adopción de soluciones digitales: sus competidores ofrecen ya experiencias integrales (app móvil, seguimiento de datos, información web completa, integración con dispositivos conectados y dinámicas gamificadas) que mejoran la fidelidad del cliente y la eficiencia operativa.

Impacto de la situación actual

Las consecuencias de mantener este modelo operativo se manifiestan en diferentes áreas:

Impacto en la experiencia del cliente. La gestión manual genera fricciones que deterioran la satisfacción de los socios. La espera por respuestas, la imposibilidad de consultar información al instante y la falta de canales digitales adecuados chocan con las expectativas de inmediatez del usuario moderno. La tendencia indica que los consumidores valoran las facilidades digitales; por ejemplo, estudios señalan que cerca del 70% de quienes incorporaron clases en línea durante la pandemia planean continuar con estas opciones (Falardeau et al., 2021), lo que implica que los clientes actuales prefieren gimnasios con servicios conectados.

Impacto operativo. El personal dedica gran parte de su tiempo a tareas administrativas que podrían automatizarse, reduciendo el tiempo disponible para actividades de mayor valor como el entrenamiento personalizado y la atención directa al cliente.

Impacto financiero. La ineficiencia operativa y la pérdida de oportunidades de captación de clientes por falta de presencia digital representan un costo de oportunidad significativo.

Impacto competitivo. La brecha tecnológica continúa ampliándose mientras los competidores avanzan en su transformación digital, poniendo en riesgo la participación de mercado de FortFit a mediano y largo plazo.

Oportunidades identificadas

A pesar de los desafíos presentados, el análisis también revela oportunidades significativas:

Base de clientes establecida. Con más de 600 miembros activos, existe un mercado prometedor para la adopción de nuevas herramientas digitales.

Infraestructura física sólida. Las instalaciones modernas y el equipamiento de calidad proporcionan una base sólida para complementar con servicios digitales.

Personal calificado. El equipo de profesionales especializados puede aportar contenido valioso para una plataforma digital.

Mercado en crecimiento. El sector fitness en Ecuador muestra un crecimiento sostenido. Un estudio reciente indicó que el valor del mercado de gimnasios superó los USD 114 millones en 2022, con un crecimiento promedio anual de aproximadamente 5,2% entre 2018 y 2022 (Restrepo et al., 2024). Este auge está impulsado por mayor conciencia de salud, aumento de la clase media y adopción de estilos de vida activos. Globalmente, la industria fitness también crece: por ejemplo, en Estados Unidos las membresías aumentaron 3,7% en 2022 (Health & Fitness Association, 2023), lo que sugiere que la demanda de servicios de gimnasios se mantiene sólida.

Esta situación actual establece claramente la necesidad de una intervención tecnológica integral que no solo cierre las brechas identificadas, sino que posicione a FortFit como líder en innovación dentro del mercado local de gimnasios.

Identificación de usuarios y stakeholders

La identificación precisa de usuarios y stakeholders es un elemento fundamental para el éxito del desarrollo del sistema web interactivo de FortFit. Este análisis permite comprender las necesidades específicas de cada grupo involucrado y garantizar que la solución tecnológica responda efectivamente a sus expectativas y requerimientos operativos.

Usuarios principales del sistema

Clientes miembros del gimnasio. Representan el grupo de usuarios más numeroso, con más de 600 miembros activos distribuidos entre las dos sedes. Este segmento incluye personas de diferentes edades y niveles de experiencia en fitness que requieren acceso a funcionalidades como reserva de citas con especialistas, consulta de rutinas personalizadas, navegación del mapa interactivo de instalaciones y acceso a la biblioteca de ejercicios. Sus necesidades se centran en la facilidad de uso, accesibilidad desde dispositivos móviles y disponibilidad de información actualizada.

Visitantes y clientes potenciales. Usuarios no registrados que acceden a la sección pública del sitio web para obtener información sobre servicios, precios, horarios y ubicación antes de tomar la decisión de inscribirse. Este grupo requiere información clara, actualizada y persuasiva que facilite su proceso de decisión.

Personal especializado. Incluye nutricionistas y fisioterapeutas que necesitan gestionar sus horarios de disponibilidad, consultar citas programadas y mantener comunicación efectiva con sus clientes. Requieren herramientas que optimicen su tiempo y mejoren la calidad de atención que brindan.

Entrenadores. Personal técnico que utilizará el sistema para consultar y actualizar rutinas de entrenamiento de los clientes, acceder a información de progreso y coordinar actividades de capacitación. Necesitan interfaces intuitivas que se integren naturalmente con su flujo de trabajo diario.

Personal administrativo. Usuarios con acceso a funcionalidades de gestión operativa, incluyendo la supervisión de reservas, mantenimiento de información del sistema y generación de reportes básicos.

Administradores del sistema. Usuarios con privilegios completos que gestionan la configuración del sistema, administran cuentas de usuario, mantienen el contenido de la biblioteca de ejercicios y supervisan el funcionamiento general de la plataforma.

Stakeholders del proyecto

Administración y propietarios de FortFit. Stakeholders principales interesados en el retorno de inversión, mejora de la competitividad y optimización de procesos operativos. Su participación es crítica para la toma de decisiones estratégicas y la asignación de recursos.

Equipo de desarrollo. Responsables de la implementación técnica que garantizan la calidad y funcionalidad del sistema.

Proveedores tecnológicos. Empresas que proporcionan servicios de hosting (Vercel), base de datos (Supabase) y otras herramientas tecnológicas necesarias para el funcionamiento del sistema.

Competidores locales. Aunque no participan directamente en el proyecto, representan un factor externo relevante cuyas estrategias digitales influyen en los requerimientos y expectativas del mercado.

La comprensión de estos grupos permite diseñar una solución que equilibre las necesidades técnicas con las expectativas de usabilidad, asegurando una adopción exitosa del sistema por parte de todos los usuarios involucrados.

Metodología utilizada para la recopilación de requisitos

Enfoque metodológico

Para el levantamiento de requisitos del sistema web interactivo de FortFit, se implementó un enfoque mixto que combinó técnicas cualitativas de investigación, aprovechando el conocimiento directo de los procesos organizacionales y la colaboración estrecha con los stakeholders clave. La metodología se diseñó para maximizar la comprensión de las necesidades reales del gimnasio mientras se minimizaba la interrupción de las operaciones diarias.

Técnicas de recopilación utilizadas

Entrevistas estructuradas con stakeholders clave. Se realizaron entrevistas dirigidas con los principales involucrados en los procesos operativos del gimnasio. Esta técnica permitió obtener información detallada sobre las necesidades específicas, problemas actuales y expectativas del sistema propuesto.

Observación participante directa. Se implementó una estrategia de observación sistemática de los procesos operativos del gimnasio, aprovechando la experiencia de uno de los desarrolladores como usuario regular de las instalaciones. Esta perspectiva privilegiada proporcionó información valiosa sobre las fricciones en la experiencia del cliente y las ineficiencias operativas que no siempre son evidentes en entrevistas formales.

Análisis de sistemas existentes. Se realizó un análisis exhaustivo del sistema administrativo actual para comprender la estructura de datos, los procesos implementados y las posibilidades de integración. Este análisis fue fundamental para determinar la estrategia de migración de datos y la arquitectura de integración óptima.

Proceso de levantamiento ejecutado

Fase de acercamiento inicial. La primera reunión con la administración del gimnasio tuvo como objetivo presentar la propuesta del proyecto como parte del trabajo de titulación y explorar la receptividad de la organización hacia una solución tecnológica. Durante esta sesión se identificaron las problemáticas generales y se estableció el contexto organizacional necesario para enfocar el desarrollo del proyecto.

Fase de levantamiento detallado. Se ejecutaron dos reuniones adicionales con el administrador del gimnasio, enfocadas específicamente en el levantamiento detallado de requisitos. Estas sesiones abordaron:

- Procesos operativos actuales y sus limitaciones
- Estructura organizacional y roles del personal
- Volumen de usuarios y patrones de uso
- Expectativas funcionales del sistema propuesto
- Restricciones técnicas y operativas

Paralelamente, se realizó una entrevista específica con un especialista en nutrición del gimnasio para comprender en profundidad el proceso de gestión de citas. Esta entrevista reveló problemas críticos como la falta de control de disponibilidad, conflictos de agendamiento y la frecuente presentación de clientes sin cita previa, información que resultó fundamental para el diseño del módulo de gestión de citas.

Fase de análisis técnico. Durante esta misma semana se realizó el análisis del sistema administrativo existente, evaluando sus capacidades, limitaciones y opciones de integración. Este análisis concluyó que la estrategia óptima de integración sería

mediante exportación e importación de datos en formato Excel, balanceando eficiencia de desarrollo con necesidades operativas.

Proceso de validación de requisitos

Una vez completado el levantamiento inicial, se elaboró un documento consolidado de requisitos que fue presentado al administrador del gimnasio para validación. El proceso de validación confirmó que los requisitos identificados respondían adecuadamente a las necesidades expresadas y se alineaban con los objetivos estratégicos de la organización.

Un aspecto destacable del proceso fue la definición clara del alcance desde el inicio del proyecto, lo que permitió evitar la expansión no controlada de requisitos y mantener el enfoque en las funcionalidades identificadas como más críticas para resolver los problemas operativos actuales.

Limitaciones metodológicas

Es importante reconocer que, por solicitud expresa de la administración del gimnasio, no se realizaron encuestas o cuestionarios directos a los usuarios finales del sistema (clientes del gimnasio). Esta decisión, aunque limitó la recopilación de retroalimentación directa de usuarios, se compensó con la observación participante y el conocimiento experiencial de los procesos desde la perspectiva del cliente.

Tabla 1

Cronograma de actividades para el levantamiento de requisitos

Actividad	Fecha	Duración	Participantes
Reunión inicial de presentación	Semana 3, Feb 2025	2 horas	Equipo desarrollo + Administrador
Primera reunión de levantamiento	Semana 4, Feb 2025	3 horas	Equipo desarrollo + Administrador
Segunda reunión de levantamiento	Semana 4, Feb 2025	2 horas	Equipo desarrollo + Administrador
Entrevista con especialista	Semana 4, Feb 2025	1.5 horas	Equipo desarrollo + Especialista fisioterapia
Análisis sistema administrativo	Semana 4, Feb 2025	4 horas	Equipo desarrollo
Validación de requisitos	Semana 4, Feb 2025	1 hora	Equipo desarrollo + Administrador

Requisitos funcionales

Requisitos de gestión de usuarios y autenticación

- El sistema debe permitir la importación usuarios mediante un Excel exportado del sistema administrativo del gimnasio
- Debe implementar autenticación segura con diferentes niveles de acceso: cliente, entrenador, especialista y administrador
- Debe incluir funcionalidad de recuperación de contraseña y gestión de perfiles de usuario

Requisitos de sección pública informativa

- El sistema debe mostrar información completa sobre servicios ofrecidos, con descripciones detalladas

- Debe presentar planes de membresía actualizados con precios, características y beneficios de cada opción
- Debe incluir horarios de operación para ambas sedes (América y El Inca) organizados por categorías
- Debe proporcionar información de contacto y ubicación

Sistema de gestión de citas

- Los clientes deben poder reservar citas con especialistas en nutrición y fisioterapia según disponibilidad
- Los especialistas deben poder gestionar sus horarios de disponibilidad y consultar citas programadas
- El sistema debe validar conflictos de horarios y enviar notificaciones de confirmación automáticas
- Debe permitir la cancelación de citas con notificación a las partes involucradas

Requisitos de gestión de rutinas personalizadas

- Los clientes deben poder consultar sus rutinas creadas
- El sistema debe organizar ejercicios por grupos musculares, tipos de entrenamiento y niveles de dificultad

Requisitos de mapa interactivo de instalaciones

- El sistema debe presentar una representación visual navegable de los tres niveles del gimnasio
- Debe permitir la exploración interactiva con información de equipos y áreas específicas

- Debe vincular la ubicación de máquinas con ejercicios correspondientes en la biblioteca de ejercicios

Requisitos de biblioteca de ejercicios

- El sistema debe categorizar ejercicios por tipo, grupo muscular, equipo requerido y nivel de dificultad
- Debe incluir descripciones textuales e imágenes o animaciones explicativas
- Debe permitir búsqueda y filtrado de contenido según preferencias del usuario

Requisitos no funcionales

Rendimiento y escalabilidad

- El sistema debe cargar las páginas principales en menos de 3 segundos bajo condiciones normales de red
- Debe soportar simultáneamente al menos 100 usuarios activos sin degradación significativa del rendimiento
- Debe optimizar automáticamente imágenes y contenido multimedia para diferentes dispositivos y velocidades de conexión

Usabilidad y accesibilidad

- La interfaz debe ser completamente responsiva, adaptándose a dispositivos móviles, tablets y ordenadores
- Debe seguir principios de diseño intuitivo con navegación clara y consistente en todas las secciones
- El tiempo de aprendizaje para nuevos usuarios no debe exceder 15 minutos para funcionalidades básicas

Seguridad y privacidad

- Debe implementar encriptación HTTPS para todas las comunicaciones entre cliente y servidor
- La información personal de usuarios debe estar protegida mediante políticas de seguridad a nivel de base de datos
- Debe incluir validación robusta de entrada de datos para prevenir inyecciones y ataques maliciosos
- Debe mantener logs de auditoría para acciones críticas del sistema

Disponibilidad y confiabilidad

- El sistema debe mantener una disponibilidad mínima del 98% mensual
- Debe incluir manejo graceful de errores con mensajes informativos para el usuario
- Debe funcionar correctamente en los navegadores más utilizados (Chrome, Firefox, Safari, Edge)

Mantenibilidad y compatibilidad

- El código debe seguir estándares de desarrollo que faciliten el mantenimiento y futuras actualizaciones
- Debe ser compatible con actualizaciones de las tecnologías base (Next.js, Supabase) sin requerir reescritura completa
- Debe incluir documentación técnica completa para facilitar el soporte y desarrollo futuro

Diagramas de casos de uso del sistema

Figura 1

Diagrama de casos de uso general

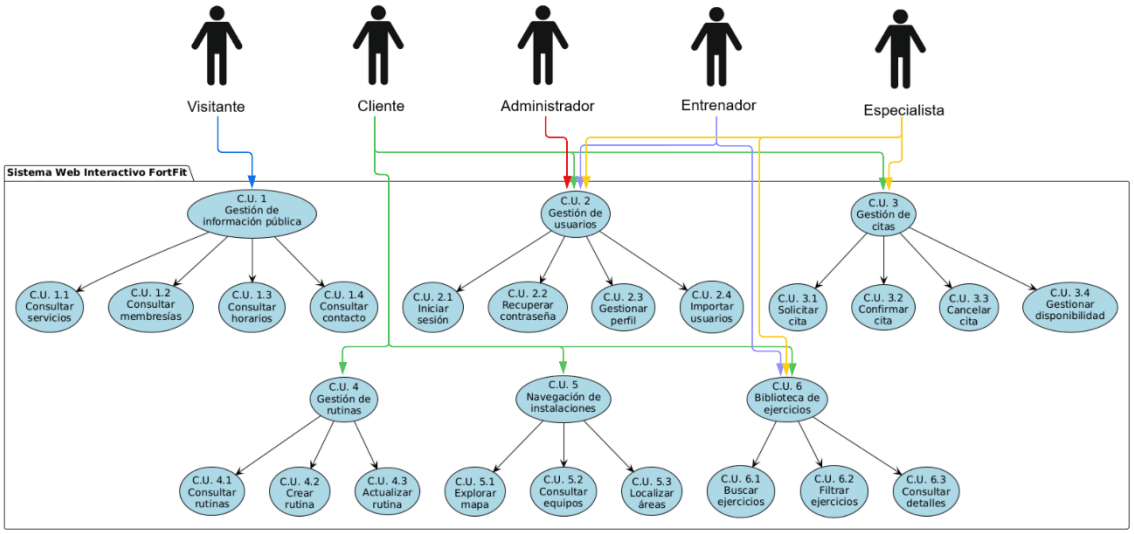


Tabla 2

Diagrama de caso de uso extendido C.U. 1.1

Nombre:	**C.U. 1.1 Consultar servicios**
Descripción:	Permite a los visitantes consultar información detallada sobre los servicios que ofrece el gimnasio FortFit, incluyendo descripciones, beneficios y características de cada servicio.
Actores:	Visitante
Precondiciones:	El visitante debe tener acceso a internet y un navegador web compatible.
Requisitos no funcionales:	La página debe cargar en menos de 3 segundos. La información debe ser accesible desde dispositivos móviles y de escritorio.
Flujo de eventos:	1. El visitante accede al sitio web de FortFit. 2. El sistema muestra la página principal con el menú de navegación. 3. El visitante selecciona la opción "Servicios" en el menú.

Nombre:	**C.U. 1.1 Consultar servicios**
	4. El sistema muestra la página de servicios con todas las opciones disponibles. 5. El visitante puede explorar cada servicio haciendo clic en ellos. 6. El sistema muestra información detallada del servicio seleccionado.
Flujo alterno:	
	4.1 Si no hay servicios activos, el sistema muestra un mensaje informativo.

Tabla 3

Diagrama de caso de uso extendido C.U. 1.2

Nombre:	**C.U. 1.2 Consultar membresías**
Descripción:	Permite a los visitantes consultar los diferentes planes de membresía disponibles, incluyendo precios, duración, características y beneficios de cada plan.
Actores:	Visitante
Precondiciones:	El visitante debe tener acceso a internet y un navegador web compatible.
Requisitos no funcionales:	La información de precios debe estar actualizada. La presentación debe ser clara y comparativa entre planes. La página debe cargar en menos de 3 segundos. La información debe ser accesible desde dispositivos móviles y de escritorio.
Flujo de eventos:	
	1. El visitante accede al sitio web de FortFit. 2. El sistema muestra la página principal con el menú de navegación. 3. El visitante selecciona la opción "Membresías" en el menú. 4. El sistema muestra todos los planes de membresía disponibles. 5. El visitante puede comparar características y precios de cada plan. 6. El sistema muestra información detallada del plan seleccionado.

Nombre:	**C.U. 1.2 Consultar membresías**
Flujo alterno:	
	4.1 Si no hay planes activos, el sistema muestra un mensaje informativo.

Tabla 4

Diagrama de caso de uso extendido C.U. 1.3

Nombre:	**C.U. 1.3 Consultar horarios**
Descripción:	Permite a los visitantes consultar los horarios de operación del gimnasio, horarios de clases grupales y disponibilidad de especialistas para ambas sedes.
Actores:	Visitante
Precondiciones:	El visitante debe tener acceso a internet y un navegador web compatible.
Requisitos no funcionales:	Los horarios deben estar organizados por categorías y sedes. La información debe ser fácil de leer y estar actualizada. La página debe cargar en menos de 3 segundos. La información debe ser accesible desde dispositivos móviles y de escritorio.
Flujo de eventos:	
	1. El visitante accede al sitio web de FortFit.
	2. El sistema muestra la página principal con el menú de navegación.
	3. El visitante selecciona la opción "Horarios" en el menú.
	4. El sistema muestra los horarios organizados por categorías (gimnasio, clases, especialistas).
	5. El visitante puede ver horarios para ambas sedes (América o El Inca).
Flujo alterno:	
	4.1 Si no hay horarios activos, el sistema muestra un mensaje informativo.

Tabla 5

Diagrama de caso de uso extendido C.U. 1.4

Nombre:	**C.U. 1.4 Consultar contacto**
Descripción:	Permite a los visitantes acceder a la información de contacto del gimnasio, incluyendo direcciones, teléfonos, correos electrónicos y ubicación en mapas.
Actores:	Visitante
Precondiciones:	El visitante debe tener acceso a internet y un navegador web compatible.
Requisitos no funcionales:	La información de contacto debe estar actualizada y ser precisa. Los mapas deben cargar correctamente. La página debe cargar en menos de 3 segundos. La información debe ser accesible desde dispositivos móviles y de escritorio.
Flujo de eventos:	
	1. El visitante accede al sitio web de FortFit.
	2. El sistema muestra la página principal con el menú de navegación.
	3. El visitante selecciona la opción "Contacto" en el menú.
	4. El sistema muestra la información de contacto de ambas sedes.
	5. El visitante puede ver direcciones, teléfonos y ubicación en mapa.
	6. El sistema permite al visitante interactuar con los mapas integrados.
Flujo alterno:	
	6.1 Si hay problemas con el servicio de mapas, el sistema muestra solo la dirección textual.

Tabla 6

Diagrama de caso de uso extendido C.U. 2.1

Nombre:	**C.U. 2.1 Iniciar sesión**
Descripción:	Permite a los usuarios registrados autenticarse en el sistema para acceder a las funcionalidades privadas según su rol.

Nombre:	**C.U. 2.1 Iniciar sesión**
Actores:	Cliente, Especialista, Entrenador, Administrador
Precondiciones:	El usuario debe tener una cuenta registrada en el sistema.
Requisitos no funcionales:	El proceso de autenticación debe completarse en menos de 5 segundos. Las credenciales deben transmitirse de forma segura mediante HTTPS.
Flujo de eventos:	
	1. El usuario accede a la página de inicio de sesión. 2. El sistema muestra el formulario de autenticación. 3. El usuario ingresa su correo electrónico y contraseña. 4. El usuario hace clic en "Iniciar sesión". 5. El sistema valida las credenciales. 6. El sistema autentica al usuario y lo redirige al dashboard correspondiente a su rol.
Flujo alterno:	
	5.1 Si las credenciales son incorrectas, el sistema muestra un mensaje de error y regresa al paso 2. 5.2.1 Si el usuario inicia sesión por primera vez, el sistema lo redirige a un formulario de cambio de contraseña. 5.2.2 El usuario ingresa y confirma su nueva contraseña y hace clic en “Cambiar contraseña”. 5.2.2.1 Si la nueva contraseña no cumple con las validaciones, se muestra un mensaje de error con las validaciones a cumplir y se regresa al paso 5.2.1 5.2.2.2 Si la nueva contraseña cumple con las validaciones, se pasa al paso 6. 6.1 Si la cuenta está inactiva, el sistema muestra un mensaje informativo en el dashboard, cierra la sesión del usuario y se regresa al paso 2.

Tabla 7

Diagrama de caso de uso extendido C.U. 2.2

Nombre:	**C.U. 2.2 Recuperar contraseña**
Descripción:	Permite al administrador reestablecer la contraseña del usuario a la contraseña por defecto (número de identificación) mediante el panel de administración.
Actores:	Administrador
Precondiciones:	El usuario debe tener una cuenta registrada y notificar a administración sobre la necesidad de recuperación de contraseña.
Requisitos no funcionales:	El proceso debe completarse en menos de 30 segundos. El sistema debe registrar todas las acciones de reinicio de contraseña por motivos de auditoría. La contraseña debe cambiarse de forma segura.
Flujo de eventos:	
	1. El administrador accede al panel de administración.
	2. El administrador selecciona "Gestión de usuarios".
	3. El sistema muestra la lista de usuarios registrados.
	4. El administrador busca al usuario que requiere reinicio de contraseña.
	5. El administrador selecciona al usuario específico.
	6. El administrador hace clic el botón de detalles y en "Reiniciar contraseña".
	7. El sistema muestra una confirmación de la acción.
	8. El administrador confirma el reinicio.
	9. El sistema establece la contraseña como el número de identificación del usuario.
	10. El sistema muestra mensaje de confirmación del reinicio exitoso.
	11. El administrador informa al usuario sobre el reinicio y la nueva contraseña temporal.
Flujo alterno:	
	4.1 Si el usuario no se encuentra, el administrador puede usar filtros de búsqueda para localizarlo.

Nombre:	**C.U. 2.2 Recuperar contraseña**
	8.1 Si el administrador cancela la confirmación, el sistema regresa al paso 5 sin realizar cambios. 9.1 Si hay errores en el proceso, el sistema muestra un mensaje de error y permite reintentar.

Tabla 8

Diagrama de caso de uso extendido C.U. 2.3

Nombre:	**C.U. 2.3 Gestionar perfil**
Descripción:	Permite a los usuarios autenticados consultar su información personal en el sistema.
Actores:	Cliente, Especialista, Entrenador, Administrador
Precondiciones:	El usuario debe estar autenticado en el sistema.
Requisitos no funcionales:	La interfaz debe ser intuitiva y responsiva.
Flujo de eventos:	1. El usuario autenticado accede a su dashboard. 2. El usuario selecciona la opción "Mi Perfil". 3. El sistema muestra la información del perfil. 4. El sistema muestra un mensaje de contacto a administración en caso de que la información no esté correcta. 5. El usuario puede hacer clic en el botón de actualizar información para traer la información más actual.
Flujo alterno:	4.1 Si la información del sistema no está correcta, está desactualizada, o incompleta, el usuario se comunica con administración. 5.1 Si hay errores en la actualización, el sistema muestra un mensaje informativo,

Tabla 9

Diagrama de caso de uso extendido C.U. 2.4

Nombre:	**C.U. 2.4 Importar usuarios**
Descripción:	Permite al administrador importar usuarios desde un archivo Excel exportado del sistema administrativo del gimnasio.
Actores:	Administrador
Precondiciones:	El administrador debe estar autenticado. Debe tener un archivo Excel con el formato correcto.
Requisitos no funcionales:	El proceso debe completarse en menos de 30 minutos para archivos de hasta 2000 usuarios. Debe validar la integridad de los datos. Debe manejar excepciones en los datos correctamente.
Flujo de eventos:	
	1. El administrador accede al panel de administración.
	2. El administrador selecciona "Importar usuarios".
	3. El sistema muestra la interfaz de importación.
	4. El administrador selecciona el archivo Excel.
	5. El administrador hace clic en "Subir archivo".
	6. El sistema valida el formato y contenido del archivo.
	7. El sistema procesa e importa los usuarios válidos.
	8. El sistema muestra un reporte de la importación.
Flujo alterno:	
	6.1 Si el archivo tiene formato incorrecto, el sistema muestra un mensaje de error y regresa al paso 3.
	7.1 Si hay usuarios duplicados, el sistema los actualiza y continúa con el proceso.

Tabla 10

Diagrama de caso de uso extendido C.U. 3.1

Nombre:	**C.U. 3.1 Solicitar cita**
Descripción:	Permite a los clientes solicitar una cita con especialistas en nutrición o fisioterapia según la disponibilidad.

Nombre:	**C.U. 3.1 Solicitar cita**
Actores:	Cliente
Precondiciones:	El cliente debe estar autenticado y tener una membresía activa.
Requisitos no funcionales:	El sistema debe validar disponibilidad en tiempo real. La interfaz debe ser intuitiva para dispositivos móviles y de escritorio.
Flujo de eventos:	
	1. El cliente accede a su dashboard. 2. El cliente selecciona "Citas" y posteriormente “Nueva”. 3. El sistema muestra las opciones de especialidad disponibles. 4. El cliente selecciona el tipo de especialista (nutrición/fisioterapia). 5. El sistema muestra fechas y horarios disponibles. 6. El cliente selecciona fecha y hora deseada. 7. El cliente ingresa el motivo de la consulta. 8. El cliente confirma la solicitud de cita. 9. El sistema registra la cita como "pendiente" y notifica al especialista.
Flujo alterno:	
	5.1 Si no hay disponibilidad, el sistema muestra un mensaje informativo. 9.1 Si hay conflicto de horario, el sistema muestra error y regresa al paso 5.

Tabla 11

Diagrama de caso de uso extendido C.U. 3.2

Nombre:	**C.U. 3.2 Confirmar cita**
Descripción:	Permite a los especialistas confirmar o rechazar las solicitudes de citas pendientes.
Actores:	Especialista
Precondiciones:	El especialista debe estar autenticado. Debe existir al menos una cita pendiente.

Nombre:	**C.U. 3.2 Confirmar cita**
Requisitos no funcionales:	Las notificaciones deben enviarse inmediatamente al cliente. La interfaz debe mostrar información clara de cada solicitud.
Flujo de eventos:	
	1. El especialista accede a su dashboard.
	2. El sistema muestra las citas pendientes de confirmación.
	3. El especialista selecciona una cita pendiente.
	4. El sistema muestra los detalles de la cita.
	5. El especialista puede agregar notas (opcional).
	6. El especialista selecciona "Confirmar" o "Rechazar".
	7. El sistema actualiza el estado de la cita.
	8. El sistema notifica al cliente la decisión del especialista.
Flujo alterno:	
	6.1 Si rechaza la cita, el especialista debe proporcionar un motivo obligatorio.

Tabla 12

Diagrama de caso de uso extendido C.U. 3.3

Nombre:	**C.U. 3.3 Cancelar cita**
Descripción:	Permite a clientes y especialistas cancelar citas confirmadas con notificación a la otra parte.
Actores:	Cliente, Especialista
Precondiciones:	El usuario debe estar autenticado. Debe existir una cita confirmada. La cita debe ser cancelable según las políticas (2 horas antes como mínimo).
Requisitos no funcionales:	El sistema debe validar las políticas de cancelación. Las notificaciones deben enviarse inmediatamente.
Flujo de eventos:	
	1. El usuario accede a su dashboard.
	2. El usuario selecciona "Citas".
	3. El sistema muestra las citas confirmadas.

Nombre:	**C.U. 3.3 Cancelar cita**
	4. El usuario selecciona la cita a cancelar. 5. El usuario hace clic en "Cancelar cita". 6. El sistema valida si la cita es cancelable. 7. El usuario ingresa el motivo de cancelación. 8. El usuario confirma la cancelación. 9. El sistema actualiza el estado y notifica a la otra parte.
Flujo alterno:	
	6.1 Si la cita no es cancelable, el sistema muestra un mensaje explicativo.

Tabla 13

Diagrama de caso de uso extendido C.U. 3.4

Nombre:	**C.U. 3.4 Gestionar disponibilidad**
Descripción:	Permite a los especialistas configurar sus horarios de disponibilidad para que los clientes puedan agendar citas.
Actores:	Especialista
Precondiciones:	El especialista debe estar autenticado en el sistema.
Requisitos no funcionales:	Los cambios de disponibilidad deben reflejarse inmediatamente en el sistema de reservas. La interfaz debe ser intuitiva para gestión de horarios.
Flujo de eventos:	
	1. El especialista accede a su dashboard. 2. El especialista selecciona "Gestionar disponibilidad". 3. El sistema muestra la configuración actual de horarios. 4. El especialista puede configurar horarios recurrentes por día de la semana. 5. El especialista puede agregar excepciones para fechas específicas. 6. El especialista guarda los cambios. 7. El sistema valida que no haya conflictos con citas existentes.

Nombre:	**C.U. 3.4 Gestionar disponibilidad**
	8. El sistema actualiza la disponibilidad y confirma los cambios.
Flujo alterno:	
	7.1 Si hay conflictos con citas existentes, el sistema muestra advertencia y permite resolverlos.

Tabla 14

Diagrama de caso de uso extendido C.U. 4.1

Nombre:	**C.U. 4.1 Consultar rutinas**
Descripción:	Permite a los clientes acceder y visualizar las rutinas de entrenamiento que han creado previamente.
Actores:	Cliente
Precondiciones:	El cliente debe estar autenticado y tener al menos una rutina creada.
Requisitos no funcionales:	La lista de rutinas debe cargarse en menos de 3 segundos. La interfaz debe ser clara y fácil de leer en dispositivos móviles.
Flujo de eventos:	
	1. El cliente accede a su dashboard.
	2. El cliente selecciona "Rutinas".
	3. El sistema muestra la lista de rutinas creadas por el cliente.
	4. El cliente selecciona una rutina específica.
	5. El sistema muestra los detalles completos de la rutina.
	6. El cliente puede navegar entre diferentes ejercicios de la rutina.
Flujo alterno:	
	3.1 Si no tiene rutinas creadas, el sistema muestra una opción para crear la primera rutina.

Tabla 15

Diagrama de caso de uso extendido C.U. 4.2

Nombre:	**C.U. 4.2 Crear rutina**
Descripción:	Permite a los clientes crear sus propias rutinas de entrenamiento seleccionando ejercicios de la biblioteca disponible.
Actores:	Cliente
Precondiciones:	El cliente debe estar autenticado. Debe existir al menos un ejercicio en la biblioteca.
Requisitos no funcionales:	El proceso debe ser intuitivo y permitir guardar en cualquier punto. La interfaz debe facilitar la búsqueda de ejercicios.
Flujo de eventos:	1. El cliente accede a su dashboard. 2. El cliente selecciona "Rutinas". 3. El cliente selecciona "Crear Rutina". 4. El sistema muestra el formulario de creación de rutina. 5. El cliente ingresa nombre y descripción de la rutina. 6. El cliente busca y selecciona ejercicios de la biblioteca. 7. El cliente configura series, repeticiones y peso para cada ejercicio. 8. El cliente guarda la rutina. 9. El sistema valida la información y guarda la rutina.
Flujo alterno:	9.1 Si falta información obligatoria, el sistema muestra errores específicos y regresa al paso correspondiente.

Tabla 16

Diagrama de caso de uso extendido C.U. 4.3

Nombre:	**C.U. 4.3 Actualizar rutina**
Descripción:	Permite a los clientes modificar rutinas existentes, agregando, eliminando o modificando ejercicios.

Nombre:	**C.U. 4.3 Actualizar rutina**
Actores:	Cliente
Precondiciones:	El cliente debe estar autenticado. Debe tener al menos una rutina creada.
Requisitos no funcionales:	Los cambios deben guardarse de forma segura. Debe permitir previsualizar cambios antes de guardar.
Flujo de eventos:	
	1. El cliente accede a su dashboard. 2. El cliente selecciona "Rutinas". 3. El cliente accede a "Mis rutinas". 4. El cliente selecciona la rutina a modificar. 5. El cliente hace clic en "Editar rutina". 6. El sistema muestra la rutina en modo de edición. 7. El cliente realiza las modificaciones deseadas. 8. El cliente guarda los cambios. 9. El sistema valida las modificaciones. 10. El sistema actualiza la rutina y muestra confirmación.
Flujo alterno:	
	9.1 Si hay errores de validación, el sistema los muestra y regresa al paso 6.

Tabla 17

Diagrama de caso de uso extendido C.U. 5.1

Nombre:	**C.U. 5.1 Explorar mapa**
Descripción:	Permite a los clientes navegar interactivamente por el mapa de las instalaciones del gimnasio distribuidas en tres niveles.
Actores:	Cliente
Precondiciones:	El cliente debe estar autenticado en el sistema.
Requisitos no funcionales:	El mapa debe cargar en menos de 5 segundos y ser responsivo. Debe funcionar correctamente en dispositivos móviles y de escritorio.

Nombre:	**C.U. 5.1 Explorar mapa**
Flujo de eventos:	
	1. El cliente accede a su dashboard. 2. El cliente selecciona "Mapa del gimnasio". 3. El sistema muestra el mapa interactivo del gimnasio. 4. El cliente puede navegar entre los tres niveles del gimnasio. 5. El cliente puede hacer zoom y desplazarse por el mapa. 6. El cliente puede hacer clic en diferentes áreas para obtener información.
Flujo alterno:	
	3.1 Si hay problemas de carga, el sistema muestra un mensaje informativo y opción de recargar.

Tabla 18

Diagrama de caso de uso extendido C.U. 5.2

Nombre:	**C.U. 5.2 Consultar equipos**
Descripción:	Permite a los clientes obtener información detallada sobre los equipos disponibles en el gimnasio y su ubicación.
Actores:	Cliente
Precondiciones:	El cliente debe estar autenticado. Debe estar en el mapa interactivo del gimnasio.
Requisitos no funcionales:	La información debe estar actualizada y ser precisa. Debe incluir instrucciones básicas de uso.
Flujo de eventos:	
	1. El cliente está navegando en el mapa del gimnasio. 2. El cliente hace clic en un área específica en el mapa. 3. El sistema muestra la lista de equipos en esa área. 4. El cliente escoge el equipo que desea. 5. El cliente puede ver especificaciones y instrucciones básicas. 6. El sistema puede mostrar ejercicios relacionados disponibles en la biblioteca.

Nombre:	**C.U. 5.2 Consultar equipos**
Flujo alterno:	
	6.1 Si no hay ejercicios relacionados, el sistema muestra solo la información del equipo.

Tabla 19

Diagrama de caso de uso extendido C.U. 5.3

Nombre:	**C.U. 5.3 Localizar áreas**
Descripción:	Permite a los clientes ubicar áreas específicas del gimnasio como vestuarios, recepción, áreas de entrenamiento, etc.
Actores:	Cliente
Precondiciones:	El cliente debe estar autenticado. Debe estar en el mapa interactivo del gimnasio.
Requisitos no funcionales:	Las áreas deben estar claramente identificadas. El sistema debe proporcionar indicaciones claras de ubicación.
Flujo de eventos:	
	1. El cliente está navegando en el mapa del gimnasio.
	2. El cliente selecciona un área específica.
	3. El sistema resalta el área solicitada en el mapa.
	4. El sistema muestra más información del área y los equipos en esta.
Flujo alterno:	
	2.1 Si el área no se selecciona, el usuario puede seguir navegando sin problemas.

Tabla 20

Diagrama de caso de uso extendido C.U. 6.1

Nombre:	**C.U. 6.1 Buscar ejercicios**
Descripción:	Permite a clientes, especialistas y entrenadores buscar ejercicios específicos en la biblioteca usando términos de búsqueda.
Actores:	Cliente, Entrenador, Especialista

Nombre:	**C.U. 6.1 Buscar ejercicios**
Precondiciones:	El usuario debe estar autenticado. Debe existir contenido en la biblioteca de ejercicios.
Requisitos no funcionales:	Los resultados de búsqueda deben mostrarse en menos de 2 segundos. El sistema debe manejar búsquedas por nombre, grupo muscular, o equipo.
Flujo de eventos:	
	1. El usuario accede a su dashboard. 2. El usuario selecciona “Biblioteca de ejercicios”. 3. El usuario ingresa términos de búsqueda en el campo correspondiente. 4. El usuario hace clic en "Buscar" o presiona Enter. 5. El sistema procesa la búsqueda y muestra resultados relevantes. 6. El usuario puede seleccionar un ejercicio de los resultados. 7. El sistema muestra los detalles completos del ejercicio seleccionado.
Flujo alterno:	
	5.1 Si no hay resultados, el sistema muestra un mensaje informativo.

Tabla 21

Diagrama de caso de uso extendido C.U. 6.2

Nombre:	**C.U. 6.2 Filtrar ejercicios**
Descripción:	Permite a usuarios aplicar filtros específicos para encontrar ejercicios por grupo muscular, tipo de entrenamiento, equipo requerido o nivel de dificultad.
Actores:	Cliente, Entrenador, Especialista
Precondiciones:	El usuario debe estar autenticado. Debe estar en la biblioteca de ejercicios.
Requisitos no funcionales:	Los filtros deben aplicarse instantáneamente. La interfaz debe ser intuitiva y permitir múltiples filtros simultáneos.
Flujo de eventos:	
	1. El usuario está en la biblioteca de ejercicios.

Nombre:	**C.U. 6.2 Filtrar ejercicios**
	2. El usuario accede a las opciones de filtrado. 3. El usuario selecciona uno o más filtros (grupo muscular, equipo, nivel, etc.). 4. El sistema aplica los filtros automáticamente. 5. El sistema muestra los ejercicios que cumplen con los criterios. 6. El usuario puede modificar o limpiar los filtros según necesite.
Flujo alterno:	
	5.1 Si no hay ejercicios que cumplan los filtros, el sistema muestra un mensaje informativo.

Tabla 22

Diagrama de caso de uso extendido C.U. 6.3

Nombre:	**C.U. 6.3 Consultar detalles**
Descripción:	Permite a los usuarios acceder a información detallada de un ejercicio específico, incluyendo descripción, instrucciones y equipo a usar.
Actores:	Cliente, Entrenador, Especialista
Precondiciones:	El usuario debe estar autenticado. Debe haber seleccionado un ejercicio específico.
Requisitos no funcionales:	El contenido multimedia debe cargar eficientemente. La información debe estar bien estructurada y ser fácil de entender.
Flujo de eventos:	
	1. El usuario ha encontrado un ejercicio mediante búsqueda o filtros. 2. El usuario hace clic en el ejercicio deseado. 3. El sistema muestra la página de detalles del ejercicio. 4. El usuario puede ver descripción, instrucciones paso a paso. 5. El usuario puede visualizar imágenes explicativas si están disponibles. 6. El usuario puede agregar el ejercicio a una rutina (si es cliente).
Flujo alterno:	

Nombre:	**C.U. 6.3 Consultar detalles**
	5.1 Si no hay material multimedia disponible, el sistema muestra solo información textual. 6.1 Si el usuario es entrenador o especialista, puede agregar el ejercicio a su propia rutina mas no a la de alguien más.

Diseño de la arquitectura del sistema

Figura 2

Diagrama de arquitectura de tecnología

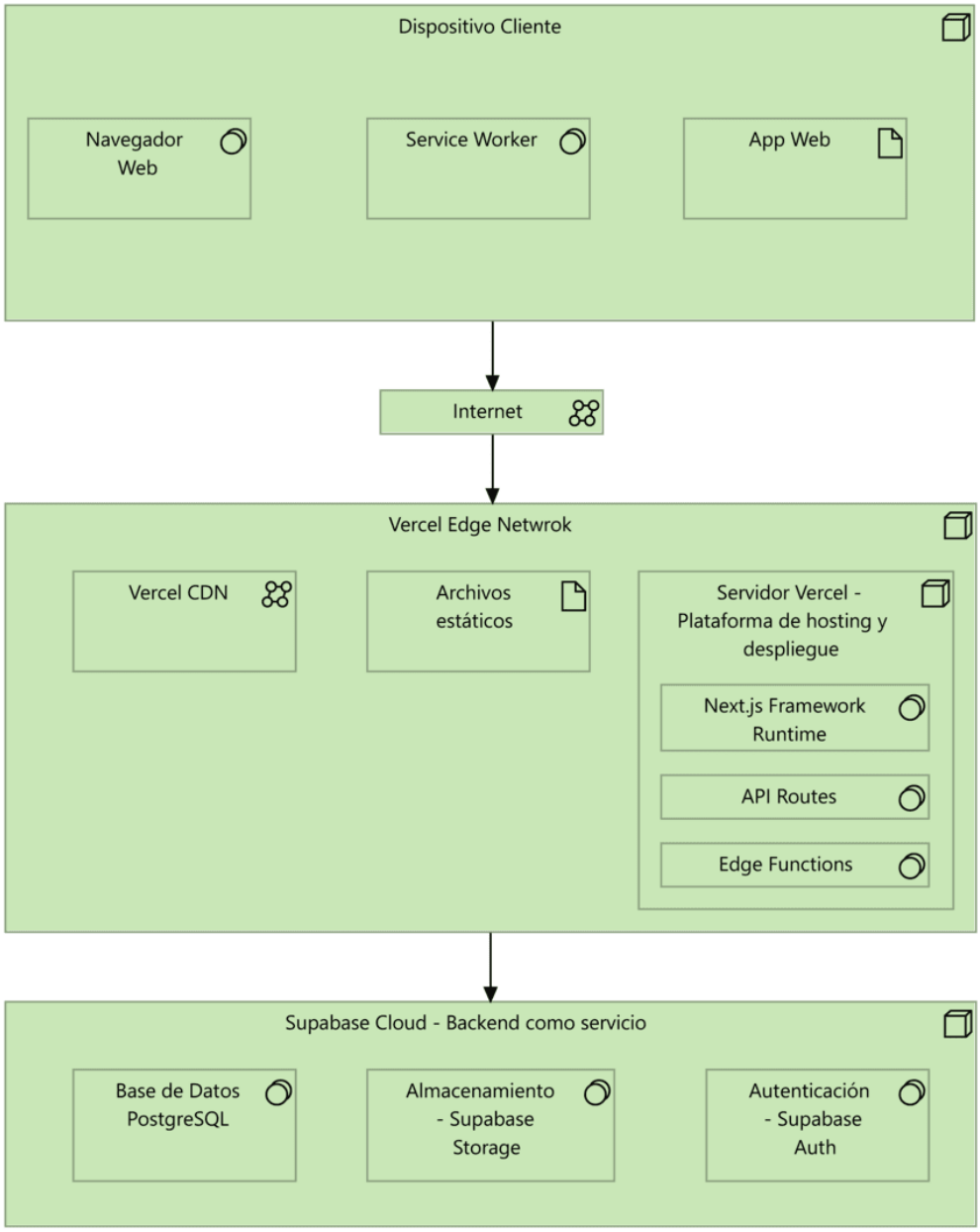
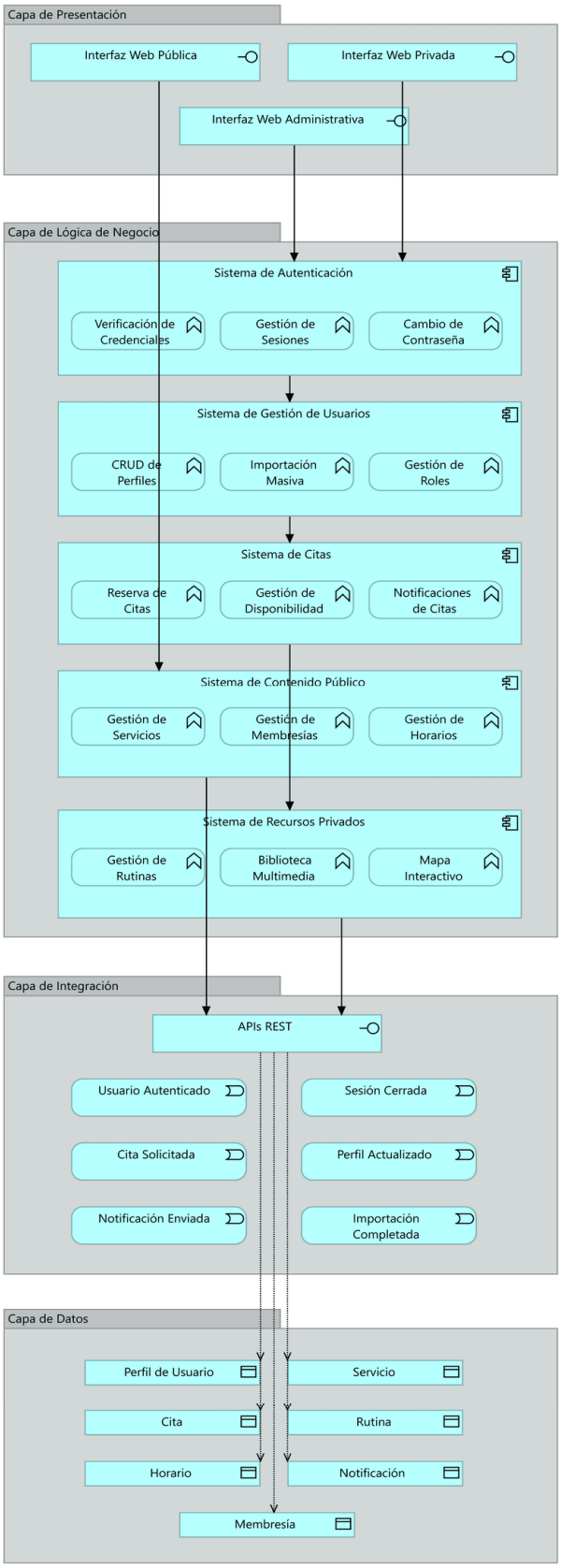


Figura 3

Diagrama de arquitectura de aplicación



Módulos del sistema

Módulo de Autenticación

Responsabilidades:

- Gestionar el proceso de inicio y cierre de sesión de usuarios
- Implementar recuperación de contraseñas
- Manejar cambio obligatorio de contraseña en primer ingreso
- Controlar el acceso a secciones según roles de usuario

Componentes principales:

- Sistema de login con validación de credenciales
- Funcionalidad de recuperación de contraseña (administrada por admin)
- Formulario de cambio de contraseña con validaciones
- Middleware de protección de rutas privadas
- Gestión de sesiones y tokens JWT

Tecnologías utilizadas: Supabase Auth, Next.js middleware

Módulo de Gestión de Citas

Responsabilidades:

- Permitir a clientes solicitar citas con especialistas
- Facilitar a especialistas la confirmación/rechazo de citas
- Gestionar cancelaciones con políticas de tiempo

- Validar disponibilidad y evitar conflictos de horarios

Componentes principales:

- Sistema de solicitud de citas por especialidad
- Panel de confirmación para especialistas
- Funcionalidad de cancelación con validaciones temporales
- Validador de disponibilidad en tiempo real
- Sistema de notificaciones de cambios de estado

Módulo de Gestión de Rutinas

Responsabilidades:

- Permitir a clientes crear rutinas personalizadas
- Facilitar consulta y actualización de rutinas existentes
- Integrar ejercicios de la biblioteca en las rutinas
- Gestionar configuración de series, repeticiones y pesos

Componentes principales:

- Constructor de rutinas con selección de ejercicios
- Visualizador de rutinas con detalles completos
- Editor de rutinas existentes
- Validador de estructura de rutinas
- Interfaz de búsqueda de ejercicios para rutinas

Integración: Se conecta estrechamente con el Módulo de Biblioteca de Ejercicios

Módulo de Biblioteca de Ejercicios

Responsabilidades:

- Proporcionar catálogo completo de ejercicios disponibles
- Implementar sistema de búsqueda y filtrado
- Mostrar detalles e instrucciones de cada ejercicio
- Categorizar ejercicios por grupos musculares y equipos

Componentes principales:

- Motor de búsqueda de ejercicios
- Sistema de filtros múltiples (grupo muscular, equipo, nivel)
- Visualizador de detalles con instrucciones
- Categorizador automático de contenido
- Interfaz de exploración por categorías

Contenido gestionado: Descripciones, imágenes, categorizaciones, niveles de dificultad

Módulo de Mapa Interactivo

Responsabilidades:

- Proporcionar navegación visual de las instalaciones
- Mostrar información de equipos por áreas
- Facilitar localización de espacios específicos
- Integrar información de equipos con ejercicios relacionados

Componentes principales:

- Visor interactivo de los tres niveles del gimnasio
- Sistema de información por zonas
- Integración con biblioteca de ejercicios
- Navegador de áreas comunes (vestuarios, recepción, etc.)

Módulo de Notificaciones Push

Responsabilidades:

- Gestionar suscripciones a notificaciones push
- Enviar recordatorios automáticos de citas
- Notificar cambios de estado en solicitudes
- Proporcionar configuración de preferencias de usuario

Componentes principales:

- Gestor de suscripciones push (registro/cancelación)
- Sistema de envío de notificaciones automáticas
- Configurador de preferencias de notificación
- Programador de recordatorios (24h y 2.5h antes de citas)
- Panel de diagnóstico para resolución de problemas

Módulo de Gestión para Especialistas

Responsabilidades:

- Proporcionar dashboard específico para especialistas
- Gestionar horarios de disponibilidad (recurrente y específica)
- Facilitar administración de citas asignadas
- Configurar preferencias profesionales

Componentes principales:

- Dashboard especializado con vista de calendario
- Configurador de disponibilidad recurrente
- Gestor de excepciones de horario (fechas específicas)
- Visor de citas programadas con detalles de clientes
- Panel de configuración de perfil profesional

Módulo de Sección Pública

Responsabilidades:

- Presentar información institucional del gimnasio
- Mostrar servicios y planes de membresía
- Proporcionar horarios e información de contacto
- Optimizar contenido para motores de búsqueda (SEO)

Componentes principales:

- Páginas informativas de servicios
- Catálogo de planes de membresía con precios
- Horarios organizados por categorías y sedes

- Información de contacto con mapas integrados
- Sistema de navegación optimizado

Módulo de Administración y Gestión de Usuarios

Responsabilidades:

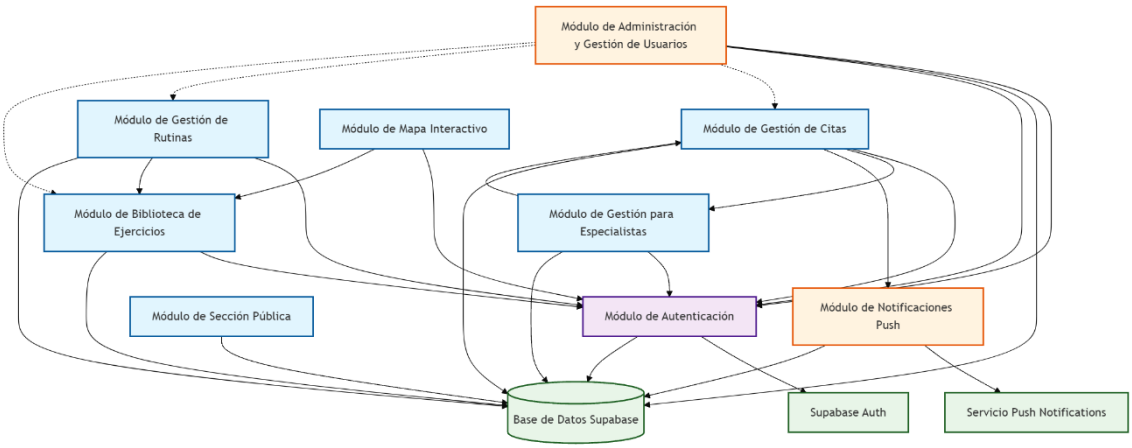
- Importar usuarios desde sistema administrativo externo
- Gestionar perfiles y roles de usuario

Componentes principales:

- Importador de usuarios vía archivos Excel
- Gestor de perfiles con actualización masiva
- Reinicio de contraseñas de usuarios

Figura 4

Diagrama de relación entre módulos del sistema



Diagramas de flujo de los procesos

Figura 5

Diagrama del Proceso de Autenticación y Primer Login

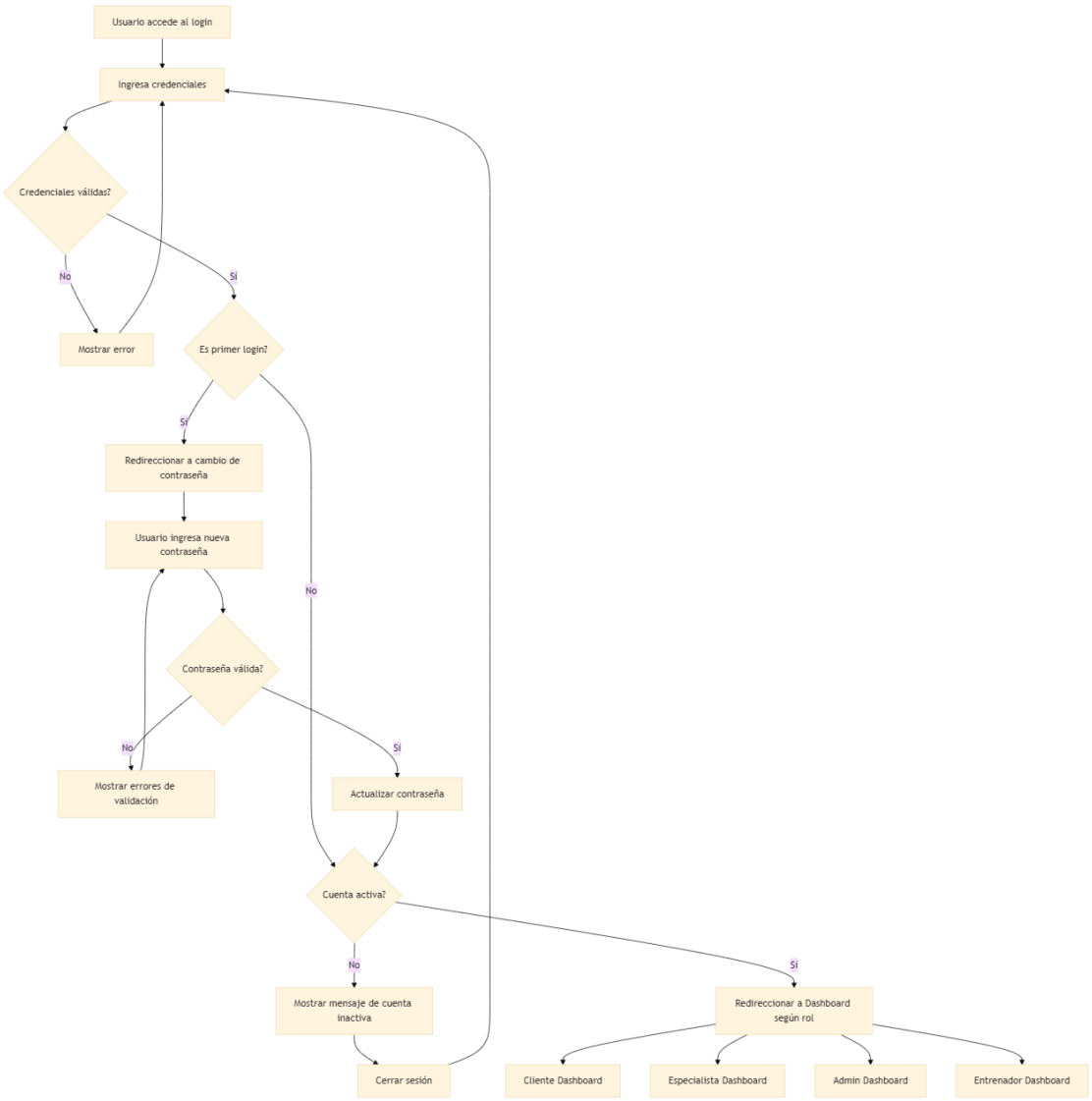


Figura 6

Diagrama del Proceso de Reserva de Citas

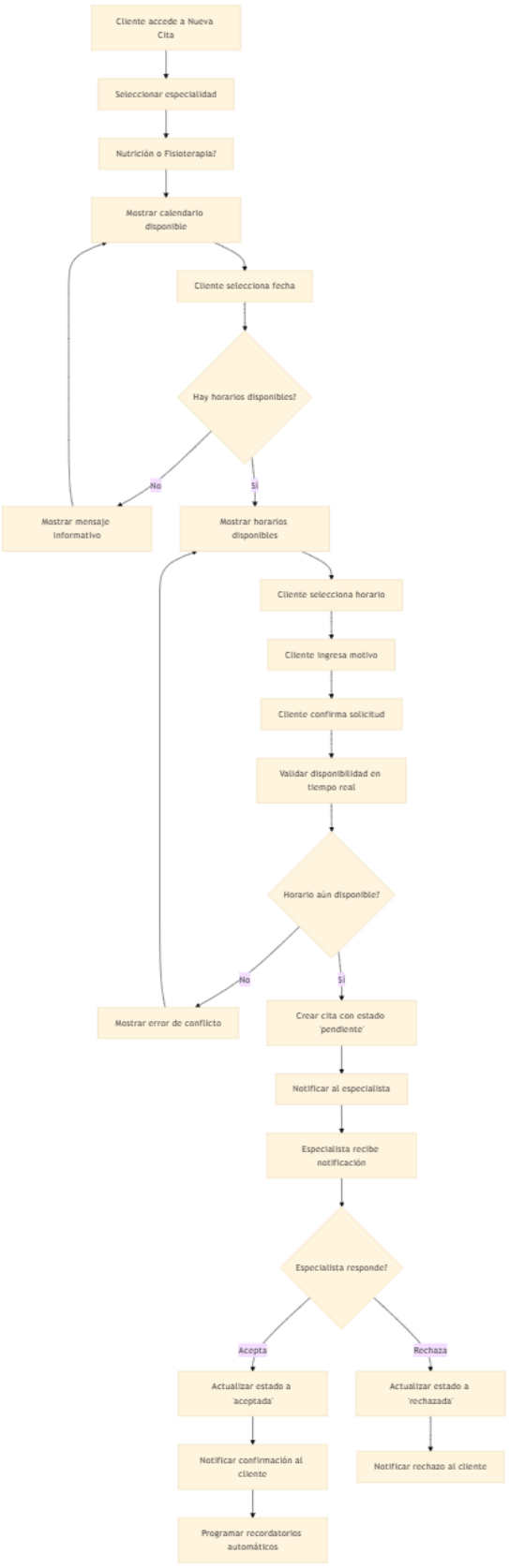


Figura 7

Diagrama del Proceso de Gestión de Rutinas

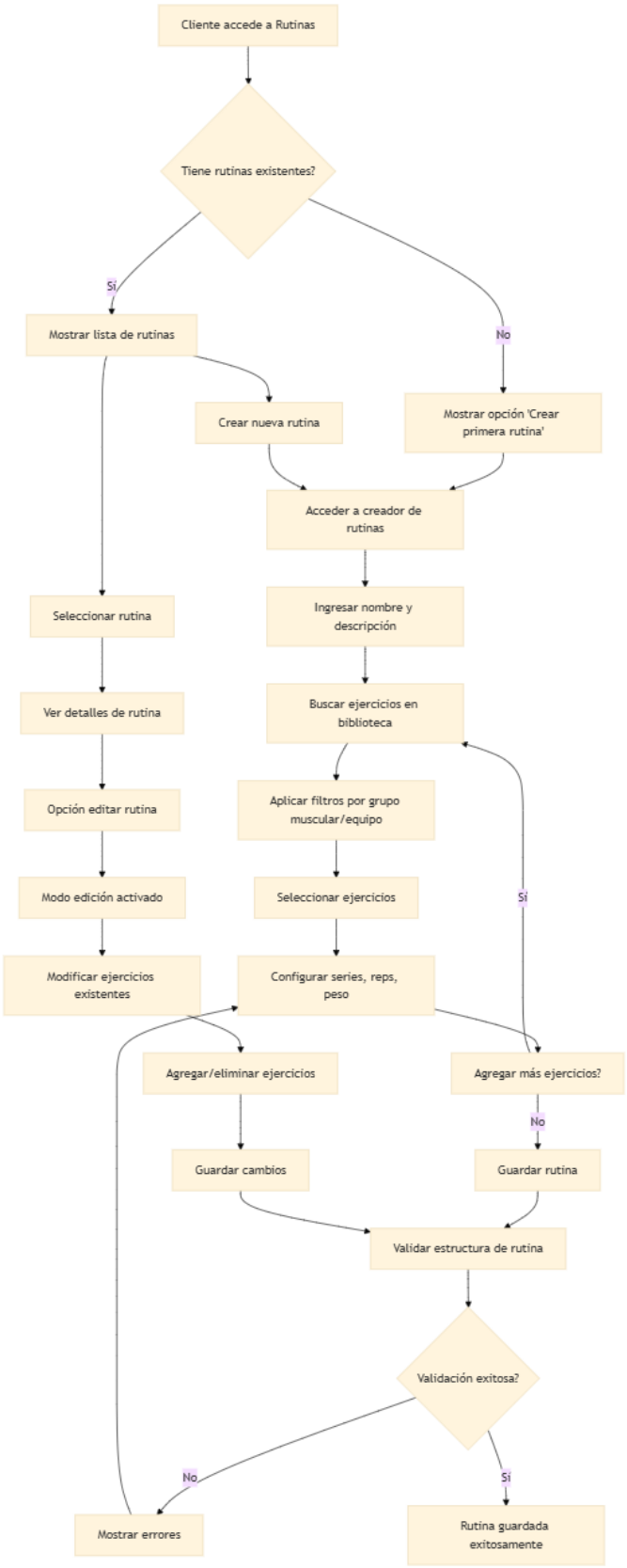


Figura 8

Diagrama del Proceso de Gestión de Disponibilidad (Especialistas)

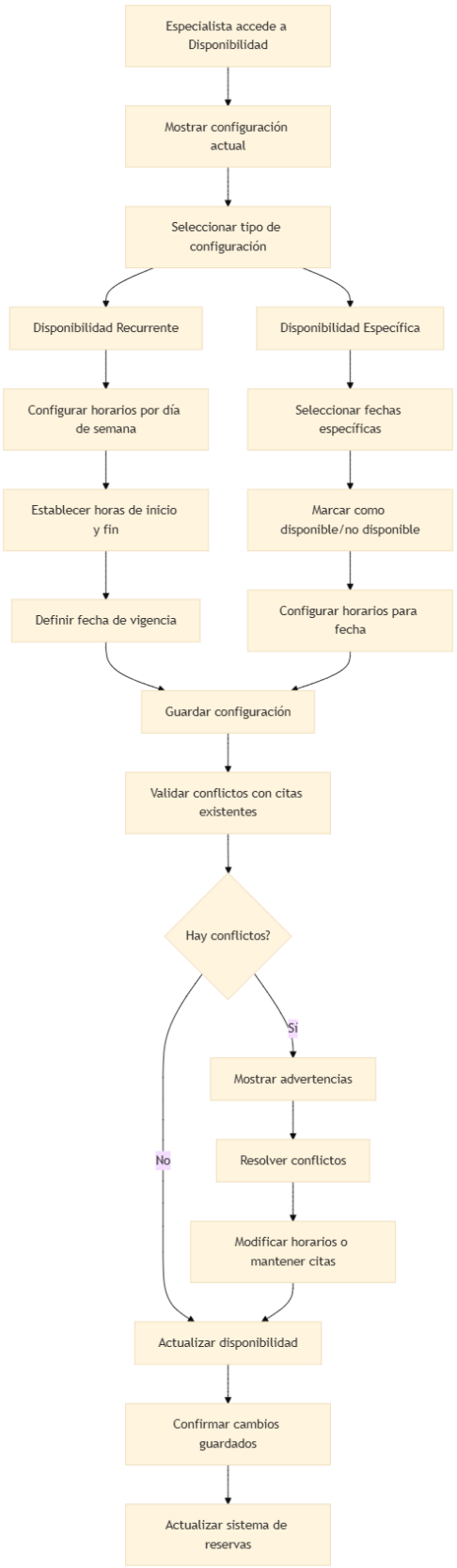
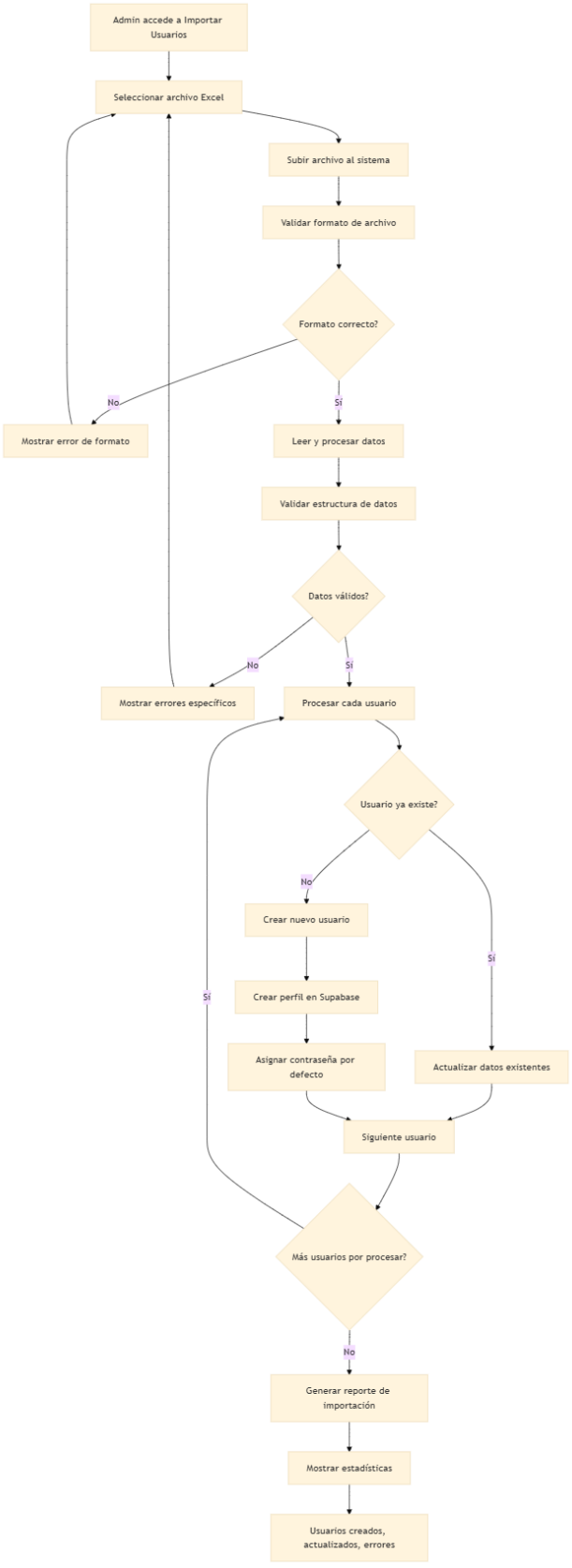


Figura 9

Diagrama del Proceso de Importación de Usuarios (Administrador)



Diagramas de clases

Figura 10

Diagrama de clases de la sección pública

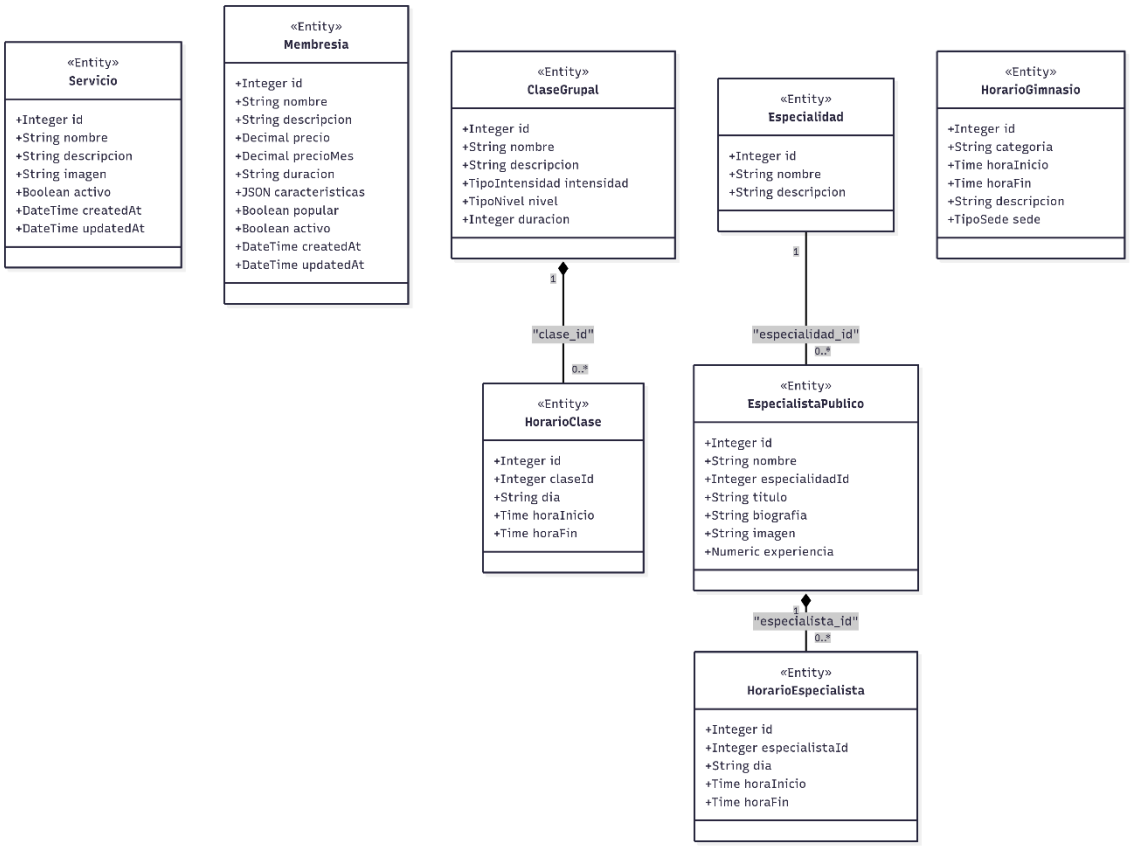
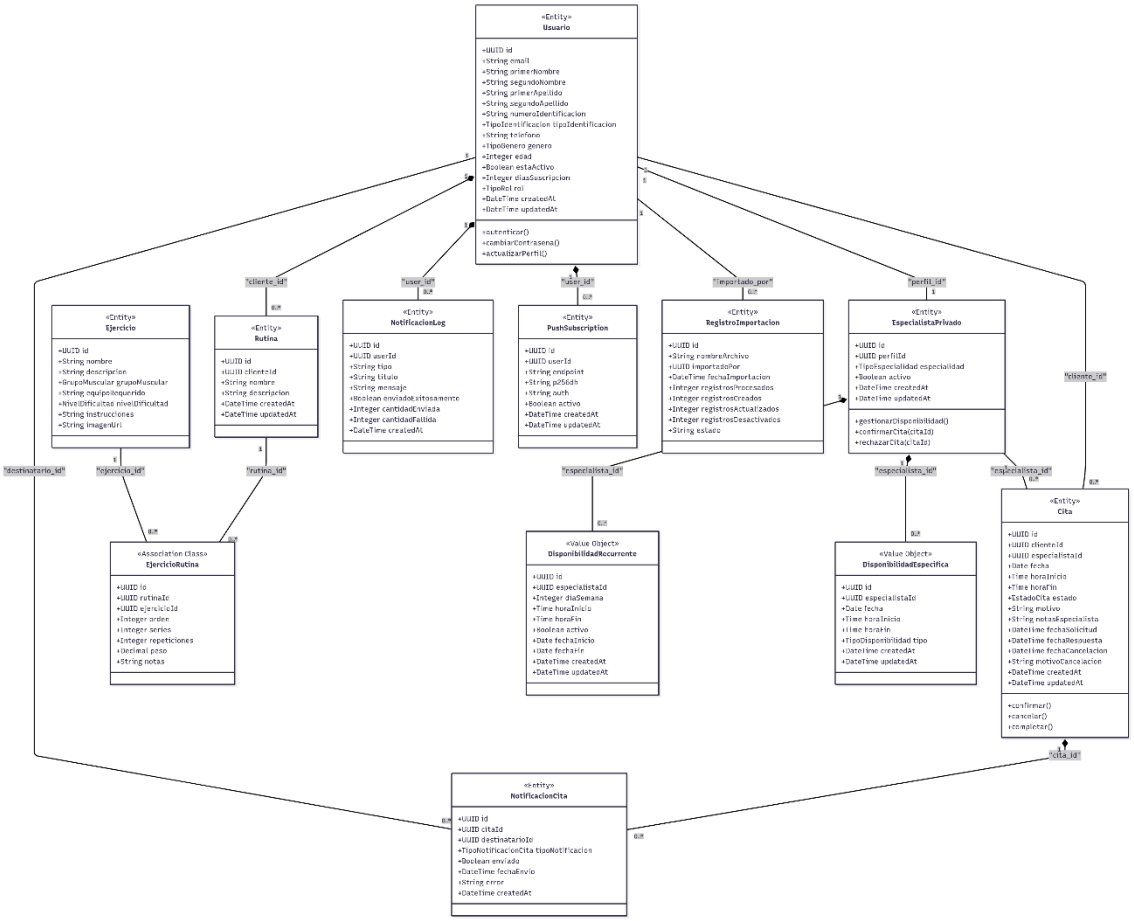


Figura 11

Diagrama de clases de la sección privada



Diseño de la base de datos

Decisiones arquitectónicas

La selección de PostgreSQL como motor de base de datos a través de la plataforma Supabase se debe a diferentes factores técnicos y operativos que se respaldan por evidencia académica y empresarial que garantizan un sistema robusto y escalable.

Justificación de PostgreSQL/Supabase. PostgreSQL fue elegido como motor de base de datos por su superioridad técnica demostrada en múltiples estudios comparativos. La investigación de Salunke y Ouda (2024) reveló que PostgreSQL es aproximadamente 13 veces más rápido que MySQL en operaciones SELECT con grandes volúmenes de datos, ejecutando consultas en 0.6-0.8 ms versus 9-12 ms de MySQL. Adicionalmente, PostgreSQL ofrece capacidades avanzadas de manejo de relaciones complejas y soporte nativo para tipos de datos JSON/JSONB que permiten combinar fortalezas relacionales con flexibilidad NoSQL (PostgreSQL Global Development Group, 2024), facilitando el almacenamiento de información estructurada flexible como las características de membresías del gimnasio.

El robusto sistema de validaciones y constraints de PostgreSQL, junto con su cumplimiento estricto de propiedades ACID, garantiza la integridad de los datos a nivel de base de datos (IBM, 2024). Su arquitectura MVCC (Multi-Version Concurrency Control) elimina la contención de bloqueos entre lecturas y escrituras, proporcionando un rendimiento superior en aplicaciones web con múltiples usuarios concurrentes (Heroku, 2024).

Supabase como plataforma de servicios complementa PostgreSQL proporcionando una arquitectura modular basada en herramientas de código abierto que incluye Kong como API Gateway, GoTrue para autenticación JWT, y PostgREST para

generación automática de APIs REST (Supabase, 2024). Las funcionalidades integradas de autenticación y autorización mediante Row Level Security (RLS) son fundamentales para un sistema interactivo como el desarrollado para FortFit, permitiendo políticas de seguridad granulares que se evalúan automáticamente en todas las consultas (AWS, 2024). Esta combinación elimina la necesidad de implementar infraestructura compleja de autenticación y autorización desde cero.

Ventajas de la solución elegida. La arquitectura elegida ha demostrado resultados cuantificables en implementaciones empresariales. Casos de estudio documentados muestran que empresas como Shotgun lograron una reducción del 83% en costos de infraestructura de datos, mientras que Good Tape obtuvo un 60% de reducción en costos operativos utilizando Supabase (Supabase, 2024). La plataforma proporciona escalabilidad automática, backup automático, y un entorno de desarrollo que facilita la iteración rápida con Edge Functions que han demostrado ser 300% más rápidas en cold starts comparado con otras alternativas (Supabase, 2024).

Supabase ofrece una interfaz de administración intuitiva que permite al equipo de FortFit gestionar datos de manera eficiente sin requerir conocimientos avanzados de administración de bases de datos, manteniendo acceso completo a todas las capacidades de PostgreSQL sin abstracciones que limiten la funcionalidad (Supabase, 2024).

Estructura general de la base de datos

El diseño de la base de datos se organiza funcionalmente en dos grandes categorías que reflejan la arquitectura dual del sistema web.

Sección pública. Incluye las tablas que almacenan información accesible a visitantes no autenticados: servicios, membresias, horarios_gimnasio, clases_grupales, horarios_clases, especialidades, especialistas, y horarios_especialistas. Estas tablas

están optimizadas para consultas de lectura frecuentes y contienen información relativamente estática que se actualiza periódicamente por administradores.

Sección privada. Comprende las tablas que gestionan la funcionalidad interactiva del sistema: perfiles, citas, ejercicio, rutina, ejercicio_rutina. especialistas_privado, disponibilidad_recurrente, disponibilidad_especifica, notificaciones_citas, push_subscriptions, notificaciones_log, y registros_importacion. Esta sección requiere controles de acceso estrictos y manejo de transacciones complejas.

Estrategias de integridad y validación

El sistema implementa múltiples capas de validación para garantizar la consistencia y confiabilidad de los datos almacenados.

Constraints de integridad. Se establecieron constraints específicos para validar reglas de negocio críticas. Por ejemplo, las citas deben tener una duración fija de 30 minutos (citas_duracion_30min), los horarios deben comenzar en intervalos de 30 minutos (citas_horario_valido), y las fechas de citas no pueden ser en el pasado (citas_horario_futuro). Estos constraints previenen la inserción de datos inválidos a nivel de base de datos.

Tipos enumerados personalizados. Se definieron tipos enum específicos como estado_cita (pendiente, aceptada, rechazada, cancelada_cliente, cancelada_especialista, completada, no_asistio), tipo_especialidad_privado (nutricion, fisioterapia), y tipo_rol (admin, entrenador, especialista, cliente) que garantizan consistencia en los valores almacenados y facilitan las consultas y validaciones.

Triggers de validación. Se implementaron triggers que ejecutan validaciones complejas antes de permitir modificaciones. El trigger trg_validar_limite_citas previene

que un cliente tenga múltiples citas pendientes o aceptadas con el mismo especialista, mientras que trg_verificar_disponibilidad_cita asegura que las citas solo se programen en horarios realmente disponibles.

Políticas de seguridad

La implementación de seguridad se basa en Row Level Security (RLS) de PostgreSQL, proporcionando un control granular de acceso a los datos.

Row Level Security (RLS). Cada tabla implementa políticas específicas que restringen el acceso según el contexto del usuario autenticado. Por ejemplo, los clientes solo pueden ver sus propias citas (cliente_id = auth.uid()), mientras que los especialistas pueden acceder a citas donde son el especialista asignado. Los administradores tienen acceso completo a través de la función is_admin().

Separación de responsabilidades. Las políticas distinguen entre operaciones de lectura, escritura, actualización y eliminación. Los clientes pueden crear citas pero solo pueden cancelar las propias, los especialistas pueden gestionar su disponibilidad pero no la de otros colegas, y solo los administradores pueden realizar operaciones de gestión masiva de usuarios.

Funciones de seguridad. Se implementó la función is_admin() que centraliza la verificación de privilegios administrativos, simplificando la gestión de permisos y facilitando futuras modificaciones en la estructura de roles.

Optimizaciones implementadas

El sistema incluye múltiples optimizaciones diseñadas para garantizar un rendimiento eficiente bajo diferentes cargas de trabajo.

Índices estratégicos. Se crearon índices específicos para las consultas más frecuentes: idx_citas_cliente optimiza las consultas de citas por cliente, idx_citas_especialista acelera las consultas de especialistas, e idx_disponibilidad_especifica_especialista_fecha mejora las consultas de disponibilidad. Estos índices reducen significativamente los tiempos de respuesta para operaciones críticas.

Vista optimizada de disponibilidad. La vista v_disponibilidad_completa precalcula la disponibilidad de especialistas combinando horarios recurrentes y excepciones específicas. Esta vista elimina la necesidad de realizar cálculos complejos en tiempo real durante el proceso de reserva de citas, mejorando la experiencia del usuario.

Funciones almacenadas. Se implementaron funciones como crear_cita_con_notificacion que encapsulan lógica de negocio compleja, garantizando consistencia transaccional y reduciendo la latencia de red al minimizar el número de consultas individuales requeridas para operaciones compuestas.

Automatización y programación

El sistema incluye elementos de automatización que reducen la carga de mantenimiento manual y garantizan operaciones consistentes.

Cron jobs automatizados. Se configuraron trabajos programados que ejecutan tareas críticas: actualización diaria de días de suscripción (restar_dias_suscripcion), verificación de recordatorios de citas cada 15 minutos, y verificación semanal de disponibilidad de especialistas. Estos trabajos aseguran que el sistema mantenga información actualizada sin intervención manual.

Triggers de mantenimiento. Los triggers update_updated_at_column mantienen automáticamente las marcas de tiempo de última modificación en todas las tablas relevantes, proporcionando auditoría automática de cambios y facilitando la sincronización con sistemas externos.

Gestión automática de notificaciones. El sistema de notificaciones opera mediante triggers y cron jobs que detectan eventos significativos (confirmación de citas, recordatorios) y programan el envío de notificaciones push correspondientes, eliminando la necesidad de gestión manual de comunicaciones con usuarios.

Wireframes de interfaces de usuario

Dispositivos escritorio

Figura 12

Wireframe de hero section en landing page escritorio

Figura 13

Wireframe de sección de servicios en landing page escritorio

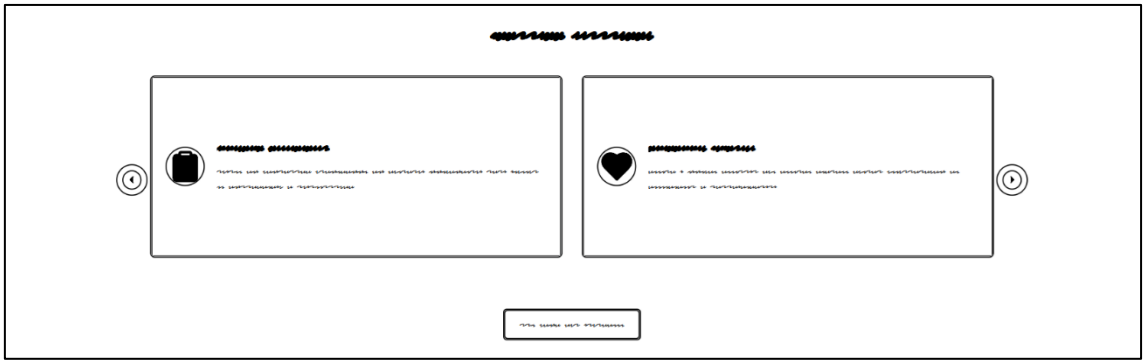


Figura 14

Wireframe de sección de instalaciones en landing page escritorio

Figura 15

Wireframe de sección de testimonios en landing page escritorio

Figura 16

Wireframe de sección de personal en landing page escritorio

Figura 17

Wireframe de sección footer escritorio

Figura 18

Wireframe de página de servicios escritorio

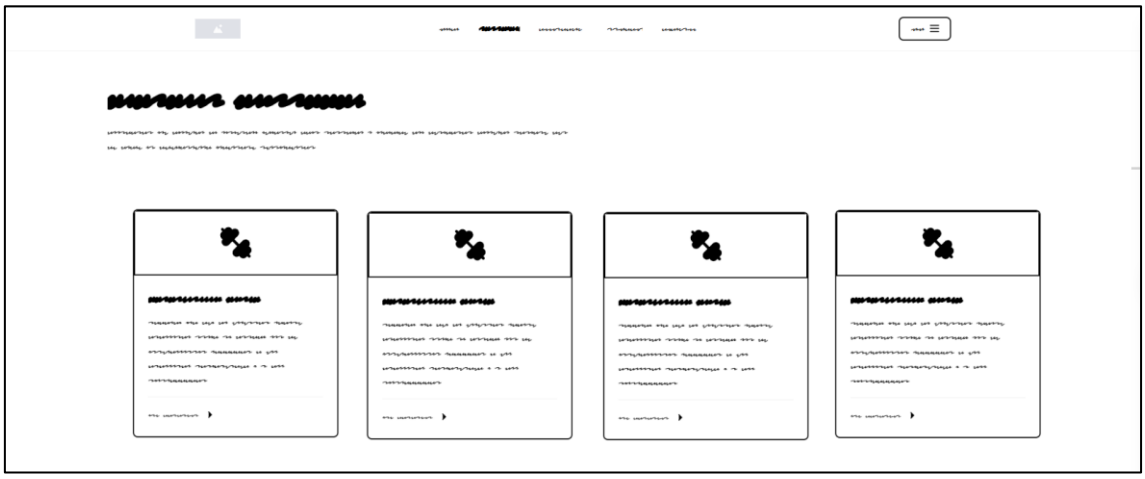


Figura 19

Wireframe de página de membresías escritorio

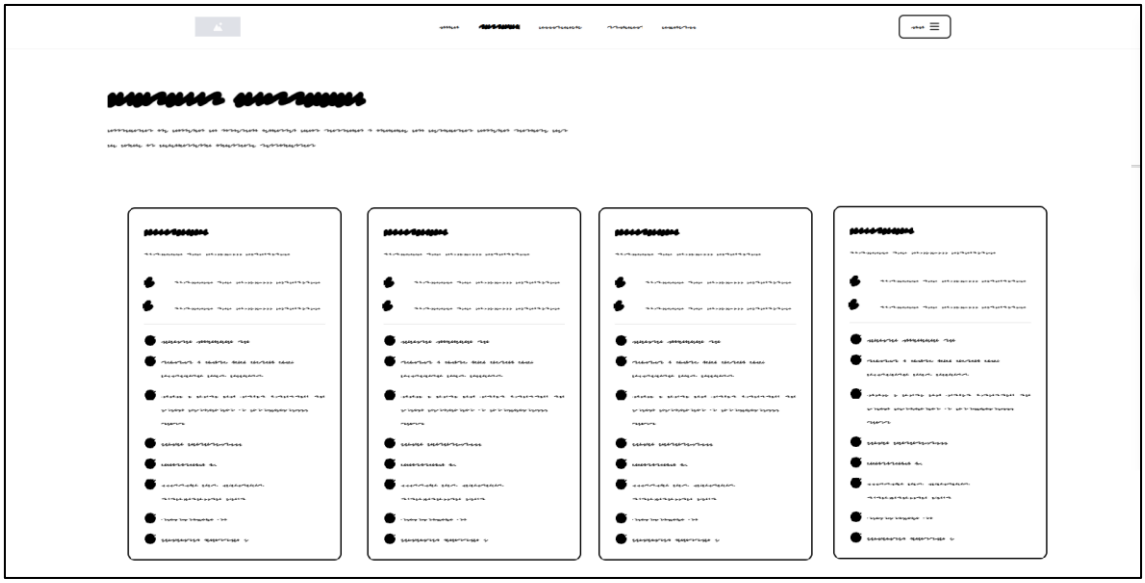
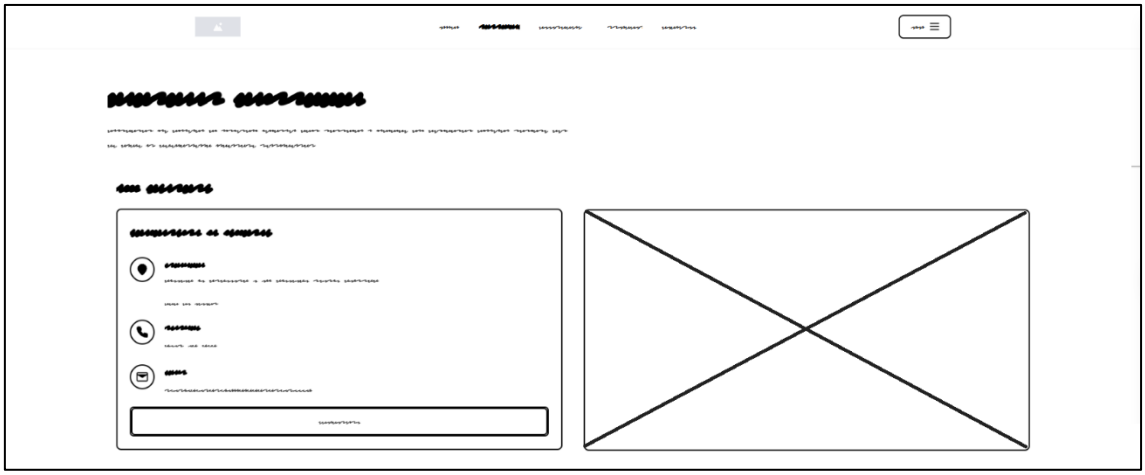


Figura 20

Wireframe de página de horarios escritorio

Figura 21

Wireframe de página de contacto escritorio



Dispositivos móviles

Figura 22

Wireframe de hero section en landing page móvil

Figura 23

Wireframe de sección de servicios en landing page móvil

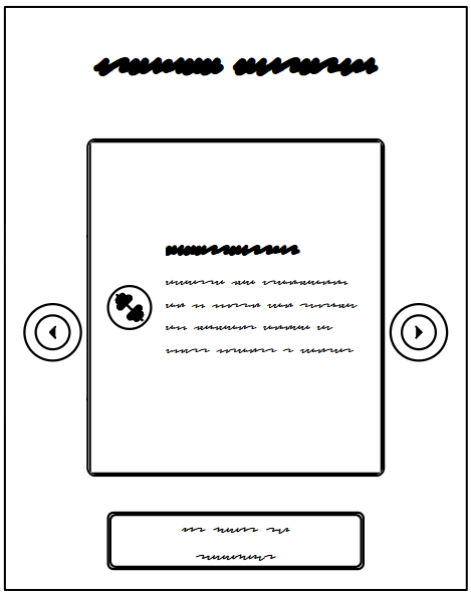


Figura 24

Wireframe de sección de instalaciones en landing page móvil

Figura 25

Wireframe de sección de testimonios en landing page móvil

Figura 26

Wireframe de sección de personal en landing page móvil

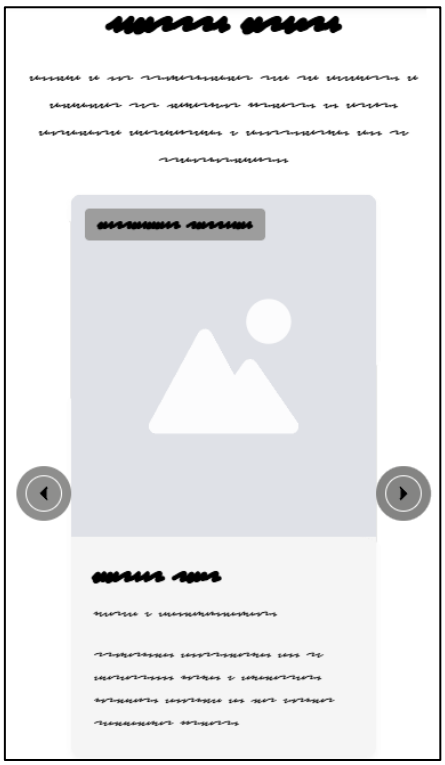


Figura 27

Wireframe de sección footer móvil

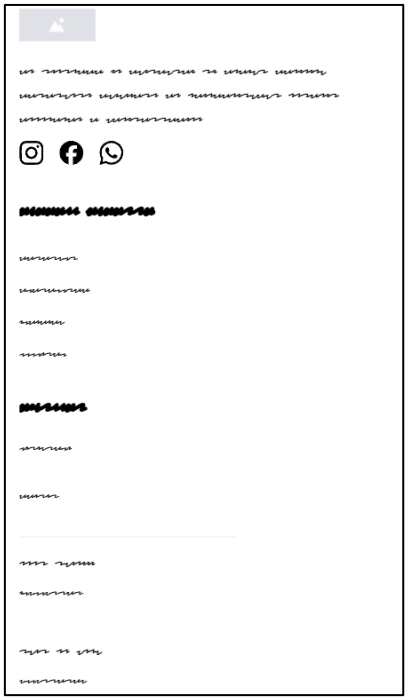


Figura 28

Wireframe de página de servicios móvil

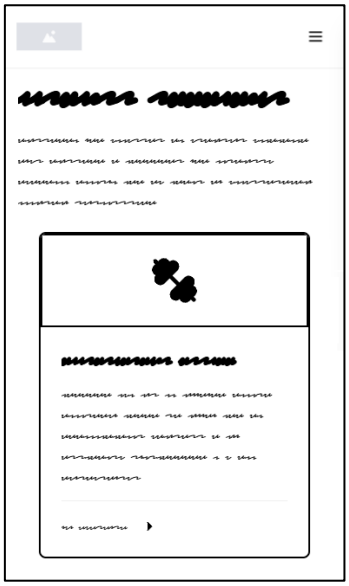


Figura 29

Wireframe de página de membresías móvil

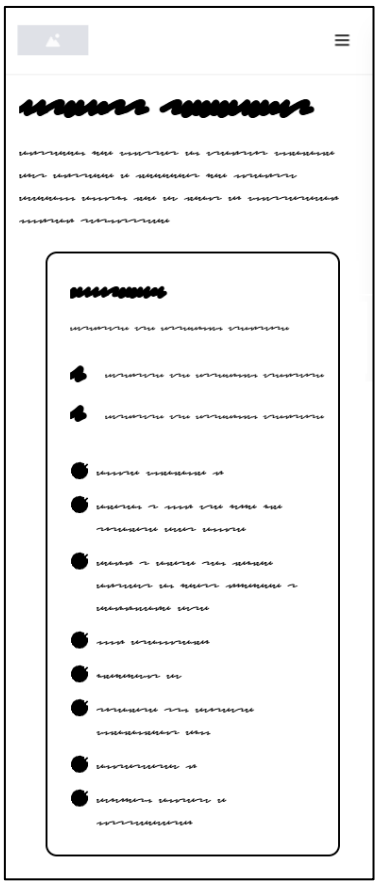


Figura 30

Wireframe de página de horarios móvil

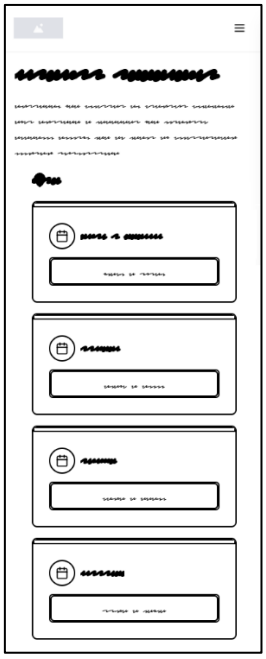
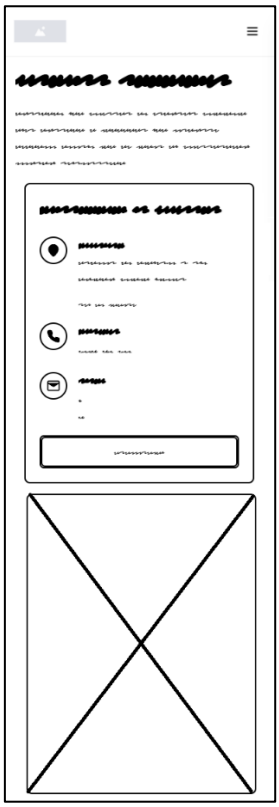


Figura 31

Wireframe de página de contacto móvil



Diseño de interfaces de usuario

Principios de UX/UI aplicados

Consistencia y patrones de diseño. Se implementó un sistema de diseño coherente basado en patrones reconocibles para que sea más intuitivo para el usuario navegar. La navegación mantiene una estructura uniforme en todas las secciones, utilizando iconografía consistente y jerarquías visuales claras que facilitan el aprendizaje y uso del sistema.

Retroalimentación visual y estados del sistema. Cada interacción del usuario genera retroalimentación visual inmediata mediante estados hover, loading, success y error claramente diferenciados. Los componentes interactivos implementan animaciones sutiles que confirman las acciones realizadas, manteniendo a los usuarios informados sobre el estado del sistema en todo momento.

Diseño responsive y mobile-first. Las interfaces se diseñaron siguiendo un enfoque mobile-first, asegurando que la experiencia en dispositivos móviles sea óptima sin comprometer la funcionalidad en pantallas más grandes. La adaptación responsive se logra mediante breakpoints estratégicos y reorganización contextual de elementos según el espacio disponible.

Accesibilidad y inclusión. Se aplicaron principios de diseño universal que garantizan la accesibilidad para usuarios con diferentes capacidades. Esto incluye contraste suficiente en los colores, tamaños de texto legibles, y áreas de clic adecuadas para dispositivos táctiles.

Sistema de diseño implementado

Paleta de colores corporativa. La paleta cromática se basa en la identidad visual existente de FortFit, adaptada para uso digital y optimizada para accesibilidad:

Colores primarios:

- **FortFit Black (#1d1d1b)**: Color principal para texto y elementos de alta jerarquía
- **FortFit Gray (#9d9d9d)**: Color secundario para elementos de soporte y estados inactivos
- **FortFit White (#ffffff)**: Fondo principal que maximiza legibilidad

Tipografía y jerarquía. Se seleccionó una familia tipográfica sans-serif moderna que garantiza legibilidad en pantallas de diferentes resoluciones:

Jerarquía tipográfica:

- **H1**: 32px/2rem - Títulos principales de página
- **H2**: 24px/1.5rem - Títulos de sección
- **H3**: 20px/1.25rem - Subtítulos y encabezados de cards
- **Body**: 16px/1rem - Texto principal
- **Small**: 14px/0.875rem - Texto secundario y metadatos

Componentes base del sistema. Se desarrolló una librería de componentes reutilizables que garantiza consistencia visual y funcional:

Componentes de entrada:

- Campos de texto con validación visual integrada

- Botones con estados clear (primary, secondary, outline, ghost)
- Selectores dropdown con búsqueda integrada
- Checkboxes y radio buttons con diseño personalizado

Componentes de navegación:

- Sidebar responsive con colapso automático en móvil
- Breadcrumbs para navegación contextual
- Tabs para organización de contenido relacionado

Componentes de contenido:

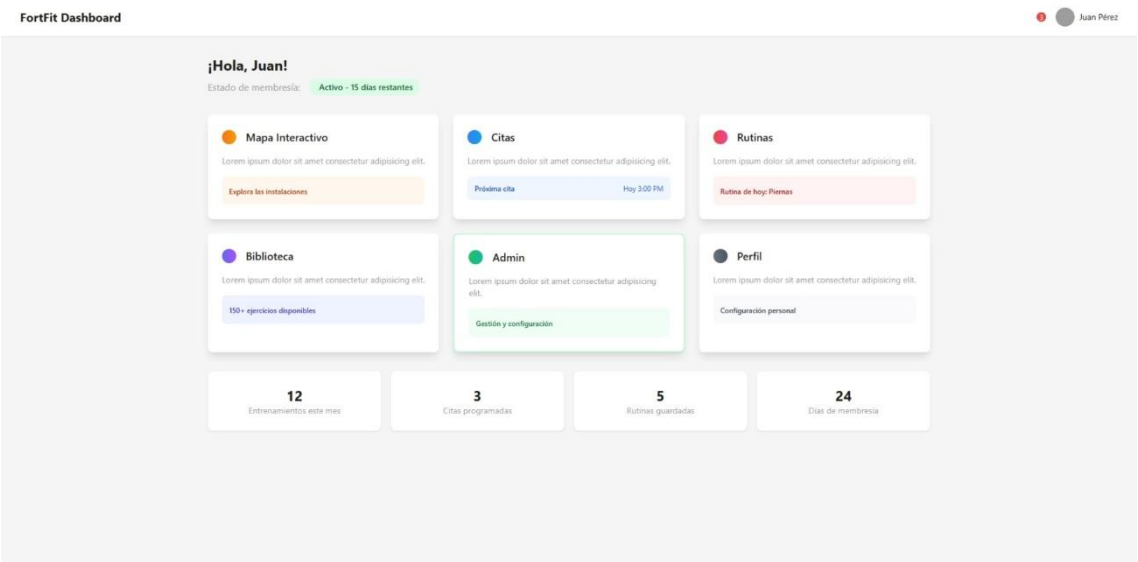
- Cards modulares para información estructurada
- Modals con overlay y gestión de foco
- Toast notifications no intrusivas
- Tablas responsive con ordenamiento y filtrado

Mockups de interfaces principales

Dashboard Principal - Vista de Escritorio

Figura 32

Mockup de página de dashboard – vista escritorio



El dashboard principal actúa como centro de control personalizado según el rol del usuario. La interfaz implementa un sidebar en el lado derecho que proporciona acceso directo a todas las funcionalidades principales del sistema. El área principal se divide en cards informativas que muestran métricas relevantes, accesos rápidos y notificaciones importantes.

Elementos clave del diseño:

- **Header superior**: Contiene el saludo personalizado y accesos rápidos al perfil y notificaciones
- **Sidebar de navegación**: Iconografía clara con etiquetas descriptivas, indicadores visuales de sección activa

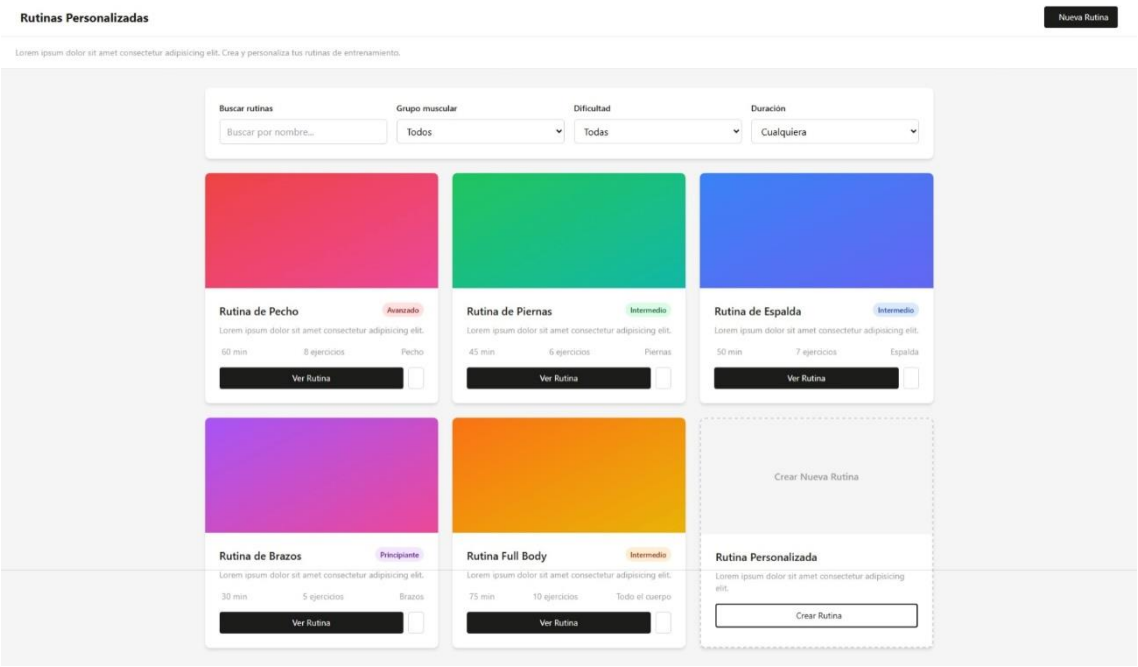
- **Grid de cards**: Distribuición responsiva que se adapta al contenido disponible según el rol del usuario
- **Área de notificaciones**: Panel lateral colapsible para alertas y actualizaciones en tiempo real

La paleta cromática se mantiene neutra con acentos de color para acciones importantes.

Gestión de Rutinas - Vista de Escritorio

Figura 33

Mockup de página de rutinas – vista escritorio



La interfaz de rutinas representa la funcionalidad más compleja del sistema, diseñada para facilitar la creación, edición y gestión de planes de entrenamiento personalizados. El diseño prioriza la visualización clara de información jerárquica (rutinas > ejercicios > series) mediante cards expandibles y una estructura visual intuitiva.

Características principales:

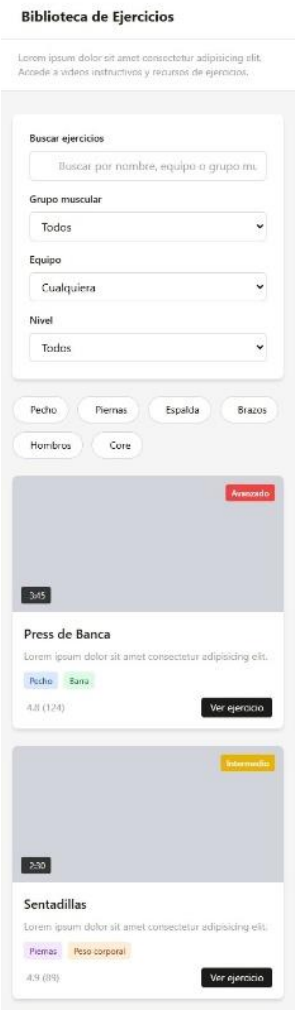
- **Lista de rutinas**: Vista de cards con información resumida, filtros por categoría y estado de progreso
- **Editor de rutinas**: Interfaz para organización de ejercicios con configuración detallada de series y repeticiones
- **Biblioteca de ejercicios**: Panel con búsqueda y filtrado por grupo muscular, equipamiento requerido y nivel de dificultad
- **Previsualización**: Vista previa en tiempo real de la rutina configurada.

El flujo de interacción se diseñó para minimizar la cantidad de clics necesarios, implementando acciones contextuales.

Biblioteca de Ejercicios - Vista Móvil

Figura 34

Mockup de página de biblioteca – vista móvil



La biblioteca de ejercicios en dispositivos móviles optimiza la experiencia de consulta durante el entrenamiento. El diseño vertical aprovecha el desplazamiento natural en móviles, presentando información de manera progresiva y enfocada.

Optimizaciones móviles implementadas:

- **Cards compactas**: Información esencial visible sin necesidad de expansión
- **Filtros desplegables**: Horizontales para filtrado rápido por categorías

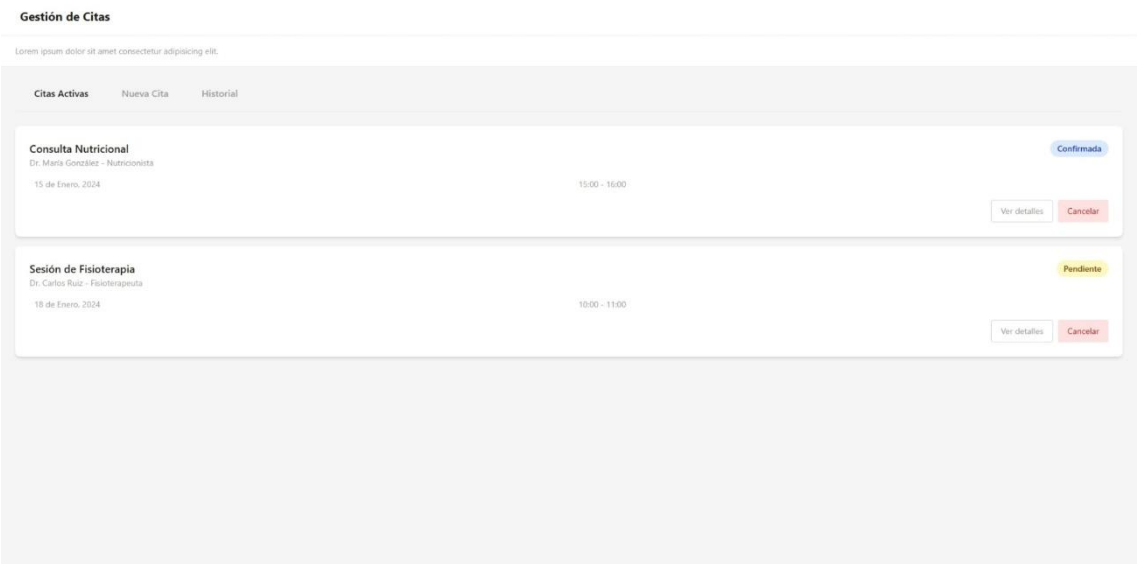
- **Imágenes optimizadas**: Carga progresiva con placeholders para conexiones lentas
- **Acciones táctiles**: Botones con área de toque aumentada

La navegación se simplifica mediante un bottom navigation que permite acceso rápido a las secciones más utilizadas sin ocupar espacio vertical valioso.

Sistema de Citas - Vista de Escritorio

Figura 35

Mockup página de citas – vista escritorio



La interfaz de gestión de citas integra un calendario visual con formularios de gestión, optimizada para diferentes roles de usuario (clientes que solicitan vs especialistas que gestionan disponibilidad).

Funcionalidades visuales destacadas:

- **Calendario interactivo**: Vista mensual con código de colores para diferentes tipos de cita y estados

- **Panel de detalles**: Información completa de citas seleccionadas con opciones contextuales según el rol
- **Gestión de disponibilidad**: Interfaz específica para especialistas con configuración de horarios disponibles
- **Validación visual**: Indicadores claros de conflictos de horario y restricciones temporales

El diseño facilita la comprensión rápida del estado de las citas mediante iconografía universalmente reconocida y códigos cromáticos consistentes.

Mapa Interactivo - Vista Móvil

Figura 36

Mockup de página de mapa interactivo – vista móvil

El mapa interactivo representa una de las funcionalidades más innovadoras del sistema, proporcionando navegación visual de las instalaciones del gimnasio mediante

una interfaz intuitiva y responsive. El diseño facilita la orientación espacial de los usuarios y optimiza el aprovechamiento de las instalaciones disponibles.

Características interactivas implementadas:

- **Navegación por plantas**: Selector de niveles con vista isométrica de cada planta del gimnasio
- **Zonas diferenciadas**: Código cromático para diferentes áreas (cardio, pesas, funcional, servicios)
- **Información contextual**: Modals desplegables con detalles de máquinas y ejercicios

La interfaz móvil optimiza la experiencia de consulta durante el recorrido físico por las instalaciones, implementando gestos de zoom y desplazamiento naturales para la exploración detallada del espacio.

Requisitos mínimos de hardware

Dispositivos de escritorio

Los usuarios que accedan desde computadoras de escritorio requieren especificaciones mínimas documentadas oficialmente.

Para Windows, se requiere Windows 10 versión 1607 o superior, con 4GB de RAM mínimo (8GB recomendado) y 1GB de almacenamiento disponible (Microsoft Learn, 2024).

En macOS, las PWAs requieren macOS Sonoma 14.0 o superior para la funcionalidad completa de "Agregar al Dock" de Safari, aunque navegadores basados en

Chromium ofrecen soporte desde macOS 10.9, requiriendo 4GB de RAM mínimo y 1GB de almacenamiento disponible (Mozilla Developer Network, 2024).

Para Linux, se necesitan distribuciones modernas como Ubuntu 18.04+ o Fedora 32+, con 2GB de RAM mínimo y 1GB de almacenamiento disponible.

Dispositivos móviles

Para smartphones y tablets, Android requiere versión 5.0+ (API Level 21) como mínimo, con 1GB de RAM mínimo (2GB recomendado), 100MB+ de almacenamiento disponible, y procesadores ARM (web.dev, 2024).

iOS requiere versión 11.3+ para soporte básico de PWAs, con 50MB de límite de almacenamiento en caché debido a las restricciones de Safari WebKit, aunque para funcionalidad completa se recomienda iOS 16.4+ que habilita notificaciones push (web.dev, 2024; Brainhub, 2024).

Navegadores web

El sistema es compatible con navegadores que cumplen estándares modernos específicos. Google Chrome establece el estándar con versión 67+ para desktop y versión 72+ para Android, proporcionando soporte completo para PWAs (web.dev, 2024).

Safari ofrece soporte desde iOS 11.3+ y macOS 14.0+, aunque con limitaciones significativas incluyendo límite de 50MB de caché (web.dev, 2024).

Microsoft Edge basado en Chromium (versión 79+) ofrece el conjunto completo de características con integración específica para entornos empresariales (Microsoft Learn, 2024).

Es importante notar que Mozilla Firefox descontinuó el soporte de instalación de PWAs en desktop desde la versión 85+ (diciembre 2020), manteniendo solo soporte para Android (Wikipedia, 2024).

Capítulo II: Construcción del Sistema

Metodología de trabajo

El desarrollo del sistema web para FortFit se basó en una adaptación de metodologías ágiles específicamente diseñada para equipos pequeños de desarrollo. La investigación académica contemporánea demuestra que los equipos de 2-3 personas en desarrollo de software logran mayor productividad y efectividad cuando implementan adaptaciones específicas de marcos ágiles tradicionales (Verwijs & Russo, 2022).

Implementación de SCRUM adaptado en el proyecto FortFit

El proyecto se estructuró utilizando una metodología basada en SCRUM adaptada para un equipo de dos desarrolladores, implementando 10 sprints consecutivos de 2 semanas cada uno, iniciando el 10 de marzo de 2025. Esta duración de sprint se seleccionó basándose en evidencia que confirma que equipos pequeños pueden efectivamente utilizar ciclos de 1-2 semanas, logrando mayor capacidad de respuesta y mejor engagement con stakeholders (Schwaber & Sutherland, 2020).

Consolidación de roles implementada. Dado el tamaño del equipo, se implementó una consolidación de roles donde ambos desarrolladores actuaron simultáneamente como Development Team y asumieron responsabilidades del Scrum Master al mismo tiempo. Las responsabilidades del Product Owner fueron compartidas entre el equipo de desarrollo y la administración del gimnasio, facilitando la toma de decisiones y la definición de prioridades en tiempo real.

Herramientas de gestión de trabajo. Se utilizó un tablero Kanban implementado en Trello como herramienta principal de gestión. El tablero se estructuró con las siguientes columnas: Backlog → To do → Desarrollo → Revisión → Completado.

Ceremonias SCRUM adaptadas

Sprint Planning. Las sesiones de planificación se redujeron de las 4 horas estándar a 1-2 horas por sprint de 2 semanas, donde se daba el refinamiento de backlog. Estas sesiones se realizaron cada lunes al inicio del sprint, con participación del equipo de desarrollo y el administrador del gimnasio para definir objetivos y prioridades.

Daily Standups. Se implementaron check-ins informales diarios de 10-15 minutos enfocados en la identificación de impedimentos y sincronización de actividades.

Sprint Reviews y colaboración con cliente. Las revisiones de sprint se realizaron presencialmente en las instalaciones del gimnasio FortFit cada 2 semanas, implementando un formato demo-interactivo donde los stakeholders podían experimentar directamente las funcionalidades desarrolladas.

Sprint Retrospectives. Al final de cada sprint se realizaron retrospectivas de 30-45 minutos enfocadas en identificar mejoras de proceso, resolver impedimentos técnicos y optimizar la colaboración entre el equipo y el cliente. La frecuencia quincenal de estas sesiones permitió implementar mejoras de proceso más rápidas comparado con ciclos mensuales tradicionales.

Integración con stakeholders

La colaboración con el cliente se estructuró siguiendo el marco de engagement de stakeholders documentado en la literatura, clasificando a la administración del gimnasio como stakeholders de alta influencia y alto interés, requiriendo involucramiento extenso y colaboración estrecha (*Creating A Stakeholder Engagement Approach*, s. f.). Se establecieron puntos de contacto regulares que incluían:

- Reuniones presenciales quincenales durante las Sprint Reviews
- Comunicación continua vía WhatsApp para consultas urgentes y clarificaciones
- Sesiones de feedback intermedio cuando se identificaban elementos que requerían validación antes del Sprint Review formal

Gestión de versiones y despliegue continuo. Se implementó un flujo de trabajo Git con ramas de desarrollo separadas para cada funcionalidad, integrando revisiones de código cruzadas entre ambos desarrolladores antes de merge a la rama principal. El despliegue se realizó utilizando Vercel con integración continua, permitiendo previsualizaciones automáticas de cada pull request que facilitaron la validación temprana con stakeholders antes del despliegue a producción.

La metodología implementada demostró efectividad en la entrega de incrementos funcionales cada 2 semanas, manteniendo alta calidad de código y satisfacción del cliente a lo largo de los 10 sprints.

Requisitos de software y hardware

Entorno de desarrollo

Herramientas de desarrollo requeridas. Para la implementación del sistema web de FortFit se establecieron requisitos específicos de software y hardware basados en las tecnologías efectivamente utilizadas en el proyecto, garantizando un entorno de desarrollo estable y una infraestructura de producción robusta.

Runtime y lenguajes de programación:

- **Node.js**: Versión 20+ requerida para compatibilidad con Next.js 15.2.2
- **pnpm**: Versión 10.8.0 como package manager principal del proyecto
- **TypeScript**: Versión 5+ con configuración strict habilitada
- **JavaScript**: Target ES2017 para compatibilidad amplia de navegadores

Framework y librerías principales:

- **Next.js**: Versión 15.2.2 con App Router y soporte completo para PWA
- **React**: Versión 19.0.0 con Server Components y hooks modernos
- **Tailwind CSS**: Versión 3.4.17 con configuración personalizada para FortFit
- **@ducanh2912/next-pwa**: Versión 10.2.9 para funcionalidades de Progressive Web App

Herramientas de gestión de código:

- **Git**: Sistema de control de versiones con ignorado específico para Next.js
- **ESLint**: Versión 9 con configuración next/core-web-vitals

- **TypeScript**: Configuración con paths alias (@/*) y strict mode habilitado
- **PostCSS**: Configuración para Tailwind CSS y autoprefixer

Dependencias principales del sistema

Autenticación y base de datos:

- **@supabase/supabase-js**: Versión 2.49.1 para cliente JavaScript
- **@supabase/ssr**: Versión 0.5.2 para Server-Side Rendering con autenticación

Componentes de interfaz de usuario:

- **@radix-ui/react-***: Suite completa de componentes accesibles sin estilos
    - react-dialog (1.1.6), react-dropdown-menu (2.1.6), react-form (0.1.2)
    - react-label (2.1.7), react-popover (1.1.14), react-select (2.1.6)
    - react-tabs (1.1.3), react-toast (1.2.6)
- **lucide-react**: Versión 0.479.0 para iconografía del sistema
- **motion**: Versión 12.6.2 para animaciones avanzadas

Utilidades y validación:

- **zod**: Versión 3.24.2 para validación de esquemas TypeScript
- **date-fns**: Versión 4.1.0 para manipulación de fechas
- **class-variance-authority**: Versión 0.7.1 para variantes de componentes
- **clsx**: Versión 2.1.1 y **tailwind-merge**: Versión 3.0.2 para gestión de clases CSS

Funcionalidades específicas:

- **xlsx**: Versión 0.18.5 para manejo de archivos Excel (importación de usuarios)

- **file-saver**: Versión 2.0.5 para descarga de archivos
- **web-push**: Versión 3.6.7 para notificaciones push
- **react-day-picker**: Versión 9.8.0 para selectores de fecha
- **next-themes**: Versión 0.4.6 para manejo de temas oscuro/claro

Configuración de desarrollo

Configuración de TypeScript

```
{
  "compilerOptions": {
    "target": "ES2017",
    "lib": ["dom", "dom.iterable", "esnext"],
    "strict": true,
    "module": "esnext",
    "moduleResolution": "bundler",
    "jsx": "preserve",
    "paths": {
      "@/*": ["./src/*"]
    }
  }
}
```

Configuración de Next.js

next.config.js configurado con:

- **PWA**: Integración completa con service worker personalizado
- **Imágenes optimizadas**: Dominios permitidos para localhost, Vercel y dominio de producción
- **React Strict Mode**: Habilitado para desarrollo riguroso
- **Variables de entorno**: Configuración para diferentes entornos

Configuración de Tailwind CSS

Paleta de colores personalizada FortFit:

```
colors: {
  fortfit: {
    black: '#1d1d1b',
    white: '#ffffff',
    gray: '#9d9d9d',
  }
}
```

Características implementadas:

- **Breakpoints extendidos**: Desde 'xs' (480px) hasta '2xl' (1536px)
- **Modo oscuro**: Configurado con clase 'dark'
- **Degradados personalizados**: Basados en colores corporativos
- **Sombras corporativas**: Utilizando colores FortFit con diferentes opacidades

Infraestructura de producción

Servicios en la nube utilizados

Plataforma de despliegue:

- **Vercel**: Plataforma principal para despliegue automático desde Git
- **Dominio de producción**: fortfit-web-app.vercel.app
- **SSL automático**: Certificados HTTPS gestionados automáticamente
- **Edge Functions**: Para APIs optimizadas globalmente

Base de datos y backend:

- **Supabase**: PostgreSQL gestionado con Row Level Security implementado
- **Edge Functions**: Funciones serverless para:
    - send-push-notification: Envío de notificaciones push
    - check-appointments-reminders: Verificación de recordatorios de citas

- archive-old-notifications: Archivado automático de notificaciones antiguas

Progressive Web App:

- **Service Worker**: Configurado en directorio 'public' con cache automático
- **Manifest**: Generado automáticamente por next-pwa
- **Instalación**: Capacidad de instalación en dispositivos móviles y escritorio

Especificaciones técnicas del entorno

Configuración de desarrollo:

- **Package Manager**: pnpm 10.8.0 con configuración de workspace
- **Scripts disponibles**: dev, build, start, lint
- **Hot Reload**: Habilitado para desarrollo ágil
- **Build optimizado**: Generación automática de archivos estáticos optimizados

Gestión de archivos:

- **Exclusiones de Git**: Configuradas para archivos de entorno, builds y dependencias
- **TypeScript**: Incremental compilation habilitado para builds más rápidos
- **PostCSS**: Procesamiento automático de CSS con Tailwind y autoprefixer

Variables de entorno requeridas

Variables de desarrollo:

```
NEXT_PUBLIC_SUPABASE_URL=[supabase-project-url]
NEXT_PUBLIC_SUPABASE_ANON_KEY=[anon-key]
SUPABASE_SERVICE_ROLE_KEY=[service-role-key]
```

Variables de producción:

```
NEXT_PUBLIC_SUPABASE_URL=[production-url]
NEXT_PUBLIC_SUPABASE_ANON_KEY=[production-anon-key]
SUPABASE_SERVICE_ROLE_KEY=[production-service-key]
NODE_ENV=production
```

Gestión de la base de datos

Configuración inicial del entorno Supabase

La implementación del sistema de base de datos se inició con la configuración de un proyecto en Supabase, aprovechando su arquitectura basada en PostgreSQL 15 con funcionalidades adicionales de autenticación, almacenamiento y real-time subscriptions. La configuración inicial incluyó la definición de tres tipos de conexiones diferenciadas según el contexto de uso.

Configuración de conexiones. Se establecieron tres clientes de Supabase específicos para diferentes contextos de la aplicación:

- **Cliente público** (client.ts): Configurado para operaciones del lado del cliente con acceso limitado a datos públicos y operaciones autenticadas básicas.
- **Cliente servidor** (server.ts): Implementado para operaciones server-side con capacidades completas de autenticación y acceso a datos privados mediante cookies de sesión.
- **Cliente administrativo** (admin.ts): Configurado con credenciales de service role para operaciones administrativas, incluyendo bypass de Row Level Security para tareas de migración y mantenimiento.

Variables de entorno y seguridad. Se configuraron las variables de entorno necesarias para el funcionamiento seguro del sistema:

```
NEXT_PUBLIC_SUPABASE_URL=https://[project-id].supabase.co
NEXT_PUBLIC_SUPABASE_ANON_KEY=[anon-key]
SUPABASE_SERVICE_ROLE_KEY=[service-role-key]
```

La separación de claves públicas y privadas garantiza que las operaciones críticas solo puedan ejecutarse desde el servidor, mientras que el cliente mantiene acceso limitado a funcionalidades seguras.

Implementación de la estructura de datos

La creación de la estructura de base de datos siguió un enfoque incremental, comenzando con las tablas fundamentales y progresando hacia entidades más complejas que dependían de las anteriores.

Tablas de la sección pública. Se implementaron primero las tablas que almacenan información accesible sin autenticación:

- servicios: Información de servicios ofrecidos por el gimnasio
- membresias: Tipos y costos de membresías disponibles
- horarios_gimnasio: Horarios de operación por sede
- clases_grupales: Catálogo de clases grupales disponibles
- horarios_clases: Programación específica de clases
- especialidades: Tipos de especialistas disponibles
- especialistas: Información pública de profesionales
- horarios_especialistas: Disponibilidad pública de especialistas

Tablas de la sección privada. Posteriormente se crearon las tablas que gestionan funcionalidades interactivas:

- perfiles: Extensión de usuarios autenticados con información adicional
- citas: Sistema de reservas entre clientes y especialistas
- ejercicios: Biblioteca de ejercicios con información detallada
- rutinas: Rutinas personalizadas para usuarios
- ejercicio_rutina: Relación many-to-many entre ejercicios y rutinas
- especialistas_privado: Información privada y configuraciones de especialistas
- disponibilidad_recurrente y disponibilidad_especifica: Gestión avanzada de horarios
- historial_ejecuciones y historial_ejercicios: Seguimiento de progreso de usuarios

Tipos de datos personalizados. Se definieron varios enums para garantizar consistencia de datos:

```
CREATE TYPE estado_cita AS ENUM ('pendiente', 'confirmada', 'cancelada',
'completada');
CREATE TYPE tipo_disponibilidad AS ENUM ('disponible', 'ocupado',
'descanso');
CREATE TYPE nivel_dificultad AS ENUM ('principiante', 'intermedio',
'avanzado');
CREATE TYPE tipo_usuario AS ENUM ('cliente', 'entrenador', 'especialista',
'admin');
```

Implementación de políticas de seguridad (RLS)

La seguridad de datos se implementó mediante Row Level Security (RLS) de PostgreSQL, estableciendo políticas granulares que se evalúan automáticamente en cada consulta.

Políticas para usuarios autenticados. Se configuraron políticas base que permiten a los usuarios acceder únicamente a sus propios datos:

--Ejemplo: Política para tabla perfiles

```
CREATE POLICY "Usuarios pueden ver su propio perfil" ON perfiles
    FOR SELECT USING (auth.uid() = usuario_id);

CREATE POLICY "Usuarios pueden actualizar su propio perfil" ON perfiles
    FOR UPDATE USING (auth.uid() = usuario_id);
```

Políticas específicas por rol. Se implementaron políticas diferenciadas según el tipo de usuario:

- **Clientes**: Acceso completo a sus rutinas, historial y citas. Lectura de información de ejercicios y especialistas.
- **Especialistas**: Acceso a sus propias citas y disponibilidad. Capacidad de gestionar rutinas de sus clientes asignados.
- **Entrenadores**: Permisos similares a especialistas con acceso adicional a gestión de rutinas.
- **Administradores**: Acceso completo para gestión del sistema.

Políticas para datos públicos. Las tablas de la sección pública se configuraron con políticas que permiten lectura sin autenticación:

```
CREATE POLICY "Información pública accesible a todos" ON servicios
    FOR SELECT USING (true);
```

Funciones y triggers de base de datos

Se desarrollaron funciones personalizadas en PL/pgSQL para encapsular lógica de negocio compleja y mantener la integridad de datos.

Funciones principales implementadas:

- actualizar_updated_at(): Trigger function que actualiza automáticamente el campo updated_at en modificaciones
- validar_disponibilidad_cita(): Valida conflictos de horarios antes de crear citas

- calcular_estadisticas_rutina(): Genera métricas de progreso para historial de ejecuciones
- notificar_cambio_cita(): Trigger que envía notificaciones automáticas en cambios de estado de citas
- limpiar_datos_expirados(): Función para mantenimiento automático de datos obsoletos

Triggers implementados:

-- Ejemplo: Trigger para actualización automática de timestamps

```
CREATE TRIGGER trigger_updated_at
    BEFORE UPDATE ON perfiles
    FOR EACH ROW EXECUTE FUNCTION actualizar_updated_at();
```

-- Trigger para notificaciones automáticas

```
CREATE TRIGGER trigger_notificar_cita
    AFTER UPDATE ON citas
    FOR EACH ROW EXECUTE FUNCTION notificar_cambio_cita();
```

Automatización y tareas programadas

Se implementaron Edge Functions de Supabase para automatizar tareas críticas del sistema, aprovechando la capacidad de ejecución programada mediante cron jobs.

Edge Function para recordatorios de citas (check-appointments-reminders): Ejecutada cada 30 minutos, identifica citas próximas (24 horas antes) y envía notificaciones push a usuarios que tengan suscripciones activas. La función consulta citas confirmadas, valida suscripciones de notificaciones y utiliza el protocolo Web Push para envío.

Edge Function para archivado de notificaciones (archive-old-notifications): Programada para ejecutarse diariamente a medianoche, elimina notificaciones de citas con más de 30 días de antigüedad para mantener optimizada la base de datos.

Edge Function para envío de notificaciones push (send-push-notification): Función de utilidad que maneja el envío de notificaciones push, incluyendo la gestión de VAPID keys, formateo de payloads y manejo de errores para suscripciones inválidas.

Migración y población inicial de datos

Scripts de migración. Se desarrollaron scripts SQL organizados por categorías para facilitar el despliegue:

- tablas_bdd: Definición completa de estructura de tablas
- funciones_bdd: Implementación de funciones personalizadas
- triggers_bdd: Configuración de triggers automáticos
- políticas_RLS: Definición de todas las políticas de seguridad
- enum_types_bdd: Creación de tipos de datos personalizados

Población inicial de datos. Se cargaron datos base del gimnasio incluyendo:

- Servicios y membresías con precios actualizados
- Información de especialistas y sus especialidades
- Horarios de operación de ambas sedes (América y El Inca)
- Biblioteca inicial de ejercicios organizados por grupos musculares
- Configuración de plantas y zonas del gimnasio para el mapa interactivo

Proceso de importación de usuarios existentes. Se implementó una funcionalidad específica para migrar usuarios del sistema administrativo existente, incluyendo validación de emails, asignación de contraseñas temporales y configuración de perfiles básicos.

Optimización y monitoreo

Índices implementados. Se crearon índices estratégicos para optimizar consultas frecuentes:

-- Índices para consultas de citas por usuario y fecha

```
CREATE INDEX idx_citas_usuario_fecha ON citas(usuario_id, fecha_hora);
CREATE INDEX idx_citas_especialista_fecha ON citas(especialista_id,
fecha_hora);
```

-- Índices para historial de ejecuciones

```
CREATE INDEX idx_historial_usuario_fecha ON historial_ejecuciones(usuario_id,
fecha_ejecucion DESC);
```

-- Índices para búsquedas de ejercicios

```
CREATE INDEX idx_ejercicios_grupo_muscular ON
ejercicios(grupo_muscular_principal);
```

Análisis de rendimiento. Se identificaron y optimizaron las consultas más críticas:

- Consultas de citas con joins complejos entre usuarios, especialistas y disponibilidad
- Recuperación de rutinas completas con ejercicios y configuraciones
- Generación de estadísticas de progreso para dashboards de usuario

Estrategias de optimización aplicadas:

- Uso de prepared statements en consultas frecuentes
- Implementación de paginación en listados extensos
- Optimización de políticas RLS para minimizar overhead de evaluación

La gestión integral de la base de datos proporcionó una base sólida y escalable para todas las funcionalidades del sistema, garantizando rendimiento, seguridad y mantenibilidad a largo plazo.

Implementación del backend

Arquitectura del backend con Next.js

El backend del sistema FortFit se implementó utilizando el patrón de API Routes de Next.js, aprovechando la capacidad del framework para unificar desarrollo frontend y backend en una sola aplicación. Esta arquitectura permitió mantener consistencia en el lenguaje de programación (TypeScript) y simplificar significativamente el proceso de desarrollo y despliegue.

Estructura de carpetas y organización del código. Las APIs se organizaron siguiendo la estructura de archivos de Next.js 13+ bajo el directorio src/app/api/, implementando una jerarquía lógica que refleja las entidades del sistema:

```
src/app/api/
├── admin/usuarios/[id]/route.ts      # Gestión administrativa de usuarios
├── auth/security/route.ts            # Seguridad y bloqueo de cuentas
├── citas/                           # Sistema de citas
├── ejercicios/[id]/route.ts         # Biblioteca de ejercicios
├── grupos-musculares/route.ts       # Categorización de ejercicios
├── historial/                       # Seguimiento de progreso
├── maquinas/route.ts                # Equipamiento del gimnasio
├── plantas/route.ts                 # Estructura física del gimnasio
├── rutinas/[id]/route.ts            # Rutinas personalizadas
└── zonas/route.ts                   # Organización espacial
```

Separación de responsabilidades. Se implementó una clara separación entre lógica de cliente y servidor, donde el backend se encarga exclusivamente de:

- Validación y sanitización de datos de entrada
- Autenticación y autorización de usuarios
- Operaciones de base de datos con Supabase
- Aplicación de reglas de negocio
- Generación de respuestas estandarizadas

El frontend consume estas APIs mediante hooks personalizados que gestionan estados de carga, error y datos, manteniendo la interfaz de usuario reactiva y desacoplada de la lógica de negocio.

Sistema de autenticación y autorización

La autenticación se implementó integrando Supabase Auth con un sistema de autorización personalizado que gestiona diferentes roles y estados de usuario.

Implementación con Supabase Auth. Se configuraron tres clientes de Supabase especializados:

- **Cliente público** (client.ts): Para operaciones básicas sin privilegios elevados
- **Cliente servidor** (server.ts): Para operaciones autenticadas con acceso completo a datos privados
- **Cliente administrativo** (admin.ts): Con credenciales de service role para operaciones críticas

Gestión de usuarios y roles. El sistema implementa cuatro tipos de usuario con permisos diferenciados:

- **Cliente**: Acceso a rutinas personales, historial y sistema de citas
- **Entrenador**: Gestión de rutinas y seguimiento de clientes asignados
- **Especialista**: Manejo de citas y disponibilidad personal
- **Administrador**: Control total del sistema y gestión de usuarios

Flujos de autenticación implementados:

1. **Registro inicial**: Los usuarios se crean con contraseñas temporales que deben cambiarse en el primer ingreso
2. **Login estándar**: Validación de credenciales con verificación de estado de cuenta y password_changed
3. **Recuperación de contraseña**: Gestionada por administradores para mayor seguridad
4. **Cambio obligatorio**: Forzado en primer acceso con validación robusta

Middleware de seguridad

Se desarrolló un middleware comprensivo que implementa múltiples capas de seguridad mediante checkpoints secuenciales.

Implementación del middleware de autenticación. El archivo src/middleware.ts implementa 9 checkpoints críticos que evalúan criterios como: si es la

primera vez que inicia sesión, si la cuenta está inactiva, si ya cambió la contraseña por primera vez, entre otros.

```
// Checkpoint 1: Verificación de sesión activa
if (!data.session) {
  return NextResponse.redirect(new URL("/login", request.url));
}

// Checkpoint 2: Obtención de metadata del usuario
const passwordChanged = userData.user?.user_metadata.password_changed ===
true;

// Checkpoint 3: Verificación de primer login
if (!passwordChanged) {
  return NextResponse.redirect(new URL("/cambiar-contrasena?first=true",
request.url));
}
```

Protección de rutas privadas. El middleware protege automáticamente todas las rutas que comienzan con /dashboard y /dashboard/admin, aplicando validaciones específicas:

- Verificación de sesión válida
- Validación de estado de cuenta activa
- Confirmación de cambio de contraseña inicial
- Verificación de roles para acceso administrativo

Validación de sesiones y tokens. Se implementó integración con el sistema de cookies de Supabase para mantener sesiones persistentes y seguras, con manejo automático de renovación de tokens y logout en caso de sesiones inválidas.

Desarrollo de APIs REST

Las APIs implementan un patrón consistente utilizando un sistema de helpers personalizado que estandariza operaciones comunes.

Sistema de helpers implementado:

- **ApiAuth**: Gestiona autenticación y autorización con métodos como requireAuth() y requireAdmin()
- **ApiDatabase**: Encapsula operaciones de base de datos con manejo automático de errores
- **ApiValidation**: Valida datos de entrada con reglas específicas por tipo de campo
- **ApiResponse**: Genera respuestas HTTP estandarizadas con formatos consistentes

APIs principales desarrolladas:

1. **APIs de Ejercicios** (/api/ejercicios/): Gestión completa de la biblioteca de ejercicios con búsqueda por máquina, grupo muscular y nivel de dificultad
2. **APIs de Rutinas** (/api/rutinas/): Creación, modificación y gestión de rutinas personalizadas con ejercicios asociados
3. **APIs de Citas** (/api/citas/): Sistema completo de reservas con validación de disponibilidad y gestión de estados
4. **APIs de Historial** (/api/historial/): Seguimiento de progreso con métricas detalladas de ejecución de rutinas
5. **APIs Administrativas** (/api/admin/): Gestión de usuarios, importación masiva y configuraciones del sistema

Ejemplo de implementación de API estándar:

```
export async function GET(request: NextRequest) {
  try {
    await ApiAuth.requireAuth();

    const { searchParams } = new URL(request.url);
    const plantaId = searchParams.get("planta_id");

    if (plantaId && !ApiValidation.isValidId(plantaId)) {
      return ApiResponse.badRequest("ID de planta inválido");
    }

    const zonas = await ApiDatabase.executeQuery(async (supabase) => {
      return await supabase.from("zonas")
        .select("*")
        .eq("activo", true)
        .order("orden_visualizacion");
    });

    return ApiResponse.success(zonas, "Zonas obtenidas exitosamente");
  } catch (error) {
    return ApiResponse.error("Error interno del servidor");
  }
}
```

Seguridad del sistema

Se implementaron múltiples capas de seguridad para proteger contra vulnerabilidades comunes y ataques maliciosos.

Sistema de bloqueo de cuentas. Se desarrolló un mecanismo sofisticado de seguridad que incluye:

- **Bloqueo temporal**: 5 minutos después de 3 intentos fallidos consecutivos
- **Bloqueo extendido**: 20 minutos después de 5 intentos fallidos totales
- **Bloqueo permanente**: Después de múltiples bloqueos temporales

Este sistema se implementó mediante funciones de base de datos que registran cada intento de login con información de IP, user agent y timestamp, permitiendo análisis forense de intentos de acceso no autorizado.

Validación y sanitización de datos. Todas las APIs implementan validación robusta mediante la clase ApiValidation:

```
static validateRequiredFields(data: Record<string, any>, requiredFields:
string[]): Record<string, string[]>
static validateStringLength(value: string, field: string, min = 0, max =
255): string[]
static validateEmail(email: string): string[]
static validateNumber(value: any, field: string, min?: number, max?: number):
string[]
static validateEnum(value: string, field: string, allowedValues: string[]):
string[]
```

Prevención de ataques comunes. Se implementaron medidas específicas contra:

- **SQL Injection**: Uso exclusivo de consultas parametrizadas a través de Supabase
- **XSS**: Sanitización automática de datos de entrada y escape de salida
- **CSRF**: Validación de tokens de sesión y verificación de origen
- **Brute Force**: Sistema de bloqueo progresivo de cuentas

Integración con servicios externos

Configuración de Supabase client/server. Se implementaron tres niveles de acceso diferenciados:

```
// Cliente público - operaciones básicas
const supabase = createBrowserClient(url, anonKey);

// Cliente servidor - operaciones autenticadas
const supabase = createServerClient(url, anonKey, cookieStore);

// Cliente administrativo - operaciones privilegiadas
const supabase = createClient(url, serviceRoleKey);
```

Gestión de Edge Functions. Se desarrollaron tres Edge Functions especializadas:

1. **check-appointments-reminders**: Ejecutada cada 30 minutos para enviar recordatorios de citas próximas

2. **archive-old-notifications**: Limpieza diaria automática de notificaciones antiguas

3. **send-push-notification**: Gestión centralizada de notificaciones push con manejo de errores

Integración con servicios de notificaciones push. Se implementó el protocolo Web Push con:

- Generación automática de VAPID keys
- Gestión de suscripciones de usuarios
- Formateo de payloads según especificaciones W3C
- Manejo robusto de errores y suscripciones inválidas

Manejo de errores y logging

Estrategias de manejo de errores. Se implementó un sistema jerárquico de manejo de errores:

```
try {
  await ApiAuth.requireAuth();
  const result = await ApiDatabase.executeQuery(/*...*/);
  return ApiResponse.success(result);
} catch (error) {
  if (error instanceof Error) {
    if (error.message === "Usuario no autenticado") {
      return ApiResponse.unauthorized();
    }
    return ApiResponse.error(error.message);
  }
  return ApiResponse.error("Error interno del servidor");
}
```

Códigos de estado HTTP apropiados. La clase ApiResponse implementa métodos específicos para cada tipo de respuesta:

- 200: Operaciones exitosas con success()
- 400: Solicitudes malformadas con badRequest()
- 401: Falta de autenticación con unauthorized()
- 403: Acceso denegado con forbidden()
- 404: Recursos no encontrados con notFound()
- 422: Errores de validación con validationError()
- 500: Errores internos con error()

Logging y monitoreo de APIs. Se implementó logging consistente en todas las APIs con información relevante para debugging y auditoría, incluyendo timestamps, IDs de usuario, parámetros de entrada y detalles de errores.

Optimización y rendimiento

Estrategias de caché implementadas. Se aprovechó el sistema de caché nativo de Supabase y las capacidades de Edge Caching de Vercel para optimizar consultas frecuentes, especialmente para datos relativamente estáticos como ejercicios, máquinas y zonas del gimnasio.

Optimización de consultas a la base de datos. Las consultas complejas se optimizaron mediante:

- Uso de select() específico en lugar de select("*")

- Implementación de joins eficientes para datos relacionados
- Aplicación de filtros en la base de datos en lugar del cliente
- Ordenamiento y paginación en consultas de listado

Gestión de conexiones. Supabase maneja automáticamente el pool de conexiones, pero se implementaron prácticas de liberación temprana de recursos y reutilización de clientes cuando es posible.

Paginación y límites de consulta. Se implementó paginación en APIs que manejan grandes volúmenes de datos, como listados de ejercicios y historial de usuario, con parámetros limit y offset para controlar el tamaño de respuesta y mejorar el rendimiento de la aplicación.

La implementación del backend proporcionó una base robusta, segura y escalable que soporta todas las funcionalidades del sistema FortFit, manteniendo altos estándares de seguridad y rendimiento mientras facilita el mantenimiento y la extensión futura del sistema.

Desarrollo del frontend

Arquitectura del frontend con Next.js y React

El frontend del sistema FortFit se desarrolló utilizando Next.js 13+ con App Router, implementando una arquitectura moderna basada en componentes de React con TypeScript para garantizar type safety y mejor experiencia de desarrollo.

Estructura de carpetas y organización de componentes. Se adoptó una organización jerárquica que separa claramente responsabilidades y facilita el mantenimiento:

```
src/
├── app/                          # App Router de Next.js 13+
│   ├── (private)/dashboard/      # Rutas privadas protegidas
│   └── (public)/                 # Rutas públicas
├── components/                   # Componentes organizados por dominio
│   ├── common/                   # Componentes generales (Logo, Loading)
│   ├── features/                 # Componentes específicos por funcionalidad
│   │   ├── landing/              # Página principal pública
│   │   ├── services/             # Servicios del gimnasio
│   │   └── auth/                 # Autenticación
│   ├── layout/                   # Componentes de layout
│   ├── rutinas/                  # Sistema de rutinas completo
│   ├── citas/                    # Sistema de citas y calendario
│   ├── mapa/                     # Mapa interactivo del gimnasio
│   ├── ui/                       # Componentes base del sistema de diseño
│   └── shared/                   # Componentes compartidos
├── lib/                          # Lógica de negocio y utilidades
│   ├── hooks/                    # Hooks personalizados
│   └── utils/                    # Funciones utilitarias
└── types/                        # Definiciones de tipos TypeScript
```

Uso del App Router de Next.js 13+. Se aprovechó completamente el nuevo sistema de routing que incluye:

- **Layouts anidados**: Implementación de layout.tsx diferenciados para secciones públicas y privadas
- **Server Components**: Optimización de rendimiento mediante renderizado del lado del servidor donde es apropiado
- **Streaming y Suspense**: Carga progresiva de contenido para mejorar perceived performance
- **Parallel Routes**: Para funcionalidades como modales y overlays

Patrón de componentes y reutilización. Se implementó un sistema de componentes de tres niveles:

1. **Componentes UI base**: Botones, inputs, modales con variantes estandarizadas
2. **Componentes de negocio**: Específicos del dominio como RutinaCard, CitaCard

3. **Componentes de página**: Agregadores que combinan múltiples componentes

Sistema de componentes y diseño

Implementación del sistema de diseño con Tailwind CSS. Se configuró Tailwind con colores corporativos personalizados que reflejan la identidad visual de FortFit:

```
:root {
  --background: #ffffff;
  --foreground: #1d1d1b;  /* fortfit-black */
  --accent: #9d9d9d;      /* fortfit-gray */
}

/* Clases de utilidad personalizadas para variaciones de opacidad */
.bg-fortfit-black-25 { background-color: rgba(29, 29, 27, 0.25); }
.bg-fortfit-gray-10 { background-color: rgba(157, 157, 157, 0.1); }
.text-fortfit-white { color: #ffffff; }
```

Componentes base reutilizables. Se desarrolló una librería completa de componentes UI que incluye:

- **Button**: Con variantes primary, secondary, outline y ghost, más estados de loading
- **Input**: Campos de texto, número, fecha con validación visual integrada
- **Card**: Contenedores estructurados con header, content y footer opcionales
- **Modal/Dialog**: Sistema modal con overlay y gestión de foco automática
- **Toast**: Notificaciones no intrusivas con tipos success, error, warning e info

Componentes específicos del dominio. Para cada módulo principal se desarrollaron componentes especializados:

- **RutinaCard**: Visualización de rutinas con progreso, dificultad y acciones rápidas
- **EjercicioViewer**: Detalle de ejercicios con imágenes, instrucciones y configuración
- **CitaCard**: Gestión visual de citas con estados y acciones contextuales
- **HistorialCard**: Visualización de sesiones completadas con métricas de rendimiento

Gestión de estado y datos

Hooks personalizados implementados. Se desarrolló un sistema comprensivo de hooks que encapsulan la lógica de negocio y gestión de estado:

```
// Hooks principales del gimnasio
export { usePlantas } from "./use-plantas";
export { useZonas } from "./use-zonas";
export { useEjercicios } from "./use-ejercicios";
export { useRutinas } from "./use-rutinas";
export { useRutina } from "./use-rutina";
export { useRutinaMutations } from "./use-rutina-mutations";
export { useHistorialEjecuciones } from "./use-historial-ejecuciones";
export { useRutinaExecution } from "./use-rutina-execution";
```

Hooks especializados por funcionalidad:

- **useRutinas**: Gestión del listado de rutinas con filtrado y paginación
- **useRutinaExecution**: Control completo del estado durante ejecución de rutinas, incluyendo timers, series completadas y métricas en tiempo real
- **useHistorialEjecuciones**: Manejo del historial con filtros avanzados por fecha, rutina y estado de completitud
- **useCitas**: Sistema completo de gestión de citas con validación de disponibilidad

- **useNotifications**: Gestión de notificaciones push con estado de lectura y persistencia

Context API para estado global. Se implementaron contextos específicos para datos que requieren acceso global:

```
// Context de usuario para información de perfil y autenticación
export const { useUserStore } = createUserStore();

// Context de título para gestión dinámica de títulos de página
export const { useTitle } = useTitleContext();
```

Gestión de formularios y validaciones. Los formularios implementan validación tanto del lado del cliente como integración con las validaciones del backend:

```
const [datosSerieModal, setDatosSerieModal] = useState<DatosSerieModal>({
  pesoUsado: "",
  repeticionesReales: "",
  notas: "",
});

// Validación en tiempo real con feedback visual inmediato
const validarFormulario = () => {
  const errores = {};
  if (!datos.nombre.trim()) errores.nombre = "Nombre requerido";
  if (datos.series < 1) errores.series = "Mínimo 1 serie";
  return errores;
};
```

Interfaces de usuario principales

Dashboard principal y navegación. El dashboard implementa un sistema de navegación adaptativo que se ajusta según el rol del usuario:

```
// Navegación específica por rol
const featureSections = [
  { href: "/dashboard/mapa", icon: Map, label: "Mapa", color: "text-orange-
500" },
  { href: "/dashboard/citas", icon: Calendar, label: "Citas", color: "text-
blue-500" },
  { href: "/dashboard/rutinas", icon: Dumbbell, label: "Rutinas", color:
"text-red-500" },
  { href: "/dashboard/biblioteca", icon: BookOpen, label: "Biblioteca",
color: "text-indigo-500" },
  // Opciones adicionales según rol de usuario
  ...(isSpecialist ? [{ href: "/dashboard/especialista", icon: Stethoscope,
label: "Especialista" }] : []),
```

```
  ...(isAdmin ? [{ href: "/dashboard/admin", icon: ShieldUser, label: "Admin"
}] : []),
];
```

Sistema de autenticación completo. Las interfaces de autenticación incluyen:

- **Login**: Con validación de bloqueo de cuenta y manejo de intentos fallidos
- **Cambio de contraseña obligatorio**: Para usuarios nuevos con validación robusta de política de contraseñas
- **Recuperación**: Gestionada por administradores con generación de contraseñas temporales

Gestión de rutinas y ejercicios. El módulo de rutinas proporciona:

- **Creador de rutinas**: Interfaz drag-and-drop para agregar ejercicios con configuración detallada
- **Ejecutor de rutinas**: Timer integrado, seguimiento de series y descansos, captura de datos reales de ejecución
- **Visualizador de ejercicios**: Detalles completos con imágenes, instrucciones paso a paso y información de máquinas

Sistema de citas y calendario. Implementa:

- **Calendario interactivo**: Vista mensual con disponibilidad de especialistas en tiempo real
- **Formulario de reserva**: Validación de conflictos y confirmación automática
- **Gestión de estados**: Seguimiento completo del ciclo de vida de citas desde solicitud hasta completitud

Funcionalidades interactivas avanzadas

Mapa interactivo SVG del gimnasio. Se desarrolló un sistema completo de mapa interactivo que incluye:

```
// Componente principal del mapa
export default function MapaContent() {
  const [plantaSeleccionada, setPlantaSeleccionada] = useState<Planta |
null>(null);
  const [zonaSeleccionada, setZonaSeleccionada] = useState<Zona |
null>(null);

  // Navegación automática desde otras secciones
  const plantaIdFromURL = searchParams.get("planta");

  // Interactividad con zonas SVG
  const handleZonaClick = (zona: Zona) => {
    setZonaSeleccionada(zona);
    setShowZonaInfo(true);
  };
}
```

Características del mapa:

- Visualización SVG responsive con zoom y navegación
- Interactividad de zonas con información detallada
- Integración con biblioteca de ejercicios para mostrar máquinas por zona
- Navegación cruzada desde rutinas para ubicar equipamiento

Sistema de notificaciones push. Implementación completa de Web Push API:

```
// Componente de instalación PWA
export function PWAInstaller() {
  const [installPrompt, setInstallPrompt] = useState<any>(null);
  const [showPrompt, setShowPrompt] = useState(false);

  // Detección de soporte y registro de service worker
  useEffect(() => {
    const handleBeforeInstallPrompt = (e: Event) => {
      e.preventDefault();
      setInstallPrompt(e);
      setShowPrompt(true);
    };
```

```
    window.addEventListener('beforeinstallprompt',
handleBeforeInstallPrompt);
  }, []);
}
```

Funcionalidades de notificaciones:

- Registro automático de suscripciones push
- Recordatorios de citas 24 horas y 2.5 horas antes
- Notificaciones de cambios de estado en citas
- Panel de configuración de preferencias de usuario

Formularios dinámicos y validación en tiempo real. Los formularios implementan validación progresiva:

```
// Validación en tiempo real durante ejecución de rutinas
const RutinaExecutor = ({ rutina }: RutinaExecutorProps) => {
  const [datosSerieModal, setDatosSerieModal] = useState({
    pesoUsado: "",
    repeticionesReales: "",
    notas: "",
  });

  // Validación inmediata con feedback visual
  const validarSerie = (datos: DatosSerieModal) => {
    const errores = {};
    if (!datos.pesoUsado || isNaN(Number(datos.pesoUsado))) {
      errores.peso = "Peso debe ser un número válido";
    }
    return errores;
  };
}
```

Responsive design y Progressive Web App

Implementación mobile-first. Todo el diseño se desarrolló con enfoque mobile-first, implementando breakpoints estratégicos:

```
/* Breakpoints implementados */
sm: '640px',   /* Tablets pequeñas */
md: '768px',   /* Tablets */
lg: '1024px',  /* Desktop pequeño */
xl: '1280px',  /* Desktop estándar */

/* Ejemplo de implementación responsive */
className="h-[300px] sm:h-[300px] group transition-all duration-300 transform
hover:scale-[1.02]"
```

Adaptación a diferentes tamaños de pantalla:

- **Móvil**: Navegación bottom-sheet, cards de ancho completo, gestos táctiles
- **Tablet**: Layout de dos columnas, navegación lateral opcional
- **Desktop**: Sidebar persistente, múltiples columnas, hover effects

Configuración de PWA. Se implementó PWA completa con:

```
// Configuración de PWA en next.config.js
const withPWA = require('next-pwa')({
  dest: 'public',
  register: true,
  skipWaiting: true,
  disable: process.env.NODE_ENV === 'development'
});
```

Características PWA implementadas:

- Manifest personalizado con iconos adaptativos
- Service Worker para caché offline
- Instalación en pantalla de inicio (Android/iOS)
- Actualizaciones automáticas con notificación al usuario
- Funcionalidad offline básica para vistas previamente cargadas

Optimización para dispositivos táctiles:

- Botones con tamaño mínimo de 44px para accesibilidad táctil
- Gestos swipe para navegación en carruseles
- Feedback táctil visual inmediato
- Espaciado adecuado entre elementos interactivos

Integración con backend y APIs

Consumo de APIs del backend. Se implementó un patrón consistente de consumo de APIs:

```
// Hook ejemplo para gestión de rutinas
export function useRutinas(filtros?: FiltrosRutinas) {
  const [rutinas, setRutinas] = useState<Rutina[]>([]);
  const [loading, setLoading] = useState(false);
  const [error, setError] = useState<string | null>(null);

  const cargarRutinas = useCallback(async () => {
    try {
      setLoading(true);
      setError(null);

      const params = new URLSearchParams();
      if (filtros?.busqueda) params.append("busqueda", filtros.busqueda);

      const response = await fetch(`/api/rutinas?${params}`);

      if (!response.ok) {
        throw new Error("Error al cargar rutinas");
      }

      const data = await response.json();
      setRutinas(data.data || []);
    } catch (err) {
      setError(err instanceof Error ? err.message : "Error desconocido");
    } finally {
      setLoading(false);
    }
  }, [filtros]);
}
```

Manejo de estados de carga y error. Todos los componentes implementan estados consistentes:

- **Loading**: Skeletons animados que mantienen la estructura del layout
- **Error**: Mensajes descriptivos con opciones de retry
- **Success**: Confirmaciones visuales no intrusivas
- **Empty**: Estados vacíos con llamadas a acción

Autenticación automática en requests. Las llamadas a APIs incluyen automáticamente:

```
// Middleware automático para autenticación
const response = await fetch('/api/rutinas', {
  headers: {
    'Content-Type': 'application/json',
    // Cookies de sesión se incluyen automáticamente
  },
  credentials: 'same-origin'
});
```

Sincronización de datos en tiempo real. Para datos críticos se implementó:

- Revalidación automática al enfocar ventanas
- Polling inteligente para notificaciones
- Invalidación de caché en mutaciones
- Optimistic updates con rollback en errores

Optimización y rendimiento

Code splitting y lazy loading. Se implementó carga diferida estratégica:

```
// Carga diferida de componentes pesados
const RutinaExecutor = lazy(() => import('@/components/rutinas/rutina-
executor'));
const MapaInteractivo = lazy(() => import('@/components/mapa/mapa-content'));

// Suspense boundaries para mejor UX
<Suspense fallback={<ComponentSkeleton />}>
```

```
  <RutinaExecutor rutina={rutina} />
</Suspense>
```

Optimización de imágenes y assets. Utilizando las capacidades nativas de Next.js:

```
// Componente Image optimizado
<Image
  src={backgroundImage}
  alt={servicio.nombre}
  fill
  sizes="(max-width: 768px) 100vw, 50vw"
  className="object-cover object-center"
  priority={isAboveFold}
/>
```

Estrategias de caché del cliente:

- **Static assets**: Caché indefinido con hashing de archivos
- **API responses**: Caché inteligente con revalidación basada en staleness
- **User data**: Invalidación inmediata en mutaciones
- **Images**: Lazy loading con intersección observer

Métricas de performance implementadas:

- Core Web Vitals tracking automático
- Medición de tiempo de respuesta de APIs
- Monitoreo de errores JavaScript
- Análisis de uso de memoria en componentes pesados

El desarrollo del frontend resultó en una aplicación moderna, responsive y altamente interactiva que proporciona una experiencia de usuario excepcional tanto en

dispositivos móviles como de escritorio, manteniendo excelentes métricas de rendimiento y accesibilidad.

Capítulo III: Pruebas y Estabilización

Introducción

El proceso de pruebas y estabilización del Sistema Web Interactivo para el Gimnasio FortFit representa una fase crítica para garantizar la calidad, funcionalidad y confiabilidad del producto final. Durante el desarrollo del sistema se realizaron pruebas por cada módulo implementado, sin embargo, esta fase establece un marco formal y estructurado para validar de manera completa todas las funcionalidades desarrolladas.

El enfoque de pruebas adoptado prioriza la verificación práctica de funcionalidades en condiciones reales de uso, considerando la diversidad de dispositivos y navegadores que utilizarán los usuarios finales del gimnasio FortFit. Las pruebas se organizan en múltiples niveles: desde la validación individual de cada módulo hasta la verificación de flujos completos de usuario, garantizando que el sistema cumpla con los requisitos funcionales y no funcionales establecidos.

Plan de Pruebas Generales

Objetivo de las pruebas

Las pruebas del sistema FortFit tienen como objetivo principal validar la funcionalidad, estabilidad y usabilidad de la aplicación web en diferentes escenarios de uso y configuraciones de dispositivos. Se busca identificar y corregir errores que puedan afectar la experiencia del usuario antes del despliegue en producción.

Estrategia de pruebas

Enfoque por módulos. Cada módulo del sistema (autenticación, citas, rutinas, biblioteca, mapa interactivo, administración, especialistas y notificaciones) será sometido a pruebas específicas que validen su funcionamiento individual y su integración con otros componentes.

Metodología de pruebas manuales. Se implementa un enfoque de pruebas manuales que simula el comportamiento real de los usuarios, permitiendo identificar problemas de usabilidad y experiencia que las pruebas automatizadas podrían pasar por alto.

Criterios de cobertura. Las pruebas cubrirán al menos el 90% de las funcionalidades implementadas, incluyendo todos los casos de uso principales y los flujos alternativos identificados en la fase de diseño.

Alcance de las pruebas

Módulos incluidos:

- Sistema de autenticación y gestión de usuarios
- Gestión de citas
- Biblioteca de ejercicios
- Sistema de rutinas personalizadas
- Mapa interactivo del gimnasio
- Panel administrativo
- Dashboard de especialistas

- Sistema de notificaciones

Tipos de usuarios:

- Clientes del gimnasio
- Especialistas (nutricionistas y fisioterapeutas)
- Administradores del sistema

Dispositivos y navegadores:

- Navegadores de escritorio (Chrome, Firefox, Safari, Edge)
- Dispositivos móviles (Android e iOS)
- Diferentes resoluciones de pantalla

Criterios de aceptación

Funcionalidad. Todas las funcionalidades deben operar según las especificaciones definidas en los casos de uso.

Usabilidad. La interfaz debe ser intuitiva y permitir que usuarios sin experiencia técnica puedan completar tareas básicas en menos de 3 intentos.

Rendimiento. Los tiempos de carga de páginas no deben exceder 3 segundos en conexiones estándar.

Compatibilidad. El sistema debe funcionar correctamente en al menos el 95% de las combinaciones dispositivo-navegador probadas.

Seguridad. Los mecanismos de autenticación y autorización deben prevenir accesos no autorizados en el 100% de los casos de prueba.

Pruebas Unitarias Funcionales

Módulo de Autenticación y Seguridad

Prueba A1: Inicio de sesión válido

- **ID:** A1
- **Objetivo:** Verificar que los usuarios puedan iniciar sesión con credenciales válidas
- **Datos de entrada:** Usuario registrado, contraseña correcta
- **Pasos a seguir:**
    1. Abrir la página principal del sistema
    2. Hacer clic en "Iniciar Sesión"
    3. Ingresar usuario
    4. Ingresar contraseña correcta
    5. Hacer clic en "Iniciar Sesión"
- **Resultado esperado:** El usuario es redirigido al dashboard correspondiente a su rol
- **Criterio de éxito:** Redirección exitosa y visualización del dashboard personalizado

Prueba A2: Bloqueo de cuenta por intentos fallidos

- **ID:** A2
- **Objetivo:** Verificar el sistema de bloqueo automático tras múltiples intentos fallidos
- **Datos de entrada:** Usuario válido, contraseña incorrecta (8 intentos)
- **Pasos a seguir:**
    1. Intentar iniciar sesión con contraseña incorrecta
    2. Repetir el proceso 8 veces consecutivas
    3. Verificar mensaje de bloqueo
    4. Intentar iniciar sesión con contraseña correcta
- **Resultado esperado:** La cuenta se bloquea permanentemente después del 8vo intento fallido
- **Criterio de éxito:** Mensaje de cuenta bloqueada y la imposibilidad de acceso

Prueba A3: Cambio de contraseña obligatorio

- **ID:** A3
- **Objetivo:** Verificar que los usuarios con contraseña temporal deben cambiarla
- **Datos de entrada:** Usuario que ingresa por primera vez
- **Pasos a seguir:**
    1. Iniciar sesión con contraseña temporal
    2. Verificar redirección automática

3. Ingresar nueva contraseña válida
4. Confirmar nueva contraseña
5. Guardar cambios

- **Resultado esperado:** Redirección forzosa al formulario de cambio de contraseña
- **Criterio de éxito:** Usuario puede acceder normalmente tras cambiar la contraseña

Módulo de Gestión de Citas

Prueba C1: Reserva de cita nueva

- **ID:** C1
- **Objetivo:** Verificar que los clientes puedan reservar citas con especialistas
- **Datos de entrada:** Usuario cliente autenticado, fecha y hora disponible
- **Pasos a seguir:**

1. Navegar a "Citas" desde el dashboard
2. Hacer clic en "Nueva Cita"
3. Seleccionar tipo de especialista (Nutrición/Fisioterapia)
4. Elegir fecha y hora disponible
5. Ingresar motivo de la consulta
6. Confirmar la reserva

- **Resultado esperado:** Cita creada con estado "pendiente" y notificación enviada al especialista
- **Criterio de éxito:** Cita visible en la lista de citas del cliente con estado correcto

Prueba C2: Confirmación de cita por especialista

- **ID:** C2
- **Objetivo:** Verificar que los especialistas puedan confirmar o rechazar citas
- **Datos de entrada:** Usuario especialista, cita pendiente de confirmación
- **Pasos a seguir:**
    1. Acceder al dashboard de especialista
    2. Revisar citas pendientes
    3. Seleccionar una cita pendiente
    4. Revisar detalles de la cita
    5. Agregar notas (opcional)
    6. Hacer clic en "Confirmar"
- **Resultado esperado:** Estado de cita cambia a "confirmada" y cliente recibe notificación
- **Criterio de éxito:** Cliente puede ver la cita confirmada en su dashboard

Prueba C3: Gestión de disponibilidad

- **ID:** C3
- **Objetivo:** Verificar que los especialistas puedan configurar su disponibilidad

- **Datos de entrada:** Usuario especialista, horarios de trabajo deseados
- **Pasos a seguir:**
    1. Acceder a "Gestión de Disponibilidad"
    2. Seleccionar días de la semana
    3. Configurar horarios de inicio y fin
    4. Guardar configuración
- **Resultado esperado:** Horarios configurados aparecen disponibles para reserva de clientes
- **Criterio de éxito:** Clientes pueden ver y reservar en los horarios configurados

Módulo de Rutinas Personalizadas

Prueba R1: Creación de rutina nueva

- **ID:** R1
- **Objetivo:** Verificar que los usuarios puedan crear rutinas personalizadas
- **Datos de entrada:** Usuario autenticado, información de rutina, ejercicios seleccionados
- **Pasos a seguir:**
    1. Navegar a "Rutinas"
    2. Hacer clic en "Crear Nueva Rutina"
    3. Ingresar nombre descriptivo y demás información
    4. Agregar ejercicios desde la biblioteca

5. Configurar series, repeticiones y peso por cada ejercicio
6. Guardar rutina

- **Resultado esperado:** Rutina creada exitosamente y visible en la lista personal
- **Criterio de éxito:** Rutina aparece en la lista y puede ser ejecutada

Prueba R2: Ejecución de rutina

- **ID:** R2
- **Objetivo:** Verificar que los usuarios puedan ejecutar rutinas y registrar progreso
- **Datos de entrada:** Rutina creada previamente, valores de ejecución
- **Pasos a seguir:**
    1. Seleccionar rutina desde la lista
    2. Hacer clic en "Ejecutar"
    3. Completar cada ejercicio registrando series y repeticiones
    4. Registrar tiempo de descanso
    5. Finalizar sesión
- **Resultado esperado:** Progreso registrado en el historial personal del usuario
- **Criterio de éxito:** Datos de ejecución visibles en historial y estadísticas

Módulo de Biblioteca de Ejercicios

Prueba B1: Búsqueda de ejercicios

- **ID:** B1
- **Objetivo:** Verificar la funcionalidad de búsqueda y filtrado de ejercicios

- **Datos de entrada:** Términos de búsqueda, filtros por grupo muscular
- **Pasos a seguir:**
    1. Acceder a "Biblioteca"
    2. Usar barra de búsqueda con término específico
    3. Aplicar filtros por grupo muscular
    4. Aplicar filtros por nivel de dificultad
    5. Aplicar filtros por máquina
    6. Revisar resultados
- **Resultado esperado:** Lista filtrada de ejercicios que coinciden con los criterios
- **Criterio de éxito:** Resultados relevantes y funciones de filtro operativas

Prueba B2: Visualización de detalles de ejercicio

- **ID:** B2
- **Objetivo:** Verificar que los usuarios puedan ver información detallada de ejercicios
- **Datos de entrada:** Ejercicio seleccionado de la biblioteca
- **Pasos a seguir:**
    1. Seleccionar un ejercicio de la lista
    2. Hacer clic para ver detalles
    3. Revisar descripción y técnica
    4. Verificar información de músculos trabajados

- **Resultado esperado:** Modal con información completa del ejercicio
- **Criterio de éxito:** Toda la información es visible

Módulo de Mapa Interactivo

Prueba M1: Navegación por plantas del gimnasio

- **ID:** M1
- **Objetivo:** Verificar la navegación entre diferentes plantas del gimnasio
- **Datos de entrada:** Usuario en la sección de mapa
- **Pasos a seguir:**
    1. Acceder al "Mapa Interactivo"
    2. Seleccionar planta
    3. Seleccionar zona
    4. Verificar carga de contenido
- **Resultado esperado:** Cambio de vista entre plantas con información actualizada
- **Criterio de éxito:** Navegación fluida y contenido específico por planta

Prueba M2: Información de máquinas y zonas

- **ID:** M2
- **Objetivo:** Verificar que se muestre información detallada de equipamiento
- **Datos de entrada:** Zona seleccionada en el mapa
- **Pasos a seguir:**
    1. Hacer clic en una zona del mapa

2. Revisar información desplegada
3. Verificar máquinas asociadas
4. Verificar ejercicios asociados a la máquina
5. Cerrar modal de información

- **Resultado esperado:** Modal con detalles completos de la zona y máquina seleccionada
- **Criterio de éxito:** Información precisa y opciones de interacción disponibles

Módulo Administrativo

Prueba AD1: Importación masiva de usuarios

- **ID:** AD1
- **Objetivo:** Verificar la funcionalidad de importación de usuarios desde Excel
- **Datos de entrada:** Archivo Excel con datos de usuarios válidos
- **Pasos a seguir:**
    1. Acceder al panel de administración
    2. Navegar a "Importar Usuarios"
    3. Seleccionar archivo Excel
    4. Revisar vista previa de proceso
    5. Confirmar importación
- **Resultado esperado:** Usuarios importados exitosamente al sistema
- **Criterio de éxito:** Usuarios visibles en la lista con credenciales generadas

Prueba AD2: Gestión de seguridad de cuentas

- **ID:** AD2
- **Objetivo:** Verificar que los administradores puedan gestionar bloqueos de cuenta
- **Datos de entrada:** Usuario con cuenta bloqueada
- **Pasos a seguir:**
    1. Acceder a "Seguridad" en el panel admin
    2. Localizar cuenta bloqueada
    3. Revisar historial de intentos
    4. Desbloquear cuenta
    5. Verificar que el usuario pueda acceder
- **Resultado esperado:** Cuenta desbloqueada y usuario puede iniciar sesión
- **Criterio de éxito:** Proceso de desbloqueo exitoso y acceso restaurado

Pruebas de Integración

Integración entre Autenticación y Dashboard

Prueba I1: Flujo completo de primer acceso

- **ID:** I1
- **Objetivo:** Verificar el flujo completo desde primer login hasta dashboard funcional
- **Datos de entrada:** Usuario nuevo con contraseña temporal

- **Pasos a seguir:**

    1. Iniciar sesión con credenciales temporales
    2. Completar cambio de contraseña obligatorio
    3. Verificar acceso a dashboard
    4. Probar navegación entre módulos

- **Resultado esperado:** Usuario accede exitosamente a todas las funcionalidades según su rol
- **Criterio de éxito:** Navegación fluida y acceso apropiado a funcionalidades

Prueba I2: Flujo de reserva y notificaciones

- **ID:** I2
- **Objetivo:** Verificar que las notificaciones se envíen correctamente en el proceso de citas
- **Datos de entrada:** Cliente y especialista en el sistema
- **Pasos a seguir:**

    1. Cliente reserva nueva cita
    2. Verificar notificación al especialista
    3. Especialista confirma la cita
    4. Verificar notificación al cliente
    5. Comprobar actualización en ambos dashboards

- **Resultado esperado:** Ambos usuarios reciben notificaciones apropiadas en tiempo real
- **Criterio de éxito:** Sincronización perfecta entre estados de cita y notificaciones

Prueba I3: Creación de rutina con ejercicios de biblioteca

- **ID:** I3
- **Objetivo:** Verificar la integración entre biblioteca de ejercicios y creación de rutinas
- **Datos de entrada:** Usuario autenticado, ejercicios disponibles en biblioteca
- **Pasos a seguir:**
    1. Iniciar creación de nueva rutina
    2. Buscar ejercicios en la biblioteca integrada
    3. Agregar múltiples ejercicios
    4. Configurar parámetros para cada ejercicio
    5. Guardar rutina completa
- **Resultado esperado:** Rutina creada con todos los ejercicios y configuraciones
- **Criterio de éxito:** Ejercicios mantienen información detallada en la rutina

Prueba I4: Persistencia de datos en tiempo real

- **ID:** I4
- **Objetivo:** Verificar que los cambios se reflejen inmediatamente en la base de datos

- **Datos de entrada:** Múltiples usuarios realizando acciones simultáneas
- **Pasos a seguir:**
    1. Usuario A crea una rutina
    2. Usuario B reserva una cita
    3. Especialista confirma la cita
    4. Usuario A ejecuta la rutina
    5. Verificar todos los datos en Supabase
- **Resultado esperado:** Todos los cambios se persisten correctamente sin conflictos
- **Criterio de éxito:** Consistencia de datos y actualizaciones en tiempo real

Pruebas de Compatibilidad de Dispositivos

Navegadores de Escritorio

Prueba D1: Google Chrome

- **ID:** D1
- **Navegador:** Google Chrome (Versión 131+)
- **Sistema Operativo:** Windows 11
- **Resoluciones probadas:** 1920x1080, 1366x768, 2560x1440
- **Funcionalidades a probar:**
    - Inicio de sesión
    - Navegación entre módulos

    - Creación de rutinas
    - Reserva de citas
    - Mapa interactivo
    - PWA (instalación)
- **Resultado esperado:** Funcionamiento completo sin errores de compatibilidad
- **Notas especiales:** Verificar funcionalidades PWA y notificaciones push

Prueba D2: Mozilla Firefox

- **ID:** D2
- **Navegador:** Mozilla Firefox (Versión 133+)
- **Sistema Operativo:** Windows 11
- **Resoluciones probadas:** 1920x1080, 1366x768
- **Funcionalidades a probar:**
    - Todas las funcionalidades principales
    - Interacciones del mapa SVG
    - Formularios
- **Resultado esperado:** Compatibilidad completa con diferencias visuales mínimas
- **Notas especiales:** Verificar CSS Grid y Flexbox en componentes complejos

Prueba D3: Safari

- **ID:** D3
- **Navegador:** Safari (Versión 18+)
- **Sistema Operativo:** macOS
- **Resoluciones probadas:** 1920x1080, 2560x1440 (Retina)
- **Funcionalidades a probar:**
    - Autenticación con Supabase
    - Todas las funcionalidades principales
    - Funcionalidades PWA
- **Resultado esperado:** Funcionamiento correcto con consideraciones específicas de Safari
- **Notas especiales:** Verificar compatibilidad de Web APIs y PWA en Safari

Prueba D4: Microsoft Edge

- **ID:** D4
- **Navegador:** Microsoft Edge (Versión 131+)
- **Sistema Operativo:** Windows 11
- **Resoluciones probadas:** 1920x1080, 1366x768
- **Funcionalidades a probar:**
    - Todas las funcionalidades del sistema
    - Integración con Windows

    - PWA y notificaciones nativas

- **Resultado esperado:** Excelente compatibilidad similar a Chrome
- **Notas especiales:** Aprovechar integraciones específicas de Windows

Dispositivos Móviles

Prueba M1: Android (Chrome Mobile)

- **ID:** M1
- **Dispositivo:** Samsung Galaxy S23, Google Pixel 7 (Emulador en Android Studio)
- **Navegador:** Chrome Mobile
- **Resoluciones:** 1080x2340, 1440x3120
- **Funcionalidades críticas:**
    - Navegación táctil
    - Formularios en pantalla pequeña
    - Mapa interactivo táctil
    - PWA (Instalación)
    - Notificaciones push
    - Modo offline básico
- **Resultado esperado:** Experiencia móvil completamente funcional
- **Criterios específicos:**
    - Botones táctiles suficientemente grandes

    - Scroll fluido

Prueba M2: iOS (Safari Mobile)

- **ID:** M2
- **Dispositivo:** iPhone 14, iPhone 15 Pro (Emulador en XCode)
- **Navegador:** Safari Mobile
- **Resoluciones:** 1170x2532, 1290x2796
- **Funcionalidades críticas:**
    - Mismas funcionalidades que Android
    - Integración con iOS
    - PWA (Agregar a pantalla de inicio)
    - Notificaciones
- **Resultado esperado:** Funcionamiento óptimo respetando limitaciones de iOS
- **Criterios específicos:**
    - Respeto por safe areas
    - Gestos nativos de iOS
    - Limitaciones PWA específicas

Pruebas de Rendimiento y Estabilización

Rendimiento de Carga

Prueba P1: Tiempo de carga inicial

- **ID:** P1

- **Objetivo:** Medir tiempos de carga de la aplicación
- **Conexiones probadas:**
    - Fibra óptica (100+ Mbps)
    - 4G (20-50 Mbps)
    - 3G (1-5 Mbps)
- **Métricas a medir:**
    - First Contentful Paint (FCP)
    - Largest Contentful Paint (LCP)
    - Time to Interactive (TTI)
    - Cumulative Layout Shift (CLS)
- **Páginas críticas:** Landing page, Dashboard, Biblioteca, Mapa
- **Objetivos de rendimiento:**
    - FCP < 1.5s
    - LCP < 2.5s
    - TTI < 3s
    - CLS < 0.1

Prueba P2: Rendimiento con múltiples usuarios

- **ID:** P2
- **Objetivo:** Evaluar el comportamiento del sistema con carga simultánea
- **Escenarios de carga:**

    - 10 usuarios simultáneos
    - 25 usuarios simultáneos
    - 50 usuarios simultáneos
- **Acciones simultáneas:**
    - Inicios de sesión
    - Creación de rutinas
    - Reserva de citas
    - Navegación en el mapa
- **Métricas a monitorear:**
    - Tiempo de respuesta de APIs
    - Errores de base de datos
    - Estabilidad de la aplicación

Estabilidad PWA

Prueba PWA1: Instalación y funcionamiento offline

- **ID:** PWA1
- **Objetivo:** Verificar funcionalidades Progressive Web App
- **Dispositivos de prueba:** Android, iOS, Windows, macOS
- **Funcionalidades PWA:**
    - Instalabilidad desde navegador
    - Funcionamiento offline básico

    - Caché de recursos estáticos
    - Splash screen personalizada
    - Iconos adaptativos
- **Pasos de prueba:**
    - Instalar PWA desde navegador
    - Verificar icono en pantalla de inicio
    - Abrir aplicación sin conexión
    - Verificar sincronización al reconectar
- **Resultado esperado:** PWA funciona correctamente como aplicación nativa

Prueba S1: Autenticación y autorización

- **ID:** S1
- **Objetivo:** Verificar robustez del sistema de seguridad
- **Escenarios de prueba:**
    - Intentos de acceso sin autenticación
    - Acceso a recursos de otros usuarios
    - Manipulación de URLs protegidas
    - Inyección de datos maliciosos
    - Escalación de privilegios
- **Herramientas:**
    - Navegador en modo incógnito

    - Herramientas de desarrollador
    - Modificación manual de requests
- **Resultado esperado:** Sistema rechaza todos los intentos de acceso no autorizado

Prueba E1: Carga de datos masivos

- **ID:** E1
- **Objetivo:** Evaluar el comportamiento con grandes volúmenes de datos
- **Escenarios:**
    - Importación de 1000+ usuarios
    - Biblioteca con 500+ ejercicios
    - Usuario con 100+ rutinas
    - 500+ citas en el sistema
- **Métricas a evaluar:**
    - Tiempo de carga de listas
    - Funcionalidad de búsqueda
    - Paginación
- **Resultado esperado:** Sistema mantiene responsividad con grandes volúmenes

Resultados de las Pruebas

Resumen de Ejecución

Durante la fase de pruebas se ejecutaron un total de 49 casos de prueba distribuidos en 5 categorías principales. El proceso de pruebas se llevó a cabo durante un período de 2 semanas, involucrando a 3 evaluadores para garantizar la objetividad de los resultados.

Distribución de pruebas:

- Pruebas unitarias funcionales: 20 casos
- Pruebas de integración: 4 casos
- Pruebas de compatibilidad: 16 casos
- Pruebas de rendimiento: 6 casos
- Pruebas de seguridad: 3 casos

Tasa de Éxito

- Pruebas exitosas: 47 de 49 casos (95.9%)
- Pruebas con incidencias menores: 2 casos (4.1%)
- Pruebas fallidas: 0 casos (0%)

Incidencias Identificadas y Resoluciones

Incidencia menor 1: Sobreposición de elementos en resoluciones muy pequeñas

Descripción: En dispositivos con pantallas muy pequeñas, como el iPhone X, se encontró que ciertos elementos de la interfaz como títulos se desbordaban de sus

contenedores y se sobreponían sobre otros elementos de su alrededor, lo que causaba un problema de experiencia de usuario.

Solución implementada: Se acortó ciertos títulos que eran bastante largos para poder acomodar su longitud en diferentes tamaños de pantallas.

Incidencia menor 2: PWA en Safari iOS

Descripción: Debido a que iOS no proporciona una forma confiable de cómo detectar si una PWA ha sido instalada en el dispositivo, el mensaje que se mostraba para invitar a los usuarios a instalar la app se mostraba incluso cuando la app ya había sido instalada.

Solución implementada: Se cambió el mensaje mostrado específicamente para iOS para incluir un botón de "No volver a mostrar" y de esta manera mejorar la experiencia del usuario.

Métricas de Rendimiento Alcanzadas

Tiempo de carga promedio:

- Landing page: 1.49s (objetivo: <1.5s)
- Dashboard: 1.73s (objetivo: <2.5s)
- Biblioteca: 1.72s (objetivo: <2.5s)
- Mapa interactivo: 1.52s (objetivo: <3s)

Core Web Vitals:

- First Contentful Paint: 0.6s
- Largest Contentful Paint: 1.1s
- Cumulative Layout Shift: 0

Compatibilidad de navegadores:

- Chrome: 100% funcional
- Firefox: 100% funcional
- Safari: 98% funcional (limitaciones PWA conocidas)
- Edge: 100% funcional

Recomendaciones de Estabilización

Monitoreo continuo. Se recomienda implementar un sistema de monitoreo de errores en producción utilizando herramientas como Sentry o LogRocket para identificar problemas en tiempo real.

Optimización progresiva. Continuar optimizando el rendimiento mediante técnicas como code splitting más granular y lazy loading de componentes no críticos.

Actualización de dependencias. Mantener las dependencias actualizadas, especialmente Next.js y Supabase, para aprovechar mejoras de rendimiento y seguridad.

Validación con Usuarios Finales

Metodología de Validación

Se realizó una sesión de validación con usuarios reales del gimnasio FortFit, incluyendo:

- 5 clientes del gimnasio (diferentes rangos etarios)
- 2 especialistas (nutricionista y fisioterapeuta)
- 1 administrador del gimnasio

Tareas de Validación

Para clientes:

1. Iniciar sesión por primera vez
2. Ver datos del perfil personal
3. Explorar la biblioteca de ejercicios
4. Crear una rutina personalizada
5. Reservar una cita con especialista
6. Navegar por el mapa del gimnasio

Para especialistas:

1. Configurar disponibilidad de horarios
2. Confirmar y gestionar citas
3. Revisar información de pacientes
4. Utilizar el calendario semanal

Para administradores:

1. Importar usuarios desde Excel
2. Gestionar cuentas bloqueadas
3. Monitorear actividad del sistema

Resultados de Validación

- Satisfacción general: 4.6/5 puntos promedio
- Facilidad de uso: 4.4/5 puntos promedio

- Utilidad percibida: 4.8/5 puntos promedio

Comentarios destacados:

- "La aplicación es muy intuitiva, pude reservar mi cita sin problemas"
- "Me encanta poder crear mis propias rutinas"
- "Como fisioterapista, me facilita mucho la gestión de mis horarios"
- "La importación de usuarios automática nos ahorrará mucho tiempo"

Sugerencias de Mejora Identificadas

1. **Notificaciones recordatorio**: Los usuarios solicitaron recordatorios automáticos para reserva de citas sugeridas
2. **Estadísticas avanzadas**: Interés en gráficos de progreso más detallados para monitoreo de administración
3. **Integración social**: Posibilidad de compartir rutinas entre usuarios

Conclusiones del Proceso de Pruebas

El proceso de pruebas y estabilización del Sistema Web Interactivo para el Gimnasio FortFit ha demostrado que la aplicación desarrollada cumple satisfactoriamente con los requisitos funcionales y no funcionales establecidos en la fase de diseño.

Fortalezas identificadas

- Alta compatibilidad entre dispositivos y navegadores
- Rendimiento de acuerdo a los objetivos establecidos
- Seguridad robusta con protecciones multicapa

- Experiencia de usuario intuitiva y satisfactoria
- Arquitectura escalable que soporta crecimiento futuro

Áreas de oportunidad

- Optimización adicional para dispositivos de recursos limitados
- Expansión de funcionalidades PWA en ecosistemas móviles
- Implementación de funcionalidades adicionales sugeridas por usuarios

La tasa de éxito del 95.9% en las pruebas realizadas, combinada con la alta satisfacción de los usuarios finales (4.6/5), confirma que el sistema está listo para su implementación en producción y cumplirá efectivamente con las necesidades del gimnasio FortFit y sus usuarios.

El sistema desarrollado representa una solución tecnológica robusta que moderniza significativamente la experiencia digital del gimnasio, estableciendo una base sólida para futuras expansiones y mejoras del servicio.

Conclusiones

El desarrollo del sistema web interactivo para FortFit permitió crear una solución tecnológica que integra herramientas digitales modernas con el objetivo de optimizar la experiencia del usuario y automatizar procesos clave dentro del gimnasio. A través de la aplicación de una metodología de levantamiento de requisitos precisa y un diseño arquitectónico modular, se lograron cubrir las necesidades detectadas en la fase de análisis, cumpliendo tanto el objetivo general como los objetivos específicos establecidos en el proyecto.

La implementación de módulos como la gestión de citas, la administración de rutinas y el acceso a una biblioteca de ejercicios categorizada ha reducido de manera significativa la carga operativa del personal administrativo y técnico. Esto ha permitido liberar recursos humanos para actividades de mayor valor añadido, mientras que el uso de tecnologías como Next.js y Supabase garantiza un alto rendimiento, seguridad y escalabilidad futura del sistema.

En cuanto a la experiencia de usuario, la incorporación de interfaces responsivas, el mapa interactivo de instalaciones y la optimización para dispositivos móviles han facilitado el acceso a la información y los servicios desde cualquier lugar, mejorando la satisfacción y fidelización de los miembros del gimnasio. Además, la nueva presencia digital posiciona a FortFit de forma competitiva en el mercado local, acercándolo a los estándares de innovación tecnológica que ya aplican sus principales competidores y abriendo el camino para futuras mejoras e integraciones.

Recomendaciones

Se recomienda fortalecer la estrategia de contenido digital mediante la creación y actualización constante de la biblioteca de ejercicios con material profesional, así como la generación de contenidos educativos y promocionales que incrementen la interacción de los usuarios con la plataforma. Este esfuerzo debe ir acompañado de la integración de pasarelas de pago seguras y de un sistema de facturación electrónica automatizada, de manera que el cliente pueda realizar todos sus trámites directamente desde el sistema sin necesidad de recurrir a procesos manuales o presenciales.

Asimismo, es aconsejable potenciar el sistema actual de notificaciones push para que, además de los recordatorios de citas, incluya alertas personalizadas sobre promociones, nuevos contenidos, eventos y cambios relevantes en el servicio.

La adopción de un plan de seguimiento con métricas de uso, tiempos de respuesta y satisfacción de los usuarios permitirá detectar de forma proactiva áreas de mejora y garantizar una evolución continua del sistema.

Finalmente, para mantener la propuesta de valor diferenciada en el mercado, se sugiere explorar a mediano plazo la incorporación de tecnologías emergentes como inteligencia artificial para el análisis de rendimiento físico, herramientas de gamificación para fomentar el compromiso de los usuarios y compatibilidad con dispositivos wearables.

Estas innovaciones no solo incrementarían el atractivo del sistema, sino que también reforzarían la imagen de FortFit como un gimnasio innovador y alineado con las tendencias globales del sector fitness.

Referencias

Aws. (2024). *Row-level security recommendations - aws prescriptive guidance*. Amazon web services. Https://docs.aws.amazon.com/prescriptive-guidance/latest/saas-multitenant-managed-postgresql/rls.html

Brainhub. (2024). *Pwa on ios*. Brainhub library. Https://brainhub.eu/library/pwa-on-ios

Creating a stakeholder engagement approach. (s. F.). Scrum.org. Https://www.scrum.org/resources/creating-stakeholder-engagement-approach

Falardeau, e., glynn, j., & ostromecka, o. (2021). *Sweating for the fitness consumer*. Mckinsey & company. Https://www.mckinsey.com/industries/consumer-packaged-goods/our-insights/sweating-for-the-fitness-consumer

Health & fitness association. (2020). *Digital transformation becomes a lifeline for health clubs*. Asociación de salud y forma física. Https://es.healthandfitness.org/improve-your-club/digital-transformation-becomes-a-lifeline-for-health-clubs

Health & fitness association. (2023). *Ihrsa releases 2023 u.s. health & fitness consumer report*. Https://www.healthandfitness.org/about/media-center/press-releases/ihrsa-releases-2023-u-s-health-fitness-consumer-report

Heroku. (2024). *Postgresql concurrency with mvcc*. Heroku dev center. Https://devcenter.heroku.com/articles/postgresql-concurrency

Ibm. (2024). *Postgresql vs. Mysql: what's the difference?* Ibm think. Https://www.ibm.com/think/topics/postgresql-vs-mysql

Lopez, j. (2024). From dumbbells to data: the role of technology in modern gyms - arch amenities group. *Arch amenity group*. Https://archamenity.com/future-of-fitness-technology-management/

M, a. (2023). *Gym online presence: expert tips for a winning marketing strategy*. Fitsw. Https://www.fitsw.com/blog/gym-online-presence-expert-tips-for-a-winning-marketing-strategy/

Más de 8 de cada 10 inscripciones a gimnasios se realizan por internet. (2024). Mercado fitness. Https://mercadofitness.com/8-de-cada-10-inscripciones-gimnasios-realizan-internet/

Microsoft learn. (2024). *Progressive web apps on microsoft edge*. Microsoft documentation. Https://learn.microsoft.com/en-us/microsoft-edge/progressive-web-apps-chromium/ux

Mozilla developer network. (2024). *Making pwas installable*. Mdn web docs. Https://developer.mozilla.org/en-us/docs/web/progressive_web_apps/guides/making_pwas_installable

Munkholm, l. (2025). *How can integrating automation tools improve your gym management experience?* Walla: fitness studio software. Https://www.hellowalla.com/blog/integrating-automation-tools-for-improved-gym-management-experience

Obregon, s. (2024). *Impact of digital fitness solutions on industry growth*. Fitness on demand. Https://www.fitnessondemand247.com/news/fitness-industry-growth

Phisique. (2024). Phisique app [aplicación móvil]. Google play. Https://play.google.com/store/apps/details?id=com.trainerize.phisiqueonline

Postgresql global development group. (2024). *Postgresql: documentation: 17: 8.14. Json types*. Postgresql.org. Https://www.postgresql.org/docs/current/datatype-json.html

Restrepo, s., moya, r., alba, f., torres, a., paredes, i., & mena, l. (2024). *Plan de negocios del gimnasio innovador fitness future, para el año 2024*. Https://repositorio.uide.edu.ec/handle/37000/7410

Salunke, s., & ouda, a. (2024). A performance benchmark for the postgresql and mysql databases. *Future internet*, 16(10), 382. Https://doi.org/10.3390/fi16100382

Schwaber, k., & sutherland, j. (2020). *The scrum guide*. Scrum guides. Https://scrumguides.org/scrum-guide.html

Smart fit. (s.f.). Smart fit app. Https://www.smartfit.com.ec/smartfitapp

Supabase. (2024). *Architecture | supabase docs*. Supabase documentation. Https://supabase.com/docs/guides/getting-started/architecture

Verwijs, c., & russo, d. (2022). A theory of scrum team effectiveness. *Acm transactions on software engineering and methodology*, *32*(3), 1-51. Https://doi.org/10.1145/3571849

Web.dev. (2024). *Progressive web apps*. Google web developers. Https://web.dev/learn/pwa/progressive-web-apps/

Wikipedia. (2024). *Progressive web app*. Wikipedia, the free encyclopedia. Https://en.wikipedia.org/wiki/progressive_web_app