

Fit Track

Equipo JetLag



ÍNDICE

Descubrimiento.....	4
Definición.....	5
Visión/ Propósito.....	5
¿Por qué lo hacemos?.....	5
¿Qué hacemos?.....	5
¿Cómo lo hacemos?.....	6
Propuesta de Valor.....	6
Estudio de competencia.....	8
Competidores directos:.....	8
Competidores indirectos:.....	9
Análisis DAFO.....	11
PRD (Product Requirements Document).....	12
Casos de uso principales.....	12
Scope inicial.....	12
Funcionalidad esencial.....	12
Criterios de aceptación.....	12
Calidad técnica mínima.....	13
Requerimientos no funcionales.....	13
MVP (Minimum Viable Product).....	13
Exclusiones del MVP.....	14
Funcionalidades incluidas.....	14
Diagramas UML.....	16
Escalabilidad inicial pensada.....	16
Definición del cliente.....	17
Usuarios principales.....	17
¿Qué piensan los usuarios acerca del producto?.....	18
Color.....	19
Recursos técnicos.....	20
Presupuesto.....	21
Monetización.....	23
Estrategias de monetización:.....	23
Planes.....	23
1. Versión gratuita (Free).....	23
2. Versión de Pago - Plan Pro.....	23

3. Versión de Pago - Plan Premium (Ilimitado).....	24
Seguimiento de progreso (Diferencias entre planes).....	25
White-Label (Marca Blanca).....	26
Ejemplo práctico:.....	26
Relación con los planes.....	27
Ventajas de incluir White-label.....	27
Gestión de proyecto.....	28
Escalabilidad y alcance.....	29
Herramientas y tecnologías:.....	32

Descubrimiento

De acuerdo con la pirámide de **MASLOW** una necesidad es aquello que debemos satisfacer para poder existir y desarrollarnos de manera plena por lo que entre tres ideas de proyecto descubrimos que queríamos ofrecer esa satisfacción a los entrenadores del mundo fitness.

Hoy, aplicaciones como *Trainerize* o *MyFitnessPal* no cubren el seguimiento técnico personalizado de entrenadores hacia clientes, por ejemplo de sus rutinas y resultados. En el mercado sólo existen soluciones individuales que abarcan más que eso y las cuales analizaremos más adelante para destacar nuestra propuesta de valor y cómo nos diferenciaremos de ello.

Lo primero que hicimos fue entender el problema que queríamos resolver usando la herramienta de **NEED STATEMENT DEFINITION**:

Usuario: Entrenadores de gimnasio

Necesidad: Crear y asignar rutinas personalizadas de forma eficiente a varios clientes por separado

Motivación: Enfocarse en la técnica y mejora continua de cada cliente

“Cuando Marbella está planeando rutinas para sus clientes quiere poder hacerlo más rápido y personalizado para poder dedicar más tiempo a corregir técnica y fomentar la mejora continua de cada cliente.”

Con esta frase de necesidad definimos y alineamos el problema que tienen los usuarios y que nuestro producto buscará resolver.

Definición

Nuestra misión es construir una herramienta para profesionales del mundo Fitness que cumpla los siguientes objetivos:

1. Agilizar la creación de rutinas personalizadas.
2. Centralizar el control y seguimiento de clientes.
3. Proveer métricas claras de progreso.

Visión/ Propósito

¿Por qué lo hacemos?

Somos un equipo de desarrollo enfocado en brindar soluciones para el ámbito del fitness y entrenamiento personalizado. Creemos en la importancia de la innovación para hacer que el trabajo de los entrenadores sea posible hacerse en un espacio amigable, simple y completo de usar. Ayudamos a los entrenadores a enfocarse en lo que más importa: la **técnica** y la **mejora continua**.

¿Qué hacemos?

Desarrollamos una aplicación diseñada para profesionales de la salud deportiva, que permite el monitoreo y seguimiento de rutinas personalizadas de cada cliente, facilitando la gestión del progreso. Reducimos tiempo y esfuerzo en creación y asignación de rutinas personalizadas según las necesidades fisiológicas de cada persona.

¿Cómo lo hacemos?

Lo lograremos mediante:

- Una plataforma intuitiva y fácil de usar, simplificando la gestión diaria del entrenador.
- Un sistema de seguimiento de datos (peso, cargas (volumen, esfuerzo percibido), repeticiones, progreso físico) con gráficas y analítica avanzada.
- Herramientas para crear rutinas personalizadas.
- Modelos de monetización flexibles (free, pro y premium) adaptados a las necesidades de cada entrenador.
- Un enfoque B2B (Business to Business) dirigido a los profesionales del fitness que buscan ofrecer un servicio de mayor calidad y valor a sus clientes

Ofrecemos un sistema simple y eficiente de control y seguimiento individual de clientes, rutinas y resultados.

En resumen los entrenadores requieren una herramienta que agilice la planificación y asignación de rutinas según las necesidades fisiológicas de cada cliente, con el fin de optimizar el tiempo de los entrenadores y puedan enfocarse en la calidad del entrenamiento y los resultados.

Propuesta de Valor

Nuestra aplicación está diseñada específicamente para profesionales de la salud deportiva, en un primer momento para entrenadores y coaches y más adelante para profesionales de otros campos.

Nuestra idea es brindarles una herramienta digital que les permita monitorizar y dar seguimiento en detalle a cada uno de sus clientes de manera personalizada.

A diferencia de las demás apps fitness que se orientan al usuario final, nuestra propuesta se centra en el coach como el protagonista, ofreciendole:

1. Gestión de clientes: cada coach puede organizar y administrar a todos sus clientes, con rutinas, objetivos y métricas individualmente.
2. Seguimiento personalizado: acceso al historial de entrenamientos, progresos, PRs (Personal Records) y adición de entrenamientos hechos en tiempo real.
3. Optimización del tiempo del entrenador: centralización de toda la información en una sola aplicación, evitando hojas de cálculo o apps no diseñadas para coaching.
4. Integración con el ecosistema Apple en un futuro (HealthKit, AppleWatch): obtención automática de datos de actividad y rendimiento para enriquecer el seguimiento de cada cliente.

En definitiva nuestra propuesta de valor se basa en dar a los entrenadores una herramienta profesional que les permita gestionar, monitorear y personalizar el entrenamiento de cada cliente de manera eficiente, organizada y tecnológica, replicando la **cercanía del entrenamiento presencial pero con la escalabilidad y comodidad de lo digital.**

ANÁLISIS FUNCIONAL

Estudio de competencia

Competidores directos:

- Trainerize
 - Plataforma todo en uno para entrenadores: creación de programas personalizados, nutrición, hábitos, comunicaciones (mensajes, video), reservas, pagos, venta de servicios y branding personalizado.
 - Ofrece Custom Branded Apps para coaches bajo planes Pro, Studio o Enterprise.
- My PT Hub
 - App para Coaches, usada para seguir el progreso de entrenamiento y nutrición de sus clientes.
 - Solución completa: creación y seguimiento de rutinas y nutrición, gestión de clientes, check-ins automáticos, calendarios, análisis financiero, branding completo e icono personalizado.
 - Permite clientes ilimitados con más de 15 funcionalidades avanzadas para coaches.
- JEFIT
 - Modo Coach (Coach Mode)
 - Ofrece un dashboard completo para entrenadores: envío y edición de rutinas personalizadas
 - Monitoreo del progreso de los clientes
 - Comunicación in-app.
 - Más de 1,300–1,400 ejercicios con videos HD, explicaciones y animaciones.

Competidores indirectos:

- Fitbod
 - Rutinas automáticas personalizadas según objetivos y equipo disponible.
 - Algoritmo de progresión adaptativa (recomienda cargas y reps).
Seguimiento de fuerza y métricas de rendimiento.
 - Integración con Apple Health.
- Strong
 - Registro rápido de entrenamientos (peso, reps, sets).
Gráficas de progreso sencillas.
Interfaz limpia y fácil de usar.
- Hevy
 - Seguimiento simple y rápido de rutinas.
 - Estadísticas de progreso.
 - Funciones sociales (compartir rutinas con amigos).
 - Sin tanta complejidad, orientada a la simplicidad.

Estudio de competencias			
	App	Ventaja	Desventajas
Directos	Trainerize	Plataforma integral (entrenos, nutrición, pagos, comunicación). Custom branded apps para coaches grandes.	Costosa \$22 USD/Mes aprox por tener máx 5 clientes. Compleja para entrenadores pequeños o individuales. Versión gratuita solo 1 cliente.
	My PT Hub	Cientes ilimitados. Gestión avanzada: rutinas, nutrición, calendario, finanzas. Branding completo.	Sobrecarga de funciones que puede abrumar. Menos intuitiva para clientes principiantes.
	JEFIT (Coach Mode)	Gran base de datos de ejercicios (1,300+). Dashboard para rutinas y progreso. Comunicación in-app básica.	Enfoque menos centrado en coaches que Trainerize/My PT Hub. Interfaz percibida como anticuada. No existe versión gratuita (cobra 16\$USD/Mes por máx 5 clientes)
Indirectos	Fitbod	Algoritmo adaptativo y rutinas personalizadas. Integración con Apple Health.	Enfoque 100% usuario final (no pensado para coaches). No permite gestión de clientes múltiples.
	Strong	Simple, rápida y fácil de usar. Buenas gráficas de progreso.	Muy limitada en funciones avanzadas. No incluye nutrición ni comunicación.
	Hevy	Orientada a la simplicidad. Funciones sociales atractivas.	Sin enfoque en coaching. Pocas herramientas de personalización avanzada.

Análisis DAFO

Fortalezas (internas, positivas)

- App especializada en entrenadores, coaching personalizado, no solo rutinas individuales genéricas.
- Uso de arquitectura CLEAN y MVVM, Server-side en Vapor + Fluent - escalabilidad y mantenibilidad.
- Swift/SwiftUI - Interfaz moderna, fluida y multiplataforma.
- Rutinas específicas, seguimiento específico y personalizado para cada uno de los clientes por separado.
- Diferenciación frente a aplicaciones de rutinas y seguimiento individual.
- Precios competitivos

Debilidades (internas, negativas)

- Requiere adopción tecnológica: coaches y especialistas deben familiarizarse con la app.
- Inicialmente limitado al ecosistema Apple.
- Compatibilidad limitada, solo disponible en iOS 17 en adelante

Oportunidades (externas, positivas)

- Diferenciación de Google sheets, excel etc. Donde los entrenadores y coaches hacen su seguimiento de entrenamiento de clientes, ahora con una app amigable y eficaz.
- Expansión hacia nutrición, fisioterapia y hábitos saludables.
- Para coaches que busquen una app intuitiva para el seguimiento profesional desde el móvil.

Amenazas (externas, negativas)

- Riesgo de poca diferenciación a pesar del valor añadido de experiencia personalizada, monitoreo y seguimiento de entrenadores para sus clientes.
- Basta competencia de apps fitness, depende mucho de sus diferenciadores.

Conclusiones del DAFO

- Personalización real (no sólo rutinas pre armadas).
- Relación directa coach-usuario como diferenciador frente a apps masivas.

PRD (Product Requirements Document)

Las funcionalidades que nuestro producto requiere para ser exitoso y las cuáles son el diferenciador en el mercado, se listan a continuación:

Casos de uso principales

- Entrenador crea una rutina con base en las necesidades del cliente.
- Entrenador asigna la rutina a un cliente específico.
- Entrenador visualiza las rutinas asignadas

Scope inicial

- Creación de rutinas (ejercicios, repeticiones, series).
- Asignación de rutinas a clientes.
- Visualización de lista de clientes y sus rutinas asignadas.

Funcionalidad esencial

- Resuelve el problema de no tener una aplicación intuitiva, fácil de usar y completa para que un coach pueda manejar y organizar sus clientes con sus rutinas, entrenamientos y su seguimiento individual.

Criterios de aceptación

- Un entrenador puede crear una rutina y guardarla.
- Un entrenador puede asignar la rutina a un cliente y queda visible en su perfil.
- El listado de rutinas muestra correctamente las asignadas a cada cliente.

Calidad técnica mínima

- Código estable
- Manejo correcto de datos (guardado/lectura confiable)
- Seguridad mínima: autenticación básica

Requerimientos no funcionales

- Seguridad en almacenamiento de credenciales (ej. llavero seguro de iOS / Keychain).
- Arquitectura escalable (Clean + MVVM) .
- Testing unitario.
- Rendimiento aceptable: acceso a rutinas, clientes y ejercicios deben cargarse sin retardos visibles.

MVP (Minimum Viable Product)

Este es el conjunto de funcionalidades mínimas que requiere nuestro producto para funcionar en su primera versión (MVP):

- **Login** de un usuario (coach) para acceder a la app
- **Pantalla principal** con lista de clientes creados con opción de crear uno nuevo si aún no existen.
- Creación de rutinas personalizadas para cada cliente.
- Destacar la acción **principal** para asignar una rutina a un cliente.
- **Vista de rutinas** en el perfil de cada cliente.
- Seguimiento de progreso: peso, repeticiones, cargas).
- Gestión y uso de app amigable y sin abrumar.
- UX y Experiencia de Usuario amigable
 - Interfaz sencilla e intuitiva.
 - Flujo de navegación limpio y sin fricción.

Exclusiones del MVP

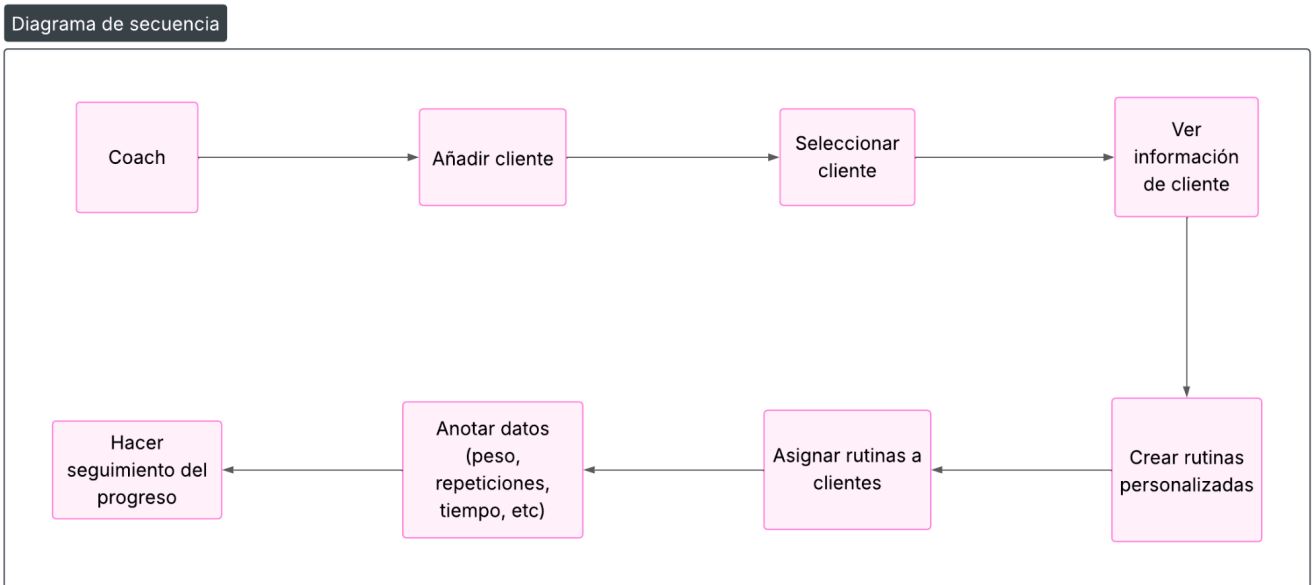
- Sin métricas avanzadas: analítica avanzada o reportes.
- Sin autenticación con herramientas de terceros (Google, Apple, etc.).
- Sin notificaciones push
- Sin branding personalizado (se reservará para funciones futuras)

Funcionalidades incluidas

- MVP (Para explicación del modelo de negocio)
 - Login de un usuario para acceder a la app
 - Pantalla principal con lista de clientes creados con opción de crear uno nuevo
 - Creación de rutinas personalizadas para cada cliente
 - Destacar acción principal para asignar una rutina a un cliente
 - Vista de rutinas para ver las rutinas asignadas a un cliente en su perfil
 - Seguimiento de progreso (peso, repeticiones, cargas)
 - Gráficas de progreso básicas
 - Gestión y uso de app amigable y sin abrumar

Diagramas UML

Diagrama de secuencia



- Arquitectura preparada para crecer.
- Base sólida que no obligue a rehacerlo.
- Base expandible (poder añadir nuevas funciones a los archivos existentes sin romper flujo).

Definición del cliente

Usuarios principales

El principal público objetivo son aquellos profesionales que desean centrarse en la salud y mejora continua de sus clientes.

Sin embargo, como se menciona anteriormente, en el MVP, hemos decidido centrarnos en primer lugar en los **entrenadores personales y coaches de fitness**.

Dentro de estos mismos podremos encontrar diferentes subcategorías, como son los coaches asociados a un gimnasio o los coaches independientes.

Nuestros usuarios, englobados en una franja de edad de entre 25 - 50 años, se encuentran cansados de haber probado varias aplicaciones y herramientas de gestión para llevar al día sus entrenamientos.

Todo ello para acabar recurriendo al papel y boli, pues ninguna agrupa la gestión de los entrenamientos con la visualización del avance de sus clientes. Y en el caso de haber encontrado una, no pueden permitirse el pago por su uso.

Tienen la necesidad de en un mismo lugar, poder visualizar los entrenamientos de sus clientes, así como su progreso. Esto les permitiría: no olvidar los entrenamientos agendados, observar el progreso de sus clientes e incluso en un futuro que los propios clientes pudieran visualizar los ejercicios.

- Entrenadores personales y Coaches independientes, pequeños estudios y eventualmente gimnasios medianos.
- En un futuro, ampliar hacia los clientes finales de los entrenadores (usuarios que reciben las rutinas)

¿Qué piensan los usuarios acerca del producto?

Durante el proyecto nos pusimos en contacto con varios entrenadores personajes, coaches y otros profesionales de la salud para tener una visión más cercana de la problemática a la que se enfrentan en su día a día.

Must have

1. Evaluación inicial: tests de condición física, movilidad, composición corporal, y fotos/medidas base.
2. Planes personalizados: periodización, progresión.
3. Generador de rutinas: por objetivos: (fuerza, pérdida de grasa, hipertrofia), por grupos musculares y por calendario.
4. Programación semanal: calendario editable, recordatorios, calentamientos y enfriamientos guiados.
5. Seguimiento de progreso: cargas, repeticiones, tiempos, PRs, métricas corporales y gráficas.
6. Temporizadores de intervalos y descansos
7. Privacidad y seguridad: permisos claros, cifrado, control de datos.

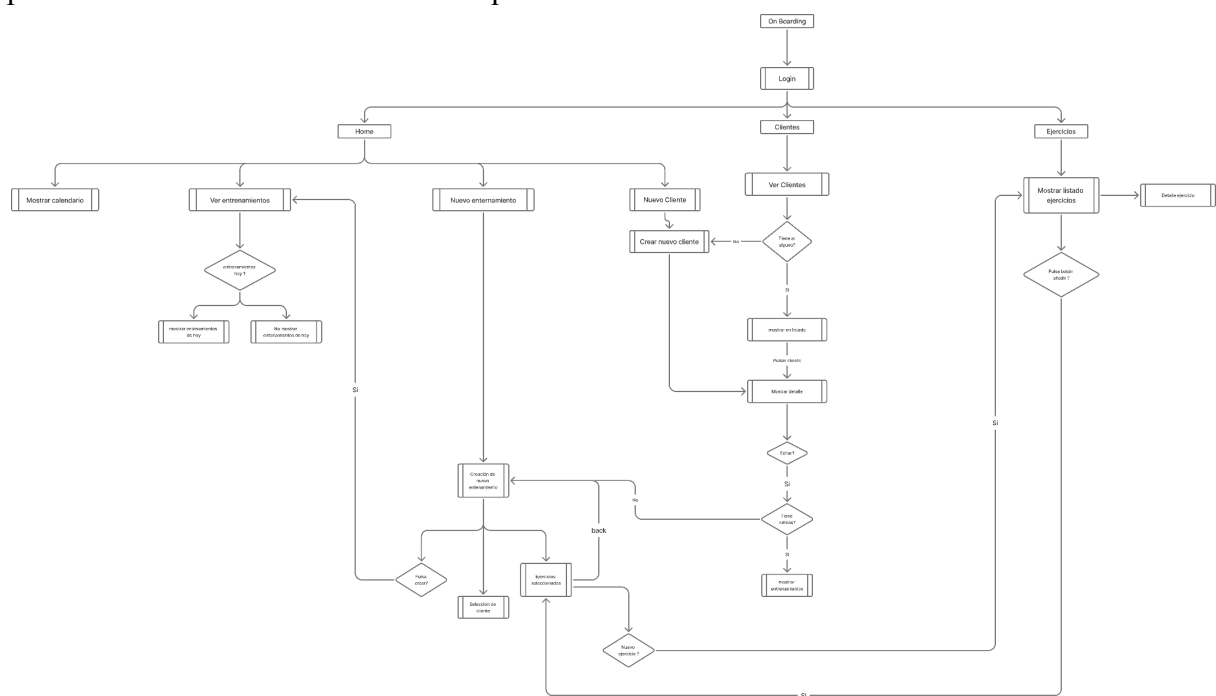
Nice to have

1. Feedback del usuario
2. Biblioteca de ejercicios HD, técnica, errores comunes, variaciones y alternativas sin equipo.
3. Comunicación coach-usuario: chat, notas de sesión, ajustes remotos, revisión de técnica por video.
4. Modo comunidad: grupos, rankings saludables, compartir logros.
5. Streaks, logros, retos, niveles o recompensas sin fomentar excesos

DISEÑO DE APLICACIÓN

User Flow

En el siguiente esquema de flujo podremos observar los diferentes caminos que el usuario podrá tomar mientras hace uso de la aplicación.



Para una mejor visualización en detalle del flujo puede encontrarlo en el siguiente enlace:

Enlace a figjam

Color

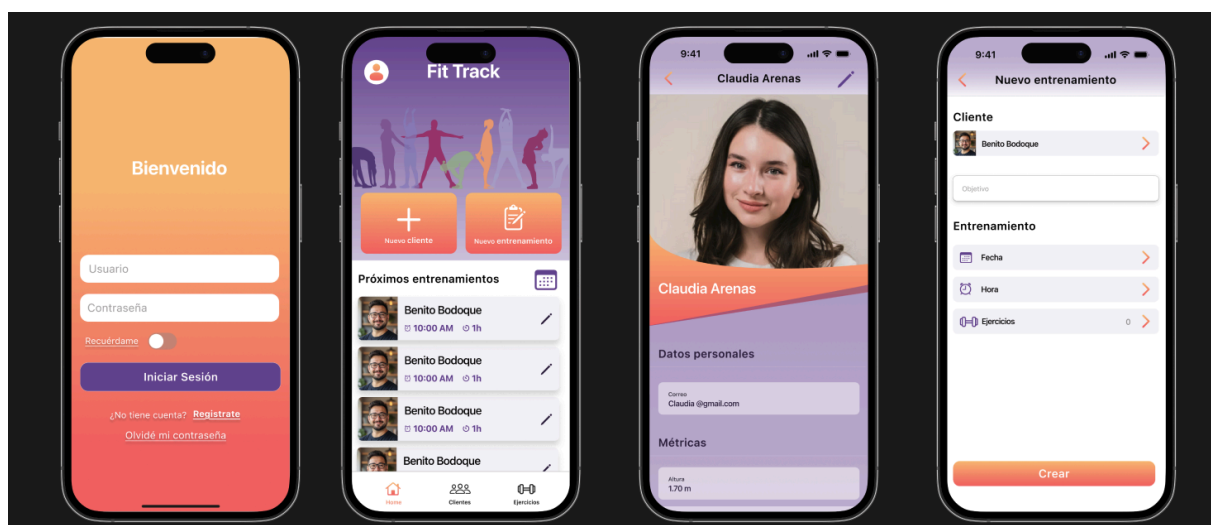
Habiendo estudiado las diferentes opciones cromáticas existentes en el mercado, nos decidimos por una gama cromática en clave alta, es decir por la predominancia de fondos claros.

Esta decisión está alineada precisamente con el público objetivo al cual va dirigida la app, el cual puede experimentar rechazo ante la oscuridad de los fondos, pues está socialmente atribuida a entrenos de alta intensidad.

Por otro lado, el fondo claro transmite una mayor sensación de transparencia y estabilidad, acorde a la tranquilidad y sensación de salud y equilibrio que nuestro usuario pretenderá dar a sus clientes.

La elección de los colores corporativos como son el **Naranja** y el **Morado**, responde a dos ideas entrelazadas: el naranja mueve a la acción mientras que su análogo en la rueda cromática, el morado, nos lleva a la creatividad y sabiduría.

Creatividad y sabiduría que serán esenciales para luchar contra la frustración que los usuarios sienten al no poder llevar un registro unificado de sus clientes.



Recursos técnicos

Los diferentes recursos técnicos que encontraremos en figma pueden ser catalogados en 5 secciones:

- Imágenes
- Estilos
- Componentes
- Buttons
- Icons

En cada una de estas áreas podremos encontrar lo que en su título se indica, siguiendo un diseño atómico donde cada área engloba a su superior.

Los márgenes dejados como safe Area lateral tienen un tamaño de 16pt, así como los espaciados entre elementos conservan una distancia de 12pt.

ANÁLISIS FINANCIERO

Presupuesto

A continuación se presenta un presupuesto estimado de los costes de producción de FitTrack, seguido de una estimación de ingresos así como el tiempo estimado en la recuperación de dicha inversión.

Costes								
	Concepto	Elemento	Precio/mes	Tiempo(mes)	Precio/persona	Efectivos	precio parcial	Total
Recursos humanos	Personal	Diseño	1.840,00 €	3	5.520,00 €	1	5.520,00 €	65.878,75 €
		Develop (4)	1.725,00 €	6	10.350,00 €	4	41.400,00 €	
Infraestructura y software	Backend	Entorno de desarrollo	30,00 €	12	-	-	360,00 €	
		Entorno de Producción	30,00 €	12	-	-	360,00 €	
		Entorno de Testing	4,15 €	12	-	-	49,78 €	
	Oficinas	Coworking	450,00 €	12			5.400,00 €	
	Equipo	Ordenadores	1300			5	6.500,00 €	
Publicación y mantenimiento inicial	Licencias	Licencias apple	-	-	300	1	300,00 €	
Contingencia e imprevistos	10%	-	-	-	-		5.988,98 €	

Ingresos								
	Concepto	Precio/mes	Estimación de personas / mes	Tiempo/meses	Total de personas	Total	Precio cubierto	
Planes	free	0,00 €	100	18	1800	0,00 €	74.430,00 €	
	Pro	45,00 €	50	18	900	40.500,00 €		

Personal	Concepto	Valor estimado	Total
Develop	Ingreso Bruto mensual	1.725,00 €	2.019,00 €
	Cuota de autonomo	294,00 €	
	IRPF (15%)	225,00 €	
	Ingreso neto mensual	1.500,00 €	
Diseño	Ingreso Bruto mensual	1.840,00 €	1.840,00 €
	IRPF (15%)	240,00 €	
	Ingreso neto mensual	1.600,00 €	

Backend	Concepto	Valor estimado/ mes	Total
Entorno de desarrollo	Docker	13,00 €	30,00 €
	Railway (Hosting)	17,00 €	
Entorno de Producción	AWS (Backup)	30,00 €	30,00 €
Entorno de Testing	BrowserStack	4,15 €	4,15 €
	TestFlight	included in develop licence	

Monetización

La app se monetiza principalmente a través de un modelo B2B (Business to Business) en este caso sería Business to Coach, donde el entrenador paga por funciones adicionales y/o premium de gestión y seguimiento para sus clientes.

Estrategias de monetización:

Planes

1. Versión gratuita (Free)

Pensada para entrenadores que quieran probar la app o tengan pocos clientes (máximo 3).

- Gestión básica de clientes (máx 3).
- Creación de 5 rutinas máximo.
- Seguimiento de progreso (peso, repeticiones, notas).
- Pequeños anuncios no invasivos (ej en pantalla de carga o sección secundaria).
- Sin acceso a:
 - Reportes avanzados
 - Gráficas
 - Mayor número de clientes
 - Creación de mayor número de rutinas
 - Guardar datos en la nube (sólo datos en memoria)

Objetivo

Captar entrenadores, permitiéndoles probar el valor y experiencia de la app y motivarlos a migrar a planes de pagos.

2. Versión de Pago - Plan Pro

Para entrenadores con una base mediana de clientes.

- Gestión de hasta 20 clientes
- Gráficas y estadísticas de progreso (PRs, cargas, medidas etc)
- Exportación básica de datos en PDF/Excel
- Integración con Apple Health/Watch para métricas automáticas.
- Sin anuncios

Este plan da el salto de “herramienta básica” a “solución profesional accesible”.

3. Versión de Pago - Plan Premium (Ilimitado)





Para entrenadores con muchos clientes y/o academias

- Clientes ilimitados

Este plan convierte la app en una herramienta de gestión y seguimiento profesional.

Característica	Free (Gratis)	Pro (Pago medio)	Premium (Pago alto)
Nº de clientes gestionados	Hasta 3	Hasta 30	Ilimitados
Creación de rutinas personalizadas	✓ Hasta 5	✓ Ilimitadas	✓ Ilimitadas
Seguimiento de progreso (peso, reps)	✓ Manual	✓ Con gráficas	✓ Con analítica avanzada
Exportación de datos (A futuro)	✗	✓ PDF/Excel	✓ Avanzado (Excel, CSV, reportes automáticos)
Estadísticas y gráficas	✗	✓ Básicas	✓ Avanzadas (históricas, comparativas)
Integración Apple Health / Watch (A futuro)	✗	✗	✓ Sí
Comunicación coach–cliente (A futuro)	✗	✗	✓ Mensajes + notificaciones avanzadas
Marca blanca (White-label) (A futuro)	✗	✗	✓ Personalización de branding
Publicidad	✗ Pequeños anuncios	✓ Sin anuncios	✓ Sin anuncios
Soporte	Básico (FAQ)	Estándar	Prioritari

Seguimiento de progreso (Diferencias entre planes)

Seguimiento de progreso (Diferencias entre planes)			
	✓ Manual	✓ Con gráficas	✓ Con analítica avanzada
Seguimiento de progreso (peso, reps)	<p>Sentadilla - Semana 1: 80kg x 8</p> <p>Sentadilla - Semana 2: 82.5kg x 8</p> <p>Sentadilla - Semana 3: 85kg x 6</p>	<p> Línea mostrando cómo subió el peso semana a semana, repeticiones máximas.</p>	<p> Línea + PR estimado, volumen máximo, Personal Records.</p> <p> Aviso: “Nuevo PR en sentadilla: 85kg”.</p> <p> Reporte mensual exportable.</p>

PLAN DE DESARROLLO

Gestión de proyecto

Durante el mes que ha durado el proyecto, el equipo decidió dividir el tiempo dado en 4 sprints.

Dentro de estos, la idea ha sido no solo rotar por cada uno de los departamentos establecidos sino también ir cambiando de TeamLead, para así fomentar una mejor comunicación entre todos al pasar por las cuatro áreas establecidas.

Estas cuatro áreas fueron las siguientes:

- Mobile
- Server
- Producto
- Diseño

Gracias también a herramientas como trello, el equipo pudo dividir las diferentes tareas de cada departamento y pasar el testigo fácilmente a la persona que durante ese sprint le tocara dicha área de trabajo.

A la tercera semana nos percatamos de que los departamentos de Diseño y Producto tienen menos volumen de trabajo que los dos restantes. Para esto decidimos que a mitad de ese sprint los dos que estuvieran en Producto y diseño entrarían como refuerzo de los otros departamentos restantes.

Escalabilidad y alcance

Una de las características que hacen competitiva FitTrack es precisamente su escalabilidad, pues a pesar de tratarse en un principio de una aplicación dirigida a entrenadores personales, esta puede **ampliarse a otros campos de la salud deportiva**.

Entre otros muchos ejemplos podríamos destacar la posibilidad futura de implementar funcionalidades que permitieran el acople a la aplicación de profesionales de la **fisioterapia** o la **nutrición**.

Esto permitiría no solo ampliar el rango del público objetivo, sino que permitiría a un mismo entrenador que supiera manejar varias áreas, aplicarlas a un mismo cliente sin necesidad de cambiar de aplicación.

Evidentemente esta **ampliación** de la aplicación, llevaría consigo el **aumento** de ciertas **tarifas** para sufragar los costes de dicha ampliación.

Por otro lado, podemos hablar de la implementación de una white label como otra forma que podría adoptar la aplicación para captar clientes.

White-Label (Marca Blanca)

La marca blanca es un modelo en el que un producto o servicio tecnológico se personaliza con la identidad visual de otra empresa o profesional, de modo que parece que fue creado por ellos.

En nuestra app:

- Un **coach de alto perfil** o un **gimnasio** podría usar tu plataforma, pero con su **logo, colores y nombre**, en lugar de la marca original de tu app.
- Desde la perspectiva del cliente final (el atleta), la app parece **propiedad del entrenador/gimnasio**.

Ejemplo práctico:

- La app se llama **FitTrack**.
- Un entrenador llamado Benito Bodocote quiere usar la plataforma, pero con su **branding**.
- A través del modelo **white-label**, la app se configura para mostrarse como:
 - **Nombre:** “Entrena con Bodoque”/ BodoqueFitness
 - **Logo:** el de su negocio
 - **Colores y tipografía:** los de su marca
- El cliente descarga la app y nunca ve el nombre original de tu producto, solo el del coach.

Relación con los planes

	Free	Pro	Premium
Marca blanca (White-label)	✗	✗	✓ Personalización de branding

Ventajas de incluir White-label

Para el coach/gimnasio:

- Refuerza su **marca personal/profesional**.
- Da una **imagen** premium y exclusiva a sus clientes.
- Permite ofrecer un **producto digital** propio sin invertir en desarrollar desde cero.

Para nosotros (como creadores de la app):

- **Monetización premium**: puedes cobrar un plan más alto por la personalización.
- Aumenta la retención de **coaches grandes** (menos probabilidad de que se cambien a otra plataforma).

En definitiva nos permitiría expandir el mercado a **gimnasios** y **academias** que quieran digitalizar su servicio.

ARQUITECTURA

En el desarrollo de aplicaciones móviles, uno de los retos más comunes que enfrentan los equipos es la **desorganización del código** cuando el proyecto comienza a crecer. Esto dificulta el mantenimiento, las pruebas y la incorporación de nuevas funcionalidades.

En nuestro análisis inicial detectamos que muchos proyectos dependen en exceso de frameworks o librerías, lo que termina generando rigidez y alto costo en cambios futuros. Para resolver este problema, decidimos adoptar el uso de **la arquitectura CLEAN**.

A continuación, se detallan las principales ventajas que justifican su uso:

1. **Separación de responsabilidades**

Cada capa de la arquitectura tiene un propósito claramente definido, lo que evita la mezcla desorganizada de lógicas. Esta división facilita la identificación de problemas y la comprensión del flujo del sistema.

2. **Facilidad para realizar pruebas**

La lógica de negocio se encuentra aislada del resto de componentes, lo que permite la creación de pruebas unitarias e integradas sin depender de la interfaz gráfica o de la base de datos. Como resultado, se obtiene un desarrollo más seguro y eficiente.

3. **Escalabilidad y mantenimiento**

CLEAN Architecture ofrece una estructura flexible que facilita la incorporación de nuevas funcionalidades sin afectar la estabilidad de las ya existentes. Además, en equipos de trabajo grandes, posibilita que varios desarrolladores colaboren de manera paralela en distintas capas, usamos CLEAN y no MVC porque el primero evita el típico “Massive ViewController” evitando un cuello de botella. Igualmente no optamos por VIPER, que también separa capas pero tiende a ser verboso y con muchos archivos por funcionalidad, puede generar sobrecarga o confusión en equipos pequeños.

4. **Independencia de frameworks**

La aplicación no queda ligada a una librería, API o SDK en particular.

5. **Legibilidad y colaboración en equipo**

El código estructurado en capas definidas sin redundancia mejora su legibilidad, permitiendo la rápida incorporación de nuevos miembros al equipo de trabajo . Esto se traduce en un aumento de la productividad y en una mayor calidad del trabajo colaborativo.

Por otro lado, y siguiendo una línea de trabajo cuyo fin es la organización óptima del código optamos como patrón de diseño principal **Model–View–ViewModel (MVVM)**.

A continuación, se presentan las principales ventajas que justifican su implementación:

1. **Separación de responsabilidades**

MVVM divide el código en tres componentes principales:

- **Model:** Encargado de la lógica de negocio y la gestión de datos.
- **ViewModel:** Procesa y transforma la información del modelo para que la vista pueda mostrarla.
- **View:** Responsable únicamente de la interfaz gráfica.

Esta división mejora la organización del proyecto y evita dependencias innecesarias.

2. **Facilidad de pruebas**

Al tener la lógica encapsulada en el ViewModel, es posible realizar pruebas unitarias sin necesidad de cargar la interfaz gráfica. Esto permite detectar errores de forma temprana y aumentar la calidad del software.

3. **Reducción del acoplamiento**

La vista no se comunica directamente con el modelo, sino a través del ViewModel. Gracias a esta independencia, se pueden modificar o reemplazar las interfaces gráficas sin alterar la lógica de negocio. A diferencia de MVP, que requiere mucha

comunicación Presenter-View, preferimos usar data binding para que la UI cambie automáticamente a cambios de estado.

4. Mantenimiento y escalabilidad

En proyectos en crecimiento, como es nuestro caso, MVVM ofrece una estructura clara que facilita la incorporación de nuevas funcionalidades. Asimismo, permite que varios desarrolladores trabajen de manera paralela sin afectar otras partes del sistema.

Herramientas y tecnologías:

En el desarrollo de la aplicación se ha optado por un conjunto de herramientas y tecnologías modernas que permiten optimizar tiempos y garantizar la calidad del producto final.

- Desarrollo de iOS

Se utiliza el lenguaje **Swift** junto con el framework **SwiftUI**, una combinación que permite desarrollar aplicaciones con un enfoque actual, seguro y eficiente.

- **Lenguaje moderno:** Swift incorpora características avanzadas como *type safety*, *optionals* e *inferencia de tipos*, lo que contribuye a la estabilidad y robustez del código.
- **Formato declarativo:** A diferencia de UIKit, donde se define paso a paso cómo construir la interfaz, SwiftUI adopta un enfoque declarativo en el que se describe **qué** se quiere mostrar, favoreciendo la simplicidad y la legibilidad del código.

Por otro lado, el uso de **macros en lugar de wrappers** se traduce en un desarrollo más **eficiente, claro y alineado con la visión declarativa de SwiftUI**. Adoptar esta práctica en proyectos con iOS 17 garantiza mantener la aplicación al día con las recomendaciones de Apple, optimizando tanto el tiempo de desarrollo como la calidad del código.

En cuanto a **reducción de tiempos de desarrollo**, SwiftUI introduce ventajas significativas frente a UIKit, entre ellas:

- **Vista previa en tiempo real**, que permite validar cambios sin necesidad de compilar.
- **Menor número de líneas de código**, lo que agiliza la implementación de interfaces.

Esto se traduce en un ciclo de entrega más corto y en una mayor capacidad de iterar sobre el diseño y la experiencia de usuario.

- Backend

Para la capa de backend hemos optado por **Vapor**, un framework para el desarrollo de aplicaciones web en Swift.

Las razones que justifican esta elección son:

- **Rendimiento y ligereza:** Vapor está construido sobre Swift, lo que le otorga alta performance y baja latencia, ideal para aplicaciones que requieren respuestas rápidas.
- **Escalabilidad:** Su arquitectura modular facilita agregar nuevas rutas, controladores y middlewares sin afectar la estabilidad del sistema.
- **Compatibilidad nativa con Swift:** Al integrarse de forma natural con las herramientas del ecosistema Apple, se reducen los problemas de interoperabilidad y se mantiene una base de código más limpia.

Tecnologías principales:

- Vapor
- Fluent + SqLite
- JWT

Estructura de carpetas:

```
FitTrack_server/  
|  
|---Controllers/  
|---DTOs/  
|---Migrations/  
|---Models/  
|---Configure  
|---Constants  
|---routes  
|---entrypoint  
|---JWTToken
```

Modelos de dominio:

·User: campos principales(id, name, email, password, isAdmin).
Relación 1...many con resto de modelos.

·Exercise: campos principales(id, name, description, repetitions, sets, trainingID).
Relación con training many to one.

·Training: campos principales (id, name, start, end, userID)
Relación con exercise 1...many.

DTOs:

En el proyecto diferenciamos entre **DTOs de entrada y DTOs de salida:**

·Los DTOs de entrada se utilizan para recibir datos del cliente, como UserLoginDTO en el proceso de login.

·Los DTOs de salida se utilizan para devolver datos al cliente de forma segura y estructurada, como UserDTO (info del usuario sin exponer contraseñas) o LoginResponseDTO (token tras la autenticación).

Así nos aseguramos de que nunca se exponen datos sensibles (passwordHash) y mantenemos claro qué datos se esperan en cada request/ response.

Autenticación:

Este servidor utiliza autenticación JWT (Json Web Tokens).

El archivo clave para esto es AuthController.swift, en la carpeta de Controllers.

Tokens generados:

- Access Token → valido durante 24h (Constants.accessTokenLifeTime).
- Refresh Token → válido durante 7 días (Constants.refreshTokenLifeTime).

Los campos del token, reflejados en Models/JWTToken.swift son:

- userID (UUID)
- userName
- expiration
- isRefresh (indica si el token es de acceso o refresh)

El flujo para este proceso quedaría del siguiente modo:

1. Registro → el usuario se crea con su contraseña hasheada (BCrypt).
2. Login → se verifica email y contraseña, se generan los tokens.
3. Refresh → con el refresh obtenemos un nuevo par de tokens.

Al usar JWT garantizamos que cada request se valide sin mantener la sesión en el servidor.

Middleware:

- AdminMiddleware: Encargado de asegurarse de que el usuario que realiza la solicitud es un coach.
- APIKeyMiddleware: Valida que la petición incluya una cabecera X-API-KEY válida. Pensado para restringir el acceso a ciertos endpoints a clientes que tengan la clave.
- RateLimitIPMiddleware: Limita el numero de peticiones que se pueden realizar en un minuto a 10. Tiene en cuenta la IP del usuario.

Aplicado en las solicitudes que no necesitan de un usuario registrado, como auth/register/coach o auth/login.

- RateLimitUserMiddleware: Limita el número de peticiones que se pueden realizar en un minuto a 10. Tiene en cuenta el token del usuario.

Controllers:

Cada controlador implementa la lógica de un recurso/ modelo y sus endpoints REST.

·UserController

- GET /users → lista de usuarios.
- GET /users/:id → devuelve un usuario por ID.
- PUT /users/:id → Actualiza un usuario existente
- DELETE /users/:id → Elimina un usuario

·AuthController

- POST /auth/register → registro de usuarios
- POST /auth/login → login y generación de tokens.
- POST /auth/refresh → refresco de tokens.

·TrainingController

- POST /training -> registro de entrenamiento
- GET /trainings -> obtener todos los entrenamientos
- GET /trainingsByMonth -> obtener los entrenamientos programados para un mes específico.
- GET /training -> Obtener training por ID
- PATCH /training -> Modificar un training
- DELETE /training -> Eliminar un training

·ExerciseController

- Seguirán una estructura CRUD similar a UserController.
- Encargados de gestionar ejercicios, entrenamientos y citas respectivamente.

Todos los controladores se registrarán en routes.swift usando app.register(collection:).

Migraciones:

Las migraciones definen las tablas de la base de datos a partir de los modelos.

·CreateUserMigration:

- Crea columnas básicas para identificación: id, name, email, password.
- Añade campos opciones a profile: goal, age, weight, height.
- Añade relación opcional 'coach_id' que referencia a otro usuario con el rol del coach.
- Incluye timestamps 'created_at' y 'updated_at'.
- La función revert elimina la tabla y sus campos.

·CreateTrainingMigration:

- Crea columnas básicas para identificación: id, name, trainee_id, scheduled_at,
- Incluye timestamps 'created_at' y 'updated_at'.
- La función revert elimina la tabla y sus campos.

Rutas:

Todas las rutas de la API se configuran en routes.swift.

De este modo organizamos la API de forma modular y cada recurso tiene su grupo de rutas.

```
try app.register(collection: UserController())
```

```
try app.register(collection: AuthController())
```

```
try app.register(collection: ExerciseController())
```

```
try app.register(collection: TrainingController())
```

Constants:

Centralización de parámetros de configuración, como el lifetime de los tokens.

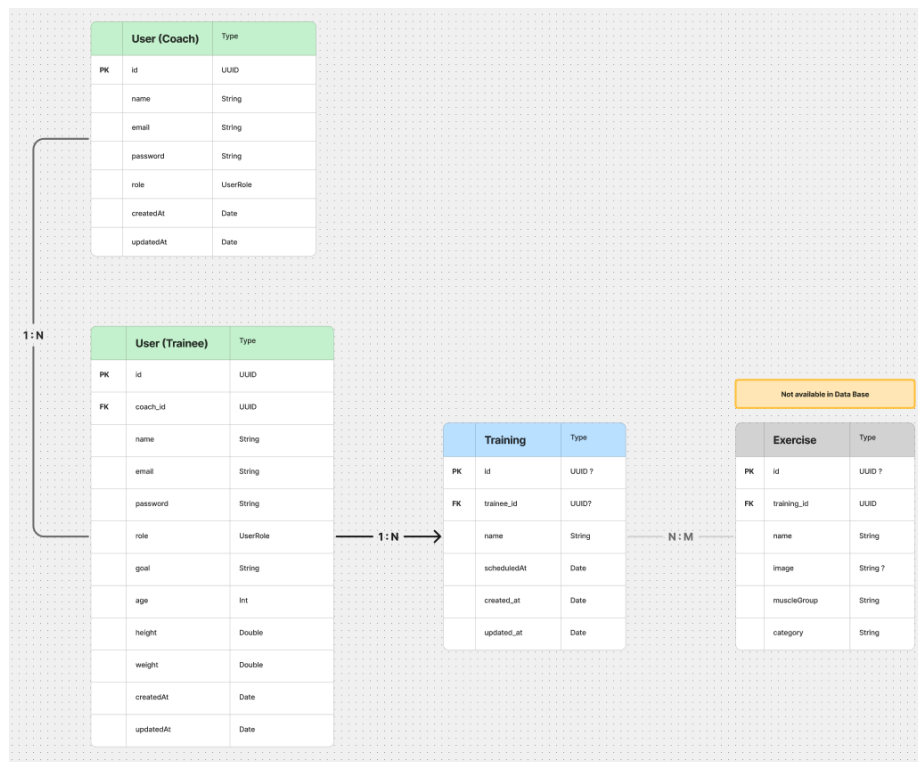
- Base de Datos

Para el almacenamiento de datos hemos decidido utilizar **SQLite** durante la fase de desarrollo y para el MVP.

Las principales razones son:

- **Rapidez y simplicidad:** SQLite permite crear, modificar y eliminar datos de forma ágil, lo que es perfecto para entornos de desarrollo y pruebas.
- **Ligereza:** No requiere la configuración de un servidor de base de datos externo, reduciendo la complejidad inicial.
- **Adecuado para el MVP:** Aunque no está diseñado para manejar grandes volúmenes de datos en entornos de producción a gran escala, su velocidad y facilidad de uso lo hacen ideal para las primeras etapas del proyecto.

Este enfoque nos permite validar el producto rápidamente y, en fases posteriores, migrar a una base de datos más robusta (por ejemplo, PostgreSQL) si la demanda lo requiere.



Para una mejor visualización en detalle del diagrama puede encontrarlo en el siguiente enlace:

[Enlace a fig jam](#)

- Control de Versiones y colaboración

Para garantizar un flujo de trabajo ordenado y colaborativo, hemos utilizado **Git** como sistema de control de versiones y **GitHub** como repositorio remoto.

Esto permite

- Diseño UX/UI

Para el diseño de interfaces hemos utilizado **Figma** como herramienta principal.

Las principales razones de dicha elección es que permite una colaboración real entre diseñadores y desarrolladores, permite un prototipado interactivo que agiliza la toma de decisiones pues también centraliza la ubicación donde los desarrolladores pueden encontrar información sobre las medidas,colores o assets del diseño para su implementación