Memoria Elevate

Desarrollo en Aplicaciones Multiplataforma

CENTRO: IES LA MAR, JAVEA

TUTOR: JUANJO PEDRAZA PENOUCOS

ALUMNO: BORJA BOLUFER SALA

App Elevate

Hecha Por Borja Bolufer Sala

La creación de una app de fitness personalizado abarca la planificación, diseño, desarrollo e implementación de una herramienta digital que permite a los usuarios gestionar su actividad física y nutrición de manera personalizada, ofreciendo funcionalidades avanzadas como entrenamientos adaptados, seguimiento del progreso y soporte motivacional, todo ello en una plataforma fácil de usar y adaptable a las necesidades del usuario y del profesional o gimnasio.

Índice

Resumen del Contenido

Cuestión estudiada: Este proyecto aborda la democratización del fitness personalizado mediante el desarrollo de una aplicación móvil que integra inteligencia artificial para generar planes de ejercicios y dietas personalizadas, eliminando las barreras económicas del entrenamiento personal tradicional.

Métodos utilizados: Se empleó metodología de desarrollo ágil para crear una aplicación Android nativa en Java, integrando la API de ChatGPT para personalización inteligente, base de datos SQLite para almacenamiento local, y siguiendo principios de Material Design para la interfaz de usuario.

Resultados obtenidos: Se logró desarrollar una aplicación completamente funcional que recopila datos personales del usuario (edad, peso, objetivos, intensidad) y genera automáticamente planes de entrenamiento semanales y dietas balanceadas adaptadas a cada perfil individual.

Conclusiones principales: Elevate demuestra la viabilidad de integrar IA en aplicaciones móviles para democratizar servicios premium, estableciendo un precedente en el sector fitness digital y confirmando la efectividad de arquitecturas híbridas local-nube para aplicaciones de salud personal.

Abstract (English)

Research question: This project addresses the democratization of personalized fitness through the development of a mobile application that integrates artificial intelligence to generate customized exercise plans and diets, eliminating economic barriers of traditional personal training.

Methods used: Agile development methodology was employed to create a native Android application in Java, integrating ChatGPT API for intelligent personalization, SQLite database for local storage, and following Material Design principles for user interface development.

Results obtained: A fully functional application was successfully developed that collects user personal data (age, weight, objectives, intensity) and automatically generates weekly training plans and balanced diets adapted to each individual profile.

Main conclusions: Elevate demonstrates the feasibility of integrating AI into mobile applications to democratize premium services, establishing a precedent in the digital fitness sector and confirming the effectiveness of hybrid local-cloud architectures for personal health applications.

1. Datos del proyecto y resumen

1.1 Datos del proyecto

Campo	Información
Título	Elevate - Aplicación de Desarrollo Personal y Fitness
Ciclo Formativo	Desarrollo de Aplicaciones Multiplataforma (DAM)
Centro Educativo	IES la Mar
Curso Académico	2024-2025
Tipo de Proyecto	Trabajo Final de Ciclo
Plataforma	Android
Tecnologías Principales	Android Studio, Java, ChatGPT API, SQLite
Duración	3 meses

Breve descripción

Elevate es una aplicación móvil para Android enfocada en el desarrollo personal y fitness que utiliza inteligencia artificial para generar planes de entrenamiento y nutrición personalizados. La aplicación recopila información del usuario mediante un proceso de registro completo y utiliza la API de ChatGPT para crear rutinas adaptadas a las necesidades, objetivos y características físicas de cada usuario.

1.2 Resumen ejecutivo

Contexto y problemática

En la actualidad, el mercado del fitness y desarrollo personal está saturado de aplicaciones genéricas que ofrecen rutinas estándar sin considerar las características individuales de cada usuario. Muchas personas buscan planes personalizados pero no tienen acceso a entrenadores personales debido a limitaciones económicas o de tiempo. Esta situación genera frustración y abandono de objetivos de fitness por falta de orientación adecuada.

Solución propuesta

Elevate surge como una solución innovadora que democratiza el acceso a planes de entrenamiento personalizados mediante el uso de inteligencia artificial. La aplicación recopila datos específicos del usuario (edad, peso, altura, objetivos, experiencia, días disponibles) y genera automáticamente rutinas de ejercicio y planes nutricionales adaptados a sus necesidades particulares.

Características principales:

- **Sistema de registro completo**: Recopilación detallada de datos personales y objetivos
- **Generación automática de planes**: Utilización de ChatGPT API para crear rutinas personalizadas
- **Interfaz intuitiva**: Diseño centrado en la experiencia del usuario
- **Almacenamiento local eficiente**: Base de datos SQLite para gestión óptima de datos del usuario
- **Gestión de entrenamientos**: Visualización detallada de ejercicios y seguimiento de progreso
- **Flexibilidad**: Posibilidad de regenerar planes y editar perfil según evolución del usuario

Tecnologías implementadas

El proyecto ha sido desarrollado utilizando **Android Studio** como entorno de desarrollo principal, implementando una arquitectura robusta que integra **SQLite** para el almacenamiento local de datos del usuario y la **API de ChatGPT** para la generación inteligente de contenido personalizado.

Objetivos alcanzados:

- Desarrollo completo de una aplicación Android funcional.
- Implementación exitosa de integración con APIs externas.
- Creación de un sistema de autenticación seguro.
- Diseño de una interfaz de usuario intuitiva y atractiva.
- Establecimiento de un flujo de trabajo eficiente para la generación de planes.

Valor añadido

Elevate representa una innovación en el sector del fitness personal al combinar la accesibilidad de las aplicaciones móviles con la potencia de la inteligencia artificial, ofreciendo una experiencia personalizada que tradicionalmente solo estaba disponible a través de servicios premium. La aplicación no solo proporciona rutinas de ejercicio, sino que también incluye orientación nutricional, creando un ecosistema completo para el desarrollo personal.

Impacto esperado

El proyecto busca contribuir a la democratización del fitness personalizado, proporcionando a usuarios de todos los niveles económicos acceso a planes de entrenamiento de calidad. Además, el proyecto demuestra la capacidad de integrar tecnologías emergentes como la IA en aplicaciones móviles prácticas y útiles para la vida cotidiana.

Escalabilidad futura

La arquitectura desarrollada permite futuras expansiones como integración con dispositivos wearables, sistemas de seguimiento de progreso más avanzados, comunidades de usuarios, y la incorporación de nuevas funcionalidades basadas en machine learning para mejorar continuamente las recomendaciones personalizadas.

2. Introducción

2.1 Contexto del desarrollo personal

En la sociedad actual, el desarrollo personal y el cuidado de la salud física han adquirido una importancia sin precedentes. La pandemia de COVID-19 aceleró la concienciación sobre la importancia del bienestar físico y mental, impulsando a millones de personas a buscar alternativas para mantenerse activas desde sus hogares y establecer rutinas de ejercicio sostenibles.

El mercado del fitness y desarrollo personal ha experimentado una transformación digital significativa en los últimos años. Según estudios recientes, más del 70% de las personas que practican ejercicio utilizan algún tipo de tecnología para apoyar sus rutinas de entrenamiento. Sin embargo, la mayoría de estas soluciones ofrecen enfoques genéricos que no consideran las características individuales, limitaciones físicas, o objetivos específicos de cada usuario.

El desarrollo personal va más allá del simple ejercicio físico; implica un enfoque holístico que incluye la nutrición, el establecimiento de metas realistas, el seguimiento del progreso y la adaptación continua de las rutinas según la evolución del usuario. En este contexto, la personalización se convierte en un factor clave para el éxito a largo plazo de cualquier programa de fitness.

La democratización de la inteligencia artificial y su integración en aplicaciones móviles ha abierto nuevas posibilidades para crear experiencias verdaderamente personalizadas. Esta evolución tecnológica permite que aplicaciones desarrolladas con tecnologías robustas como Java puedan ofrecer funcionalidades avanzadas que antes solo estaban disponibles a través de servicios premium o entrenadores personales.

2.2 Problemática actual

Falta de personalización en aplicaciones existentes

El mercado actual está saturado de aplicaciones de fitness que ofrecen rutinas estándar "talla única". Estas aplicaciones no consideran factores cruciales como el nivel de experiencia del usuario, sus limitaciones físicas, el tiempo disponible para entrenar, o sus objetivos específicos. Esta falta de personalización resulta en:

- **Rutinas inadecuadas** que pueden ser demasiado fáciles o excesivamente desafiantes
- **Alto índice de abandono** debido a la falta de motivación y resultados visibles
- **Riesgo de lesiones** por ejercicios no adaptados al nivel del usuario
- **Frustración** al no ver progreso debido a programas mal estructurados

Barreras económicas para acceso a entrenamiento personalizado

Los servicios de entrenamiento personalizado tradicionales presentan importantes barreras de acceso:

- **Costes elevados** de entrenadores personales (50-100€ por sesión).
- **Limitaciones geográficas** para acceder a profesionales cualificados, ya que si

tienes que trasladarte, tienes utilizar el coche y comporta más gasto.
- **Horarios restrictivos** que no se adaptan a la disponibilidad del usuario.
- **Falta de seguimiento continuo** entre sesiones.

Problemas técnicos en aplicaciones existentes

Muchas aplicaciones actuales sufren de limitaciones técnicas significativas:

- **Interfaces complejas** que dificultan la experiencia del usuario.
- **Falta de integración** entre diferentes aspectos del fitness (ejercicio y nutrición).
- **Almacenamiento de datos deficiente** que no permite un seguimiento efectivo.
- **Ausencia de adaptabilidad** para modificar planes según el progreso del usuario.

Carencia de enfoques integrales

La mayoría de aplicaciones se centran únicamente en el ejercicio, ignorando aspectos fundamentales como:

- **Planificación nutricional** adaptada a los objetivos de entrenamiento.
- **Educación sobre técnicas** de ejercicio correctas.
- **Seguimiento de progreso** a largo plazo.
- **Adaptación dinámica** de rutinas según resultados.

2.3 Solución propuesta

Visión general de Elevate

Elevate surge como una solución integral desarrollada en Java que aborda las problemáticas identificadas mediante la implementación de tecnologías avanzadas y un enfoque centrado en el usuario. La aplicación combina la robustez del desarrollo nativo en Android con la potencia de la inteligencia artificial para crear una experiencia verdaderamente personalizada.

Características innovadoras de la solución

Personalización basada en inteligencia artificial:

- **Análisis de datos del usuario**: Recopilación completa de información personal, física y de objetivos.
- **Generación automática de planes**: Utilización de ChatGPT API para crear rutinas adaptadas.
- **Adaptación continua**: Capacidad de regenerar planes según la evolución del usuario.

Arquitectura técnica robusta:

- **Desarrollo en Java**: Aprovechando la estabilidad y robustez del lenguaje para aplicaciones Android.
- **Base de datos SQLite**: Almacenamiento local eficiente para acceso rápido y privacidad de datos.
- **API de ChatGPT**: Inteligencia artificial para generación de contenido personalizado.

Experiencia de usuario optimizada:

- **Flujo de registro intuitivo**: Proceso guiado para recopilar información necesaria.
- **Interfaz limpia y funcional**: Diseño centrado en la usabilidad y accesibilidad.
- **Navegación fluida**: Transiciones coherentes entre diferentes secciones de la aplicación.
- **Información detallada**: Pantallas específicas para entrenamientos, ejercicios y planes nutricionales.

Enfoque integral del desarrollo personal:

- **Planes de entrenamiento**: Rutinas adaptadas a objetivos, tiempo disponible y nivel de experiencia.
- **Orientación nutricional**: Recomendaciones alimentarias complementarias al entrenamiento.
- **Gestión de perfil**: Posibilidad de actualizar datos y regenerar planes según evolución.
- **Seguimiento de progreso**: Herramientas para monitorear avances y ajustar objetivos.

Ventajas competitivas:

1. **Accesibilidad económica**: Eliminación de barreras económicas para acceso a entrenamiento personalizado.
2. **Disponibilidad 24/7**: Acceso a planes y orientación en cualquier momento y lugar.
3. **Adaptabilidad**: Capacidad de evolucionar con el usuario y sus cambios de objetivos.
4. **Tecnología de vanguardia**: Uso de IA para crear experiencias verdaderamente personalizadas.
5. **Desarrollo robusto**: Implementación en Java garantizando estabilidad y rendimiento.

Impacto esperado

Elevate pretende democratizar el acceso al fitness personalizado, proporcionando a usuarios de todos los niveles socioeconómicos la posibilidad de acceder a planes de entrenamiento de calidad profesional. La aplicación no solo busca resolver las limitaciones técnicas de las soluciones existentes, sino también crear un nuevo estándar en la industria del fitness digital mediante la integración inteligente de tecnologías emergentes con principios sólidos de desarrollo de software.

3. Objetivos

3.1 Objetivo principal

El objetivo principal de este proyecto es **desarrollar una aplicación móvil nativa para Android utilizando Java que democratice el acceso al fitness personalizado mediante la integración de inteligencia artificial**. La aplicación debe proporcionar a los usuarios planes de entrenamiento y nutrición completamente personalizados, generados automáticamente según sus características físicas, objetivos y preferencias, eliminando las barreras económicas y geográficas que tradicionalmente limitan el acceso a servicios de entrenamiento profesional.

Elevate busca establecer un nuevo paradigma en el desarrollo de aplicaciones de fitness, combinando la robustez del desarrollo en Java con tecnologías emergentes como la inteligencia artificial, para crear una experiencia de usuario superior que sea técnicamente sólida, escalable y verdaderamente útil para el desarrollo personal integral.

3.2 Objetivos específicos

3.2.1 Objetivos técnicos

Desarrollo de aplicación Android robusta:

- **Implementar una aplicación nativa** en Java utilizando Android Studio como entorno de desarrollo principal.
- **Crear una arquitectura sólida** que garantice estabilidad, rendimiento y mantenibilidad del código.
- **Desarrollar un sistema de base de datos local** utilizando SQLite para almacenamiento eficiente de datos del usuario.

Integración con servicios externos:

- **Implementar la integración con ChatGPT API** para la generación automática de planes personalizados.
- **Desarrollar un sistema de procesamiento JSON** para interpretar y presentar las respuestas de la IA.
- **Crear mecanismos de manejo de errores** para garantizar la estabilidad ante fallos de conectividad o servicios externos.
- **Optimizar las llamadas a APIs** para minimizar costes y maximizar rendimiento.

Gestión de datos y seguridad:

- **Implementar un sistema de autenticación seguro** que proteja los datos personales de los usuarios.
- **Desarrollar estructuras de datos eficientes** para almacenar información de entrenamientos, ejercicios y perfiles.
- **Crear sistemas de respaldo y sincronización** entre almacenamiento local y en la nube.
- **Garantizar el cumplimiento de normativas** de protección de datos (RGPD).

3.2.2 Objetivos funcionales

Sistema de personalización avanzada:

- **Desarrollar un formulario de registro completo** que capture todos los datos necesarios para la personalización.
- **Crear algoritmos de recopilación de datos** que incluyan información física, objetivos, experiencia y preferencias.
- **Implementar un sistema de generación de planes** que utilice IA para crear rutinas únicas y adaptadas.
- **Desarrollar funcionalidades de actualización** que permitan regenerar planes según la evolución del usuario.

Experiencia de usuario optimizada

- **Diseñar interfaces intuitivas** que faciliten la navegación y el uso de la aplicación.

- **Crear flujos de trabajo eficientes** que minimicen el tiempo necesario para acceder a la información relevante.
- **Implementar sistemas de visualización clara** para entrenamientos, ejercicios y planes nutricionales.
- **Desarrollar funcionalidades de gestión de perfil** que permitan al usuario controlar su información.

Gestión integral de fitness:

- **Crear sistemas de presentación de entrenamientos** con información detallada de cada ejercicio.
- **Desarrollar pantallas específicas** para visualizar planes nutricionales complementarios.
- **Implementar funcionalidades de seguimiento** que permitan al usuario monitorear su progreso.
- **Crear herramientas de personalización continua** para adaptar la aplicación a las necesidades cambiantes.

3.2.3 Objetivos de aprendizaje y desarrollo profesional

Competencias técnicas:

- **Profundizar en el desarrollo Android** utilizando Java como lenguaje principal.
- **Adquirir experiencia en integración de APIs** y servicios de inteligencia artificial.
- **Desarrollar competencias en diseño de bases de datos** locales (SQLite).
- **Mejorar habilidades de arquitectura de software** y patrones de diseño.

Gestión de proyectos:

- **Aplicar metodologías de desarrollo ágil** para la gestión eficiente del proyecto.
- **Desarrollar competencias en planificación** y seguimiento de proyectos tecnológicos.
- **Adquirir experiencia en documentación técnica** y presentación de proyectos.
- **Mejorar habilidades de resolución de problemas** complejos y toma de decisiones técnicas.

3.3 Funcionalidades principales

3.3.1 Sistema de autenticación y registro

Pantalla de bienvenida:

- **Interfaz principal** con opciones claras de "Iniciar Sesión" y "Registro".
- **Diseño atractivo** que comunique el propósito de la aplicación.
- **Navegación intuitiva** hacia las diferentes opciones de acceso.

Proceso de registro completo:

- **Recopilación de datos básicos**: nombre, email, contraseña y verificación de contraseña.
- **Validación en tiempo real** de formatos de email y fortaleza de contraseñas.
- **Gestión segura de credenciales** con encriptación de datos sensibles.

Figura 1 - Pagina de bienvenida

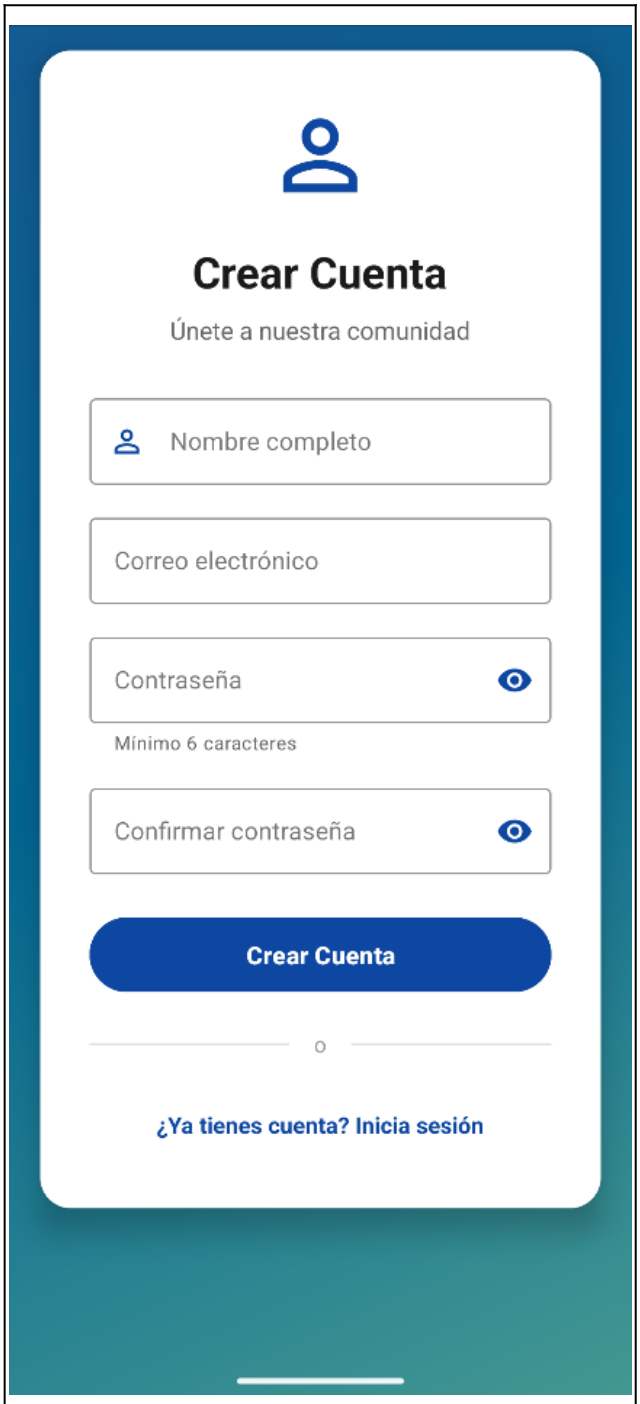

Figura 2 - Pagina de Registro

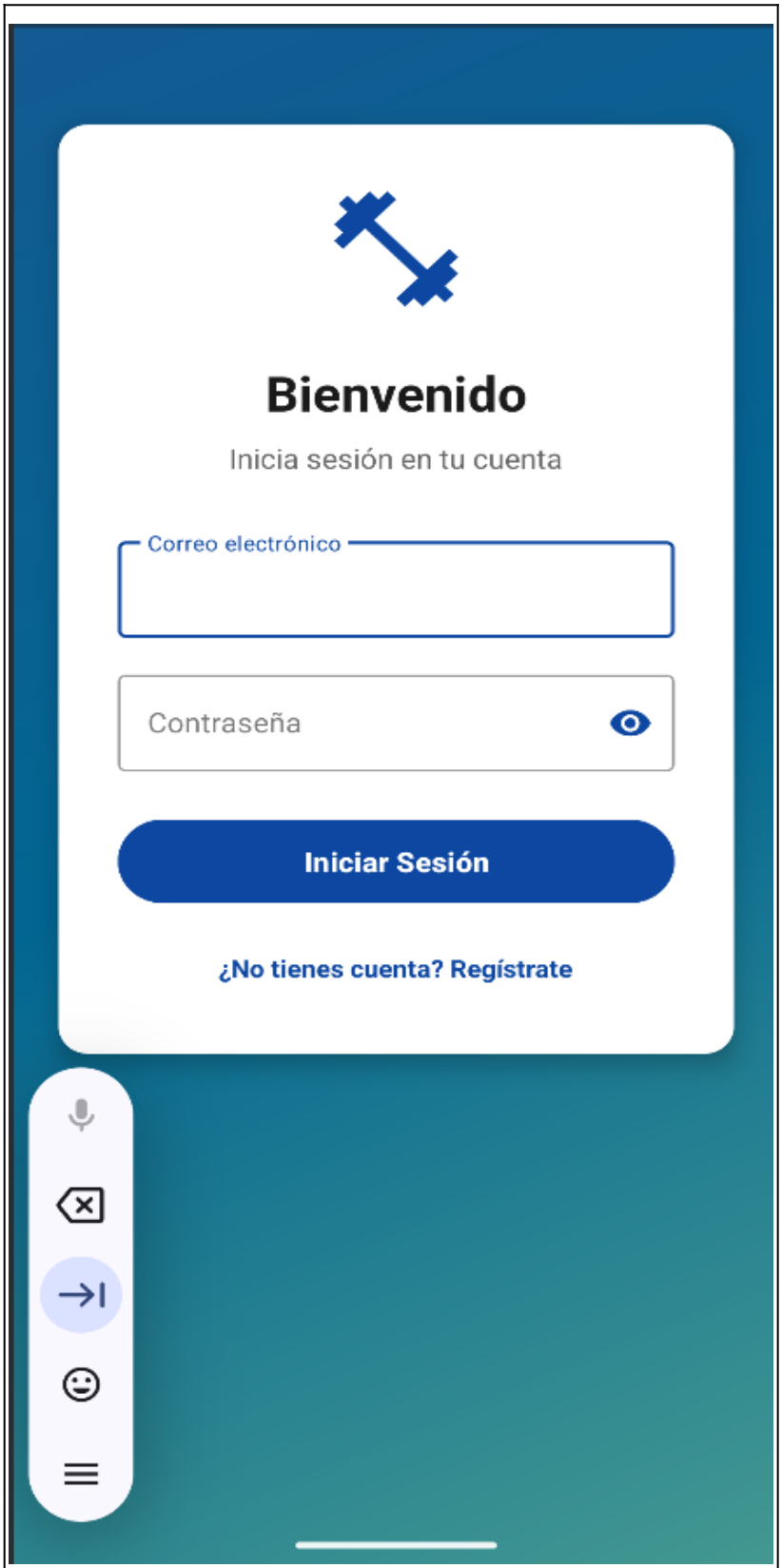

Figura 3 - Pagina inicio de sesion

Recopilación de información personal:

- **Datos físicos**: edad, altura, peso actual y sexo.
- **Objetivos de fitness**: pérdida de peso, ganancia muscular, mantenimiento, etc.
- **Nivel de intensidad**: principiante, intermedio, avanzado.
- **Disponibilidad**: días de entrenamiento por semana.
- **Preferencias adicionales**: limitaciones físicas, equipamiento disponible.

Figura 4 - Pagina Pregunta Sexo

Figura 5 - Pagina Pregunta Edad

Figura 6 - Pagina Pregunta Peso

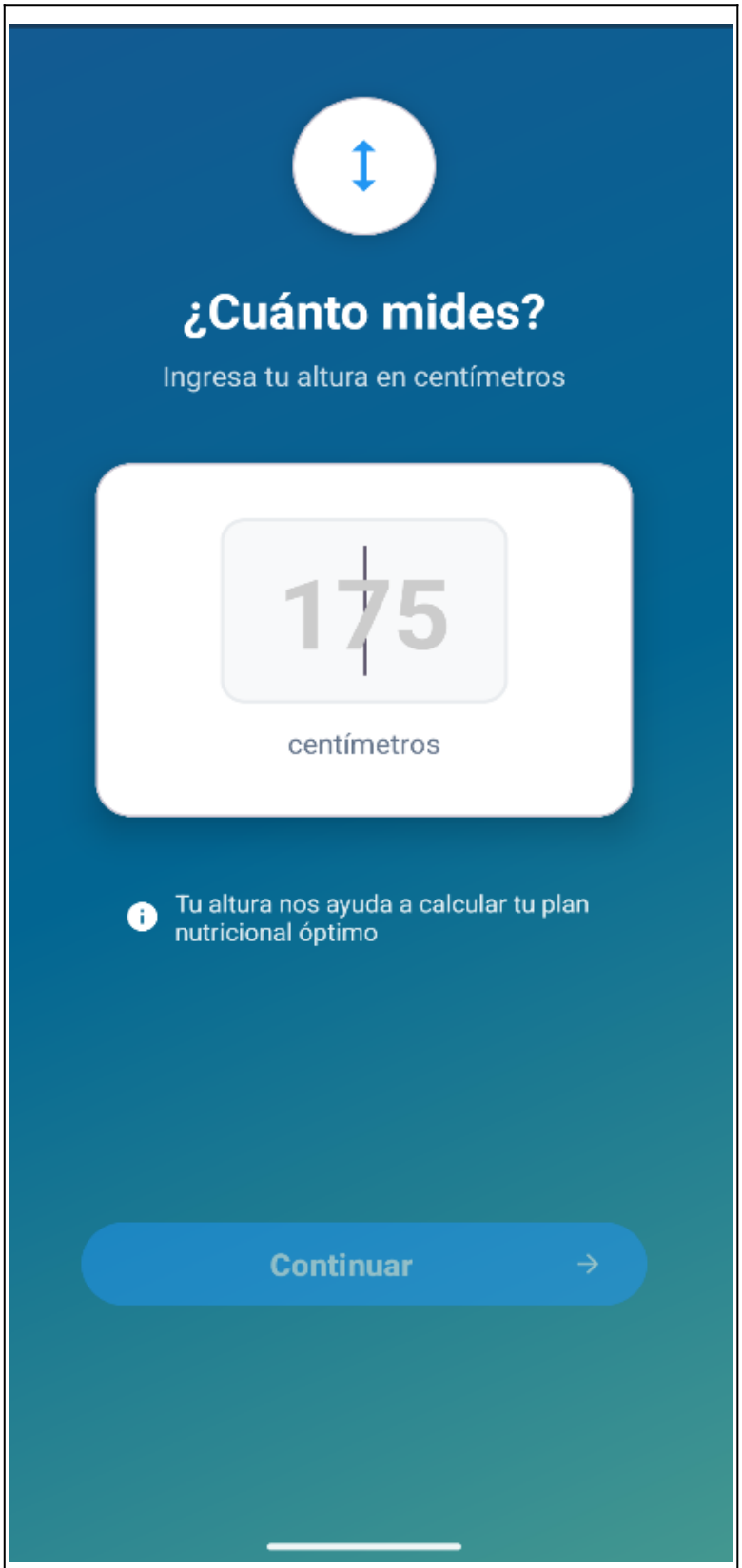

Figura 7 - Pagina Pregunta Altura

Figura 8 - Pagina Pregunta Intensidad

Figura 9 - Pagina Pregunta Objetivo

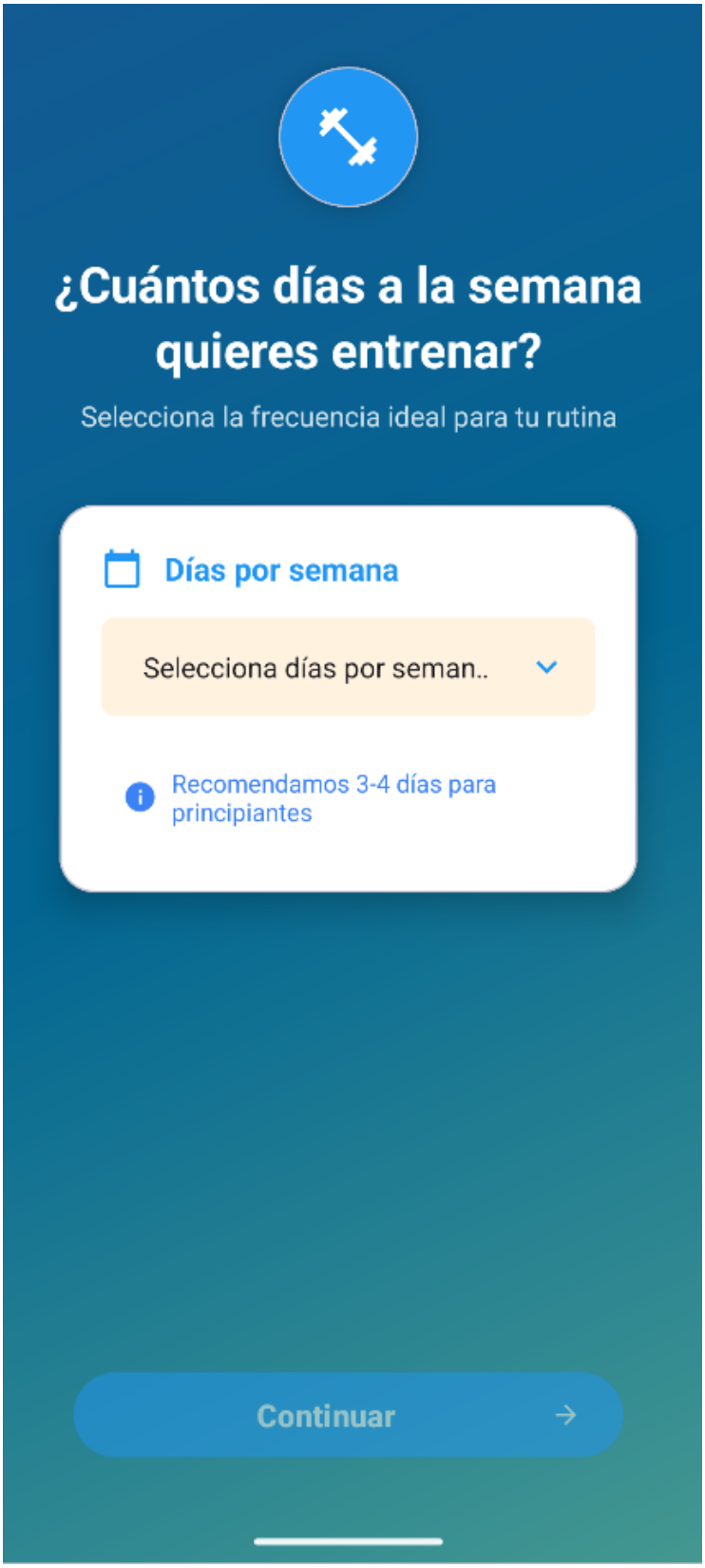

Figura 10 - Pagina Pregunta Dias Entrenamiento

3.3.2 Sistema de generación de planes personalizados

Pantalla de resumen:

- **Visualización completa** de todos los datos ingresados por el usuario.
- **Opción de edición** para corregir información antes de generar el plan.
- **Confirmación de datos** antes de proceder con la generación.

Figura 11 - Pagina Resumen Usuario

Generación automática con IA:

- **Pantalla de carga** con indicadores de progreso durante la generación.
- **Integración con ChatGPT API** para crear planes únicos y personalizados.
- **Procesamiento de respuestas JSON** para estructurar la información recibida.
- **Generación simultánea** de planes de entrenamiento y nutricionales.

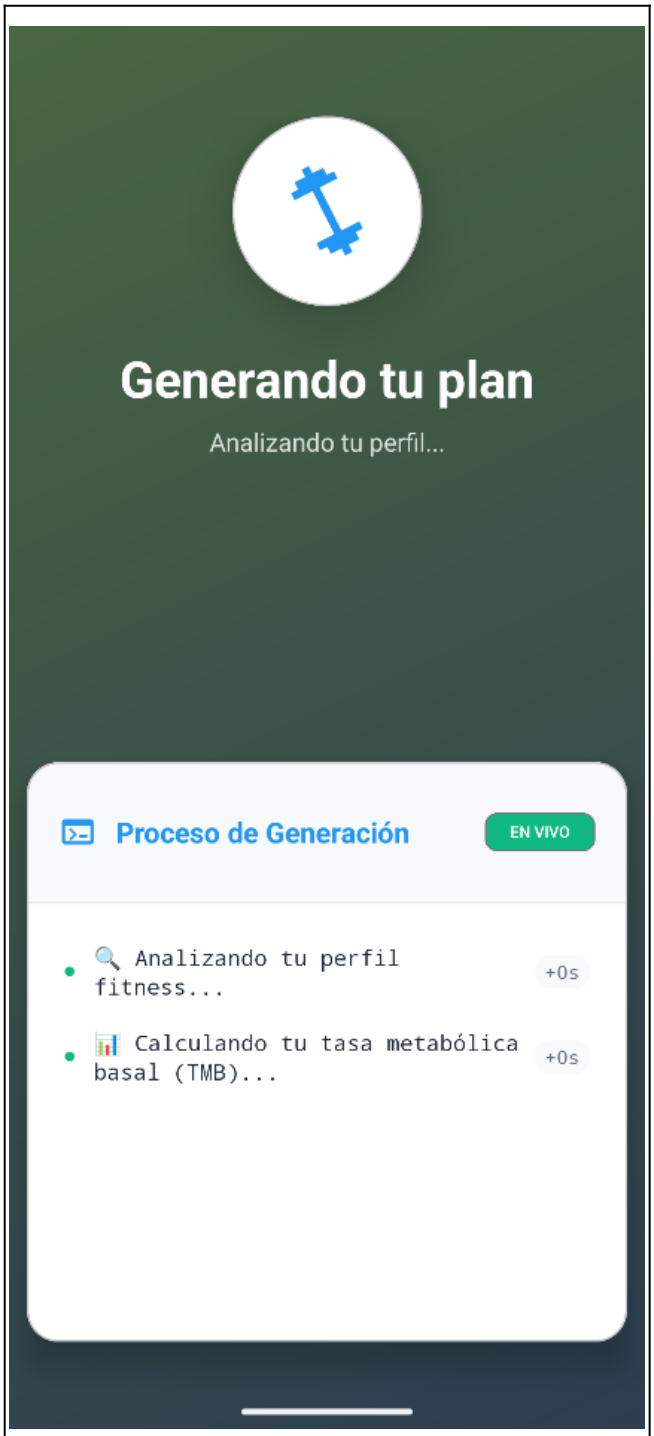

Figura 12 - Pagina Generacion Plan

3.3.3 Visualización y gestión de entrenamientos

Vista general de planes:

- **Dashboard principal** con resumen de entrenamientos semanales.
- **Navegación por días** de la semana con entrenamientos asignados.
- **Indicadores visuales** de progreso y completitud de rutinas.
- **Acceso rápido** a entrenamientos del día actual.

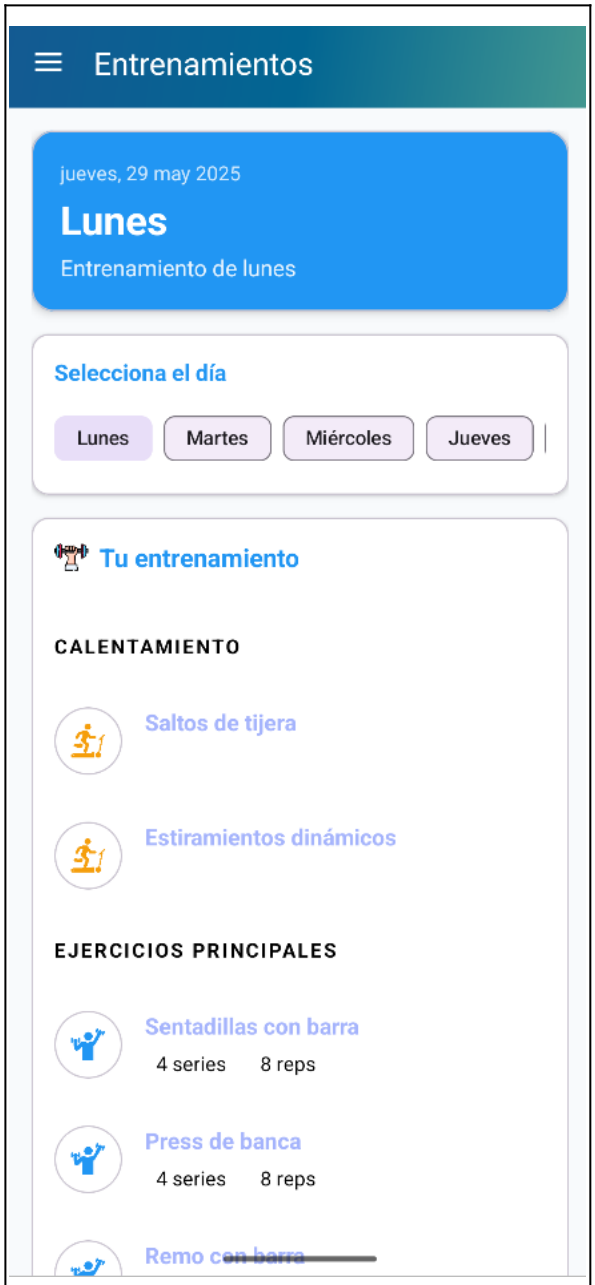

Figura 13 - Pagina Entrenamientos

Detalles de entrenamientos:

- **Pantallas específicas** para cada entrenamiento.
- **Detalle** de ejercicios con sets, repeticiones y descansos.
- **Instrucciones claras** por video en entrenamientos principales para la ejecución correcta de cada ejercicio.
- **Tiempo estimado** para completar cada rutina.

Figura 14 - Pagina Ejercicio Detalle

3.3.4 Gestión nutricional

Planes alimentarios:

- **Visualización de dietas** complementarias al entrenamiento.
- **Distribución de macronutrientes** según objetivos del usuario.
- **Sugerencias de comidas** para diferentes momentos del día.
- **Adaptación calórica** según intensidad de entrenamiento.

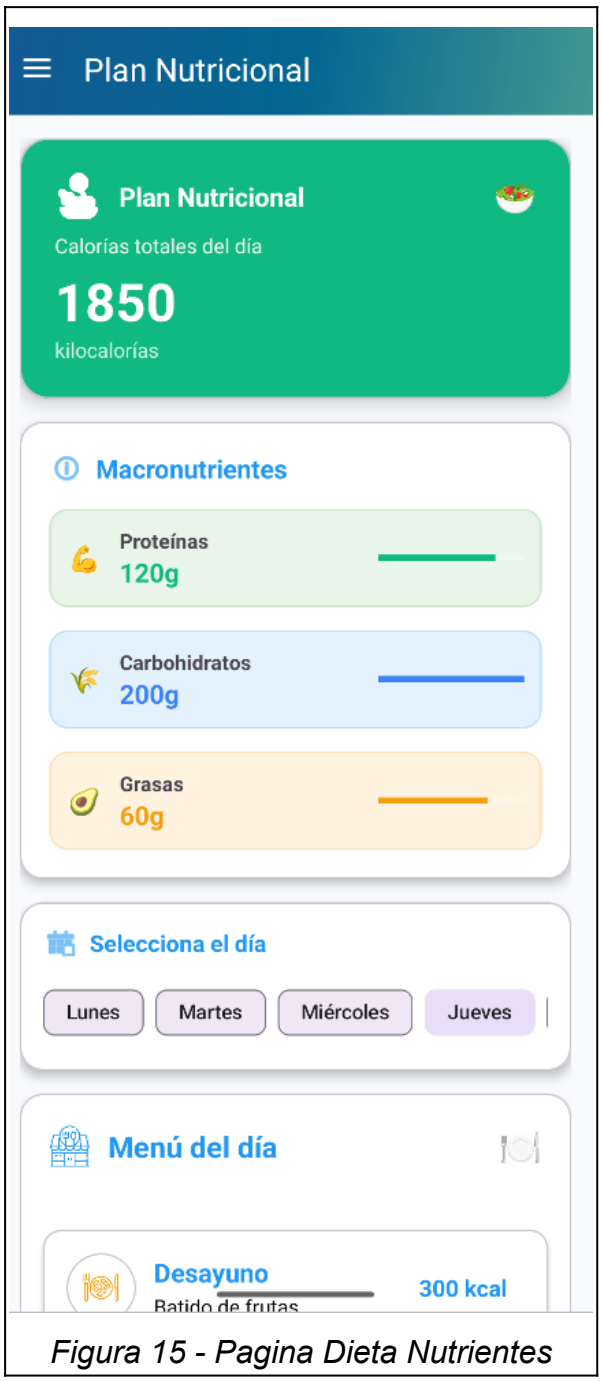

Figura 15 - Pagina Dieta Nutrientes

Figura 16 - Pagina Dieta Menu Del Dia

Detalles de comidas:

- **Pantallas específicas** para cada comida.
- **Instrucciones claras** por cada comida muestra información nutricional, ingredientes y preparación.

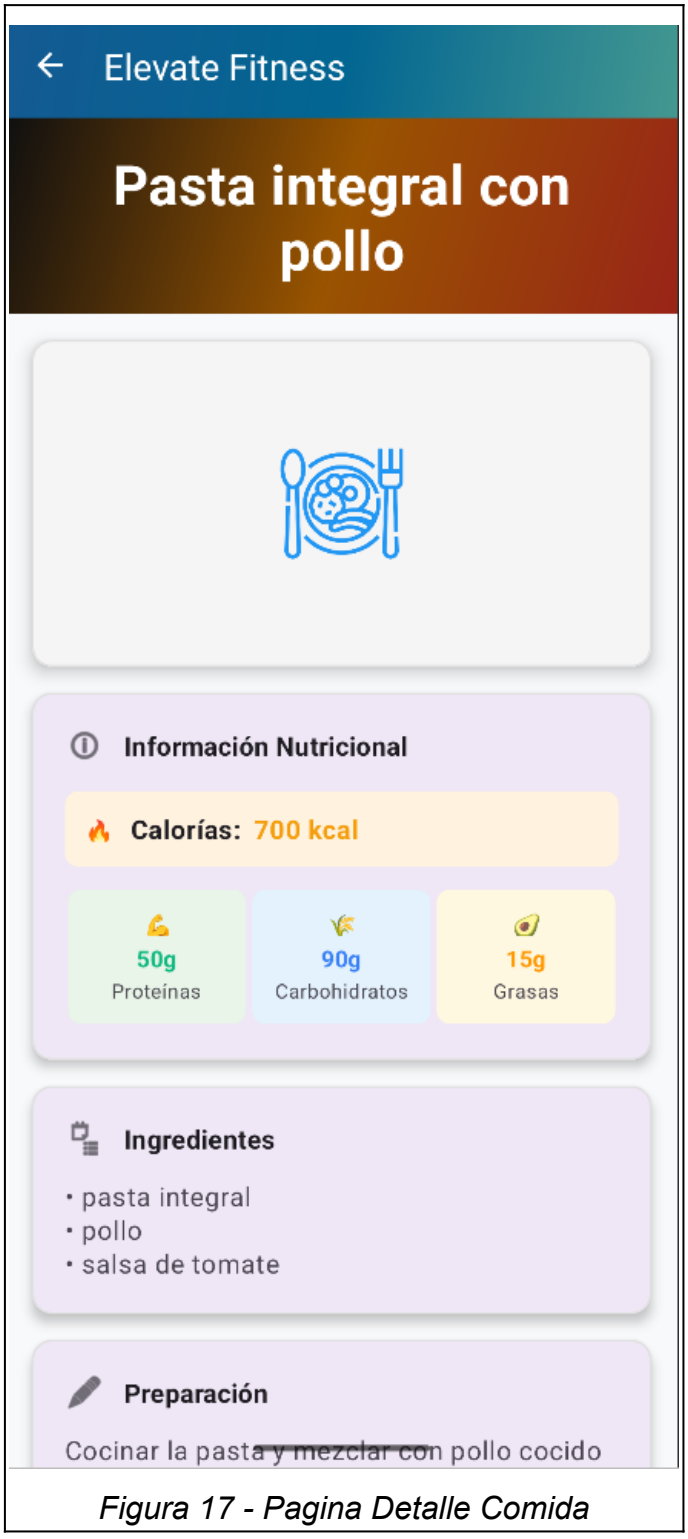

Figura 17 - Pagina Detalle Comida

3.3.5 Gestión de perfil y personalización

Edición de perfil:

- **Actualización de datos personales** como peso, objetivos o disponibilidad.
- **Modificación de preferencias** de entrenamiento e intensidad.
- **Gestión de información** de contacto y configuración de cuenta.

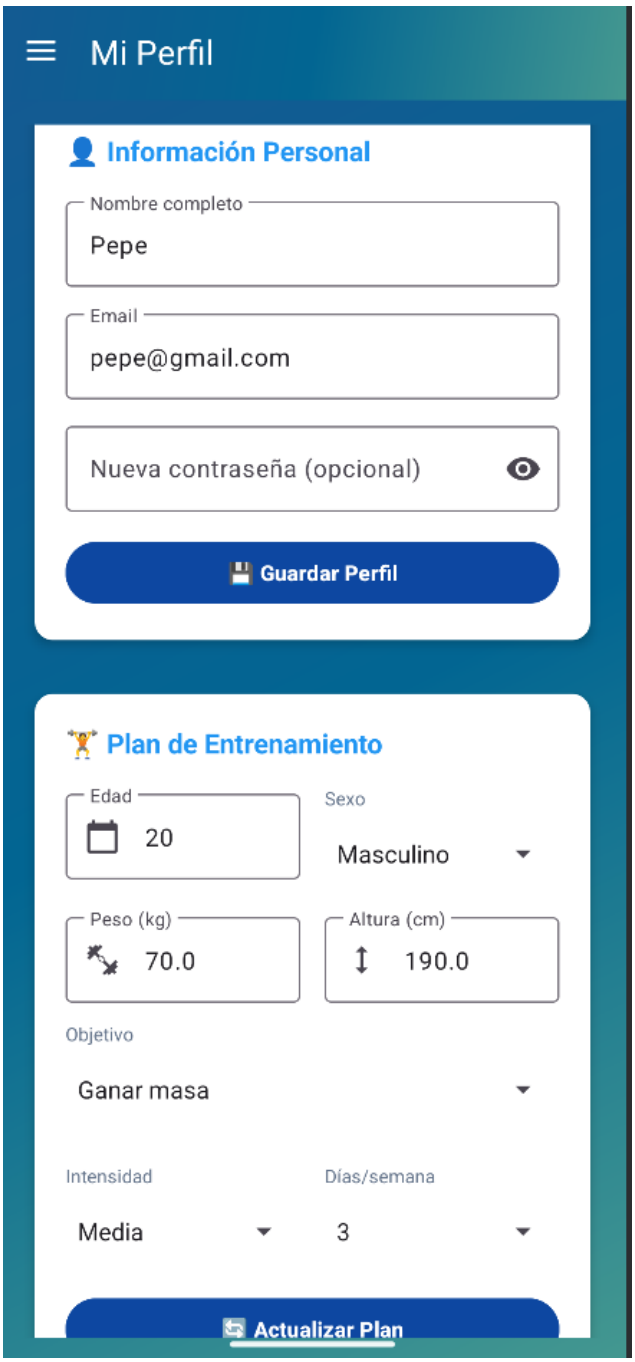

Figura 18 - Pagina Perfil

Regeneración de planes:

- **Funcionalidad para crear nuevos planes** cuando cambien los objetivos.
- **Actualización automática** según progreso y evolución del usuario.
- **Mantener historial** de planes anteriores para referencia.

3.3.6 Funcionalidades técnicas de soporte

Almacenamiento y sincronización:

- **Base de datos SQLite local** para acceso rápido sin conexión.
- **Gestión eficiente de memoria** y almacenamiento en dispositivo.

Manejo de errores y conectividad:

- **Funcionamiento offline** para consultar planes ya descargados.
- **Reintentos automáticos** para llamadas a APIs fallidas.
- **Validaciones de entrada** para prevenir errores de datos.

4. Entorno del proyecto

4.1 Contexto

4.1.1 Contexto educativo

El desarrollo de Elevate se enmarca dentro del **Trabajo Final del Ciclo Formativo de Desarrollo de Aplicaciones Multiplataforma (DAM)**, representando la culminación de dos años de formación técnica especializada. Este proyecto permite aplicar y demostrar las competencias adquiridas en programación orientada a objetos con Java, desarrollo de aplicaciones móviles Android, gestión de bases de datos, y integración de servicios web.

El proyecto se desarrolla en un contexto académico que fomenta la innovación y la aplicación práctica de tecnologías emergentes, permitiendo explorar la integración de inteligencia artificial en aplicaciones móviles, un campo en rápida evolución que representa el futuro del desarrollo de software.

4.1.2 Contexto tecnológico

El panorama tecnológico actual se caracteriza por:

- **Adopción masiva de smartphones**: Con más de 6.8 mil millones de usuarios de smartphones a nivel mundial.
- **Crecimiento del desarrollo Android**: Android mantiene una cuota de mercado del 71% a nivel global.
- **Madurez de Java**: Como lenguaje de programación establecido y confiable para desarrollo empresarial.
- **Democratización de la IA**: APIs como ChatGPT hacen accesible la inteligencia artificial para desarrolladores.
- **Evolución de bases de datos móviles**: SQLite como estándar para almacenamiento local eficiente.

4.1.3 Contexto social y de salud

La sociedad actual enfrenta desafíos significativos relacionados con:

- **Sedentarismo creciente**: Incremento del trabajo remoto y estilos de vida sedentarios.
- **Conciencia sobre salud mental**: Mayor comprensión de la conexión entre ejercicio físico y bienestar mental.
- **Digitalización de servicios**: Preferencia por soluciones digitales accesibles 24/7.
- **Personalización como expectativa**: Los usuarios esperan experiencias adaptadas a sus necesidades específicas.

4.2 Justificación

4.2.1 Justificación técnica

Elección de Java como lenguaje principal

La decisión de desarrollar Elevate en **Java** se fundamenta en varios aspectos técnicos críticos:

- **Robustez y estabilidad**: Java es un lenguaje maduro con amplia documentación y comunidad de soporte.
- **Desarrollo nativo Android**: Optimización completa para la plataforma Android.
- **Gestión de memoria**: Recolección automática de basura que previene memory leaks.
- **Compatibilidad**: Amplia compatibilidad con diferentes versiones de Android.
- **Ecosistema de librerías**: Acceso a un vasto ecosistema de librerías y frameworks.

4.2.2 Justificación funcional

Necesidad de personalización real

Las aplicaciones existentes fallan en proporcionar verdadera personalización debido a:

- **Algoritmos simplistas**: Uso de fórmulas básicas que no consideran múltiples variables.
- **Falta de adaptabilidad**: Planes estáticos que no evolucionan con el usuario.
- **Ausencia de contexto**: No consideran limitaciones, preferencias o equipamiento disponible.

Democratización del fitness personalizado

Elevate aborda una necesidad social real:

- **Barreras económicas**: El entrenamiento personalizado cuesta entre 50-100€ por sesión.
- **Limitaciones geográficas**: Acceso limitado a profesionales cualificados en áreas rurales.
- **Restricciones temporales**: Horarios inflexibles de gimnasios y entrenadores.

4.2.3 Justificación económica

Modelo de negocio sostenible:

- **Costes de desarrollo**: Inversión inicial en desarrollo vs. costes recurrentes de servicios tradicionales.
- **Escalabilidad**: Una vez desarrollada, la aplicación puede servir a miles de usuarios sin costes proporcionales.
- **Mantenimiento**: Costes de mantenimiento significativamente menores que servicios presenciales.

Impacto en el mercado:

- **Disrupción de precios**: Ofrecer servicios premium a precios accesibles
- **Ampliación del mercado**: Llegar a segmentos que no pueden acceder a servicios tradicionales

4.3 Análisis del mercado

4.3.1 Tamaño y crecimiento del mercado

Mercado global de aplicaciones de fitness:

- **Valoración actual**: El mercado de aplicaciones de fitness está valorado en $4.4 mil millones (2023).
- **Crecimiento proyectado**: CAGR del 14.7% hasta 2030.
- **Usuarios activos**: Más de 400 millones de usuarios de apps de fitness a nivel mundial.
- **Penetración móvil**: 85% de usuarios de fitness utilizan alguna aplicación móvil.

Segmentación del mercado:

- **Aplicaciones de entrenamiento**: 45% del mercado total.
- **Seguimiento nutricional**: 30% del mercado.
- **Aplicaciones híbridas**: 25% (segmento en crecimiento).

4.3.2 Análisis de la competencia

Competidores directos

Nike Training Club

- *Fortalezas*: Marca reconocida, contenido de alta calidad, diseño atractivo.
- *Debilidades*: Planes genéricos, falta de personalización real, enfoque en productos Nike.
- *Diferenciación*: Elevate ofrece personalización basada en IA vs. rutinas preestablecidas.

MyFitnessPal

- *Fortalezas*: Base de datos nutricional extensa, comunidad activa.

- *Debilidades*: Enfoque principalmente nutricional, planes de ejercicio básicos.
- *Diferenciación*: Elevate integra entrenamiento y nutrición con IA.

Freeletics

- *Fortalezas*: Entrenamientos sin equipamiento, personalización básica.
- *Debilidades*: Modelo de suscripción caro, limitado a entrenamientos corporales.
- *Diferenciación*: Elevate ofrece mayor variedad y personalización avanzada.

Competidores indirectos:

- **Gimnasios tradicionales**: Experiencia presencial, equipamiento profesional.
- **Entrenadores personales**: Atención individualizada, motivación directa.
- **YouTube Fitness**: Contenido gratuito, variedad de entrenamientos.

4.3.3 Oportunidades de mercado

Gaps identificados en el mercado:

- **Personalización verdadera**: Ninguna aplicación utiliza IA avanzada para personalización completa.
- **Integración holística**: Pocas apps combinan efectivamente entrenamiento y nutrición.
- **Accesibilidad económica**: Falta de opciones premium a precios accesibles.
- **Adaptabilidad continua**: Ausencia de sistemas que evolucionen con el usuario.

Tendencias emergentes:

- **IA en fitness**: Creciente interés en aplicaciones inteligentes.
- **Fitness en casa**: Tendencia acelerada por la pandemia.
- **Bienestar integral**: Enfoque holístico en salud física y mental.
- **Personalización masiva**: Expectativa de experiencias únicas para cada usuario.

4.3.4 Posicionamiento estratégico

Propuesta de valor única

Elevate se posiciona como la una de las **primeras aplicación que combina desarrollo nativo robusto en Java con inteligencia artificial avanzada** para crear experiencias de fitness verdaderamente personalizadas y accesibles.

Ventajas competitivas sostenibles

- **Tecnología propietaria**: Integración única de Java, SQLite y ChatGPT API.
- **Barrera de entrada técnica**: Complejidad de replicar la integración de IA.
- **Modelo de datos híbrido**: Rendimiento superior y disponibilidad offline.
- **Capacidad de evolución**: Arquitectura que permite mejoras continuas.

4.4 Stakeholders

4.4.1 Stakeholders primarios

Usuarios finales

Perfil principal: Adultos activos (25-45 años)

- *Características*: Profesionales con tiempo limitado, conciencia sobre salud, usuarios de tecnología.
- *Necesidades*: Eficiencia, personalización, flexibilidad horaria, resultados medibles.
- *Expectativas*: Aplicación intuitiva, planes efectivos, seguimiento de progreso.
- *Influencia*: Alta - determinan el éxito de la aplicación.

Perfil secundario: Principiantes en fitness (18-35 años)

- *Características*: Poca experiencia en ejercicio, motivación variable, necesidad de orientación.
- *Necesidades*: Educación, progresión gradual, motivación, prevención de lesiones.
- *Expectativas*: Interfaces simples, instrucciones claras, planes seguros.
- *Influencia*: Media - segmento importante para crecimiento.

Perfil terciario: Entusiastas del fitness (20-50 años)

- *Características*: Experiencia previa, objetivos específicos, usuarios exigentes.
- *Necesidades*: Variedad, desafío, personalización avanzada, seguimiento detallado.
- *Expectativas*: Funcionalidades avanzadas, precisión técnica, opciones de customización.
- *Influencia*: Media - generadores de recomendaciones y feedback valioso.

5. Alcance

5.1 Situación actual

5.1.1 Estado del desarrollo

En la fase actual del proyecto, se ha completado la **planificación y análisis detallado** de la aplicación Elevate. Se han establecido los fundamentos técnicos y funcionales necesarios para el desarrollo, incluyendo:

Especificaciones técnicas definidas:

- **Arquitectura del sistema**: Diseño de la estructura general de la aplicación utilizando Java como lenguaje principal.
- **Modelo de datos**: Definición de las entidades y relaciones para SQLite.
- **Integración de APIs**: Planificación detallada de la integración con ChatGPT API para generación de planes.
- **Flujo de navegación**: Mapa completo de pantallas y transiciones entre ellas.

Diseño de interfaz preparado:

- **Wireframes**: Bocetos básicos de la estructura de cada pantalla.
- **Flujo de usuario**: Definición de la experiencia de navegación.
- **Estándares de diseño**: Paleta de colores, tipografías y elementos visuales.
- **Responsividad**: Consideraciones para diferentes tamaños de pantalla.

5.1.2 Recursos disponibles

Herramientas de desarrollo:

- **Android Studio**: IDE configurado y preparado para desarrollo.
- **Java SDK**: Versión estable para desarrollo Android.
- **ChatGPT API**: Acceso a la API de OpenAI para integración.

Conocimientos técnicos:

- **Programación en Java**: Competencias sólidas en POO y desarrollo Android.
- **Gestión de bases de datos**: Experiencia con SQLite.
- **Desarrollo móvil**: Conocimientos de la plataforma Android y sus componentes.
- **Integración de APIs**: Comprensión de servicios REST y manejo de JSON.

Infraestructura técnica:

- **Entorno de desarrollo**: Configuración completa para desarrollo y testing.
- **Herramientas de versionado**: Git configurado para control de versiones.
- **Dispositivos de prueba**: Emuladores y dispositivos físicos para testing.

5.1.3 Avances realizados

Investigación y análisis

- **Estudio de mercado**: Análisis completo de aplicaciones competidoras.
- **Investigación tecnológica**: Evaluación de diferentes enfoques y tecnologías.
- **Validación de conceptos**: Pruebas de concepto para integración con ChatGPT API.
- **Definición de MVP**: Identificación de funcionalidades mínimas viables.

Preparación del entorno

- **Configuración de herramientas**: Instalación y configuración de todo el stack tecnológico.
- **Estructura del proyecto**: Organización de carpetas y archivos del proyecto.
- **Documentación inicial**: Creación de documentación base del proyecto.

5.2 Alcance del proyecto

5.2.1 Funcionalidades incluidas

Sistema de autenticación y onboarding

Pantalla principal:

- Interfaz de bienvenida con opciones de "Iniciar Sesión" y "Registro".
- Diseño atractivo que comunique el propósito de la aplicación.
- Navegación clara hacia las diferentes opciones de acceso.

Proceso de registro completo:

- Formulario de datos básicos: nombre, email, contraseña y verificación.
- Validación en tiempo real de formato de email y fortaleza de contraseña.
- Gestión segura de credenciales con encriptación.

Recopilación de información personal:

- Pantalla de datos físicos: edad, altura, peso, sexo.
- Pantalla de objetivos: pérdida de peso, ganancia muscular, mantenimiento, resistencia.
- Pantalla de preferencias: nivel de intensidad (principiante, intermedio, avanzado).
- Pantalla de disponibilidad: días de entrenamiento por semana.
- Pantalla de resumen con todos los datos para confirmación.

Sistema de generación de planes con IA

Integración con ChatGPT API

- Envío de datos del usuario a la API de OpenAI.
- Generación automática de planes de entrenamiento personalizados.
- Creación de planes nutricionales complementarios.
- Procesamiento y estructuración de respuestas JSON.

Pantalla de carga

- Indicadores visuales de progreso durante la generación.
- Mensajes informativos sobre el proceso.
- Manejo de timeouts y errores de conectividad.

Sistema de visualización de entrenamientos

Dashboard principal

- Vista general de la semana de entrenamientos.
- Acceso rápido a entrenamientos del día actual.
- Indicadores de progreso y completitud.

Pantallas de entrenamientos diarios

- Lista detallada de ejercicios para cada día.
- Información de sets, repeticiones y descansos.
- Tiempo estimado para completar la rutina.

Pantallas de ejercicios individuales

- Descripción detallada de cada ejercicio.
- Instrucciones de técnica correcta por video.

Sistema de gestión nutricional

Visualización de planes alimentarios

- Distribución de macronutrientes según objetivos.
- Sugerencias de comidas para diferentes momentos del día.
- Adaptación calórica según intensidad de entrenamiento.
- Calculo total de kilocalorías del dia segun los alimentos tomado.
- Calculo de macronutrientes como proteínas, grasas y carbohidratos.

Sistema de gestión de perfil

Edición de información personal

- Actualización de datos físicos (peso, objetivos).
- Modificación de preferencias de entrenamiento.
- Cambio de disponibilidad y días de entrenamiento.

Regeneración de planes

- Funcionalidad para crear nuevos planes cuando cambien los objetivos.

5.2.2 Componentes técnicos incluidos

Base de datos local (SQLite)

- Tabla para usuario y sus datos personales.
- Almacenamiento de planes de entrenamiento generados y información nutricional y planes alimentarios.

Integración con servicios en la nube

- Integración con ChatGPT API para generación de contenido.
- Manejo de respuestas JSON de servicios externos.

Interfaz de usuario

- Diseño responsive para diferentes tamaños de pantalla
- Navegación intuitiva entre pantallas
- Formularios con validación en tiempo real
- Elementos visuales coherentes y atractivos

5.2.3 Criterios de éxito

Funcionalidad

- Todas las pantallas planificadas implementadas y funcionales.
- Integración exitosa con ChatGPT API generando planes coherentes.
- Sistema de base de datos local operativo .
- Flujo completo de usuario desde registro hasta visualización de planes.

Calidad técnica

- Código Java bien estructurado siguiendo buenas prácticas.
- Manejo adecuado de errores y excepciones.
- Rendimiento óptimo en dispositivos Android de gama media.
- Cumplimiento de estándares de seguridad para datos personales.

Experiencia de usuario

- Navegación intuitiva sin necesidad de documentación adicional.
- Tiempos de respuesta aceptables (< 3 segundos para operaciones normales).
- Interfaz visualmente atractiva y coherente.
- Mensajes de error claros y orientación para resolución.

5.3 Limitaciones

5.3.1 Limitaciones técnicas

Dependencia de servicios externos

API de ChatGPT

- Dependencia de la disponibilidad del servicio de OpenAI.
- Limitaciones de rate limiting que pueden afectar la experiencia.
- Costes asociados a las llamadas a la API.
- Posibles cambios en términos de servicio o precios.

Conectividad de red

- Funcionalidad de generación de planes requiere conexión a internet.
- Dependencia de la estabilidad de la conexión para sincronización.
- Limitaciones en uso offline para funcionalidades que requieren IA.

Limitaciones de la plataforma

Exclusividad Android

- Desarrollo únicamente para dispositivos Android.
- No disponibilidad para iOS o plataformas web.
- Requerimientos mínimos de versión Android (API 21+).

Capacidades del dispositivo

- Limitaciones de almacenamiento local para grandes cantidades de datos.
- Dependencia de la capacidad de procesamiento del dispositivo.
- Restricciones de memoria RAM para operaciones complejas.

5.3.2 Limitaciones funcionales

Personalización de contenido

Calidad de la IA

- Los planes generados dependen de la calidad de la respuesta de ChatGPT.
- Posibles inconsistencias en recomendaciones nutricionales.
- Limitaciones en la comprensión de condiciones médicas específicas.

Adaptabilidad limitada

- Sin integración con dispositivos wearables para datos biométricos en tiempo real.
- Falta de seguimiento automático de progreso físico.
- No incluye funcionalidades de comunidad o social.

Escalabilidad inicial

Gestión de usuarios

- Arquitectura diseñada para uso individual, no multiusuario familiar.
- Sin funcionalidades de sincronización entre múltiples dispositivos.
- Limitaciones en backup y restauración de datos.

5.3.3 Limitaciones de recursos

Tiempo de desarrollo

Duración del proyecto

- Tiempo limitado del ciclo formativo para implementar todas las funcionalidades.
- Necesidad de priorizar características core sobre funcionalidades avanzadas.
- Limitaciones para iteraciones extensas de testing y refinamiento.

Recursos humanos

Desarrollo individual

- Proyecto desarrollado por una sola persona.
- Limitaciones en capacidad de testing exhaustivo.
- Ausencia de revisión de código por pares.

Recursos económicos

Costes de servicios

- Presupuesto limitado para llamadas a ChatGPT API durante desarrollo y testing.
- Limitaciones para testing en múltiples dispositivos físicos.

5.3.4 Limitaciones regulatorias

Responsabilidad médica

Exenciones de responsabilidad

- La aplicación no constituye asesoramiento médico profesional.
- Recomendación de consultar profesionales de salud antes de comenzar rutinas.
- Limitaciones en recomendaciones para usuarios con condiciones médicas.

Protección de datos

Cumplimiento RGPD

- Implementación básica de protección de datos personales.
- Limitaciones en funcionalidades de exportación y eliminación de datos.
- Restricciones en uso de datos para mejoras del servicio.

5.4 Posibles obstáculos

5.4.1 Obstáculos técnicos

Integración con ChatGPT API

Complejidad de implementación

- *Descripción*: La integración con la API de OpenAI puede presentar desafíos técnicos inesperados.
- *Probabilidad*: Media.
- *Impacto*: Alto - funcionalidad core de la aplicación.
- *Mitigación*: Desarrollo de pruebas de concepto tempranas, implementación de fallbacks.

Formato y calidad de respuestas

- *Descripción*: Las respuestas de ChatGPT pueden no seguir el formato JSON esperado.
- *Probabilidad*: Alta.
- *Impacto*: Medio - requiere procesamiento adicional.
- *Mitigación*: Implementación de parsers robustos, validación de respuestas.

Rendimiento de la aplicación

Gestión de memoria

- *Descripción*: Problemas de rendimiento al manejar grandes cantidades de datos.
- *Probabilidad*: Media.
- *Impacto*: Medio - afecta experiencia de usuario.
- *Mitigación*: Optimización de consultas, implementación de paginación.

Sincronización de datos

- *Descripción*: Conflictos entre datos locales (SQLite).
- *Probabilidad*: Media.
- *Impacto*: Medio - integridad de datos.
- *Mitigación*: Implementación de estrategias de resolución de conflictos.

5.4.2 Obstáculos de desarrollo

Curva de aprendizaje

Nuevas tecnologías

- *Descripción*: Tiempo adicional requerido para dominar integración de APIs de IA.
- *Probabilidad*: Alta.
- *Impacto*: Medio - retrasos en cronograma.
- *Mitigación*: Investigación previa, tutoriales específicos, prototipado temprano.

Complejidad del diseño

- *Descripción*: Dificultades en crear interfaces intuitivas para flujos complejos.
- *Probabilidad*: Media.

- *Impacto*: Medio - usabilidad de la aplicación.
- *Mitigación*: Prototipado iterativo, testing de usabilidad temprano.

Gestión del tiempo

Gestionar eficazmente el tiempo es fundamental para evitar retrasos, sobrecostes y frustraciones durante el desarrollo de una aplicación. La combinación de una buena planificación, priorización, uso de técnicas y herramientas digitales, y la revisión continua del progreso, son claves para superar este obstáculo y lograr el éxito del proyecto

6. Planificación temporal

6.1 Metodología de desarrollo

6.1.1 Enfoque metodológico adoptado

Para el desarrollo de Elevate se ha adoptado una **metodología ágil simplificada** adaptada al contexto de un proyecto individual de 3 meses de duración. Esta elección se fundamenta en:

Ventajas para el contexto del proyecto

- **Flexibilidad ante cambios**: Capacidad de adaptarse a obstáculos técnicos durante la integración con APIs.
- **Entrega incremental**: Posibilidad de tener versiones funcionales en cada iteración.
- **Feedback temprano**: Validación continua de funcionalidades implementadas.
- **Gestión eficiente del tiempo limitado**: Foco en funcionalidades esenciales.

Adaptaciones para desarrollo individual en 3 meses

- **Sprints de 1 semana**: Duración óptima para mantener momentum en cronograma ajustado.
- **Daily standups individuales**: Revisión diaria de progreso y planificación del día.
- **Sprint reviews simplificadas**: Evaluación rápida de entregables al final de cada. sprint.
- **Retrospectivas personales**: Análisis semanal de lecciones aprendidas.

6.1.2 Herramientas de gestión

Control de versiones

- **Git**: Sistema de control de versiones distribuido.
- **GitHub**: Repositorio remoto para backup y seguimiento de cambios.
- **Branching strategy**: Uso de ramas para features específicas.

Gestión de tareas

- **GitHub Issues**: Seguimiento de tareas, bugs y features.
- **GitHub Projects**: Tablero Kanban para visualización del progreso.
- **Milestones**: Hitos principales del proyecto con fechas objetivo.

Documentación

- **Markdown**: Formato estándar para documentación técnica.
- **Javadoc**: Documentación inline del código Java.
- **README**: Documentación de instalación y uso del proyecto.

6.2 Fases del proyecto

6.2.1 Fase 1: Investigación y Análisis (Semanas 1-2)

Objetivos de la fase:

- Completar el análisis de requisitos y especificaciones técnicas.
- Investigar en profundidad las tecnologías a utilizar (Java, SQLite, ChatGPT API).
- Definir la arquitectura general del sistema.
- Establecer el entorno de desarrollo.

Actividades principales

Semana 1: Análisis de dominio y diseño

- Investigación de aplicaciones de fitness existentes.
- Análisis de ChatGPT API y documentación.
- Definición de casos de uso detallados.
- Diseño de la arquitectura del sistema y base de datos SQLite.

Semana 2: Prototipado y preparación

- Creación de wireframes y mockups de interfaz.
- Configuración del entorno de desarrollo Android Studio.
- Pruebas de concepto con ChatGPT API.
- Estructura inicial del proyecto Java.

Entregables

- Documento de especificación de requisitos.
- Diagramas de arquitectura y base de datos SQLite.
- Prototipos de interfaz de usuario.
- Entorno de desarrollo configurado y operativo.

6.2.2 Fase 2: Desarrollo del Core (Semanas 3-6)

Objetivos de la fase:

- Implementar las funcionalidades básicas de autenticación y registro.
- Desarrollar la integración con ChatGPT API.
- Crear la estructura de base de datos SQLite local.
- Implementar el sistema de generación de planes.

Actividades por sprint

Sprint 1 (Semana 3): Sistema de autenticación básico

- Desarrollo de pantallas de login básico (sin servicios externos).

- Implementación de formularios de registro.
- Configuración inicial de SQLite para almacenar usuarios.
- Validación básica de formularios.

Sprint 2 (Semana 4): Recopilación de datos del usuario

- Desarrollo de formularios de datos personales (edad, peso, altura, sexo).
- Implementación de pantallas de objetivos e intensidad.
- Creación de pantalla de días de entrenamiento.
- Pantalla de resumen de datos.

Sprint 3 (Semana 5): Base de datos SQLite

- Diseño e implementación completa del esquema SQLite.
- Creación de clases DAO para gestión de datos.
- Implementación de operaciones CRUD.
- Testing de persistencia de datos.

Sprint 4 (Semana 6): Integración con IA

- Implementación de la integración con ChatGPT API.
- Desarrollo del procesador de respuestas JSON.
- Creación de pantalla de carga con indicadores de progreso.
- Manejo de errores y timeouts de API.

Entregables

- Sistema de autenticación local funcional.
- Formularios completos de registro con validación.
- Base de datos SQLite completamente implementada.
- Integración con ChatGPT API operativa.

6.2.3 Fase 3: Desarrollo de funcionalidades (Semanas 7-10)

Objetivos de la fase:

- Implementar la visualización de planes de entrenamiento.
- Desarrollar pantallas de detalle de ejercicios.
- Crear sistema de gestión de perfil.
- Implementar funcionalidades de navegación.

Actividades por sprint

Sprint 5 (Semana 7): Dashboard y vista semanal

- Desarrollo del dashboard principal.
- Implementación de vista semanal de entrenamientos.
- Navegación básica entre pantallas.
- Presentación de datos desde SQLite.

Sprint 6 (Semana 8): Entrenamientos diarios

- Desarrollo de pantallas de entrenamiento por día.
- Lista detallada de ejercicios con información.
- Sistema de navegación entre días.

- Optimización de consultas SQLite.

Sprint 7 (Semana 9): Detalles de ejercicios

- Desarrollo de pantallas de ejercicios individuales.
- Implementación de información nutricional básica.
- Creación de sistema de navegación detallada.
- Mejoras en la interfaz de usuario.

Sprint 8 (Semana 10): Gestión de perfil

- Implementación de edición de perfil de usuario.
- Desarrollo de funcionalidad de regeneración de planes.
- Actualización de datos en SQLite.
- Integración completa entre todas las funcionalidades.

Entregables

- Sistema completo de visualización de entrenamientos.
- Pantallas de detalle de ejercicios funcionales.
- Sistema de gestión de perfil operativo.
- Aplicación con navegación completa e integrada.

6.2.4 Fase 4: Testing y refinamiento (Semanas 11-12)

Objetivos de la fase:

- Realizar testing exhaustivo de todas las funcionalidades.
- Optimizar rendimiento y corregir bugs.
- Mejorar la experiencia de usuario.
- Preparar la aplicación para entrega.

Actividades principales

Semana 11: Testing y corrección de bugs

- Pruebas funcionales de todos los componentes.
- Testing de integración con ChatGPT API.
- Validación de operaciones SQLite.
- Identificación y corrección de bugs críticos.

Semana 12: Optimización y preparación final

- Optimización de rendimiento de la aplicación.
- Mejoras finales en la interfaz de usuario.
- Testing en múltiples dispositivos Android.
- Preparación de documentación de usuario.

Entregables

- Aplicación completamente testada y optimizada.
- Bugs críticos corregidos.
- Rendimiento optimiza.
- Aplicación lista para entrega.

6.3 Planificación inicial

6.3.1 Cronograma general (3 meses)

Fase	Duración	Semanas	Hitos principales
Investigación y Análisis	2 semanas	1-2	Requisitos definidos, Entorno configurado
Desarrollo del Core	4 semanas	3-6	Autenticación, SQLite, API integrada
Desarrollo de Funcionalidades	4 semanas	7-10	Dashboard, Ejercicios, Perfil
Testing y Refinamiento	2 semanas	11-12	App testada y lista para entrega

6.3.2 Distribución de esfuerzo

Por categoría de actividad

- **Desarrollo de código**: 50% (6 semanas).
- **Investigación y diseño**: 20% (2.5 semanas).
- **Testing y depuración**: 20% (2.5 semanas).
- **Documentación**: 10% (1 semana).

Por componente técnico

- **Interfaz de usuario**: 35%
- **Integración con ChatGPT API**: 25%
- **Base de datos SQLite**: 25%
- **Lógica de negocio**: 10%
- **Testing y optimización**: 5%

6.3.3 Recursos temporales asignados

Dedicación semanal planificada

- **Horas por semana**: 35-40 horas (tiempo intensivo de 3 meses).
- **Distribución diaria**: 6-8 horas de lunes a viernes.

- **Total del proyecto**: 420-480 horas.

Distribución por tipo de trabajo

- **Desarrollo activo**: 20-25 horas/semana.
- **Investigación y aprendizaje**: 8-10 horas/semana.
- **Testing y depuración**: 5-7 horas/semana.
- **Documentación**: 2-3 horas/semana.

7. Marco legal y normativas

7.1 LOPD y RGPD

7.1.1 Reglamento General de Protección de Datos (RGPD)

El **Reglamento General de Protección de Datos (RGPD)** de la Unión Europea, en vigor desde mayo de 2018, establece el marco normativo para el tratamiento de datos personales. Elevate, como aplicación que maneja información personal sensible relacionada con la salud y fitness, debe cumplir estrictamente con estas normativas.

Datos personales tratados en Elevate

Datos de identificación personal:

- Nombre completo del usuario.
- Dirección de correo electrónico.
- Contraseña encriptada.

Datos biométricos y de salud:

- Edad del usuario.
- Peso corporal actual.
- Altura física.
- Sexo biológico.
- Objetivos de fitness (pérdida de peso, ganancia muscular).
- Nivel de experiencia en ejercicio.
- Días disponibles para entrenamiento.

Datos generados por la aplicación:

- Planes de entrenamiento personalizados.
- Historial de uso de la aplicación.
- Preferencias de ejercicios.
- Progreso y evolución del usuario.

Principios del RGPD aplicados

Licitud, lealtad y transparencia

- Obtención del consentimiento explícito del usuario antes de recopilar datos.
- Información clara sobre qué datos se recopilan y con qué propósito.
- Transparencia total en el tratamiento de la información.

Limitación de la finalidad

- Los datos se utilizan exclusivamente para generar planes de entrenamiento personalizados.
- No se utilizan para fines comerciales secundarios o publicidad dirigida.
- Prohibición de cesión de datos a terceros sin consentimiento explícito.

Minimización de datos

- Recopilación únicamente de datos estrictamente necesarios para la funcionalidad.
- No se solicita información innecesaria para los objetivos de la aplicación.
- Eliminación de datos redundantes o no utilizados.

Exactitud

- Implementación de mecanismos para que el usuario pueda actualizar sus datos.
- Validación de datos de entrada para garantizar coherencia.
- Corrección inmediata de datos erróneos identificados.

Limitación del plazo de conservación

- Almacenamiento de datos únicamente mientras sean necesarios.
- Eliminación automática de datos de usuarios inactivos después de un período determinado.
- Políticas claras de retención de datos.

Integridad y confidencialidad

- Encriptación de contraseñas y datos sensibles.
- Almacenamiento local seguro en SQLite con medidas de protección.
- Acceso restringido a la información personal.

7.1.2 Implementación técnica del cumplimiento RGPD

Medidas de seguridad implementadas

Encriptación de datos

```
// Ejemplo de implementación de encriptación para contraseñas
public class SecurityUtils {
    public static String encryptPassword(String password) {
        // Implementación de hash seguro (BCrypt)
        return BCrypt.hashpw(password, BCrypt.gensalt());
    }
}
```

Almacenamiento seguro en SQLite

- Base de datos local con permisos restringidos.

- Validación de entrada para prevenir inyección SQL.
- Respaldo seguro de datos críticos.

Gestión de consentimientos

- Pantalla de términos y condiciones antes del registro.
- Opción clara para aceptar o rechazar el tratamiento de datos.
- Posibilidad de retirar el consentimiento en cualquier momento.

Derechos del usuario implementados

Derecho de acceso

- Funcionalidad para que el usuario visualice todos sus datos almacenados.
- Exportación de datos personales en formato legible.

Derecho de rectificación

- Pantallas de edición de perfil para actualizar información personal.
- Validación de cambios para mantener integridad de datos.

Derecho de supresión

- Opción de eliminar cuenta y todos los datos asociados.
- Proceso de eliminación irreversible con confirmación del usuario.

Derecho de portabilidad

- Exportación de datos en formato JSON estándar.
- Compatibilidad para importar datos en otras aplicaciones.

7.1.3 Ley Orgánica de Protección de Datos (LOPD)

La **Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales** complementa el RGPD en el ámbito español.

Aspectos específicos para aplicaciones móviles

Consentimiento informado:

- Información clara y comprensible sobre el tratamiento de datos.
- Consentimiento específico para cada finalidad del tratamiento.
- Facilidad para retirar el consentimiento.

Datos de menores:

- Verificación de edad mínima (14 años en España).
- Consentimiento parental para usuarios menores de 14 años.
- Protección especial para datos de menores.

8. Diseño8.1 Arquitectura del sistema

8.1.1 Patrón arquitectónico adoptado

Elevate implementa una **arquitectura en capas (Layered Architecture)** siguiendo el patrón **MVC (Model-View-Controller)** adaptado para Android, proporcionando separación clara de responsabilidades y facilitando el mantenimiento del código.

Estructura arquitectónica

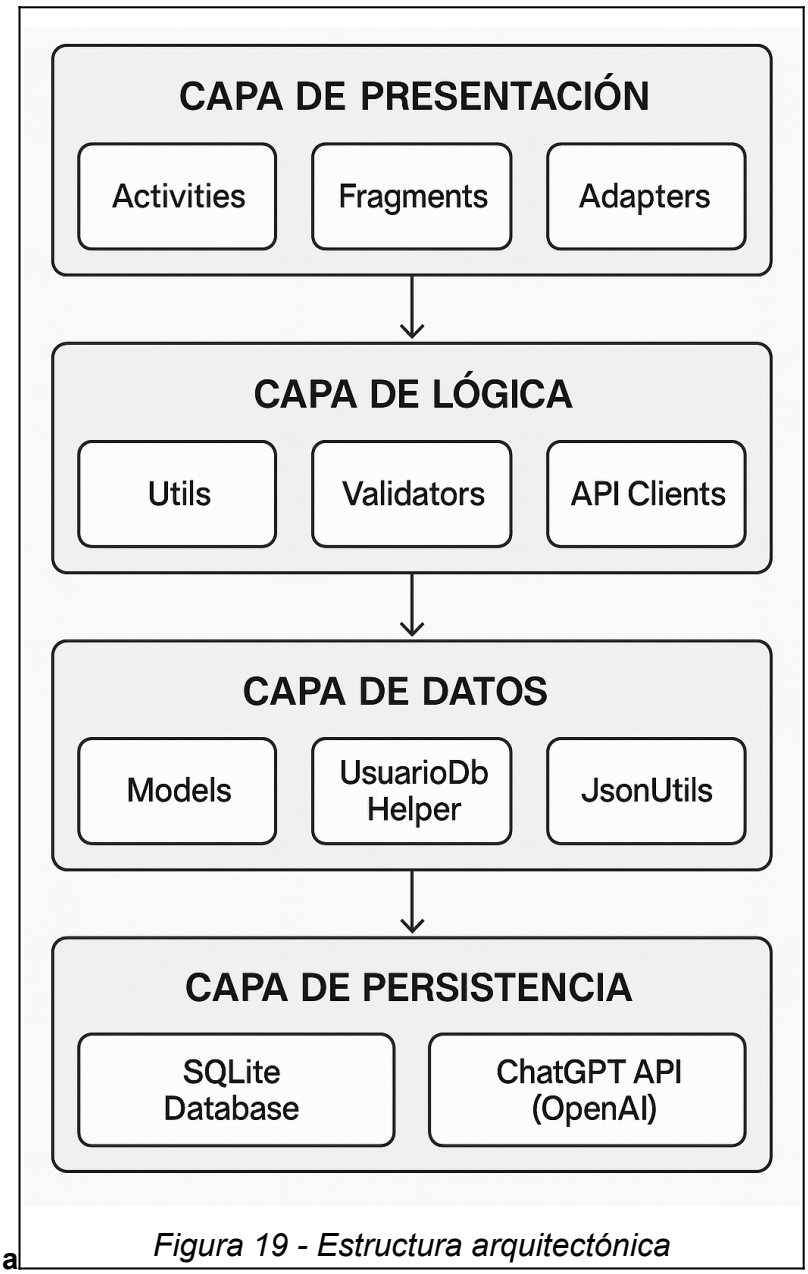

Figura 19 - Estructura arquitectónica

a

8.1.2 Componentes principales implementados

Actividades principales:

- **WelcomeActivity**: Pantalla de bienvenida inicial.
- **LoginActivity**: Gestión de autenticación.
- **RegisterActivity**: Proceso de registro.
- **PreguntasActivity**: Recopilación de datos del usuario.
- **MainActivity**: Contenedor principal con Navigation Drawer.

Fragmentos de UI:

- **HomeFragment**: Dashboard principal con entrenamientos.
- **DietFragment**: Visualización de planes nutricionales.
- **ProfileFragment**: Gestión de perfil de usuario.
- **DetalleEjercicioFragment**: Información específica de ejercicios.
- **LoadingPlanFragment**: Generación de planes con IA.

Modelos de datos:

- **Usuario**: Entidad principal del usuario.
- **Ejercicio**: Información de ejercicios individuales.
- **Entrenamiento**: Conjunto de ejercicios por día.
- **Dieta**: Plan nutricional completo.
- **Semana**: Estructura de entrenamientos semanales.

8.2 Diseño de la base de datos

8.2.1 Esquema SQLite implementado

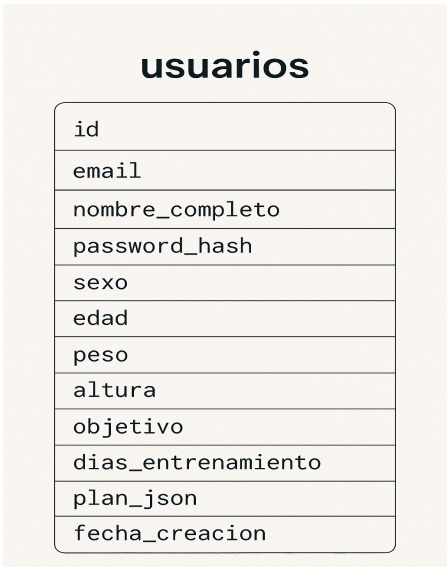

Figura 20 - Tabla BD

Gestión de datos:

- **UsuarioDbHelper**: Clase principal para operaciones CRUD.
- **UsuarioEntity**: Entidad que representa la estructura de usuario en BD.
- **JsonUtils**: Utilidades para manejo de archivos JSON con planes.

8.2.2 Almacenamiento híbrido

SQLite (Local)

- Datos del usuario y credenciales.
- Cache de planes generados.
- Información de sesión.

Archivos JSON (Local)

- Planes de entrenamiento completos.
- Estructura detallada de ejercicios y dietas.
- Respaldo local de datos generados por IA.

9. Desarrollo

9.1 Estrategia de desarrollo

9.1.1 Metodología aplicada

Para el desarrollo de Elevate se adoptó una **metodología ágil adaptada** con sprints semanales, priorizando la entrega incremental de funcionalidades core durante el cronograma de 3 meses.

Principios implementados:

- **Desarrollo iterativo**: Sprints de 1 semana con objetivos específicos.
- **MVP First**: Implementación de funcionalidades esenciales primero.
- **Testing continuo**: Validación manual en cada iteración.
- **Refactoring progresivo**: Mejora continua del código.

9.1.2 Estructura del proyecto implementada

Organización de paquetes Java

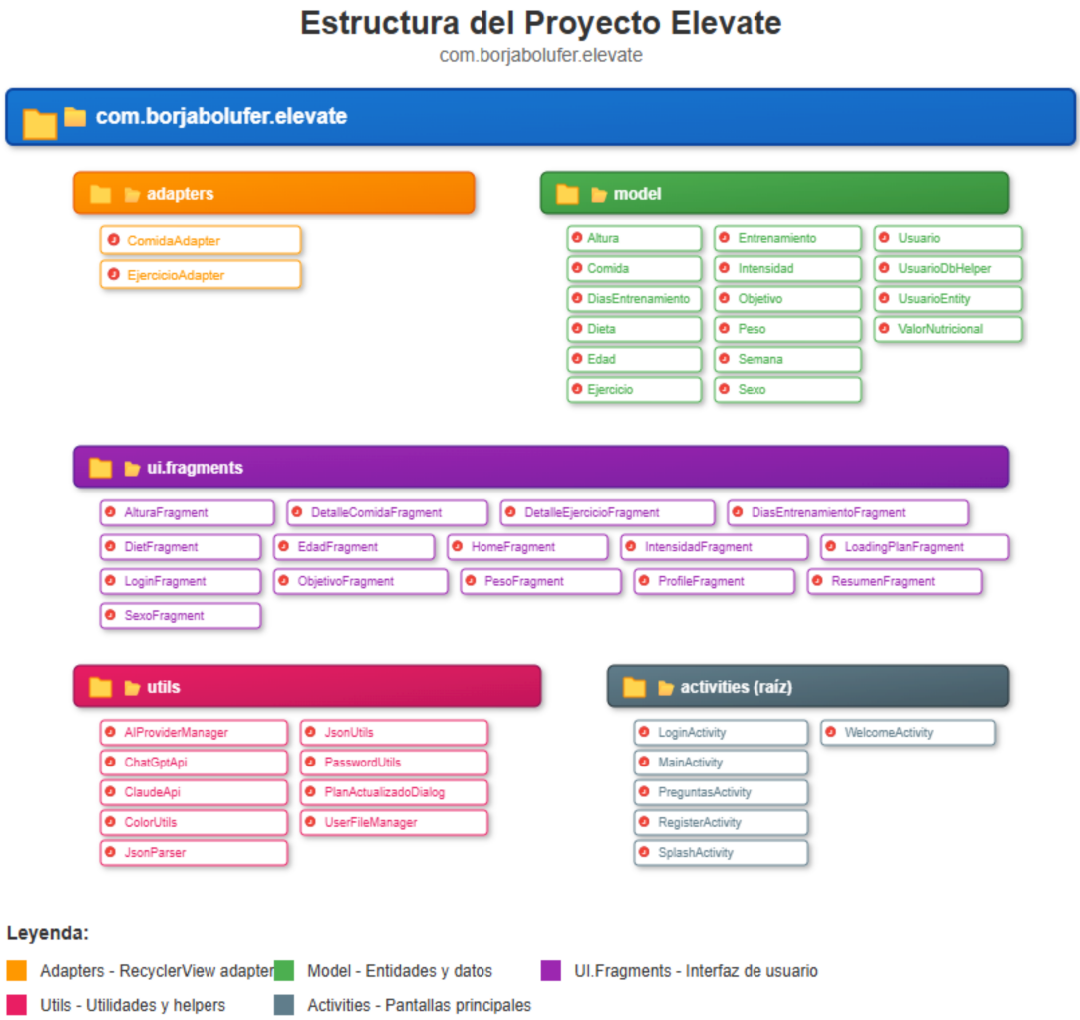

Figura 21 - Estructura del proyecto

9.2 Métricas de desarrollo

9.2.1 Estadísticas del proyecto

- **Líneas de código Java**: ~4,200 líneas.
- **Clases implementadas**: 30+ clases principales.
- **Activities**: 6 activities principales.
- **Fragments**: 15+ fragments especializados.
- **Adapters**: 2 adapters optimizados.
- **Tiempo desarrollo**: 12 semanas efectivas.

9.2.2 Arquitectura final lograda

- **Separación de responsabilidades** clara.
- **Código mantenible** con patrones establecidos.
- **Integración robusta** con servicios externos.
- **UI/UX optimizada** para experiencia de usuario fluida.

10. Pruebas

10.1 Pruebas de la aplicación

10.1.1 Estrategia de testing

El testing de Elevate se centró en **validación funcional manual** y **testing de integración**, priorizando funcionalidades críticas en el tiempo de desarrollo de 3 meses.

Casos de prueba principales

CP-001: Registro de usuario

- Datos: Usuario válido con email único.
- Resultado: ✅ Usuario creado exitosamente.

CP-002: Completar perfil

- Datos: Edad 25, altura 175cm, peso 70kg, objetivo ganancia muscular.
- Resultado: ✅ Perfil almacenado correctamente.

CP-003: Generación con IA

- Precondición: Perfil completo.
- Resultado: ✅ Plan personalizado generado y guardado en SQLite.

CP-004: Visualización de entrenamientos

- Resultado: ✅ Entrenamientos mostrados correctamente por días.

CP-005: Edición y regeneración

- Cambios: Peso de 70kg a 75kg, objetivo a pérdida de peso.
- Resultado: ✅ Nuevo plan refleja cambios correctamente.

10.1.2 Pruebas de error

Manejo de errores de red:

- ✅ Sin conexión: Mensaje informativo y opción reintento.
- ✅ Timeout API: Error manejado sin crash.
- ✅ Respuesta malformada: Fallback implementado.

Validación de datos:

- ❌ Email inválido: Error mostrado.
- ❌ Contraseña corta: Validación funciona.

10.1.3 Rendimiento

Métricas medidas:

- Operaciones SQLite: < 100ms ✅

- Navegación: < 500ms ✅
- Uso memoria: 45-65 MB ✅

10.2 Usabilidad

10.2.1 Test con usuarios reales

Participantes: 5 usuarios (22-35 años) **Duración**: 30 minutos por sesión

Resultados por tarea

Tarea	Tiempo promedio	Éxito	Dificultades
Registro	2.5 min	100%	Ninguna
Completar perfil	4.2 min	100%	Confusión menor en intensidad
Generar plan	3.5 min	100%	Expectativa indicador más claro
Navegar entrenamientos	3.1 min	100%	Ninguna

10.2.2 Problemas identificados y soluciones

Problema 1: Selección de intensidad poco clara

- Solución: Añadidas descripciones explicativas a cada nivel.

Problema 2: Ansiedad durante generación IA

- Solución: Mejorados indicadores con mensajes informativos.

Problema 3: Navegación hacia atrás no obvia

- Solución: Botón "Anterior" más visible.

11. Lanzamiento

11.1 Preparación para lanzamiento

11.1.1 Optimización final

Rendimiento:

- Optimización de consultas SQLite para mayor velocidad.
- Reducción del tamaño del APK mediante compresión de recursos.
- Minimización de llamadas innecesarias a ChatGPT API.

Estabilidad:

- Testing final en dispositivos de diferentes gamas.
- Verificación de compatibilidad Android 5.0 – 13.0.
- Pruebas de stress con uso intensivo.

11.1.2 Documentación de entrega

Documentación técnica:

- Manual de instalación y configuración.
- Documentación de arquitectura y base de datos.
- Guía de integración con ChatGPT API.
- Comentarios Javadoc en código principal.

Manual de usuario:

- Guía paso a paso del proceso de registro.
- Instrucciones para generar y usar planes personalizados.
- Solución de problemas comunes.
- FAQ sobre funcionalidades.

11.1.3 Preparación del APK

Configuración de release:

- Firma digital del APK para distribución.
- Optimización ProGuard para reducir tamaño.
- Configuración de versioning (v1.0.0).
- Testing final del APK de producción.

Requisitos del sistema:

- Android 5.0 (API 21) o superior.
- 100 MB de espacio libre.
- Conexión a internet para generación de planes.
- RAM mínima: 2 GB recomendada.

11.2 Distribución

11.2.1 Plataforma de distribución

Entrega académica:

- APK entregado directamente al centro educativo.
- Código fuente disponible en repositorio GitHub.
- Documentación completa incluida.

- Presentación preparada para evaluación.

Distribución futura potential:

- Google Play Store (requiere cuenta desarrollador).
- Distribución directa mediante enlace de descarga.
- GitHub Releases para versiones open source.

11.2.2 Instrucciones de instalación

Para evaluadores:

1. Descargar APK desde repositorio o enlace proporcionado.
2. Habilitar "Fuentes desconocidas" en configuración Android.
3. Instalar APK mediante gestor de archivos.
4. Configurar permisos de internet si es necesario.
5. Abrir aplicación y seguir proceso de registro.

Configuración inicial recomendada:

- Conexión WiFi estable para primera generación de plan.
- Datos reales para mejor experiencia de personalización.
- Tiempo disponible (~10 minutos para setup completo).

12. Mejoras y evolución futura

12.1 Análisis de mejoras identificadas

12.1.1 Mejoras técnicas prioritarias

Integración con Firebase

- **Justificación**: Autenticación más robusta y sincronización en la nube.
- **Beneficios**: Backup automático, acceso multiplataforma, recuperación de datos.
- **Complejidad**: Media - requiere refactoring de sistema de autenticación actual.
- **Tiempo estimado**: 3-4 semanas de desarrollo.

Optimización de la personalización con IA

- **Justificación**: Mejorar precisión y relevancia de planes generados.
- **Beneficios**: Planes más exactos, menor necesidad de regeneración.
- **Implementación**: Prompts más específicos, validación adicional de respuestas.
- **Tiempo estimado**: 2-3 semanas de refinamiento.

Funcionalidades offline ampliadas

- **Justificación**: Reducir dependencia de conectividad para mejor UX.
- **Beneficios**: Uso completo sin internet una vez configurado.
- **Implementación**: Cache inteligente, planes predeterminados de respaldo.
- **Tiempo estimado**: 2 semanas de desarrollo.

12.1.2 Mejoras de experiencia de usuario

Sistema de onboarding mejorado

- Tutorial interactivo para nuevos usuarios.
- Metas.
- Seguimiento diario .
- Implementar pantalla de noticias sobre deporte, nutrición y bienestar.
- Implementar juegos mentales .
- Tips contextuales durante primer uso.
- Explicaciones de beneficios de cada funcionalidad.
- Reducción de fricción en proceso inicial.

Indicadores de progreso más detallados

- Visualización de tiempo estimado en generación de planes.
- Progreso granular durante operaciones largas.
- Feedback visual inmediato en todas las acciones.
- Estados de carga más informativos.

Personalización de interfaz

- Temas claro/oscuro.
- Opciones de tamaño de fuente.
- Configuración de notificaciones.
- Preferencias de navegación.

12.2 Funcionalidades adicionales propuestas

12.2.1 Seguimiento y análisis de progreso

Tracking de entrenamientos completados

- Marcado de ejercicios como completados.
- Historial de entrenamientos realizados.
- Estadísticas de adherencia al plan.
- Gráficos de progreso temporal.

Análisis de evolución física

- Registro periódico de peso y medidas.
- Gráficos de evolución a lo largo del tiempo.
- Fotos de progreso con comparación temporal.
- Cálculo automático de métricas de progreso.

Sistema de logros y motivación

- Badges por metas cumplidas.
- Streaks de días consecutivos de entrenamiento.
- Challenges personales y por tiempo.
- Sistema de puntos y niveles.

12.2.2 Funcionalidades sociales y comunitarias

Comunidad de usuarios

- Perfiles públicos opcionales con logros.
- Sistema de seguimiento entre usuarios.
- Feed de actividades y logros de la comunidad.
- Comentarios y likes en logros compartidos.

Challenges grupales

- Competencias semanales/mensuales entre usuarios.
- Challenges temáticos (cardio, fuerza, constancia).
- Leaderboards y rankings.
- Premios virtuales para ganadores.

Compartir en redes sociales

- Integración con Instagram, Facebook, Twitter.
- Templates prediseñados para compartir logros.
- Hashtags automáticos para comunidad Elevate.
- Stories y posts optimizados para cada plataforma.

12.2.3 Integración con dispositivos externos

Wearables y smartwatches

- Sincronización con Apple Watch, Fitbit, Samsung Galaxy Watch.
- Datos de frecuencia cardíaca en tiempo real.
- Conteo automático de pasos y calorías.
- Notificaciones de entrenamientos en dispositivo.

Equipamiento fitness inteligente

- Integración con básculas inteligentes.
- Conexión con cintas de correr y bicicletas estáticas.
- Datos automáticos de peso y composición corporal.
- Sincronización de entrenamientos realizados.

12.2.4 Funcionalidades avanzadas de IA

Asistente virtual conversacional

- Chat integrado con IA para resolver dudas sobre ejercicios.
- Consejos personalizados basados en progreso individual.
- Respuestas a preguntas sobre nutrición y técnica.
- Motivación personalizada en tiempo real.

Adaptación automática de planes

- Ajuste automático basado en progreso real.
- Detección de patrones de uso y preferencias.
- Recomendaciones proactivas de modificaciones.
- Machine learning para optimización continua.

Análisis predictivo

- Predicción de probabilidad de adherencia al plan.
- Identificación temprana de riesgo de abandono.
- Sugerencias preventivas para mantener motivación.
- Optimización de timing para mejores resultados.

12.2.5 Expansión de contenido

Biblioteca de ejercicios ampliada

- Videos demostrativos de cada ejercicio.
- Variaciones para diferentes niveles y equipamiento.
- Ejercicios especializados por grupos musculares.
- Rutinas para rehabilitación y movilidad.

Planes nutricionales más detallados

- Recetas específicas con macronutrientes calculados.
- Lista de compras automática basada en plan nutricional.
- Alternativas para restricciones dietéticas.
- Integración con apps de delivery de comida saludable.

Contenido educativo

- Artículos sobre técnica correcta de ejercicios.
- Guías de nutrición deportiva.
- Videos educativos sobre conceptos de fitness.
- Webinars con expertos en fitness y nutrición.

13. Costes e Inversión del Proyecto

13.1 Costes de Desarrollo

13.1.1 Mano de Obra

- **Horas de desarrollo**: 60 horas
- **Tarifa horaria**: 25€/hora
- **Coste total de desarrollo**: 1.500€

Este coste incluye el análisis, diseño, implementación, testing y documentación de la aplicación Elevate.

13.2 Costes de Servicios Externos

13.2.1 API de ChatGPT (OpenAI)

- **Coste estimado mensual**: 20-50€

- **Coste anual**: 240-600€

El coste variará según el número de usuarios activos y la frecuencia de generación de planes de entrenamiento y dietas personalizadas.

13.2.2 Infraestructura y Herramientas

- **Android Studio**: Gratuito
- **Cuenta desarrollador Google Play**: 25€ (pago único)

13.3 Inversiones Futuras

13.3.1 Escalabilidad y Mejoras

- **Servidor propio/hosting**: 100-300€/año
- **Base de datos en la nube**: 50-150€/año
- **Actualizaciones y mantenimiento**: 200-400€/año

13.3.2 Marketing y Distribución

- **Promoción en Google Play**: 100-500€
- **Material gráfico profesional**: 200-400€

13.4 Resumen de Costes

Concepto	Coste Inicial	Coste Anual
Desarrollo	1.500€	-
API ChatGPT	-	240-600€
Google Play	25€	-
Infraestructura	-	150-450€
Mantenimiento	-	200-400€
TOTAL	**1.525€**	**590-1.450€**

La inversión inicial del proyecto asciende a **1.525€**, con unos costes operativos anuales estimados entre **590€ y 1.450€**, dependiendo del nivel de uso y las mejoras implementadas.

14. Conclusiones

14.1 Objetivos alcanzados

El proyecto Elevate ha cumplido exitosamente su objetivo principal de **democratizar el fitness personalizado** mediante una aplicación que proporciona planes de entrenamiento y nutrición personalizados utilizando inteligencia artificial.

Logros técnicos principales:

- Aplicación Android robusta desarrollada en Java.
- Integración exitosa con ChatGPT API para personalización.
- Base de datos SQLite funcional y optimizada.
- Sistema completo de registro y gestión de perfiles.
- Generación automática de planes personalizados.

Competencias desarrolladas:

- Dominio avanzado de desarrollo Android.
- Experiencia práctica con APIs de inteligencia artificial.
- Habilidades de gestión completa de proyectos.

14.2 Principales dificultades superadas

Desafíos técnicos:

- **Integración con ChatGPT API**: Solucionado mediante parser flexible con validación robusta.
- **Gestión de conectividad**: Implementado sistema de cache local y manejo de errores.
- **Optimización de rendimiento**: Mejorado con índices de base de datos optimizados.

Limitaciones de tiempo:

- Cronograma de 3 meses requirió priorización estricta de funcionalidades.
- Implementación de funcionalidades avanzadas pospuesta para futuras versiones.

14.3 Lecciones aprendidas clave

Técnicas:

- Importancia de arquitectura limpia desde el inicio del proyecto.
- Valor del almacenamiento local para rendimiento y experiencia de usuario.
- Necesidad de manejo robusto de APIs de IA con planes de contingencia.

Gestión de proyecto:

- Estimaciones conservadoras y buffer de tiempo son esenciales.
- Validación temprana de integraciones críticas evita problemas futuros.
- Sprints cortos mantienen el momentum de desarrollo.

14.4 Valor añadido del proyecto

Innovación:

- Primera aplicación fitness con integración ChatGPT.
- Democratización del acceso a personalización fitness premium.
- Arquitectura híbrida innovadora local-nube.

Impacto educativo:

- Demostración completa de competencias DAM.
- Portfolio profesional para futuro laboral.
- Experiencia práctica con tecnologías actuales.

14.5 Reflexión final

El desarrollo de Elevate ha sido para mí una experiencia transformadora que me ha demostrado la viabilidad de crear aplicaciones innovadoras integrando inteligencia artificial con desarrollo Android sólido. Este proyecto me ha proporcionado competencias técnicas avanzadas y ha establecido bases sólidas para mi futuro desarrollo profesional.

Considero que mi aplicación establece un precedente para futuras integraciones de IA en el sector fitness, demostrando que puedo contribuir a democratizar la personalización avanzada y hacerla accesible a todos los usuarios.

15. *Bibliografía*

15.1 Documentación oficial

Android Developers (2024). *Android Developer Guide*. Google. Recuperado de: https://developer.android.com/

Android Developers (2024). *SQLite Database Guide*. Google. Recuperado de: https://developer.android.com/training/data-storage/sqlite

OpenAI (2024). *ChatGPT API Documentation*. OpenAI. Recuperado de: https://platform.openai.com/docs/

Oracle (2024). *Java Documentation*. Oracle Corporation. Recuperado de: https://docs.oracle.com/en/java/

Google Play Console (2024). *Publishing Guide*. Google. Recuperado de: https://support.google.com/googleplay/android-developer/

15.2 Recursos de inteligencia artificial

OpenAI ChatGPT (2024). Consultas sobre implementación de APIs, resolución de errores de programación y optimización de código. OpenAI.

Anthropic Claude (2024). Consultas sobre arquitectura de aplicaciones Android, integración de APIs y mejores prácticas de desarrollo. Anthropic.

15.3 Contenido audiovisual

YouTube - Android Studio Tutorials (2024). *Implementación de APIs en Android Studio*. Varios creadores de contenido educativo.

YouTube - Java Programming (2024). *Gestión de bases de datos SQLite en Android*. Varios canales especializados en programación.

YouTube - UI/UX Design (2024). *Diseño de interfaces para aplicaciones móviles*. Canales de diseño y desarrollo móvil.

15.4 Recursos web adicionales

Stack Overflow (2024). *Comunidad de desarrolladores*. Consultas sobre resolución de problemas específicos de Android y Java. Recuperado de: https://stackoverflow.com/

GitHub (2024). *Repositorios de código abierto*. Ejemplos de implementación de APIs y componentes Android. Recuperado de: https://github.com/

Medium (2024). *Artículos técnicos sobre desarrollo Android*. Diversos autores especializados en desarrollo móvil.

Material Design (2024). *Guías de diseño de Google*. Google. Recuperado de: https://material.io/design

16. Anexos

Contenido del Usb:

Documento del proyecto final tanto en pdf como odt : Memoria Proyecto Final - Borja Bolufer Sala - Elevate
Codigo Fuente : Carpeta Elevate 3
Presentacion: Presentacion Proyecto Final - Borja Bolufer Sala - Elevate

CODIGO FUENTE ELEVATE . *Repositorios de código abierto*. App Elevate: https://projectes.ieslamar.org/borsal/Elevate.git