UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA
INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

GRADO EN INGENIERÍA DEL SOFTWARE

Plataforma web para la planificación y seguimiento de rutinas de entrenamiento deportivo y de dietas

Platform for the planning and monitoring of sports training routines and diets

Realizado por
Christian Berdejo Sánchez

Tutorizado por
Eduardo Guzmán de los Riscos

Departamento
Lenguajes y Ciencias de la Computación
UNIVERSIDAD DE MÁLAGA

MÁLAGA, (JULIO 2024)

UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
GRADUADA EN INGENIERÍA DEL SOFTWARE

Plataforma web para la planificación y seguimiento de rutinas de entrenamiento deportivo y de dietas

Platform for the planning and monitoring of sports training routines and diets

Realizado por
Christian Berdejo Sánchez

Tutorizado por
Eduardo Guzmán de los Riscos

Departamento
Lenguajes y Ciencias de la Computación

UNIVERSIDAD DE MÁLAGA
MÁLAGA, JULIO DE 2024

Fecha defensa: Julio de 2024

Resumen

En la actualidad, la salud se ha convertido en una prioridad esencial para las personas de todas las edades, lo que ha incrementado la demanda de soluciones tecnológicas que promuevan un estilo de vida activo y saludable. Como ingeniero, es fundamental estar a la vanguardia en el desarrollo de páginas web y aplicaciones, no solo para satisfacer esta necesidad creciente, sino también para consolidar habilidades técnicas en un entorno digital en constante evolución.

Este proyecto nace de la convicción de crear una aplicación *full stack* dedicada al deporte que sirva como una herramienta poderosa para fomentar hábitos saludables y, al mismo tiempo, fortalecer competencias cruciales en el ámbito del desarrollo web y móvil. El resultado es una web y aplicación móvil a la que se ha denominado como "Fit-Match" que facilita el registro del ejercicio físico, permite la búsqueda y creación de plantillas de entrenamiento y da la opción a compartirlas con otros usuarios.

Para la implementación se han utilizado tecnologías modernas y robustas. En el *back-end*, se ha empleado "Express" y "Node.js", junto con "TypeScript" para mejorar la calidad del código. En el *front-end*, se ha optado por "Flutter", permitiendo crear interfaces nativas para múltiples plataformas. La base de datos se ha gestionado con "MySQL", asegurando un almacenamiento eficiente y confiable. Esta combinación garantiza una aplicación robusta y de alto rendimiento.

Palabras clave: Salud, Entrenamiento, Flutter, MySql, Node.js

Abstract

Nowadays, health has become a fundamental priority for people of all ages, increasing the demand for technological solutions that promote an active and healthy lifestyle. As an engineer, it is crucial to be at the forefront of web and application development, not only to meet this growing need but also to consolidate technical skills in an ever-evolving digital environment.

This project arises from the conviction to create a full-stack application dedicated to sports, serving as a powerful tool to foster healthy habits while simultaneously strengthening essential competencies in web and mobile development. The result is a web and mobile application named "Fit-Match," which facilitates the recording of physical exercise, allows the search and creation of workout templates, and gives the option to share them with other users.

For the implementation, modern and robust technologies have been utilized. In the back-end, "Express" and "Node.js" have been employed, along with "TypeScript" to improve code quality. In the front-end, "Flutter" was chosen, allowing the creation of native interfaces for multiple platforms. The database has been managed with "MySQL," ensuring efficient and reliable storage. This combination guarantees a robust and high-performance application.

Keywords: Health, Training, Flutter, MySQL, Node.js

Índice

1

Introducción

La importancia del ejercicio físico para la salud y el bienestar general de las personas es un hecho ampliamente reconocido y aceptado en la comunidad científica y médica. Según la organización mundial de la salud la actividad física regular puede ayudar a prevenir y controlar una gran variedad de enfermedades y afecciones, tales como la obesidad, la diabetes, la hipertensión arterial, las enfermedades cardiovasculares, la osteoporosis y la depresión, entre otras. Además, el ejercicio también puede mejorar la calidad de vida, aumentar la energía y la resistencia, promover el sueño y el descanso, y mejorar el estado de ánimo y la autoestima [1].

Sin embargo, a pesar de los numerosos beneficios del ejercicio físico, muchas personas todavía no realizan la cantidad recomendada de actividad física, ya sea por falta de tiempo, motivación, conocimientos o recursos. Esto de forma progresiva está provocando un lento aumento de la popularidad del ejercicio y por tanto a la búsqueda de nuevas formas de hacerlo más accesible, atractivo y eficaz.

En este contexto, el desarrollo de aplicaciones de entrenamiento se ha convertido en una herramienta clave para promover y facilitar el ejercicio físico. Las aplicaciones de entrenamiento pueden ofrecer una gran variedad de ventajas, como la personalización, la gamificación, la socialización, la monitorización y la retroalimentación, que pueden ayudar a motivar, guiar e informar a los usuarios en su práctica del ejercicio.

El desarrollo de una aplicación de entrenamiento no es una tarea sencilla ni trivial. Requiere el conocimiento y la aplicación de diversas herramientas, técnicas y metodologías, que abarcan desde el diseño y la programación, hasta la usabilidad y la

accesibilidad, pasando por la interfaz de usuario y la interacción, la seguridad y la privacidad, la analítica y el análisis de datos.

Por ello, este trabajo fin de grado (TFG) busca desarrollar una aplicación de entrenamiento para web y dispositivos móviles, que cubra una necesidad detectada en el mercado actual y que satisfaga las demandas y expectativas de los usuarios. La aplicación estará diseñada y desarrollada siguiendo buenas prácticas y los estándares, y tendrá en cuenta los aspectos técnicos, funcionales y estéticos.

La elección de desarrollar la aplicación para web y dispositivos móviles no es casual, sino que responde a una estrategia de alcance y accesibilidad, ya que estas plataformas permiten llegar a un gran número de usuarios, independientemente de su lugar, momento o dispositivo.

En definitiva, este TFG supone un reto y una oportunidad para demostrar y aplicar los conocimientos y habilidades adquiridos durante la formación académica, y para contribuir al campo del desarrollo de aplicaciones de entrenamiento, con el fin de mejorar la salud y el bienestar de las personas.

1.1 Motivación

En la era digital actual, el desarrollo de páginas web y aplicaciones no solo se ha convertido en una necesidad imperante para empresas y emprendedores, sino también en una plataforma vital para la interacción humana, el comercio, y la educación, entre otros. Sin embargo, la complejidad de estos sistemas digitales aumenta a medida que avanza la tecnología. La primera gran motivación de este proyecto es explorar y aplicar tecnologías y metodologías innovadoras en el **desarrollo de aplicaciones** *full stack*, poniendo especial énfasis en la implementación de buenas prácticas y la modularización mediante el uso arquitecturas en capas para facilitar la escalabilidad y mantenimiento a largo plazo y poseer un *software* eficaz y de calidad. La implementación de estas técnicas es esencial para enfrentar los desafíos que presenta el mundo dinámico del desarrollo web y de aplicaciones, lo que permite la creación de soluciones sólidas, seguras y adaptables.

Por otro lado, en un mundo cada vez más consciente de la importancia de la salud y el bienestar**, promover el ejercicio físico** se ha convertido en un objetivo crucial para las comunidades globales. El sedentarismo plantea riesgos significativos para la salud física y mental de las personas. Aquí radica la segunda motivación de este trabajo: utilizar la

tecnología para fomentar hábitos de vida saludables, especialmente el ejercicio físico. Mediante el desarrollo de una aplicación que incorpore seguimiento de rendimiento en ejercicios realizados y que permita crear y compartir distintas rutinas de entrenamiento. Se busca no solo captar la atención de los usuarios, sino también motivarlos de manera efectiva hacia la adopción de un estilo de vida más activo. Esta aplicación aspira a ser un puente entre la tecnología y el bienestar físico, demostrando cómo las herramientas digitales pueden ser aliadas en la mejora de la calidad de vida.

1.1.2 Objetivos del proyecto

El objetivo principal es la creación de una plataforma web integral y altamente funcional, diseñada específicamente para ser un pilar en la planificación y seguimiento de rutinas de entrenamiento. Este espacio digital buscará ser una herramienta útil para quienes deseen mejorar su estado físico y bienestar general. Otros objetivos serán

O1. **Desarrollo de *back-end* y *front-end***: El proyecto es un desafío técnico en el que se construirán los cimientos y la fachada de nuestra plataforma; es decir, el *back-end* (lado del servidor) y el *front-end* (lado del cliente). El *back-end* actuará como el cerebro detrás de operaciones, manejando datos, asegurando la lógica de la aplicación y protegiendo la privacidad de nuestros usuarios. Paralelamente, el *front-end* se convertirá en el rostro amigable de nuestra plataforma, ofreciendo una interfaz que invite a la interacción intuitiva.

O2. **Planificación de rutinas de entrenamiento personalizadas**: El objetivo es desarrollar un sistema adaptable que permita a los usuarios personalizar y crear sus propias rutinas de entrenamiento. Esta funcionalidad será el centro del proyecto y permitirá a cada persona configurar su propio camino hacia el bienestar, adaptándose completamente a sus necesidades y objetivos personales, desde seleccionar ejercicios hasta definir la cantidad de series y repeticiones.

O3. **Seguimiento de progreso y estadísticas personalizadas**: Los usuarios al registrar su progreso en los ejercicios a lo largo del tiempo podrán visualizar gráficas y un historial de los registros de sus entrenos con el fin de proporcionar una visión completa del progreso. Este seguimiento incluirá elementos como el peso levantado o las repeticiones realizadas.

O4. **Retroalimentación de usuarios**: Los usuarios deberán poder utilizar rutinas de entrenamiento planificadas por otros usuarios y proporcionar retroalimentación a base de reseñas o indicadores de "me gusta".

O5. **Seguridad de Datos**: La información de los usuarios será manipulada de forma segura, incluyendo encriptación de datos necesarios y validaciones para evitar la entrada de datos peligrosos o erróneos en la base de datos.

O6. **Interfaz de Usuario Amigable**: Se busca diseñar una interfaz atractiva e intuitiva y accesible para poder verse en la mayoría de los dispositivos. Esta interfaz será la puerta de entrada a todas las funcionalidades de la plataforma.

O7. **Optimización de Rendimiento**: Se buscará asegurar que la aplicación funcione ágil y eficientemente incluso

1.1.2 Antecedentes

Cuando se trata de aplicaciones que permiten a los usuarios crear y compartir entrenamientos, así como registrar sesiones de ejercicio, algunas de las más destacadas en el mercado son:

- "Keelo": Esta aplicación se centra en entrenamientos personalizados, adaptando los planes según el nivel de experiencia y el historial de entrenamientos del usuario. Se especializa en entrenamientos de fuerza y rutinas de "HIIT", y permite a los usuarios tanto organizar rutinas personalizadas como interactuar con entrenadores para consejos y seguimiento [2].
- "Training Peaks": Es una herramienta avanzada para atletas y entrenadores que ofrece una planificación detallada de los entrenamientos y metas [3].
- "Virtuagym": Ofrece una experiencia de entrenamiento virtual con un extenso catálogo de ejercicios categorizados por grupo muscular y nivel de dificultad, incluyendo videos tutoriales. Además, tiene funciones de planificación de entrenamientos y monitorización de actividades [4].
- "Trifecta": Se orienta hacia los aficionados del "Crossfit" y proporciona una colección de entrenamientos personalizados de cardio y fuerza. Una característica distintiva es su enfoque en la nutrición, permitiendo a los usuarios rastrear su dieta y el balance calórico [5].

- "7 Minute Workout Challenge": Esta aplicación promueve rutinas de ejercicio rápidas y eficientes de 7 minutos diarios, adecuadas para personas con horarios apretados [6].

1.1.3 Organización del documento

La memoria se ha dividido en apartados para facilitar la lectura. A continuación, se detallan los apartados de forma ordenada y sus propósitos:

1. **Introducción**: Este capítulo se ha centrado en describir la motivación del proyecto y se detallarán los objetivos que se buscan alcanzar con el desarrollo del proyecto.
2. **Tecnologías y herramientas utilizadas**: En este capítulo se enumeran y explican a alto nivel las tecnologías utilizadas.
3. **Especificación y análisis:** Este capítulo va dedicado a definir el proyecto a alto nivel y nombrar los requisitos funcionales y no funcionales. Este apartado se encarga también de desarrollar los principales casos de uso y diagramas de secuencia.
4. **Diseño del sistema:** Este apartado define claramente la arquitectura del sistema. Este apartado explicará la división del sistema en tres grandes subsistemas (base de datos, cliente y servidor) y explicará el diseño de cada subsistema.
5. **Maquetado de interfaz**: En este capítulo se comenta la importancia de maquetar interfaces mediante esquemas de pantallas y se dan unos pocos ejemplos.
6. **Implementación y pruebas**: Este capítulo se enfoca en abordar detalles de la implementación y las pruebas, destacando por las buenas prácticas y las soluciones efectivas seleccionadas para afrontar determinados desafíos.
7. **Conclusiones y Trabajos Futuros**: En este capítulo se resumen los principales puntos y se mencionan los objetivos que se han cumplido, las dificultades encontradas y algunas de las posibles ampliaciones del proyecto.

Por último, encontraremos un apartado de "Referencias", donde se podrán encontrar las fuentes citadas en el documento y dos apéndices que corresponden a un manual de instalación y manual de usuario.

2

Tecnologías y herramientas utilizadas

El proyecto se divide en tres partes muy definidas: base de datos, *back-end* y *front-end*. El *front-end* es la parte del proyecto de programación del lado del cliente, en otras palabras, se encarga de hacer interfaces, animaciones y todos los componentes visuales. Por otra parte, el *back-end* es el lado del servidor y se encarga de conectar la base de datos y otros servicios de terceros con el lado del cliente.

Para el *back-end* se ha utilizado "Node.js" [7], que es un entorno de ejecución que, de forma asíncrona, es capaz de enviar información desde el servidor al cliente [8]. El lenguaje de programación que se ha utilizado en este TFG es *TypeScript* [9] que hereda el comportamiento de *JavaScript* [10], lenguaje de programación orientado a objetos, y añade nuevas funcionalidades de las cuales destaca el tipado sensitivo al flujo. El tipado sensitivo al flujo es un sistema que permite indicar el tipo esperado en una variable. Es una herramienta muy poderosa que asegura la persistencia de datos y hace viable ver errores en tiempo de compilación y evitar problemas a largo plazo. Ayuda a reducir tanto el error

humano como la cantidad de problemas al desplegar el producto en producción. Es un recurso que facilita mucho el mantenimiento y corrección de posibles errores. También se ha usado "npm" [11] que es un gestor de paquetes que recomienda "Node.js" y admite la posibilidad de instalar funcionalidades de forma rápida y quedan indicadas en un fichero de dependencias, de forma que otros usuarios al instalar el proyecto puedan añadir esos paquetes rápidamente. "npm" actualmente es uno de los gestores de paquetes más grandes y actualmente posee más de 1.3 millones de paquetes con más 75 millones de descargas diarias [12]. En el contexto del *back-end*, se suele utilizar herramientas de mapeo relacional de objetos (ORM) ya que permite tener modelos tipados con el que realizar la mayoría de las operaciones y asegurar la persistencia de los datos. Además facilitan mucho las operaciones de acceso a la base de datos proporcionando una interfaz sencilla. Para este proyecto, el software ORM que se ha usado se denomina *PrismaORM* [13].

El *front-end* se ha desarrollado con *Flutter* [14], que es un kit de desarrollo de software con el que se puede escribir las interfaces para el lado del cliente utilizando Dart [15], lenguaje de programación de código abierto creado y mantenido por Google. Flutter consigue convertirse en una herramienta muy eficaz debido al uso de *widgets* que son componentes que te permiten interactuar con la aplicación. Se puede crear y utilizar estos componentes para darle forma a la aplicación y aprovechar la gran librería de *widgets* que ofrece Flutter.

Para programar tanto en el back-end como en el *front-end* se ha utilizado el editor de código Visual Studio Code [16] debido al gran número de extensiones que posee que agilizan el proceso de desarrollo y sobre todo por la facilidad que tiene para implementar un control de versiones. El control de versiones es imprescindible para la realización de un proyecto de gran tamaño, en este caso se realizará con Git [17] mediante el uso del repositorio GitHub [18].

Para la base de datos se ha utilizado MySQL [19], uno de los sistemas de gestión de bases de datos más populares en parte por su gran eficiencia, fiabilidad y facilidad a la hora de usarlo, junto con el hecho de que es de código abierto. MySQL posibilita gestionar modelos para una base de datos de forma relacional. Se ha utilizado el software HeidiSQL [20] para la comunicación con MySQL y creación de tablas.

Las imágenes se han guardado en la nube, en un proyecto en producción podría ser interesante tener servidores propios, pero para mantener la consistencia en el desarrollo a

la hora de programar en múltiples máquinas se ha optado por usar Cloudinary [21], en vez de guardar las imágenes en local.

3

Especificación y Análisis

3.1 Actores del sistema

En el sistema existen dos roles principales que son los de **cliente** y **administrador**. El cliente tendrá acceso a la herramienta y sus funcionalidades y el administrador se encargará de gestionar a los usuarios de rol de cliente, sus accesos a la aplicación y gestionará los ejercicios de la base de datos.

3.2 Requisitos funcionales

Con el fin de ofrecer un panorama general de la aplicación, se presentan los requisitos funcionales que guían el desarrollo de la solución, asegurando que cada elemento esté alineado con los objetivos generales de proporcionar una buena experiencia de usuario.

3.2.1 Requisitos funcionales comunes a todos los roles

RF1. Un usuario podrá iniciar sesión.

RF2. Un usuario podrá cerrar sesión.

RF3. Un usuario podrá crearse una cuenta.

RF4. Un usuario podrá editar sus credenciales, los datos de su perfil y preferencias.

RF5. Un usuario podrá visualizar las plantillas de entrenamiento que sean públicas y filtrarlas por nombre, intereses, objetivos, experiencia necesaria y equipamiento disponible.

3.1.2 Requisitos funcionales de usuario de rol cliente

RF6. Un usuario con rol de cliente podrá gestionar (crear, editar, eliminar) plantillas de entrenamiento.

RF7. Para cada plantilla de entrenamiento un usuario con rol de cliente podrá crear, editar y eliminar sesiones de entrenamiento.

RF8. Para cada sesión de entrenamiento, un usuario con rol de cliente podrá añadir ejercicios y podrá indicar un objetivo de rendimiento (número de repeticiones, tiempo ejecutando el ejercicio, ...).

RF9. Un usuario con rol de cliente podrá hacer pública o privada una plantilla de entrenamiento.

RF10. Un usuario con rol de cliente podrá crear, editar y eliminar ejercicios en su propia lista privada. Estos ejercicios solo serán visibles y accesibles por el usuario creador y no aparecerán en la lista de ejercicios de otros usuarios.

RF11. Un usuario con rol de cliente podrá guardar en la sección de "plantillas de entrenamientos activas" una plantilla de entrenamiento publicada por otro usuario.

RF12. Un usuario con rol de cliente podrá archivar una plantilla que este la sección de "plantillas de entrenamientos activas".

RF13. Un usuario con rol de cliente podrá registrar los entrenos.

RF14. Un usuario con rol de cliente podrá visualizar los registros a lo largo del tiempo.

RF15. Un usuario con rol de cliente podrá apuntar medidas corporales.

RF16. Un usuario con rol de cliente podrá visualizar a lo largo del tiempo sus medidas corporales.

RF17. Un usuario con rol de cliente podrá recibir notificaciones.

RF18. Un usuario con rol de cliente podrá crear reseñas para plantillas de entrenamiento que sean públicas.

RF19. Un usuario con rol de cliente podrá darle "Me gusta" a reseñas y comentarios.

RF20. Un usuario con rol de cliente podrá poner un comentario una reseña.

3.1.3 Requisitos funcionales de los usuarios de rol administrador

RF21. Un usuario con un rol de administrador podrá visualizar todos los intentos de acceso a la web o aplicación y si han sido exitosos.

RF22. Un usuario con rol de administrador podrá visualizar las *IPs* bloqueadas.

RF23. Un usuario con rol de administrador podrá visualizar todos los usuarios y filtrarlos por correo electrónico o nombre.

RF24. Un usuario con rol de administrador podrá banear usuarios.

RF25. Un usuario con rol de administrador podrá gestionar (crear, editar, borrar) ejercicios de forma global.

RF26. Un usuario con rol de administrador podrá crear credenciales para un nuevo usuario de rol de administrador.

3.3 Requisitos no funcionales

Para complementar el entendimiento de la aplicación, es crucial también detallar los requisitos no funcionales.

RNF1. El sistema deberá contar con cifrado de contraseñas y nunca guardarlas en texto plano.

RNF2. El sistema deberá recordar la sesión y preferencias del usuario.

RNF3. El sistema deberá tener validaciones en los formularios para asegurarse que los datos introducidos en la base de datos sean correctos.

RNF4. El sistema deberá ser eficiente a la hora de cargar múltiples registros, utilizando la carga perezosa.

RNF5. El sistema deberá contar con un diseño que adapte su disposición en función del tamaño de la pantalla o dispositivo utilizado.

3.4 Casos de uso

Los casos de uso principales para que desempañaran todos los roles son los siguientes:

Nombre	Descripción
Caso de uso	Inicio de sesión.
Actores	Usuario registrado.
Descripción	Un usuario podrá iniciar sesión.

Precondiciones	El usuario no ha iniciado sesión y posee una cuenta creada.
Escenario Principal	1. El usuario escribe sus credenciales. 2. El usuario le da al botón de "Iniciar sesión" 3. La aplicación redirige al usuario a la vista inicial de la aplicación
Escenario Alternativo	3a. La aplicación muestra un mensaje de error debido a que las credenciales son incorrectas o el usuario está bloqueado.

Nombre	Descripción
Caso de uso	Cierre de sesión.
Actores	Usuario registrado.
Descripción	Un usuario con la sesión iniciada podrá cerrarla.
Precondiciones	El usuario ha iniciado sesión y se encuentra navegando en la aplicación.
Escenario Principal	1. El usuario navega al menú de opciones de su perfil. 2. El usuario selecciona la opción "Cerrar sesión". 3. La aplicación cierra la sesión del usuario y lo redirige a la pantalla de inicio de sesión.
Escenario Alternativo	

Nombre	Descripción
Caso de uso	Registro de Nuevo Usuario.
Actores	Usuario potencial.
Descripción	El usuario se registra para obtener una nueva cuenta en la aplicación, completando el proceso de verificación de correo electrónico.
Precondiciones	El usuario no tiene una cuenta existente y accede a la página de registro.

Escenario Principal	1. El usuario selecciona la opción “Regístrate aquí”. 2. El usuario ingresa su correo electrónico, contraseña, confirmación de contraseña y nombre de usuario en la pantalla de credenciales. 3. El usuario puede escoger foto de perfil y elige su fecha de nacimiento en el selector de fecha. 4. El usuario revisa los datos ingresados y selecciona “Continuar”. 5. La aplicación envía un código de verificación al correo electrónico proporcionado. 6. El usuario introduce el código de verificación recibido en el campo correspondiente. 7. La aplicación valida el código y completa el registro. 8. El usuario es redirigido a la pantalla principal de la aplicación.
Escenario Alternativo	2a. Si el usuario ingresa un correo electrónico ya registrado o deja sin completar algún valor obligatorio. El sistema muestra un mensaje de error. 6a. El código de verificación no es introducido correctamente o ha expirado. 6b. El usuario selecciona “Enviar de nuevo” si no ha recibido el correo con el código. 7a. La verificación falla y el usuario debe solicitar un nuevo código o verificar los datos ingresados.

Nombre	Descripción
Caso de uso	Crear de Plantillas de Entrenamiento.
Actores	Usuario con rol de cliente.
Descripción	El usuario con rol de cliente crea una plantilla de entrenamiento personalizada proporcionando detalles específicos, como el nombre, descripción, objetivos y el tipo de ejercicio.
Precondiciones	El usuario ha iniciado sesión con un rol de cliente y accede a la sección de creación de nuevas plantillas de entrenamiento.
Escenario Principal	1. El usuario selecciona la opción "Nuevo Programa", representado con un botón con un icono de agregar. 2. El usuario ingresa el nombre del programa en el campo correspondiente. 3. El usuario puede proporcionar una descripción detallada del programa. 4. El usuario puede seleccionar una imagen o miniatura representativa para la plantilla. 5. El usuario marca uno o más objetivos del programa que desea alcanzar. 6. El usuario selecciona al menos un deporte o disciplina que se utilizarán en la plantilla. 7. El usuario elige el nivel recomendado de experiencia para la rutina de entrenamiento (principiante, intermedio o avanzado).

	8. El usuario indica el equipo necesario para realizar la rutina. 9. El usuario especifica la duración aproximada de las sesiones de entrenamiento. 10. Tras completar todos los campos requeridos, el usuario selecciona "Siguiente" para finalizar la creación de la plantilla. 11. La aplicación redirige al usuario a la pantalla de gestión de sesiones de entrenamiento de esa plantilla.
Escenario Alternativo	10a. El usuario le da al botón de volver. 10a.1. La aplicación le redirige a la anterior pantalla y no se guardan los cambios. 11a. Si el programa no posee nombre o no hay al menos un elemento seleccionado para cada categoría (objetivos, disciplinas usadas, equipo necesario, duración aproximada), la aplicación muestra un mensaje de error y solicita al usuario que complete la información necesaria antes de guardar la plantilla.

Nombre	**Descripción**
Caso de uso	Crear Sesión de Entrenamiento.
Actores	Usuario con rol de cliente.
Descripción	Un usuario podrá iniciar sesión.
Precondiciones	El usuario ha iniciado sesión y ha accedido a una plantilla de entrenamiento en la aplicación.
Escenario Principal	1. El usuario selecciona la opción "Crear sesión de entrenamiento"

	dentro de su programa de entrenamiento. 2. La aplicación redirige al usuario a la vista de edición de una sesión de entrenamiento. 3. El usuario introduce el nombre de la nueva sesión y cualquier instrucción específica para la sesión en los campos correspondientes. 4. El usuario selecciona "Añadir Ejercicio" para buscar y añadir ejercicios a la sesión. 5. El usuario define el método de registro del rendimiento para cada ejercicio, eligiendo entre opciones como "rango de repeticiones", "objetivo de repeticiones", "repeticiones máximas", "amrap" (*as many repetitions as possible*), "tiempo" o "rango de tiempo". A continuación, especifica las series y los valores objetivo correspondientes al método de registro seleccionado, como el número exacto de repeticiones o la duración del tiempo por serie. 6. El usuario puede añadir notas adicionales o ajustar la disposición de los ejercicios utilizando la función de reordenamiento si está disponible. 7. Una vez completada la sesión, el usuario guarda la sesión

	seleccionando “Guardar sesión de entrenamiento”. 8. Después de guardar, la aplicación muestra automáticamente la vista general de la plantilla de entrenamiento, donde se muestran todas las sesiones que forman parte de la plantilla, incluyendo la nueva permitiendo al usuario una revisión rápida o la creación de más sesiones.
Escenario Alternativo	7a. El usuario le da al botón de volver. 7a.1. La aplicación le redirige a la anterior pantalla y no se guardan los cambios. 8a. Si el usuario intenta guardar la sesión sin añadir ningún ejercicio, la aplicación muestra un mensaje de error indicando que se debe añadir al menos un ejercicio a la sesión.

Nombre	**Descripción**
Caso de uso	Restringir o autorizar acceso a un usuario.
Actores	Usuario con rol de administrador.
Descripción	El administrador restringe el acceso a la aplicación a un usuario específico debido a infracciones de las políticas de la comunidad o comportamiento indebido.
Precondiciones	El administrador ha iniciado sesión en la aplicación.
Escenario Principal	1. El administrador accede a la sección “Gestión de usuarios”. 2. El administrador busca al usuario específico por nombre o correo electrónico usando los filtros disponibles.

	3. El administrador selecciona al usuario que desea banear. 4. El administrador elige la opción "restringir acceso" o "autorizarlo" o similar representada por un icono. 5. El sistema procede a restringir el acceso del usuario a la aplicación y actualiza el estado del usuario a "Baneado: Sí".
Escenario Alternativo	No existen escenarios alternativos.

Nombre	Descripción
Caso de uso	Crear Ejercicio.
Actores	Usuario registrado.
Descripción	El usuario añade un nuevo ejercicio personalizado, en caso de ser de rol cliente lo añade a su lista de ejercicios y si es de rol administrador lo añade de forma global a la lista de todos los usuarios, proporcionando detalles como nombre, descripción, grupo muscular objetivo, materiales necesarios y un enlace a un vídeo de referencia.
Precondiciones	El usuario ha iniciado sesión y si es de rol administrador se encuentra en la sección de ejercicios de la aplicación y si es de rol cliente se encuentra en la sección para añadir ejercicios en una sesión de entrenamiento.
Escenario Principal	1. El usuario escribe el nombre del nuevo ejercicio en el campo correspondiente. 2. El usuario proporciona una descripción clara y concisa del ejercicio.

	3. El usuario selecciona el grupo muscular que se trabajará con el ejercicio. 4. El usuario elige el material o equipo necesario para realizar el ejercicio. 5. Si dispone de ello, el usuario añade la URL de un vídeo de YouTube que muestra cómo se realiza el ejercicio correctamente. 6. El usuario revisa la información ingresada y presiona "Guardar" para añadir el ejercicio a su biblioteca personal de ejercicios. 7. La aplicación redirige al usuario a la pantalla que muestra la lista de ejercicios.
Escenario Alternativo	6a. El usuario le da al botón de volver. 6a.1. La aplicación le redirige a la anterior pantalla y no se guardan los cambios. 7a. Si existe algún campo que falte, a excepción de la descripción y la URL o si la URL no es válida, la aplicación muestra un mensaje de error.

Nombre	**Descripción**
Caso de uso	Añadir Medidas Corporales.
Actores	Usuario con rol de cliente
Descripción	El usuario registra y actualiza sus medidas corporales y puede subir imágenes de su progreso físico.
Precondiciones	El usuario ha iniciado sesión en la aplicación y se encuentra en la pantalla de inicio.

Escenario Principal	1. El usuario selecciona la opción “Toma de medidas corporales” en el menú principal. 2. La aplicación redirige al usuario a una vista que enumera las medidas tomadas por fecha. 3. El usuario elige “Añadir medidas”. 4. El usuario ingresa las medidas de diferentes partes del cuerpo como peso, cintura, cuello, pecho, entre otros, en los campos correspondientes. 5. Si desea añadir imágenes para un seguimiento visual, selecciona “Añadir imágenes” y sube fotos desde su dispositivo. 6. El usuario guarda la información de las medidas y las imágenes seleccionando el botón “Guardar”. 7. La aplicación redirige al usuario a la vista donde se muestran todas las medidas por fecha y se puede observar las nuevas medidas añadidas para el día de hoy.
Escenario Alternativo	6a. El usuario le da al botón de volver atrás. 6a.1. La aplicación le redirige a la anterior pantalla y no se guardan los cambios. 7a. El usuario intenta guardar sin especificar ninguna medida y la aplicación muestra un mensaje de error.

Nombre	Descripción

Caso de uso	Añadir Registros a una Sesión de Entrenamiento.
Actores	Usuario registrado con rol de cliente.
Descripción	El usuario añade detalles de su rendimiento en ejercicios específicos durante una sesión de entrenamiento para realizar un seguimiento de su progreso.
Precondiciones	Precondiciones: El usuario ha iniciado sesión y ha abierto una plantilla de entrenamiento.
Escenario Principal	1. El usuario selecciona el botón de "Empezar sesión de entrenamiento". 2. La aplicación lo redirige al registro de rendimiento para esa sesión de entrenamiento. 3. Para cada serie de cada ejercicio, el usuario puede apuntar el valor objetivo correspondiente (repeticiones, peso, ...). 4. El usuario puede añadir registros de serie extras a los ejercicios y apuntar los rendimientos correspondientes. 5. El usuario presiona Terminar. 6. La aplicación guarda los datos y redirige a la vista de la plantilla de entrenamiento, ahora está vista habilitará la posibilidad de hacer la siguiente sesión de entrenamiento de la plantilla.
Escenario Alternativo	5a. El usuario le da al botón de volver atrás.

	5a.1. La aplicación le redirige a la anterior pantalla y no se guardan los cambios.

Nombre	Descripción
Caso de uso	Ver estadísticas de sesión de entrenamiento.
Actores	Usuario registrado con rol de cliente.
Descripción	Un usuario podrá iniciar sesión.
Precondiciones	Precondiciones: El usuario ha iniciado sesión y se encuentra la pantalla de inicio.
Escenario Principal	1. El usuario selecciona la opción "Historial de entrenamiento" la página de inicio. 2. La aplicación muestra un resumen de los ejercicios realizados, junto con las repeticiones, pesos y tiempos registrados. 3. El usuario puede filtrar por fecha. 4. El usuario selecciona una sesión de entrenamiento para ver su progreso. 5. La aplicación redirige al usuario a una interfaz con un gráfico de líneas que muestra la evolución de las repeticiones o pesos a lo largo del tiempo para cada ejercicio. 6. El usuario puede cambiar la vista de gráfica a una vista de lista, donde puede ver por ejercicio los registros de forma ordenada por fecha.
Escenario Alternativo	5.a. La aplicación no mostrará gráfica para los ejercicios que tengan menos de dos registros.

Nombre	Descripción
Caso de uso	Ver estadísticas de medidas corporales
Actores	Usuario registrado con rol de cliente
Descripción	Un usuario podrá iniciar sesión.
Precondiciones	Precondiciones: El usuario ha iniciado sesión y se encuentra la pantalla de inicio.
Escenario Principal	1. El usuario selecciona la opción "Toma de medidas corporales" la página de inicio. 2. La aplicación muestra un resumen de las medidas corporales realizadas. 3. El usuario selecciona "Ver estadísticas". 4. La aplicación redirige al usuario a una interfaz con un gráfico de líneas que muestra la evolución de las medidas a lo largo del tiempo para cada grupo muscular medido. 5. El usuario puede cambiar la vista de gráfica a una vista de lista, donde puede ver por ejercicio los registros de forma ordenada por fecha.
Escenario Alternativo	4.a. La aplicación no mostrará gráfica para aquellos grupos musculares que tengan menos de dos registros.

Nombre	Descripción
Caso de uso	Crear una reseña
Actores	Usuario registrado con rol de cliente.
Descripción	Un usuario podrá escribir una reseña en una plantilla de entrenamiento

Precondiciones	El usuario ha iniciado sesión, está en la pantalla de una plantilla de entrenamiento.
Escenario Principal	1. El usuario pulsa el botón de "Reseñas". 2. La aplicación lo redirige al apartado de reseñas 3. El usuario pulsa el botón de escribir reseñas. 4. La aplicación lo redirige a la pantalla de escribir reseña. 5. El usuario selecciona una calificación y escribe un comentario. 6. El usuario selecciona al botón para enviar la reseña. 7. La aplicación comprueba que los valores son correctos y manda la información al servidor, redirigiendo al usuario a la pantalla de visualización de reseñas
Escenario Alternativo	7a. La reseña está vacía y la aplicación muestra un mensaje de error.

3.5 Diagramas de secuencia

Para facilitar el entendimiento de cómo interactúan las funciones de la aplicación con los elementos del sistema, se han creado diagramas de secuencia que destacan los requisitos críticos en términos de interacción usuario-sistema, representados en las **Figuras 1 a 12**. Estos diagramas asumen que el usuario ya se encuentra en la página adecuada para realizar las acciones descritas, simplificando así la visualización de los procesos y se omiten transiciones obvias o que se salgan del contexto del diagrama evitando que empeore la legibilidad de estos diagramas.

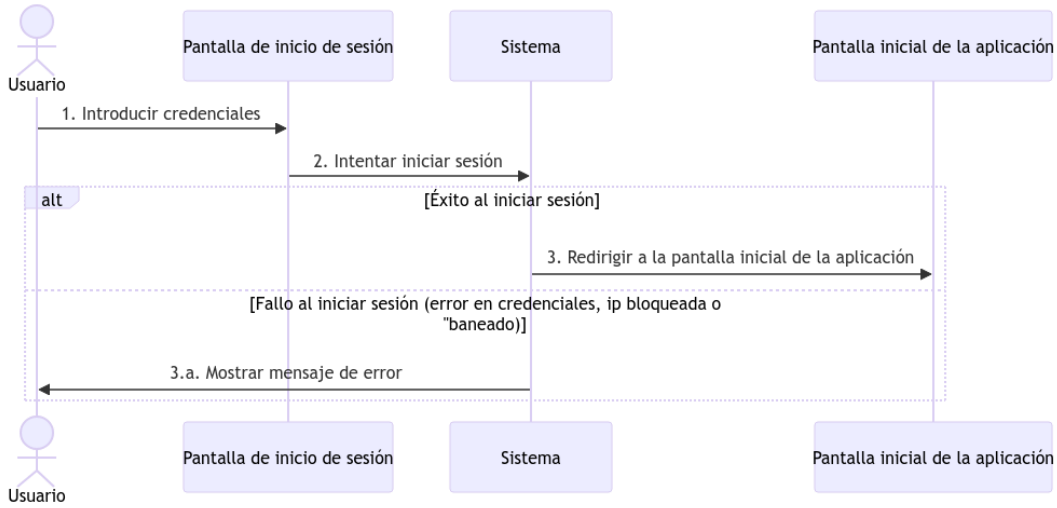

Figura 1 *Diagrama de secuencia de inicio de sesión*

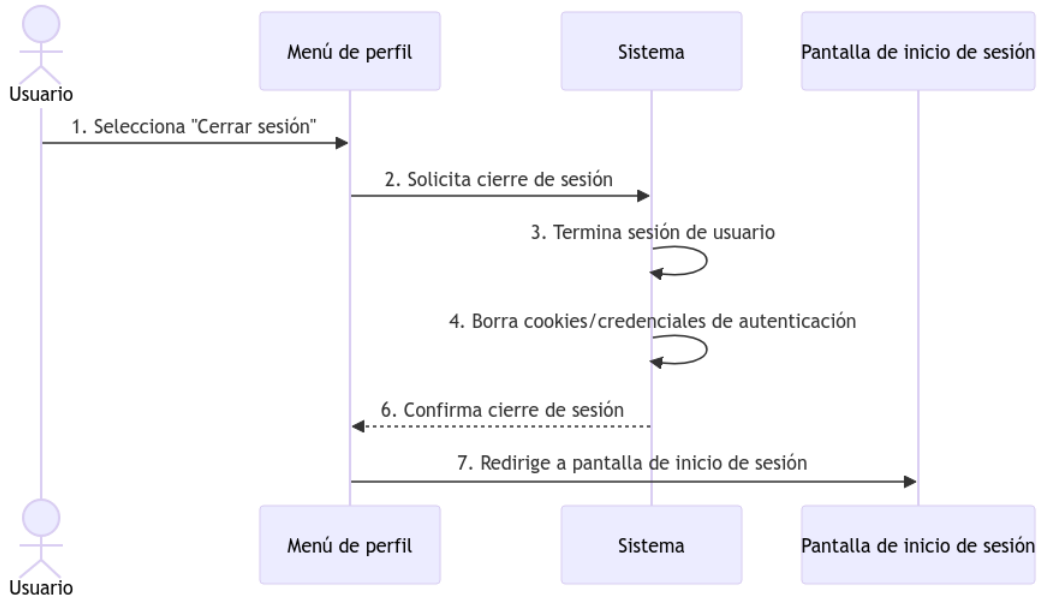

Figura 2 Diagrama de secuencia de cierre de sesión

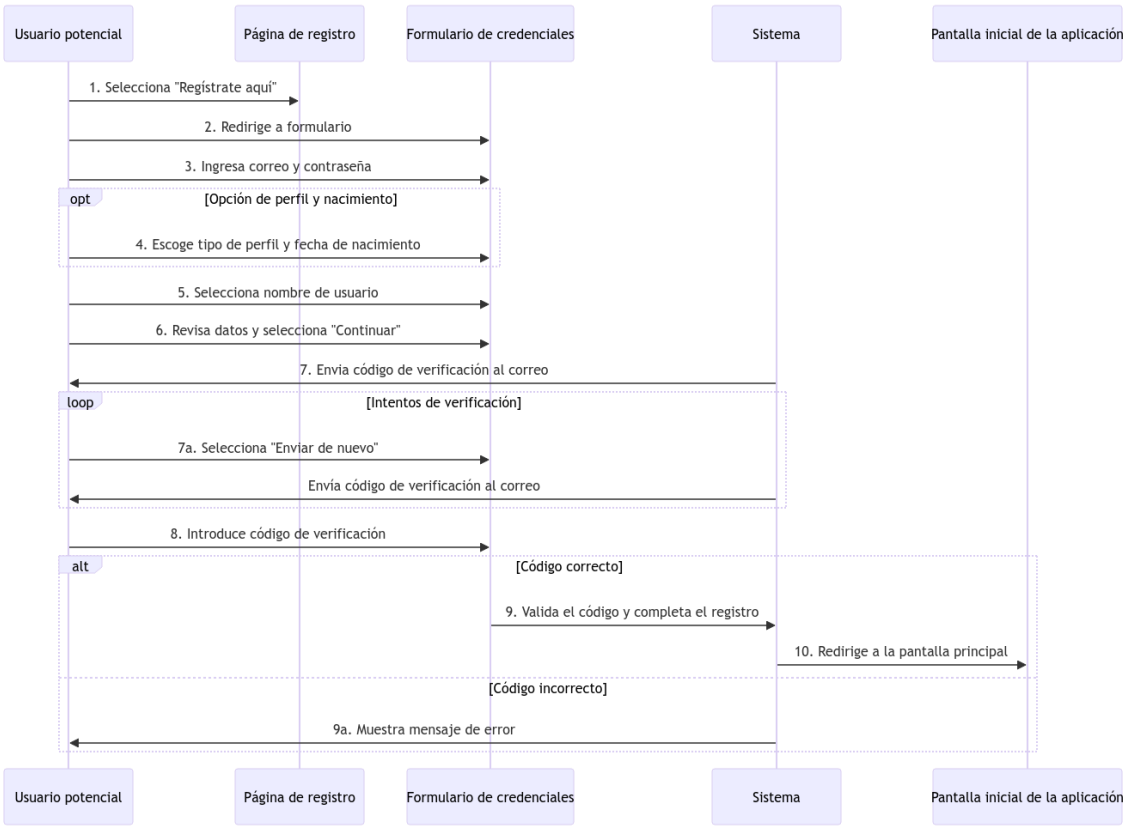

Figura 3 Diagrama de secuencia de registro de usuario.

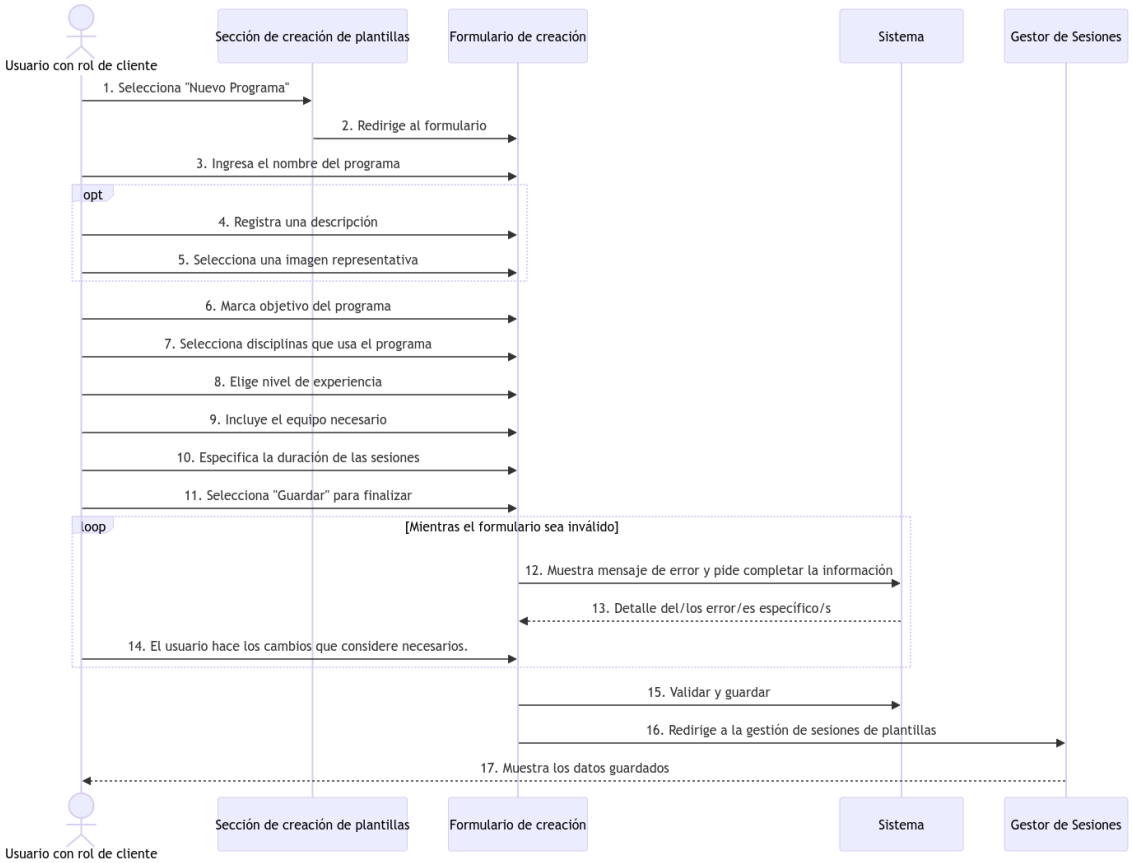

Figura 4 Diagrama de secuencia de creación de plantilla de entrenamiento.

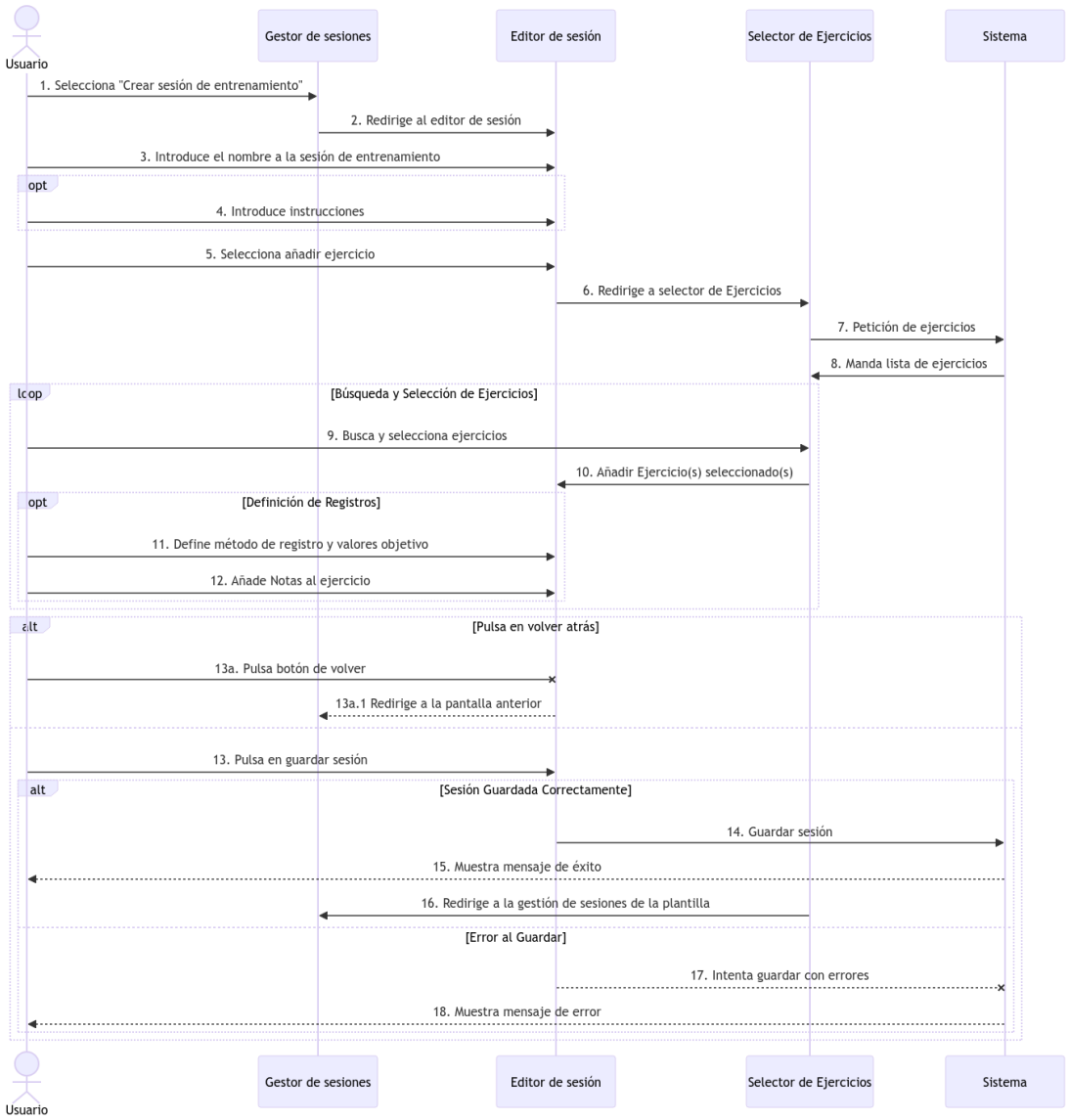

Figura 5 Diagrama de secuencia de creación de sesión de entrenamiento.

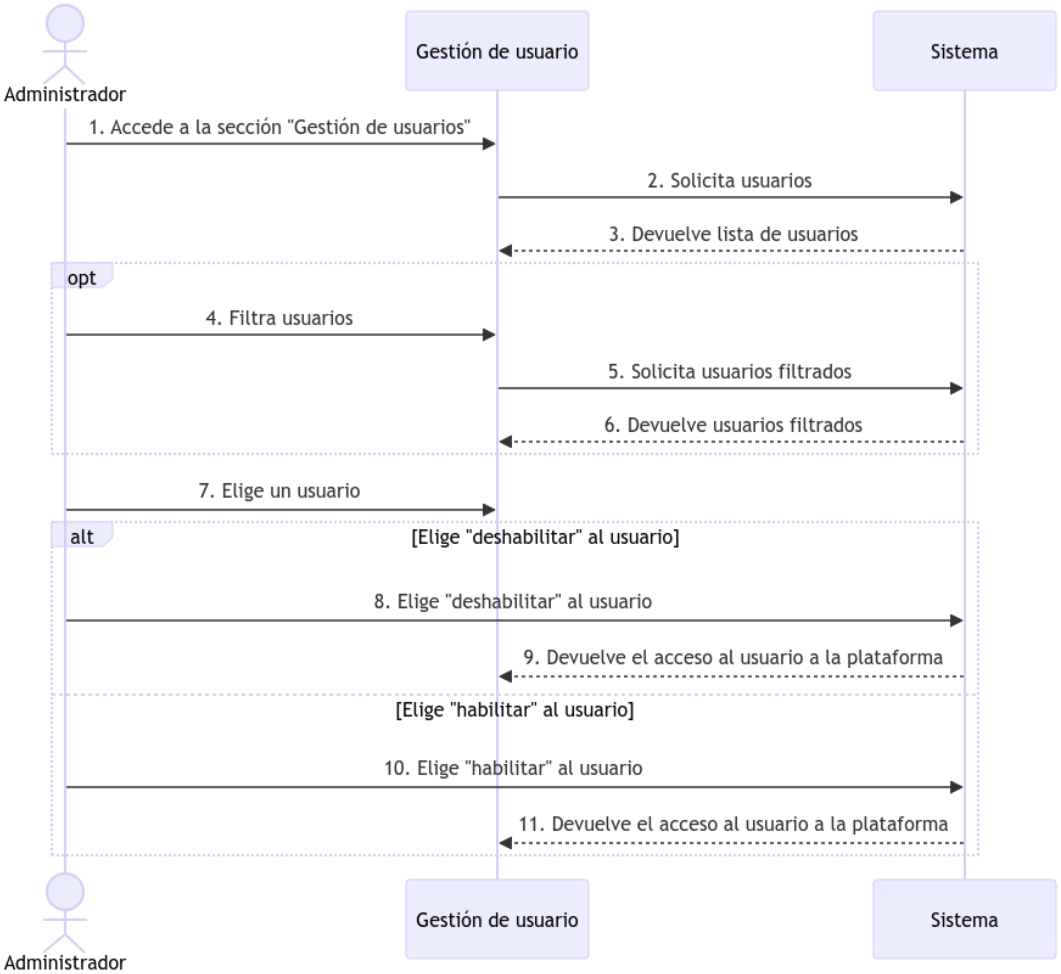

Figura 6 Diagrama de secuencia de restricción de acceso a usuarios.

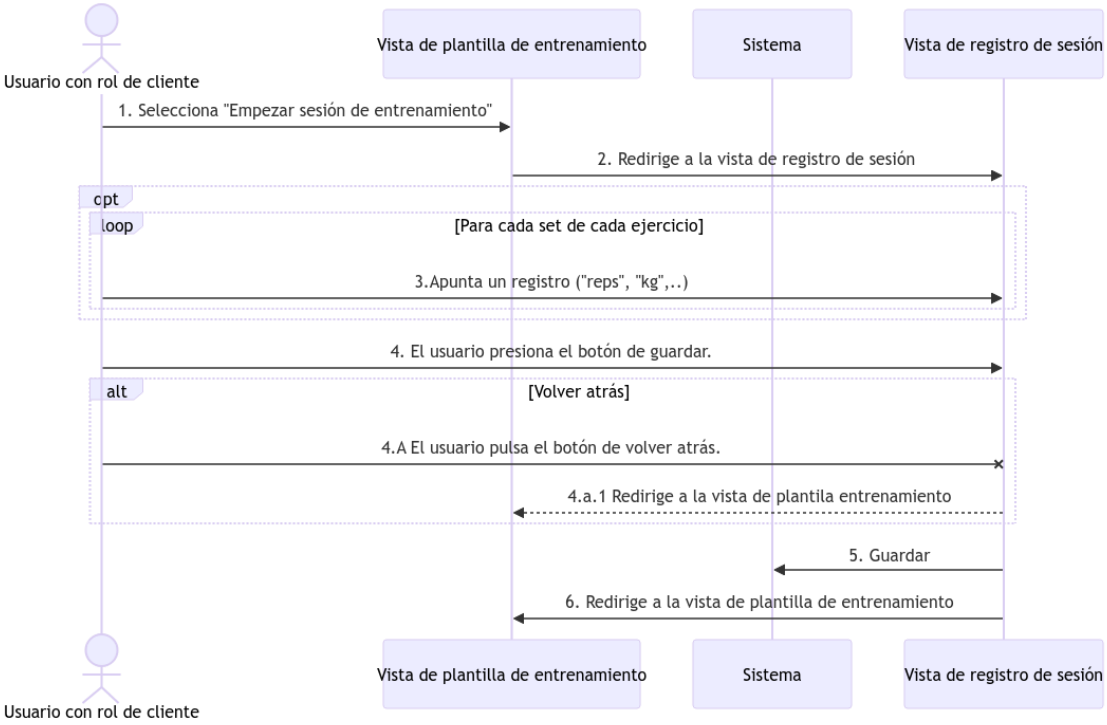

Figura 7 Diagrama de secuencia para añadir un registro a una sesión de entrenamiento.

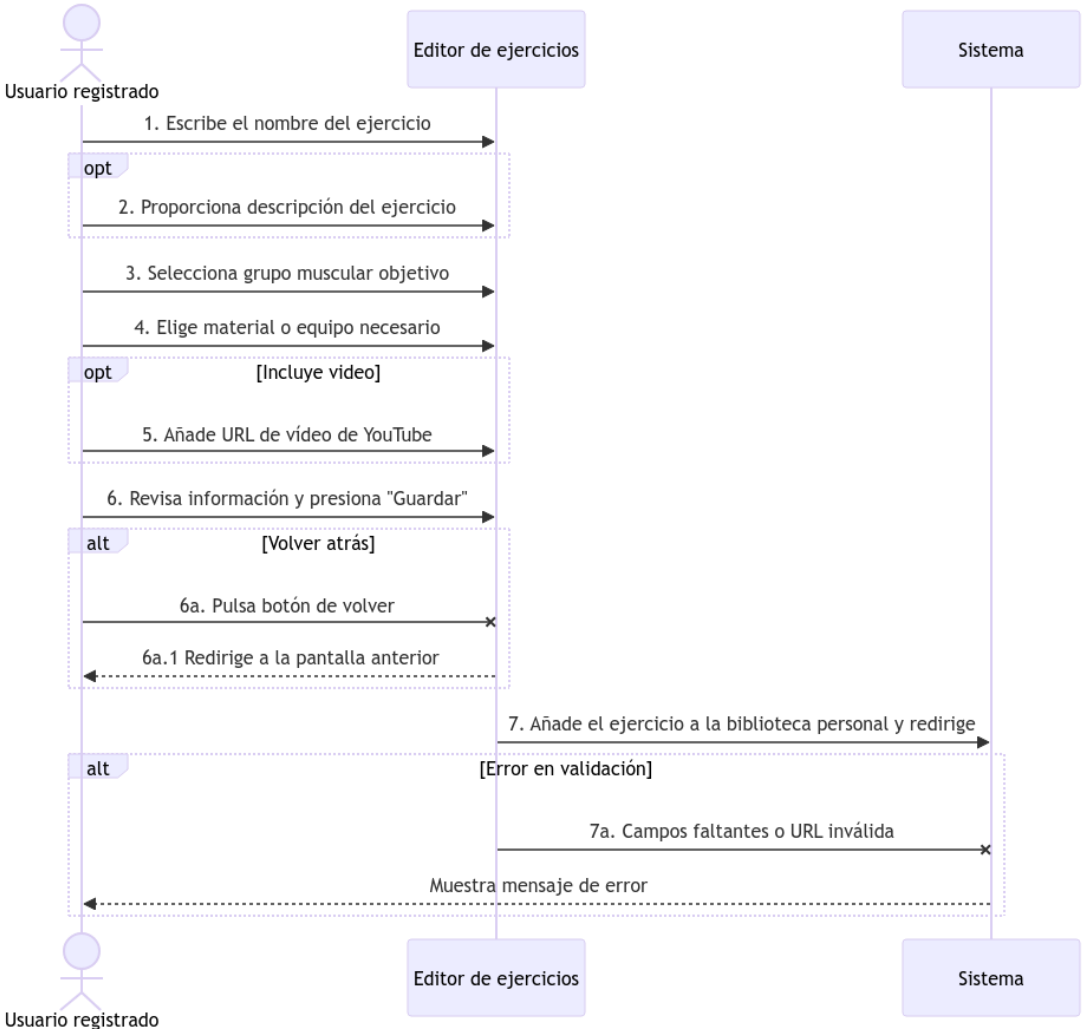

Figura 8 Diagrama de secuencia para crear un ejercicio

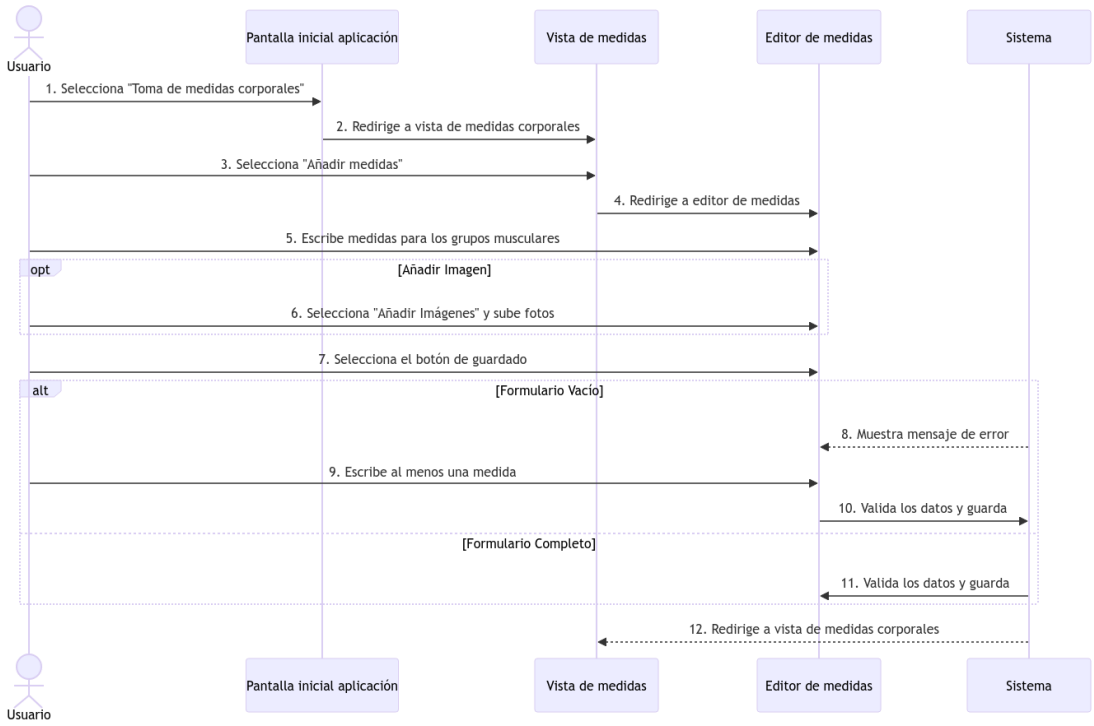

Figura 9 Diagrama de secuencia de crear medida

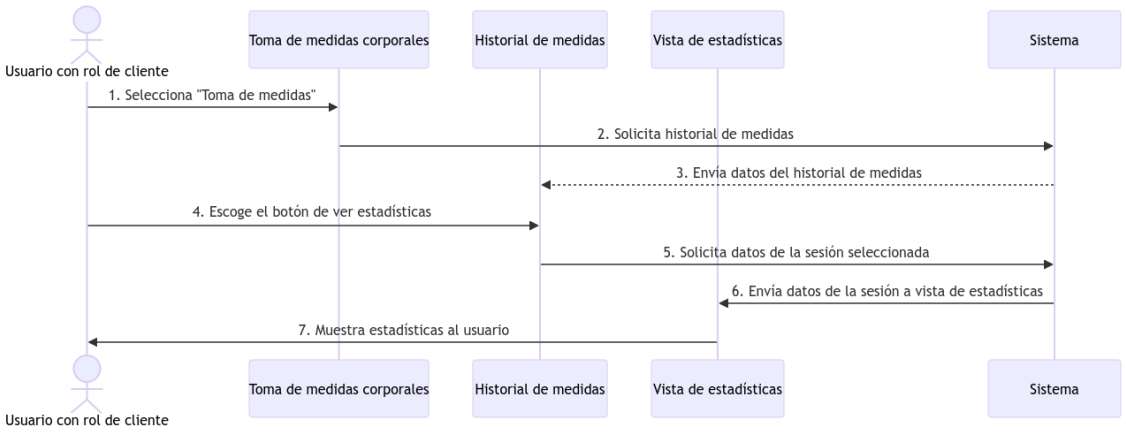

Figura 10 Diagrama de secuencia de visualización de estadísticas de medidas corporales

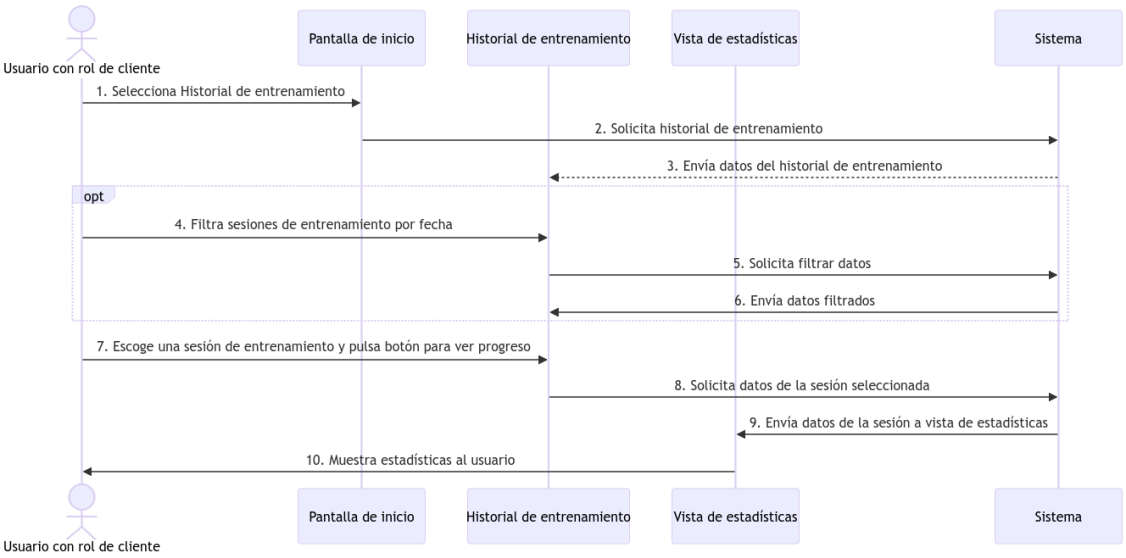

Figura 11 Diagrama de secuencia de visualización de sesiones de entrenamiento

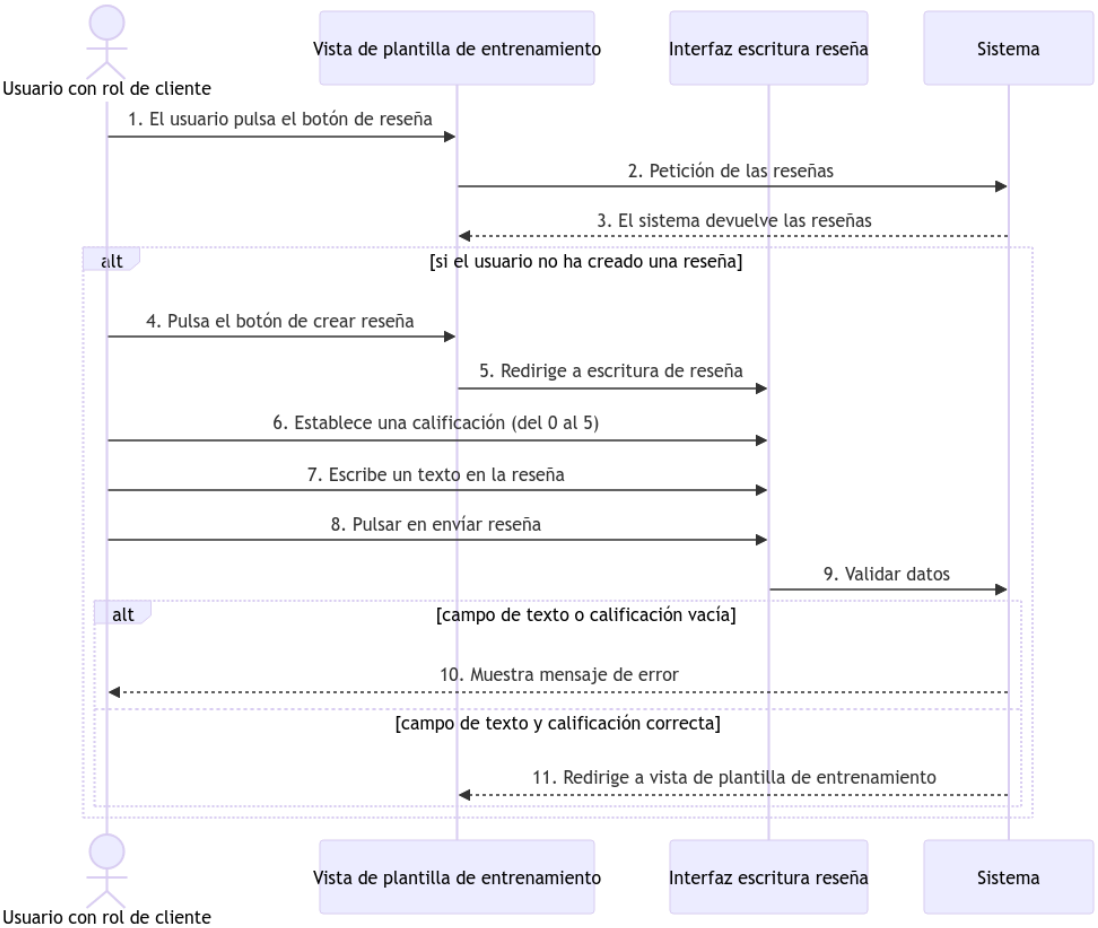

Figura 12 Diagrama de secuencia para crear una reseña

4

Diseño del sistema

El proyecto se divide en 3 módulos: el módulo de **almacenamiento**, el ***back-end*** o el módulo del servidor y ***front-end*** o módulo del cliente.

El módulo de almacenamiento se encarga de guardar todos los datos y archivos que sean necesarios y especificar cómo se relacionan. Uno de sus componentes es la base de datos que guardará la información de los usuarios, ejercicios, plantillas de entrenamiento gestionadas por los usuarios y además los registros sobre el rendimiento que apuntará cada usuario tras realizar las sesiones de entrenamiento.

El *back-end* constituye el núcleo de la aplicación, incluyendo gran parte de la lógica y centrándose en la interacción con la base de datos y otros servicios. Su principal función es enviar los datos necesarios al lado del cliente o *front-end*, lo que implica recuperar información almacenada en la base de datos y mostrarla al usuario. Asimismo, en esta capa se guardarán registros en las tablas de la base de datos.

El *front-end* es el módulo que interactúa con los usuarios del sistema. Esta capa se centra en mostrar interfaces, animaciones y todos los componentes visuales. La lógica suele estar centrada en un aspecto más creativo y en buscar una experiencia de usuario amigable, intuitiva y atractiva para los usuarios de la aplicación. Se comunica con el *back-end* y hace peticiones indicando los datos que requiere de la base de datos o cualquier otra acción que requiera consultar, editar, crear o borrar cualquier información de la base de datos.

Una interacción con esta arquitectura podría verse de la siguiente manera: Un usuario de la aplicación busca registrarse en la aplicación. En el *front-end* se muestra un formulario atractivo e intuitivo con los campos necesarios para que el usuario se cree una cuenta (nombre de usuario, correo, foto de perfil, contraseña...), si los datos son correctos y no están vacíos, el *front-end* manda una petición al *back-end* para que envíe un correo electrónico con un código único *OTP* (One Time Password). En otras palabras, el *back-end* genera un código aleatorio y lo envía al correo que el usuario debe de haber indicado en el formulario de la primera etapa. El *front-end* ahora mostraría un campo de entrada para que el usuario en busca de crear una cuenta inserte el código generado *OTP*. Al escribir el código en el campo de entrada, el *front-end* enviaría al servidor el código escrito. El *back-end* por último comprobaría que el código introducido es igual al enviado y en el caso de ser así, crearía el usuario en la base de datos.

En resumen, el proyecto se divide en 3 módulos conectados entre sí. El módulo de almacenamiento es un repositorio para guardar los datos e imágenes. Consta de una base de datos relacional y de un servidor de archivos en la nube, proporcionado a través de Cloudinary. Esta capa se conecta con el *back-end* que consulta, modifica, elimina y crea datos para la capa de bases de datos y está en espera de que el *front-end* realice las peticiones según sea necesario para realizar estas acciones (**Figura 13**).

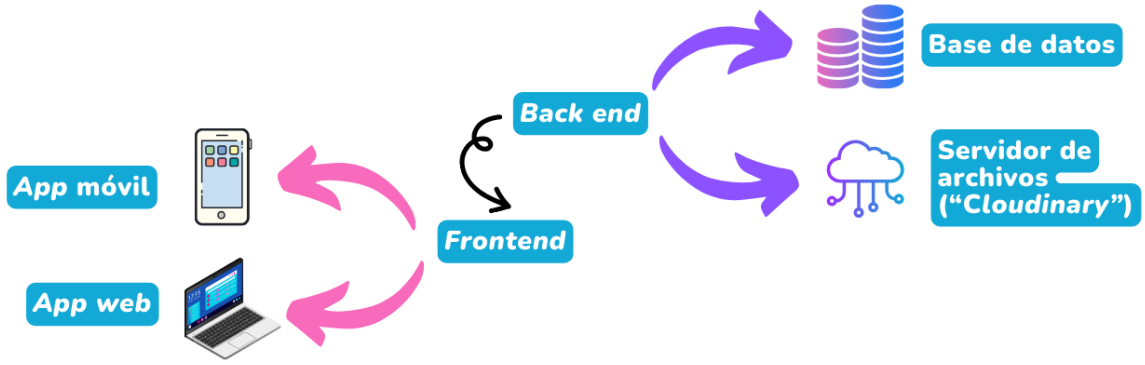

Figura 13 Esquema de arquitectura del sistema y cómo se conectan las capas.

4.1 Base de datos

La base de datos está desarrollada con *MySQL*, es decir, es una base de datos relacional y, por tanto, formada por tablas. Se ha diseñado el modelo relacional por partes o módulos. Como se puede ver en la **Figura 14,** cada módulo puede describirse como un modelo entidad relación que funciona de forma independiente y se agrupa con el resto de los

módulos para formar el modelo completo de la base de datos. Los módulos diseñados son los siguientes:

- **Módulo de gestión de usuarios**: Este sistema hace referencia a los datos necesarios que posee el usuario y como se establecen los permisos. Se constituye principalmente de dos modelos, los cuales son, el modelo de usuario, que incluye datos básicos como nombre, email, contraseña cifrada, imagen de perfil, fecha de nacimiento, preferencia de sistema de medidas, biografía, opción de compartir estadísticas, y estado de expulsión y el modelo del perfil que se encarga de especificar el rol del usuario entre administrador y cliente, donde el administrador tiene permisos adicionales para gestionar la aplicación y expulsar usuarios. Esta estructura es clave para la seguridad y administración de usuarios.

- **Módulo de notificaciones**: Especifica modelos que facilitan la gestión correcta de las notificaciones recibidas por cada usuario. El modelo "notificación" establece una relación de uno a muchos con el usuario, lo que significa que un usuario puede recibir múltiples notificaciones.

- **Módulo de control de acceso**: Controla el acceso con múltiples tablas que registran los intentos de acceder a la aplicación guardando la dirección *IP* de donde se origina el intento. Dependiendo del número de intentos fallidos se podrá bloquear la dirección *IP*. Este sistema también se encarga de almacenar códigos que serán enviados por correo electrónico para confirmar que estos mismos son correctos.

- **Módulo de gestión de medidas corporales**: Permite al usuario registrar y monitorear su progreso físico a través de mediciones corporales y está descrito por dos modelos principales El primero, "medidas", almacena datos detallados del usuario como medidas de brazos, hombros, cuello, pecho, cintura, piernas y pantorrillas, así como peso y otras métricas relevantes. El segundo, "fotos_progreso", se conecta al modelo "medidas" y almacena *URLs* del servidor de Cloudinary que hacen referencia a imágenes de progreso subidas por el usuario. La relación entre ambos modelos permite analizar el progreso en estas métricas a lo largo del tiempo. Este método fundamental es esencial para los usuarios que buscan monitorizar su progreso y ajustar sus rutinas de ejercicios y dietas basándose en datos precisos y actualizaciones visuales de su progreso.

- **Módulo de gestión de plantillas de entrenamiento**: La funcionalidad de este módulo es principalmente la creación y gestión de rutinas de entrenamiento individualizadas para los usuarios. Utiliza el modelo "plantillas_de_entrenamiento", que es un concepto abstracto que **agrupa sesiones de entrenamiento**. El modelo además almacena información como su título, descripción, imagen representativa que sirve como miniatura y configuraciones para establecer el acceso a otros usuarios. Cada plantilla tiene una fecha de creación específica y está vinculada directamente al usuario que la creó. Se vincula con la entidad "etiquetas" y permite clasificar las plantillas según experiencia, intereses, objetivos y equipo disponible, facilitando a los usuarios encontrar o diseñar entrenamientos que se ajusten a sus necesidades específicas. Cada sesión de entrenamiento tendrá indicado ejercicios para realizar y detalles sobre cómo realizarlos, a modo de ejemplo, algunos de estos detalles podrían ser objetivo de repeticiones en cada serie, instrucciones para realizar el ejercicio de alguna forma específica, ...

- **Módulo de registros de rendimiento:** Este módulo se utiliza para que el usuario sea capaz de registrar su desempeño después de realizar las sesiones de entrenamiento. Cada sesión se registra utilizando "registro_de_sesion", marcando el inicio, el final. Estos registros se utilizarán para la creación de gráficas de forma que el usuario entrenando puede ver la progresión en los ejercicios en un periodo de tiempo.

- **Módulo de almacenamiento de tipos de rutinas**: Este módulo gestiona el almacenamiento de las rutinas de entrenamiento guardadas por los usuarios. Además, gracias a este módulo, un usuario será capaz de archivar una plantilla de entrenamiento cuando no desee hacerla más veces, pero le interese mantener su historial.
- **Módulo de gestión de reseñas**: Este módulo gestiona las reseñas a las plantillas de entrenamiento y todo lo relacionado con ellas. Cada reseña tendrá una valoración del 0 al 5 y un comentario del usuario que realiza la reseña. Esto se hace con la entidad "*review*" la cual está relacionada de uno a muchos con las plantillas de entrenamiento (una plantilla de entrenamiento puede tener varias reseñas) y a su vez estará relacionada con otras dos entidades que hacen referencia

a los comentarios que se realizan por parte de otros usuarios a la reseña y los “me gusta” que recibe.

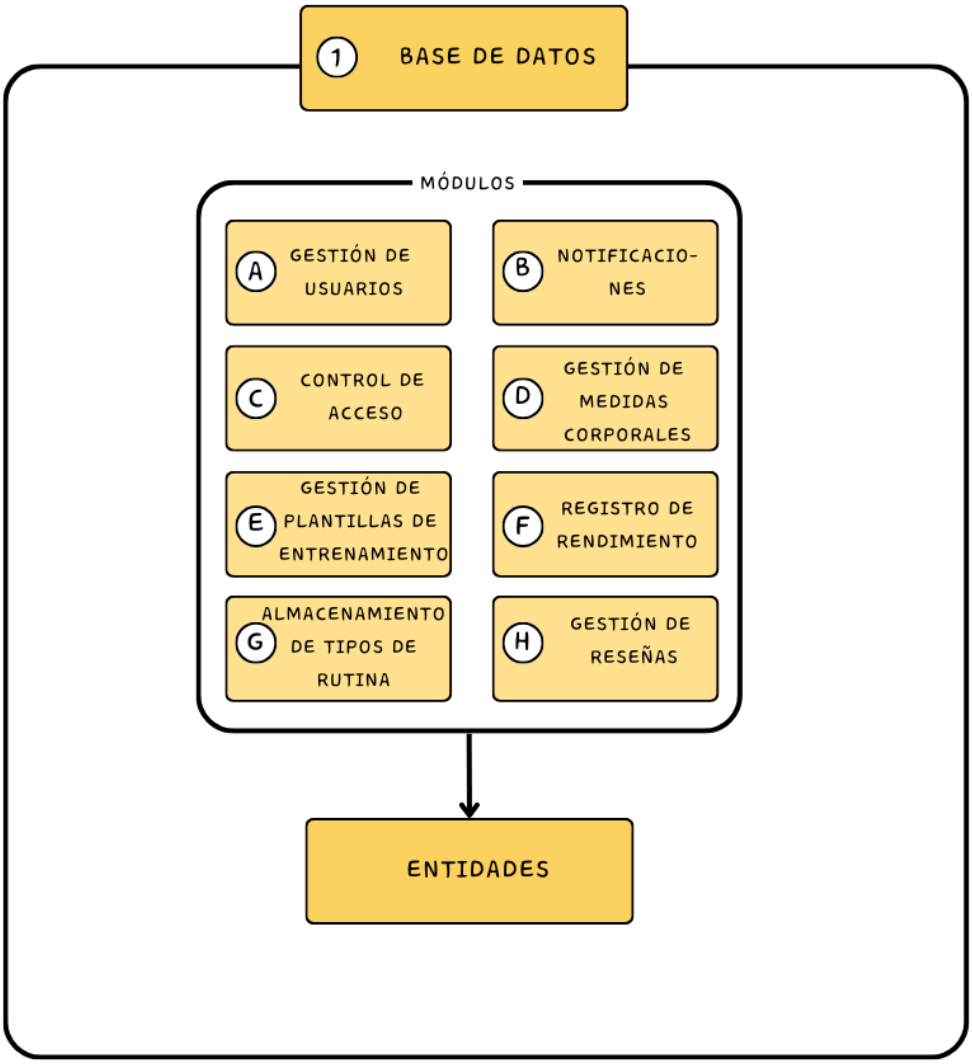

Figura 14 Esquema de módulos de la base de datos

4.1.1 Módulo de gestión de usuarios

Los modelos "usuario" y "perfil" representan a un usuario de la aplicación y sus permisos en la aplicación (**Figura 15**). Se puede observar que existe una relación en la que todos los usuarios deben tener obligatoriamente un perfil.
La tabla "usuario" contiene los siguientes campos:

- *username*: Almacena el nombre de usuario, con una cadena de texto con un límite de 255 caracteres.
- *email*: Para almacenar la dirección de correo electrónico del usuario.

- *password*: Una cadena de texto que hace referencia al resultado de cifrar la contraseña del usuario.
- *profile_picture*: Una cadena de texto con una referencia para la *URL* o ruta de la imagen de perfil del usuario que se aloja en "Cloudinary".
- *birth*: Una columna que registra la fecha de nacimiento del usuario.
- *system*: Este valor indica la preferencia del usuario entre sistema métrico o imperial para mostrar los resultados en kilogramos o en libras.
- *bio*: Un campo de texto para una biografía o descripción del usuario.
- *public*: Es un campo que servirá en un futuro para indicar si el usuario quiere compartir sus estadísticas e historial de progreso o no.
- *banned*: Un parámetro que indica si el usuario ha sido expulsado de la aplicación.

La tabla "perfil", define el tipo de usuario que usa la aplicación por el momento existen dos roles: "*admin*" o "cliente". El administrador tendrá otros permisos y podrá gestionar otras pantallas, así como poder expulsar a otros usuarios. Tendrá como parámetro:

- *rol*: Describe el rol del usuario dentro de la aplicación o el sistema.

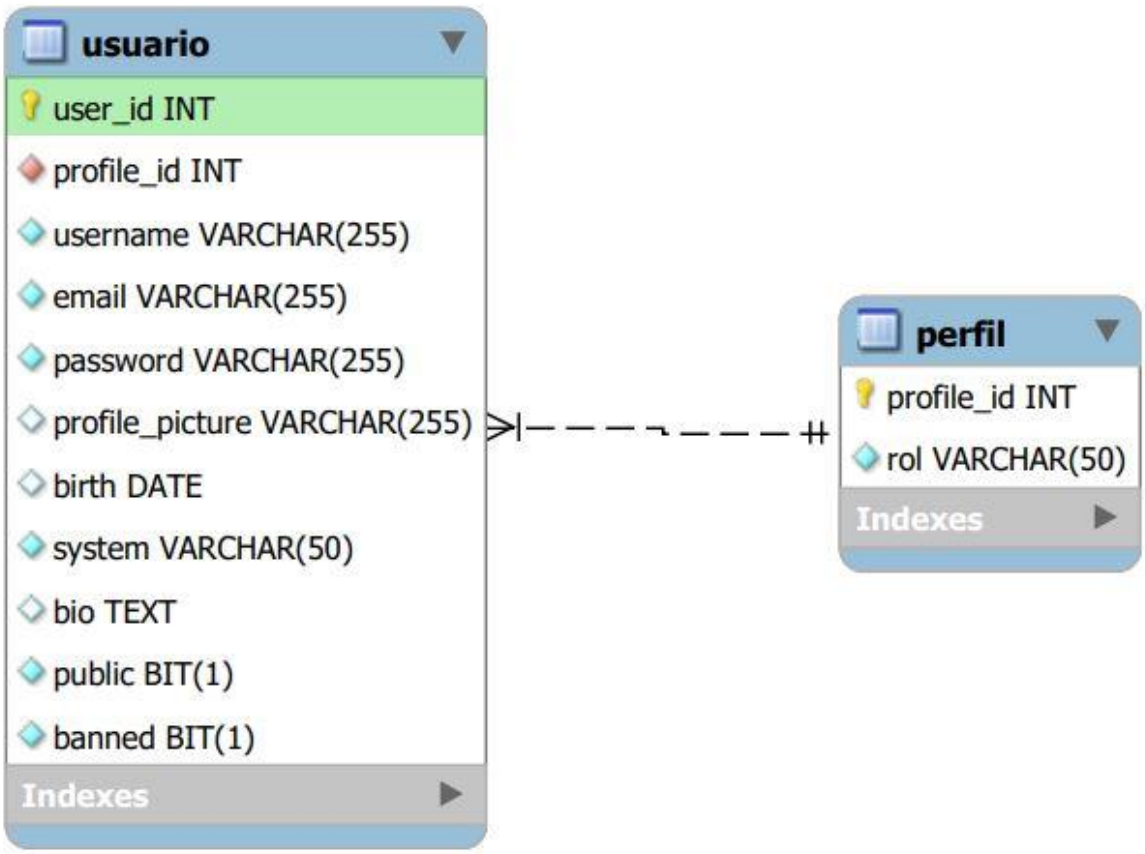

Figura 15 Tabla de "usuario" y su relación con la tabla "perfil

4.1.2 Módulo de gestión de notificaciones

El siguiente modelo o entidad viene dado en la tabla "notificación" (**Figura 16**), que se relaciona con el usuario con una relación de uno a muchos. Es decir, que un usuario puede tener muchas notificaciones.

Una notificación tendrá los siguientes parámetros:

- *type:* Este parámetro hace referencia al tipo de la notificación, equivale a un pequeño resumen de porqué se envía la notificación.
- *mensaje*: Una cadena de texto que representa el contenido de la notificación.
- *read*: Un valor de tipo de dato lógico que representa si el mensaje ha sido leído por el usuario al que se ha mandado la notificación.
- *timestamp:* Un valor de tipo fecha que indica cuando se ha enviado la notificación.

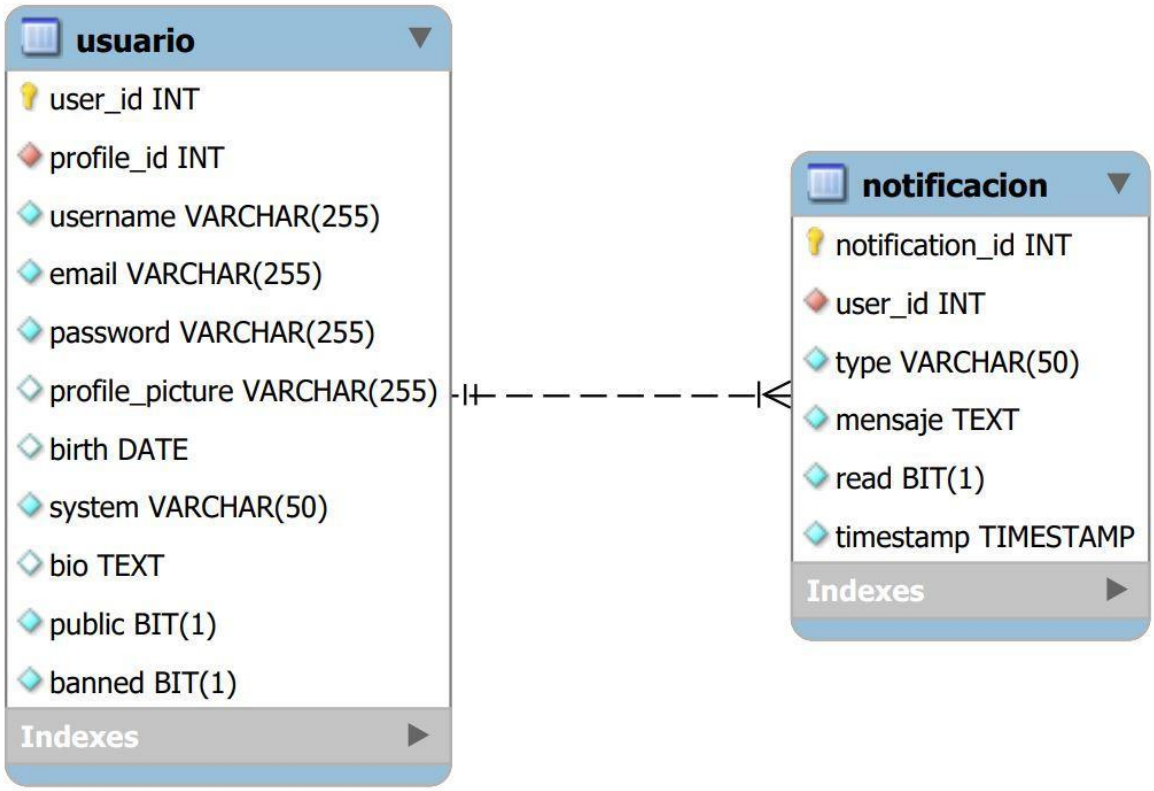

Figura 16 Tabla de "usuario" y su relación con la tabla "notificación"

4.1.3 Módulo de control de acceso

A continuación, tres entidades importantes para la gestión de autenticación son las tablas de "*codigo_otp*", "*logs*" y "bloqueos" (**Figura 17**). Estas tres tablas son independientes y no están relacionadas con ninguna otra. La primera tabla "*codigo_otp*", como su nombre indica hace referencia al almacenamiento de los códigos generados de forma aleatoria y enviados por correo electrónico, estos códigos se generan y envían por correo electrónico, por ejemplo, a la hora de registrarse o recuperar la contraseña en caso de olvido. Los parámetros de esta primera tabla son:

- *valor*: Es la combinación de dígitos del código *OTP*.
- *fecha_caducado*: Este parámetro es una fecha que indica cuando el código *OTP* deja de ser válido.

La siguiente tabla *"logs"* hace referencia a un modelo que guarda un registro cada vez que se intenta acceder a la aplicación, guardará desde la dirección *IP* que se intenta acceder y si el intento de inicio de sesión ha sido exitoso. Los parámetros de esta tabla son:

- *fecha*: Atributo que indica cuándo se intentó entrar en la aplicación.
- éxito: Un valor de tipo lógico que indica si el intento de entrar en la aplicación ha sido exitoso.
- *ip_address*: Guarda la dirección *IP* desde donde se ha intentado acceder a la aplicación.
- *email*: Correo con el que ha intentado acceder a la aplicación.

La tabla de "bloqueos" es importante para evitar el robo de cuentas mediante el uso de fuerza bruta. La idea es registrar los intentos de inicio de sesión de cada *IP*, en caso de que haya muchos intentos fallidos en breve periodo de tiempo desde la misma dirección *IP*, se puede entender que el usuario detrás de la dirección *IP* puede estar intentando entrar a una cuenta que no es suya a base de usar aplicaciones que prueban infinidad de combinaciones distintas hasta encontrar la contraseña. Por tanto, si se detecta, se estima un bloqueo de un periodo de tiempo. Los atributos de esta entidad son:

- *ip_address*: Cadena de texto que representa la dirección IP bloqueada.
- *fecha_hasta*: Parámetro que indica la fecha de cuando termina el bloqueo.
- *timestamp*: Fecha en la que se realiza el bloqueo.

Figura 17 Tablas de "codigo_otp", "logs", "bloqueos"

4.1.4 Módulo de gestión de medidas corporales

Un usuario es capaz de escribir sus medidas corporales, con el objetivo de poder analizar su progreso a lo largo del tiempo mediante gráficas y tablas. Para ello se han creado los modelos "medidas" que hace referencia a los registros de medidas corporales en una fecha determinada. Cada registro de medidas, si el usuario lo desea, puede ser acompañada con una o más fotos, para ello se tiene una relación de uno a muchos con la tabla "fotos_progreso" (**Figura 18**).

Los parámetros de la entidad "medidas" son:

- *left arm, right arm*: Medidas del brazo izquierdo y derecho.
- *shoulders*: Ancho de hombros.
- *neck*: Medida del cuello.
- *chest*: Medida del pecho.
- *waist*: Medida de la cintura.
- *upper_left_leg, upper_right_leg*: Medidas de la parte superior de la pierna izquierda y derecha.
- *left_calf, right_calf*: Medidas de la pantorrilla izquierda y derecha.
- *weight*: Peso del usuario.
- *left_forearm, right_forearm*: Medidas del antebrazo izquierdo y derecho.
- *timestamp*: Marca de tiempo de cuándo se tomaron las medidas.

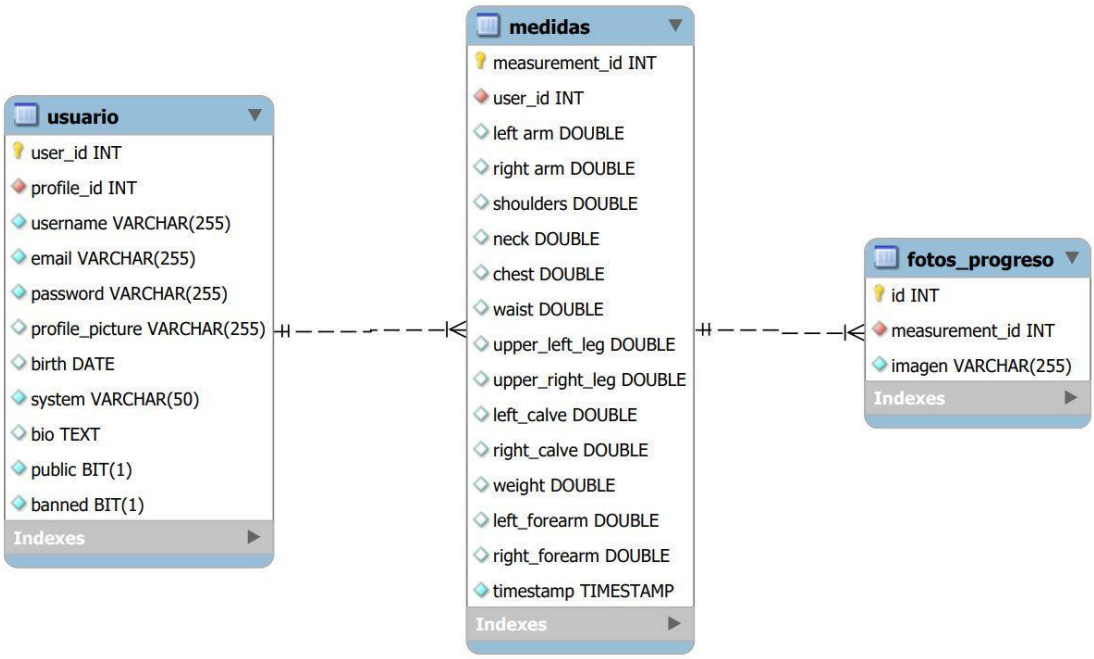

Figura 18 Tablas de "usuario", "medidas "y "fotos_progreso" y cómo están relacionadas entre sí.

4.1.5 Módulo de gestión de plantillas de entrenamiento

Dentro de la estructura funcional de la plataforma, se destaca el módulo de gestión de plantillas de entrenamiento, cuya finalidad es permitir a los usuarios la creación, personalización y administración de sus propias rutinas de entrenamiento. Este módulo se estructura sobre varios modelos interconectados que facilitan una experiencia de usuario coherente y dinámica (**Figura 19**).

Las entidades principales son, "plantillas_de_entrenamiento" y "sesión de entrenamiento, son el eje sobre el que gira la creación de las rutinas. Este modelo alberga la información esencial de cada plantilla, incluyendo:

- *user_id*: El identificador del usuario creador de la plantilla.
- *template_name*: Nombre descriptivo de la plantilla de entrenamiento.
- *description*: Un texto que detalla los objetivos y métodos de la plantilla.
- *picture*: La imagen representativa que funciona como miniatura.
- *public*: Un indicador lógico que define si la plantilla es pública o privada.
- *hidden*: Indica si la plantilla debe estar oculta o visible para otros usuarios.
- *fecha_creacion*: La fecha en que la plantilla fue creada.

La entidad “etiquetas", vinculada a las plantillas, organiza y clasifica cada una basándose en diferentes criterios que reflejan necesidades de los usuarios para facilitar encontrarlos posteriormente. Los atributos de esta tabla incluyen:

- *experience, interests, objectives, equipment*: Atributos que describen la experiencia necesaria para realizar la rutina, intereses temáticos, objetivos específicos del entrenamiento y el equipo necesario.

Por otro lado, la entidad “sesion_de_entrenamiento” es una instancia específica de entrenamiento relacionadas con las plantillas que hacen referencia a la agrupación de todas las superseries de una sesión de entrenamiento. Una plantilla de entrenamientos puede tener varias sesiones de entrenamiento. Estas sesiones están definidas por:

- *template_id*: Vínculo con la plantilla de entrenamiento correspondiente.
- *session_name*: El nombre asignado a la sesión específica.
- *session_date*: Fecha planeada para la sesión.
- *notes*: Anotaciones adicionales relativas a la sesión.
- *order*: La secuencia en la que la sesión aparece dentro de la plantilla.
- *activa*: Estado de la sesión, si está activa o no.

El resto de las entidades se relacionan de la siguiente forma:

- La entidad “ejercicios” se relaciona con “grupo_muscular” y “material” a través de las claves foráneas muscle_group_id y material_id, respectivamente. Esto indica que cada ejercicio pertenece a un grupo muscular y requiere un material específico.
- “ejercicios_detallados” está relacionada con “ejercicios” y representa un ejercicio al que se le asocian detalles o instrucciones para realizarlos en la sesión de entrenamiento. Estas instrucciones van desde notas, hasta tipo de registro (rango de tiempo, tiempo, *amrap* (*as many repetitions as possible*), número de repeticiones, rango de repeticiones y repeticiones máximas) representadas con la relación de la entidad de “tipo_de_registro”. Además, cada ejercicio detallado puede estar relacionado con la entidad “sets_ejercicios_entrada” en el que dependiendo del tipo de registro se indican instrucciones para cada serie del ejercicio detallado.

- "ejercicios_detallados_agrupados" es una superserie (*super set*) de ejercicios con detalles o agrupación de varios ejercicios con detalles pensados para realizarse uno detrás del otro sin descanso. Se relacionan directamente con la entidad de "sesion_de_entrenamiento".

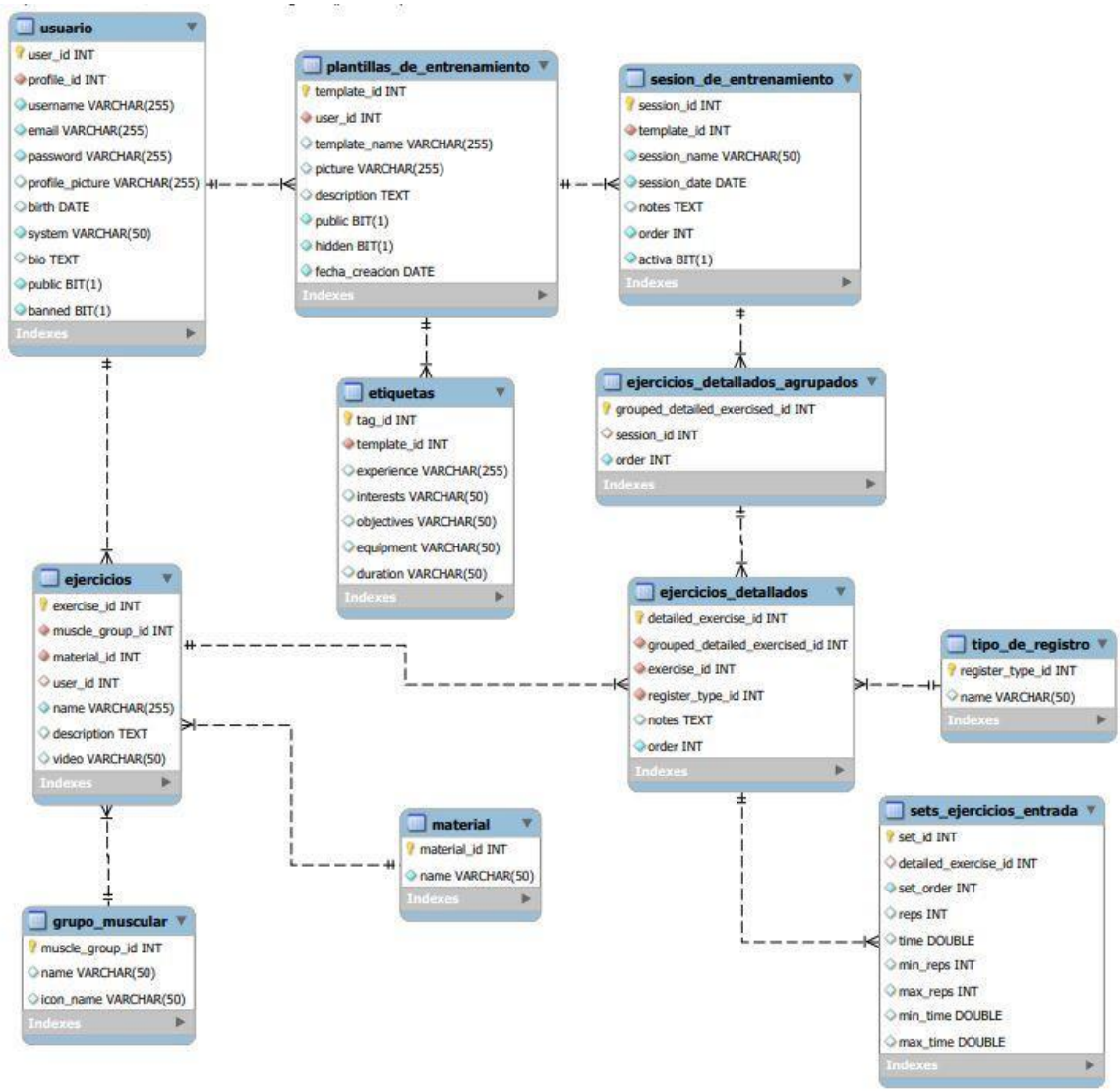

Figura 19 *Tablas y relaciones usadas para la gestión de plantillas de entrenamiento a la hora de crearlas y visualizarlas.*

4.1.6 Módulo de registro de rendimiento

En la **Figura 20**, se presenta el módulo de registro de rendimiento. Esta parte del sistema se encarga de almacenar la información relativa a la realización de los ejercicios y las sesiones de entrenamiento, permitiendo a los usuarios rastrear su progreso.

La entidad “registro_de_sesion” sirve para registrar las instancias en que un usuario lleva a cabo una sesión de entrenamiento. Sus atributos son:

- *date*: Fecha y hora en que comenzó la sesión de entrenamiento.
- *final_date*: Fecha y hora en que terminó la sesión de entrenamiento.
- *finished*: Indicador de si la sesión fue completada o no.

La entidad “registro_set” documenta los datos específicos de cada conjunto de ejercicios realizados durante la sesión. Los parámetros podrán ser nulos y se rellenarán dependiendo del tipo de registro relacionado con la entidad “set_ejercicios_entrada”. Los parámetros serán:

- *reps*: Número de repeticiones realizadas en la serie.
- *weight*: Peso utilizado en el ejercicio de la serie.
- *time*: Duración de la serie.
- *timestamp*: Marca de tiempo cuando se realizó la serie.

Las relaciones entre las entidades de la **Figura 20** y las restantes entidades funcionan de la siguiente manera:

- “registro_de_sesion” se vincula con un usuario que hace referencia al autor de la sesión que está siendo registrada. Asimismo, se conecta con la entidad “sesion_de_entrenamiento” proporcionando información sobre qué plantilla y sesión se está registrando.
- “registro_set” está asociado con “registro_de_sesion”, estableciendo que cada serie pertenece a una sesión registrada. También está conectado a “sets_ejercicios_entrada” relacionando la serie registrada con los detalles específicos del ejercicio realizado como parte de la sesión de entrenamiento.

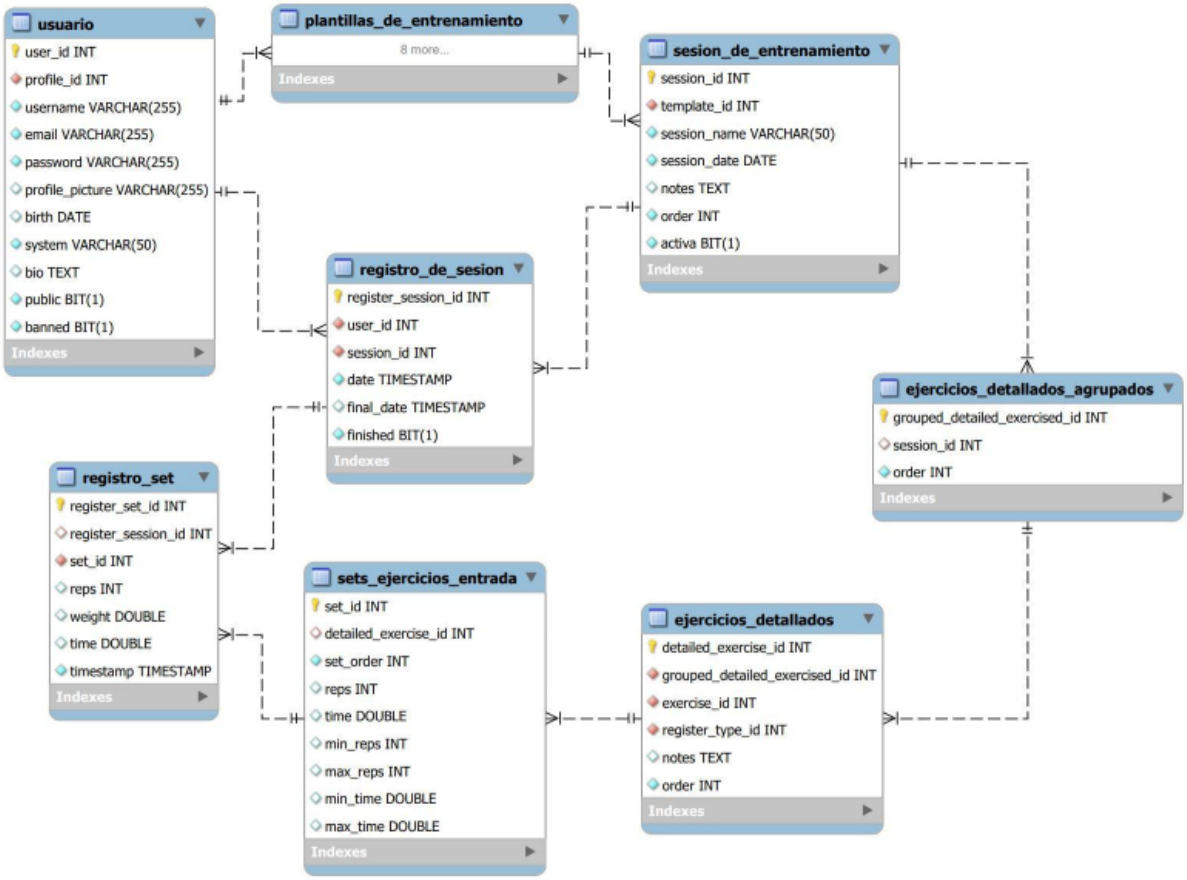

Figura 20 Tablas y relaciones usadas para registrar el rendimiento de los ejercicios en las sesiones de entrenamiento de las plantillas de entrenamiento.

4.1.7 Módulo de almacenamiento de tipos de rutina

En la **Figura 21,** se muestran dos entidades relacionadas con la gestión de las plantillas de entrenamiento en la plataforma: “rutinas_archivadas” y “rutinas_guardadas”. Ambas tienen la función de manejar diferentes estados de las rutinas de los usuarios, ya sea para archivarlas o guardarlas para un uso futuro. Tiene un parámetro “hidden” el cual es un indicador lógico que determina si la rutina está oculta o visible para otros usuarios dentro de la plataforma. Este parámetro es útil porque a veces no conviene borrar un dato de la base de datos por completo ya que puede estar siendo usado por otros usuarios.

Aunque ambas entidades comparten los mismos parámetros, su funcionalidad es distinta. La entidad “rutinas_archivadas” se utiliza para almacenar rutinas que el usuario ha decidido no usar actualmente, pero desea mantener en un registro histórico, quizás para referencia futura o para revisar el progreso a largo plazo. La entidad “rutinas_guardadas” se emplea para rutinas que el usuario desea conservar activamente para acceder y utilizar con mayor facilidad en el corto plazo, sin necesidad de buscarlas nuevamente o reconstruirlas desde cero.

La existencia de ambas entidades ofrece a los usuarios la flexibilidad de organizar sus rutinas de entrenamiento y mantener un control sobre la información que desean tener a mano o almacenar para referencia futura.

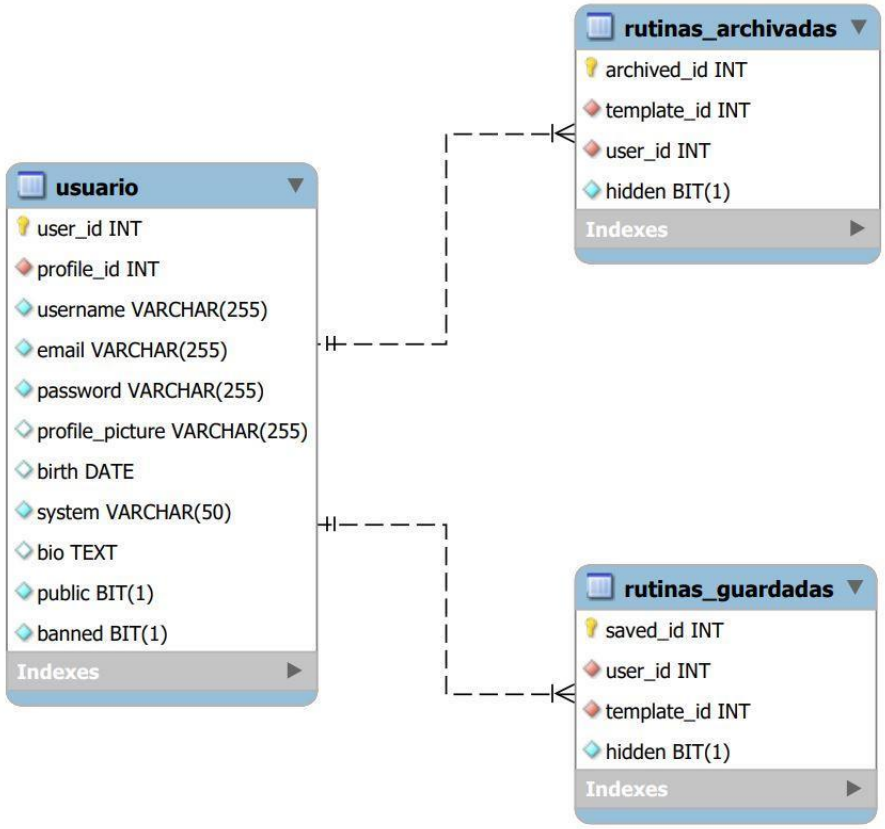

Figura 21 Tablas de "usuario", "rutinas_archivadas "y "rutinas_guardadas" y cómo están relacionadas entre sí.

4.1.8 Módulo de gestión de reseñas

La **Figura 22** representa el módulo de gestión de reseñas enfocado en la interacción de los usuarios con las reseñas y comentarios dentro de la plataforma. Este diseño permite que los usuarios compartan sus opiniones sobre las plantillas de entrenamiento y participen en discusiones, mejorando la experiencia comunitaria de la plataforma.

La entidad "reviews" se encarga de almacenar las reseñas y valoraciones que los usuarios realizan sobre las plantillas de entrenamiento. Se relaciona con el usuario creador de la reseña y con la plantilla de entrenamiento a la que se realiza la reseña. Los parámetros de "reviews" son:

- *rating*: Calificación numérica que el usuario asigna a la plantilla.
- review_*content*: Texto de la reseña donde el usuario expresa su opinión sobre la plantilla.
- *timestamp*: Marca de tiempo de cuándo se realizó la reseña.

El resto de las entidades y sus relaciones funcionan de la siguiente manera:

- "me_gusta_reviews": Permite a los usuarios expresar que les gusta una reseña específica. Relaciona el usuario que ha dado "me gusta" con la reseña.
- "comentario_review": Contiene los comentarios que los usuarios pueden hacer en cada reseña y está relacionada con el usuario que crea el comentario y la reseña a la que se realiza el comentario.
- "me_gusta_comentarios": Similar a "me_gusta_reviews", esta entidad permite a los usuarios dar "me gusta" a comentarios individuales, asociando al usuario que da el "me gusta" con el comentario que recibe la aprobación.

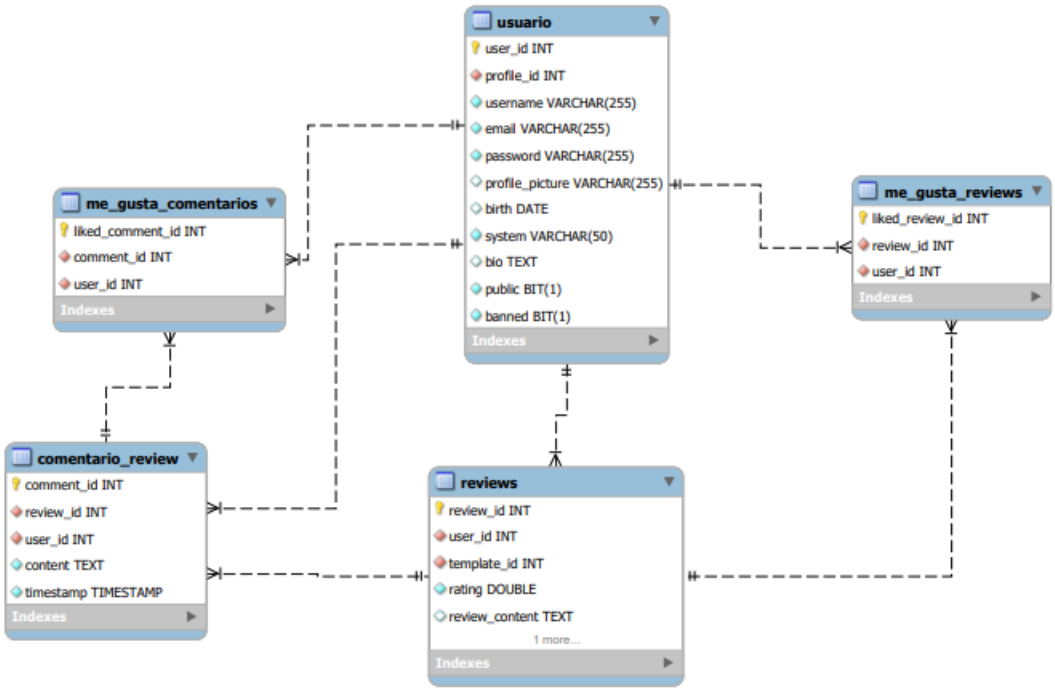

Figura 22 Tablas y relaciones usadas para las *reviews* y su gestión.

4.2 Módulo de *back-end*

La capa del *back-end* tiene una arquitectura bien definida. Tiene un componente principal o índice que inicia la conexión y posee las configuraciones necesarias. Se conecta a los módulos. Los módulos en el *back-end* son una división inicial en el que se separa la

lógica para facilitar su mantenimiento, normalmente cada módulo corresponderá con una o varias entidades muy relacionadas entre sí de la base de datos. Cada módulo se compone a su vez de subcapas (**Figura 23**), que incluyen:

- **Rutas**: Definen los puntos de acceso o *endpoints* a los cuales se accede desde el *front-end*.
- **Agente Intermediario o *middleware***: Procesa las solicitudes antes de que lleguen a los controladores, realizando diversas tareas, entre ellas, validación de datos, y manejo de errores.
- **Controladores**: Encargados de gestionar la lógica de aplicación, los controladores procesan las solicitudes recibidas y determinan las respuestas a enviar.
- **Servicio**: Esta capa se ocupa de la interacción con la base de datos.

La gran ventaja de tener una arquitectura tan modular es la facilidad para mantener el código y su escalabilidad, en otras palabras, la capacidad para expandir el sistema en un futuro de forma eficiente sin comprometer la integridad del código existente.

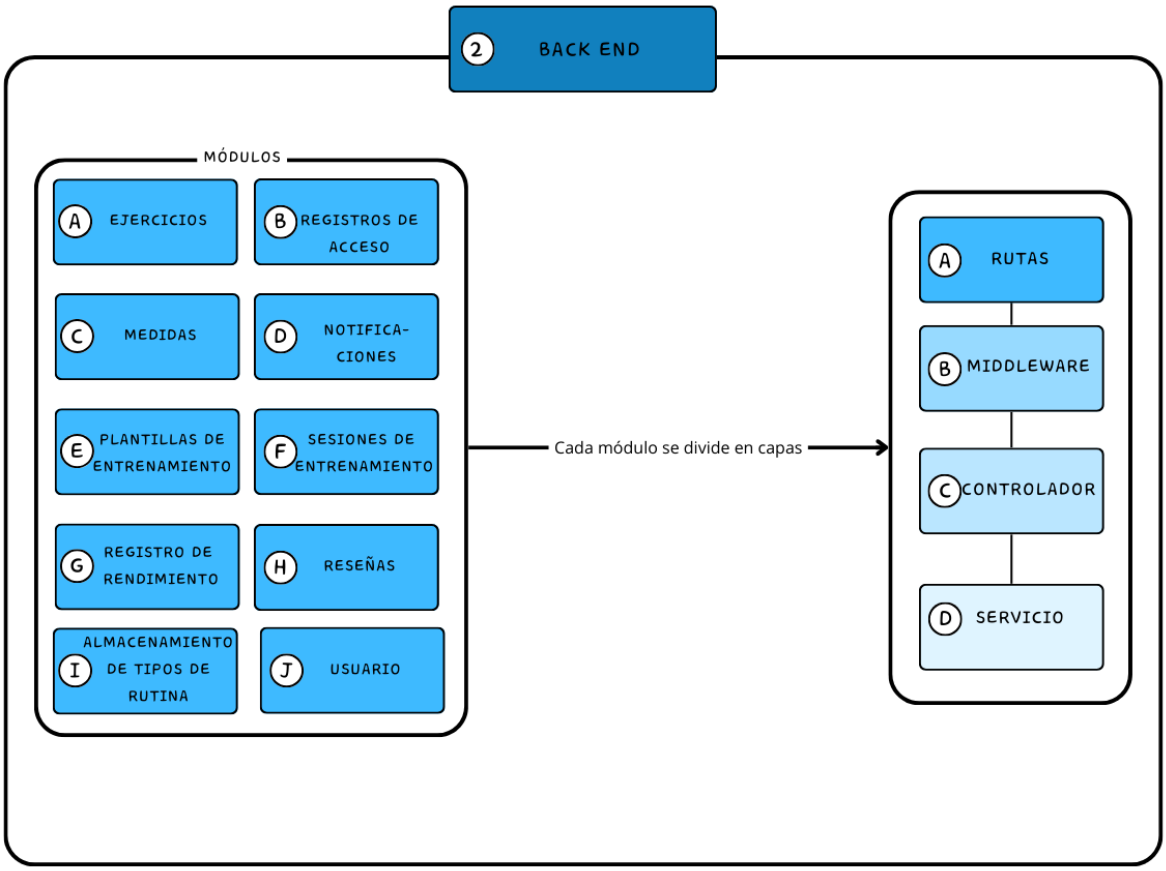

Figura 23 Arquitectura para cada módulo del *back-end*.

4.2.1 Capa de rutas

La capa de enrutamiento es el primer nivel de la arquitectura *back-end*, responsable de definir las *URL* y los métodos *HTTP* para acceder a los recursos de su aplicación desde el *front-end*. Estas rutas se encargan de asignar solicitudes *HTTP* y las conectan al resto de capas intermedias del *back-end*.

Se podría decir que el sistema muestra el patrón de diseño "**Facade**" o fachada y la capa de rutas sería la fachada, representando las funcionalidades del sistema de forma simple y el resto de las capas serían el subsistema complejo detrás de la fachada.

4.2.2 Capa de agente intermediario (*Middleware*)

La subcapa de agente intermedio o *middleware* añade funcionalidades a las peticiones. Esta subcapa en la práctica son funciones que se indican en las rutas, en orden de ejecución. Es opcional, pero muy útil, en concreto, se puede usar para añadir **validaciones**. Las validaciones verifican, en caso de que sea necesario, los datos que llegan desde el lado del cliente. En caso de que se cumplan unas reglas establecidas, permite el paso a las siguientes capas intermedias, en caso contrario, manda un error al lado del cliente indicando como se esperan los datos. Ejemplos de su uso se pueden observar en ciertas funciones en las que es crucial que ciertos parámetros no sean vacíos o que ciertos parámetros como un correo o una fecha tengan sentido. Es interesante añadir validadores por seguridad y para verificar que no se añada a la base de datos información incorrecta o peligrosa. Además, muchas veces las validaciones se repiten en varias partes del código y es una forma de tener el código estructurado y modularizado y, por tanto, mucho más fácil de mantener y trabajar en caso de añadir nuevas funcionalidades.

Otro *middleware* recurrente es aquellos que se encargan de recibir los ficheros enviados desde el *front-end* para poder tratarlos en el *back-end*.

Tal y como se ha señalado anteriormente, la gran ventaja de usar esta arquitectura y en concreto los agentes intermediarios es la posibilidad de expandir el funcionamiento en un futuro, en particular, habilita la posibilidad de añadir nuevos agentes de forma sencilla a la ruta, además permite reutilizar las funciones en varias rutas, haciendo su mantenimiento muy cómodo.

Por lo tanto, esta capa podríamos verla como si fuese un patrón decorador, donde vamos envolviendo las rutas con funcionalidades a nuestro gusto.

4.2.3 Capa de controlador

La capa de controlador definitivamente es la subcapa que posee la verdadera esencia de la lógica del *back-end*. En esta subcapa se ejecutan las principales acciones que determinan cómo se comporta el sistema. El controlador se responsabiliza de procesar las peticiones recibidas del lado del cliente y orquestar las operaciones que sean necesarias para satisfacerlas.

En el controlador, se llevan a cabo numerosas tareas, entre ellas destacan:

1. **Validación adicional**: Aunque en la mayoría de los casos las validaciones preliminares se pueden haber realizado en el *middleware*, el controlador también válida parámetros según las necesidades de las operaciones, de forma que asegura la integridad y consistencia de los datos previos a su procesamiento.
2. **Gestión de la lógica de negocio:** En el controlador se genera la lógica real de la aplicación, tomando decisiones basadas en los datos obtenidos, haciendo cálculos complejos, aplicando reglas preestablecidas y transformando o manipulando los resultados para devolver al cliente la respuesta deseada.
3. **Control de flujo**: Los controladores son capaces de escoger el flujo de la aplicación, eligiendo acciones en función de los datos recibidos. Son los que interactúan con la capa de servicio o con funciones auxiliares como cifrado, envío de correos o gestión de imagen con "Cloudinary". Además, devolverán una respuesta que llegará al cliente. Normalmente la respuesta que se enviará al lado del cliente tendrá un código de estado de respuesta y un *JSON* con datos si es que son necesarios. El *front-end* escogerá su forma de actuar en función de la respuesta que reciba del controlador.

Al ser una capa independiente del resto, el controlador puede funcionar para la ruta que escojamos siempre y cuando le lleguen los datos que el controlador necesite. También en caso de que cambie alguna capa como la de los servicios, por ejemplo, en el caso de que surja la necesidad de cambiar de base de datos, la lógica del controlador no tendría por qué cambiar.

4.2.4 Capa de servicios

La capa de servicios lleva a cabo las llamadas necesarias para la interacción con la base de datos. Esta capa recibe datos de los controladores y hace las llamadas a la base de

datos. Al estar separada la lógica, se fomenta la reutilización de código y se facilita la realización de pruebas y depuración. Esta separación también permite poder cambiar la lógica de cualquiera de las capas con un impacto mínimo en el resto del sistema.

4.3 Programación en lado del cliente

En el diagrama proporcionado de la **Figura 25**, el *front-end* se desglosa en tres grandes bloques de acuerdo con el tipo de usuario y la plataforma utilizada:

1. **Disposiciones**: Existen dos disposiciones que permiten el acceso a las mismas funcionalidades y pantallas. La disposición se seleccionará según el dispositivo para optimizar la experiencia del usuario. Las pantallas contendrán el contenido correspondiente y se ajustarán a la disposición del dispositivo. Por tanto, la arquitectura de la capa de cliente está diseñada para ser flexible y adaptable, proporcionando una experiencia de usuario coherente y eficiente tanto en dispositivos móviles como en plataformas web (**Figura 24**).

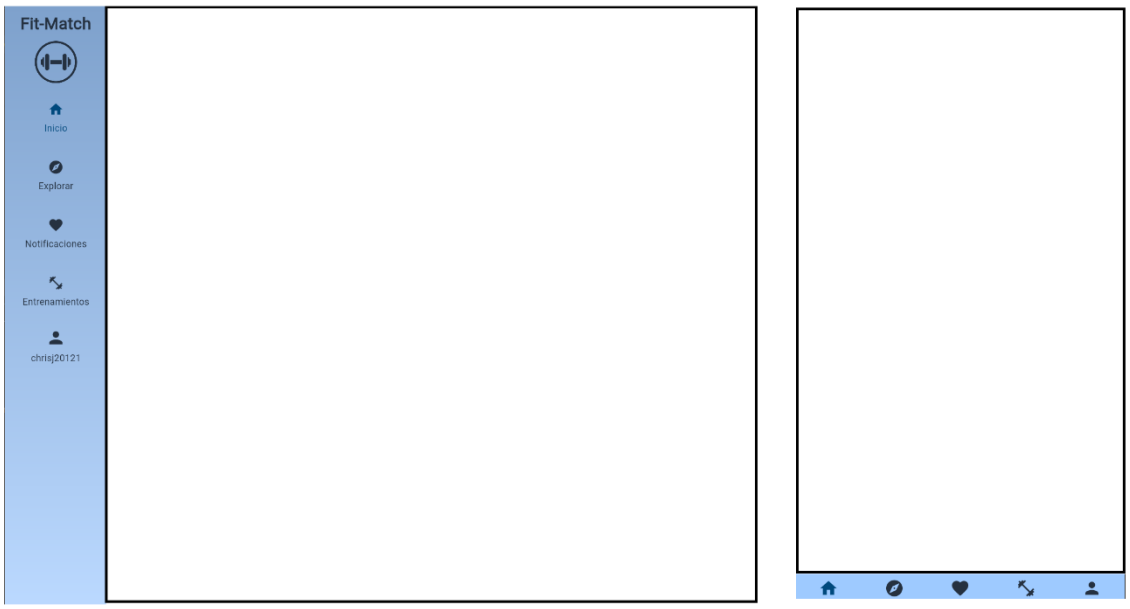

Figura 24 Disposición para pantalla ancha y estrechas.

2. **Pantallas**: Este bloque muestra las diferentes vistas o pantallas que están disponibles para cada tipo de usuario en la aplicación:

- Cliente: Las pantallas que un usuario cliente puede ver y con las que puede interactuar, como exploración de plantillas, perfil, manejo de sesiones de entrenamiento y recuperación de contraseña.
- Compartidas: Interfaces que son comunes para ambos roles de usuario. Aquí se podrían incluir las funcionalidades de acceso, inicio de sesión y registro, las cuales son fundamentales para cualquier tipo de usuario.
- Administrador: Pantallas específicas para administradores, que permiten la gestión de registros de acceso, ejercicios y usuarios. Estas pantallas proporcionan controles adicionales para la administración y el mantenimiento de la plataforma.

3. Las pantallas se forman utilizando:
    - *Widgets*: Un *widget* es un componente de la interfaz de usuario. Una pantalla es ella mismo un *widget* con otros *widgets* anidados. Los *widgets* se pueden dividir en aquellos que se usan para una pantalla o en aquellos más generales que son usados múltiples pantallas.
    - Proveedores: Gestionan parámetros y estados globales de la aplicación. Por ejemplo, gestionan la navegación entre los apartados de la barra de navegación, la sesión del usuario (datos, fecha de expiración) y el tema o paleta de colores seleccionado (oscuro o claro).
    - Servicios: Los servicios se encargan de hacer peticiones a la base de datos. Múltiples pantallas llaman a las funciones de los servicios.
    - Modelos: Especifica un tipo a los datos con los que se trabajan. De forma que las entidades de la base de datos sean más fácilmente manejadas en las pantallas y *widgets*.

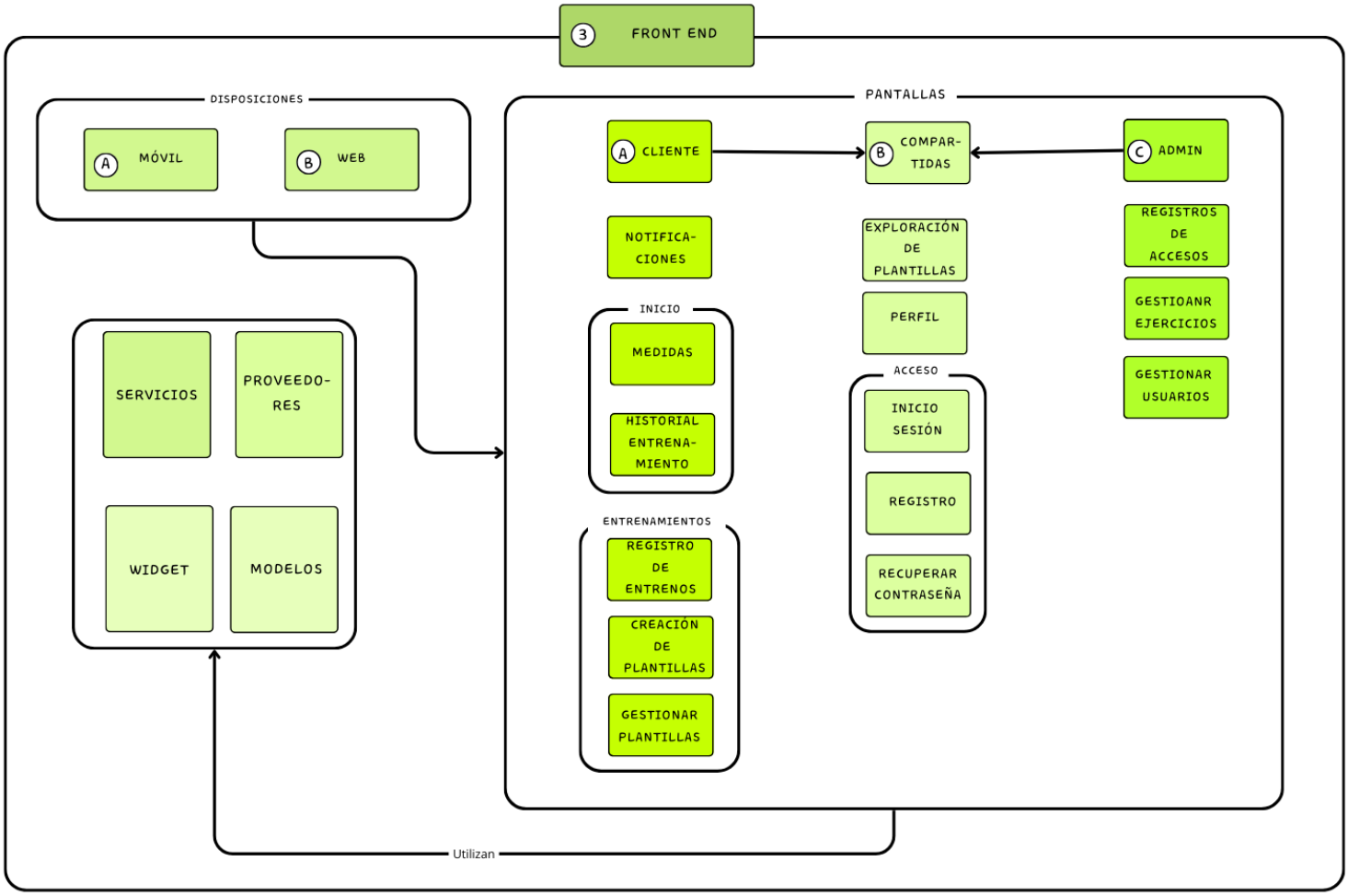

Figura 25 Esquema de diseño de la capa del cliente.

5

Diseño de la interfaz

Los esquemas de página a alto nivel o *wireframes* son esenciales para diseñar las interfaces más complejas, siendo un puente entre las ideas conceptuales e implementación real. Además, son clave para asegurar que se cumplen los casos de uso viendo a alto nivel como interactúan los usuarios con la aplicación y asegura que las interfaces sean intuitivas.

Es importante entender que un esquema a alto nivel no tendrá un gran número de detalles, muchas veces se obvian cosas que se dan por sentado y no tienen por qué coincidir al 100% con el resultado final, pero sí guían en términos generales su implementación. Por poner unos cuantos ejemplos:

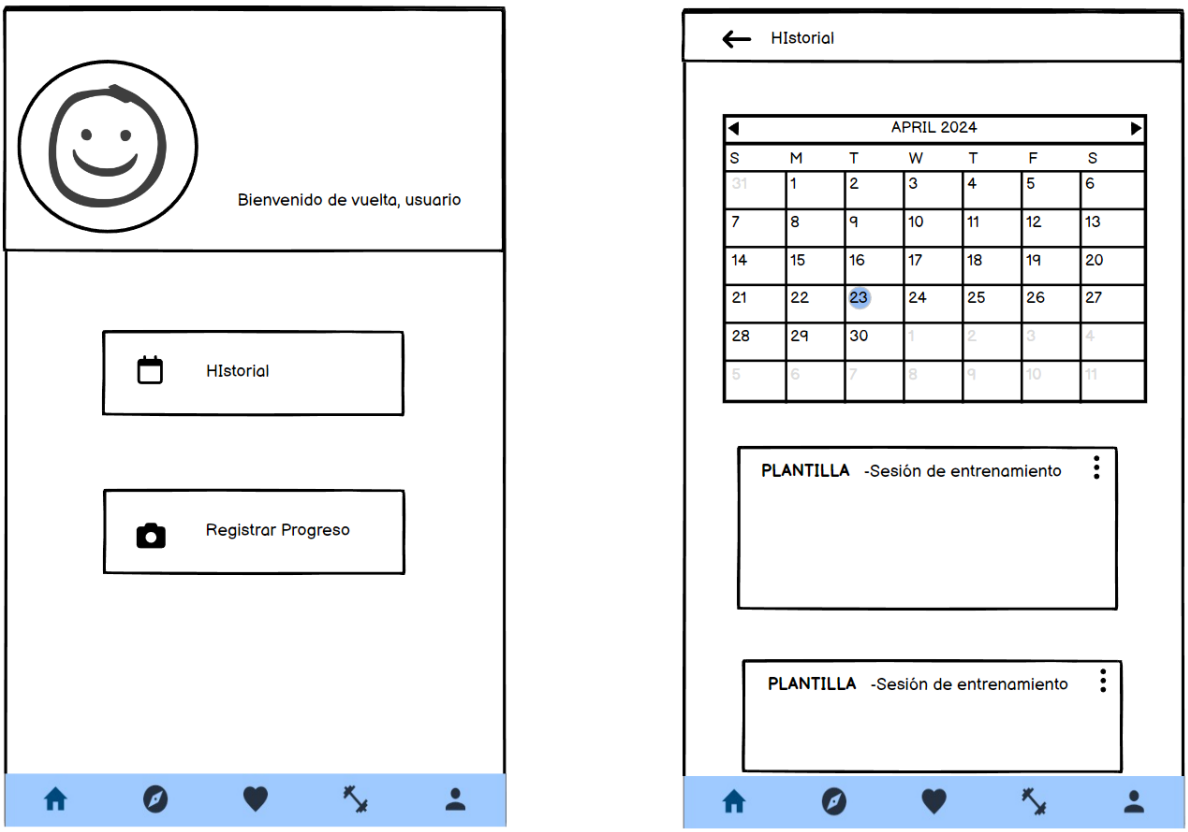

Figura 26 Esquemas de la interfaz inicio e historial de registros de entrenamiento

La **Figura 26** representa esquemas de la interfaz de inicio e historial de registros de entrenamiento, las cuales se pueden observar ya implementadas en el manual de usuario al final del documento en la **Figura 38**.

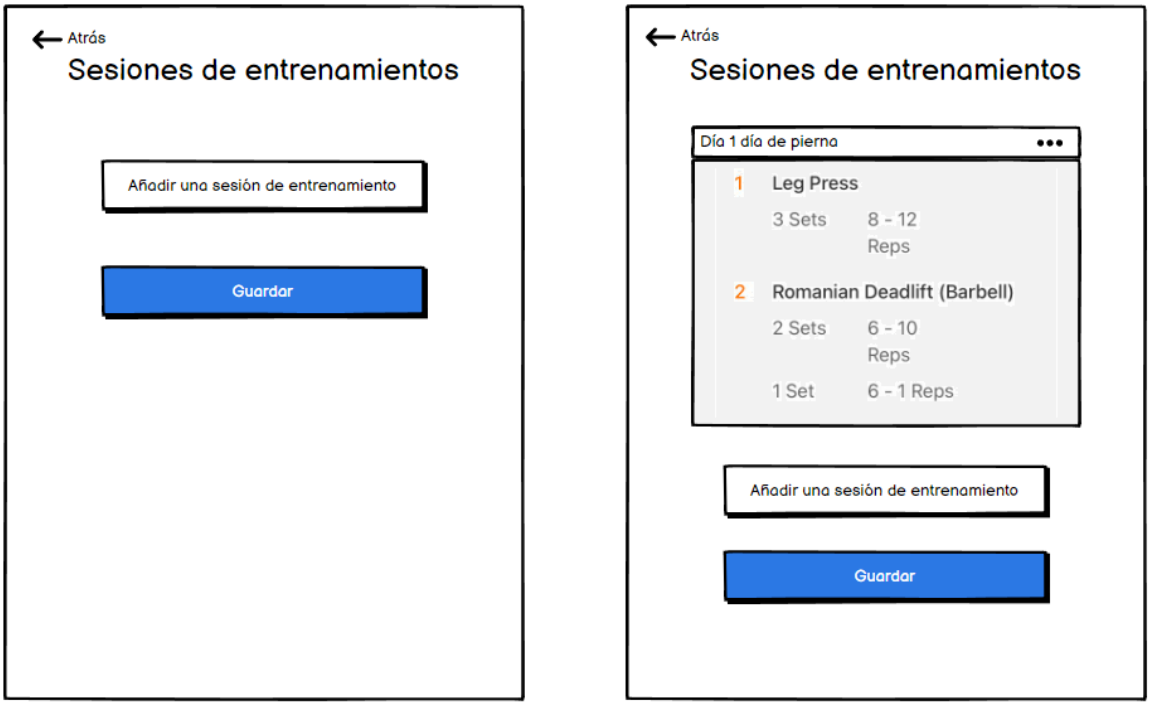

Figura 27 Esquema de la interfaz de plantilla de entrenamiento y gestión de sus sesiones

La **Figura 27** muestra un plano de la pantalla de gestión de sesiones de entrenamiento y cómo sería la visualización cuando existe una creada, en la versión final, para cada día se puede desplegar para ver su información, la **Figura 53** en el manual de usuario muestra el diseño final.

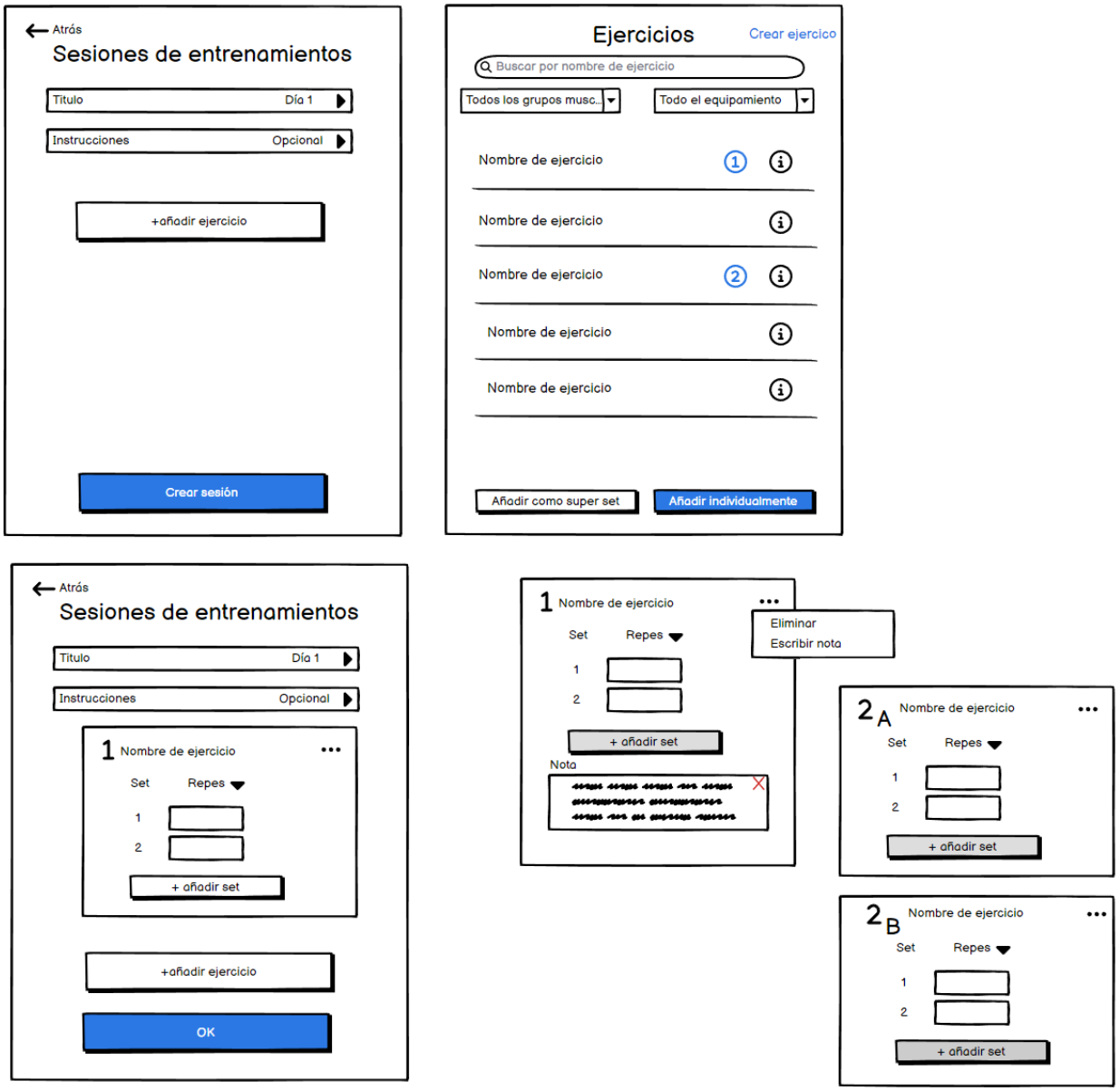

Figura 28 Esquema de la interfaz de creación de sesiones de entrenamiento

La **Figura 28** muestra el diseño de la interfaz para crear una sesión de entrenamiento. Este esquema de pantalla guía de manera intuitiva a través del proceso, desde seleccionar un ejercicio hasta configurar las tarjetas de actividad. Además, destaca las dos maneras en que podemos elegir un ejercicio: individualmente o por grupos, conocidos como superseries. El resultado final puede encontrarse en el manual de usuario en la **Figura 55**.

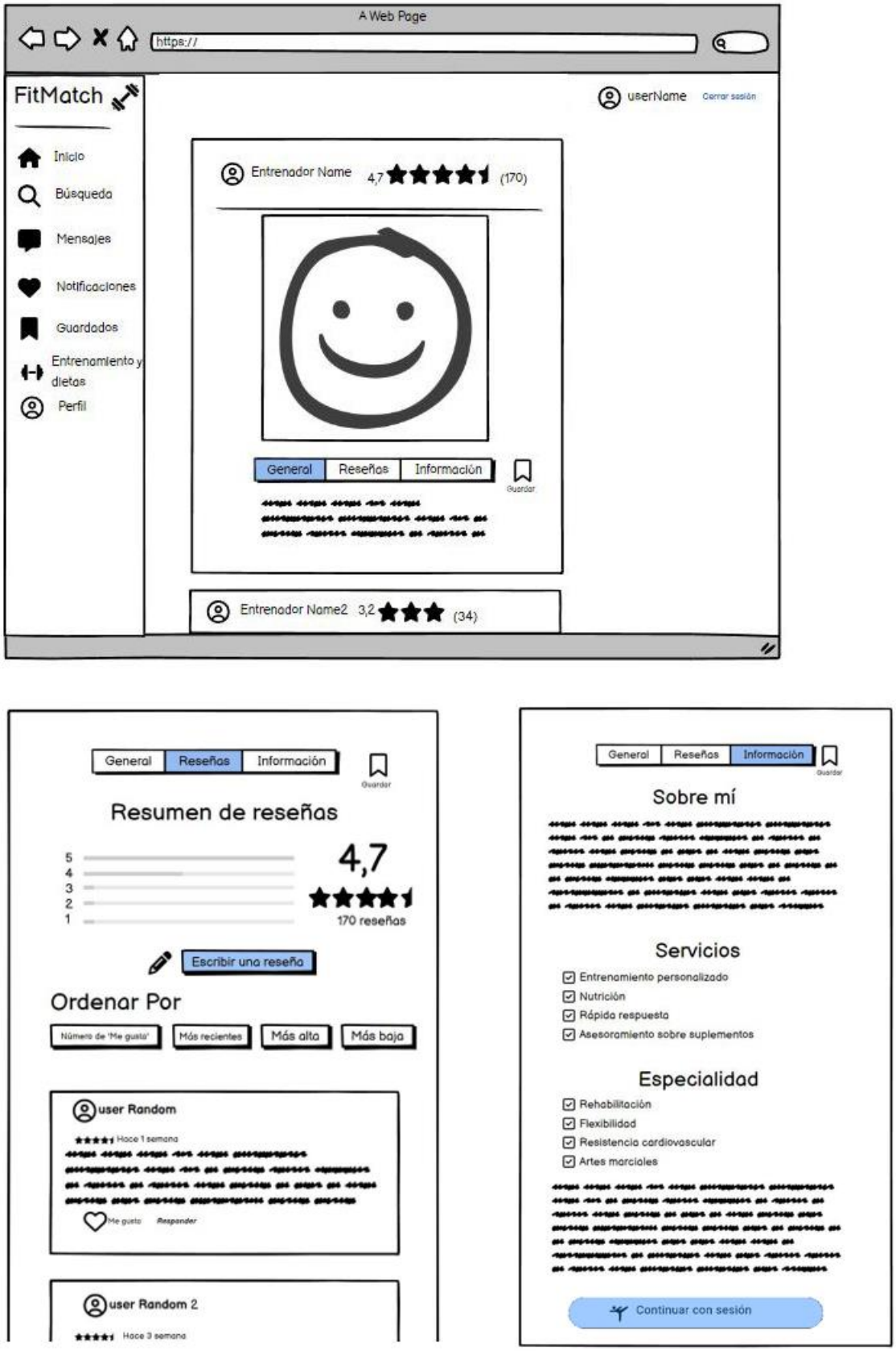

Figura 29 Esquema de la interfaz de una tarjeta de plantilla de entrenamiento

Como último ejemplo, se puede ver en la **Figura 29** un esquema sobre las tarjetas de las plantillas de entrenamiento y sus secciones. Para verla ya desarrollada se puede acudir al manual de usuario a la **Figura 47 y 48.**

6

Implementación y pruebas

6.1 Implementación

A lo largo del documento se ha discutido y desarrollado sobre las primeras fases del ciclo de vida del desarrollo de un proyecto software (**Figura 30)**, abarcando desde la definición de requisitos y planificación de interfaces hasta el diseño del sistema y arquitecturas. Es el momento de profundizar en las fases de implementación y pruebas, correspondiente a la cuarta y quinta etapa de las etapas del desarrollo visibles en la **Figura 30.**

Figura 30 Esquema que muestra las etapas del desarrollo que se han seguido para la creación del desarrollo del *software*.

La implementación de la aplicación mediante el uso de técnicas *software* es un proceso largo y costoso. Para ilustrar esto, se describirán algunos desarrollos específicos dentro del proyecto: la carga perezosa, gestión de seguridad y *tokens* de sesión, el almacenamiento de imágenes, los registros de intentos de acceso a la aplicación, así como el envío y recepción de notificaciones. Además, se incluirán detalles sobre la eliminación dinámica de plantillas de entrenamiento, la edición de sesiones de entrenamiento y el registro del rendimiento de las sesiones y, por último, se abordará la creación de formulario y su diseño, junto con la generación de estadísticas.

6.1.1 Carga perezosa

La carga perezosa es una técnica de optimización en el desarrollo web y de aplicaciones que consiste en cargar datos o componentes solamente cuando son necesarios. Esta técnica se ha utilizado para mejorar la experiencia del usuario al reducir el tiempo de carga inicial y utilizar los recursos de manera más eficiente, especialmente útil cuando se manejan grandes volúmenes de datos. En este proyecto, en específico se ha utilizado para la carga de los siguientes elementos:

- Los registros de los intentos de acceso.
- Los registros de los bloqueos de las *IPs*.

- Las plantillas de entrenamiento públicas en el apartado de compartir.
- Los ejercicios a la hora de seleccionar uno para una sesión de entrenamiento o gestionarlos desde el rol de administrador.
- En la carga de reseñas de una plantilla.
- Los usuarios a la hora de gestionarse desde el rol de administrador.

La implementación de la carga perezosa consiste en dividir todos los datos en lotes o páginas y se ha desarrollado mediante la gestión de la paginación en el *back-end* y la detección del desplazamiento o deslizamiento en el *front-end*.

En el ***front-end***, la gestión de la carga perezosa se realiza utilizando un objeto "ScrollController", que controla el deslizamiento o desplazamiento de la pantalla, cuando se va acercando al final, el objeto "ScrollController" se activa la función "loadMoreContentOnScroll". Esta función comprueba dos condiciones importantes antes de proceder a cargar más contenido: si aún hay más publicaciones por cargar ("hasMore") y si el sistema no está ocupado cargando otros datos ("isLoading"). Si ambas condiciones se cumplen, se procede a invocar la función "loadMoreContent".

"loadMoreContent" es una función que envía al *back-end* el número de lote o página actual que se está visualizando, representado por la variable "currentPage". Cada vez que se realiza una carga exitosa de más publicaciones, esta variable se incrementa, preparando el sistema para solicitar el siguiente lote de datos en la próxima acción de desplazamiento al final.

En el ***back-end*** el controlador recibe parámetros como el número de página o lote y el tamaño de cada página ("page" y "pageSize"). Según estos parámetros, se calcula el número de publicaciones a omitir ("offset") multiplicando el tamaño del lote o página con el número de lote actual y se realiza una consulta a la base de datos para recuperar solo el segmento o lote específico de datos requeridos. Este enfoque asegura que el servidor solo envíe los datos que son actualmente visibles o próximos a ser mostrados en la interfaz de usuario, en lugar de cargar todos los datos de una sola vez.

Este enfoque permite además enviar más filtros si son necesarios y que queden paginados en lotes, dando una gran flexibilidad y asegurando una gran eficiencia en la carga de datos.

6.1.2 Gestión de seguridad y Tokens de sesión

La gestión de seguridad para las contraseñas y los tokens *JWT* [22] ("JSON Web Token") juega un papel crucial para proteger la información de los usuarios y asegurar la comunicación entre el cliente y el servidor.

En primer lugar, en el lado del servidor se realiza un **cifrado de contraseña,** cuando el usuario crea una cuenta o cambia su contraseña, la contraseña en texto plano es cifrada usando el algoritmo "SHA-256". "SHA-256" se crea utilizando un secreto o "clave *hash*", guardado en las variables de entorno y transforma el texto en uno que no puede descifrarse y es único para cada combinación de caracteres. (**Figura 31**). Para verificar que la contraseña es correcta al iniciar sesión, la contraseña en texto plano se introduce de nuevo por el algoritmo con el mismo secreto "hash" y se compara con la contraseña cifrada guardada en la base de datos. Por lo que en ningún momento se expone en el proceso de verificación la contraseña en texto plano.

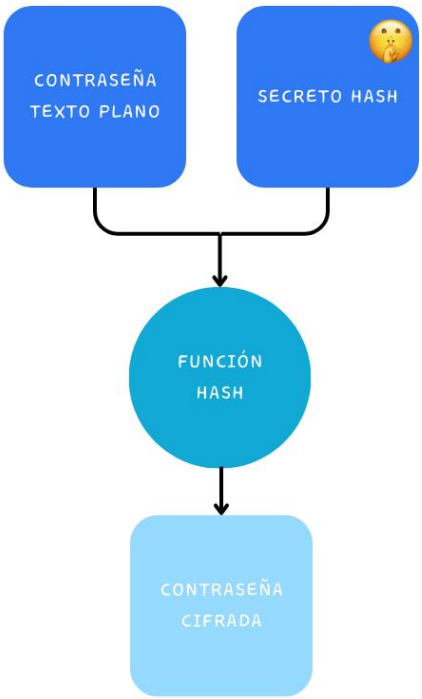

Figura 31 Funcionamiento de cifrado de contraseñas con "SHA-256"

Al verificar que las contraseñas coinciden en el proceso de inicio de sesión, se creará un ***token*** "JWT" (*Json Web Token*) en el *back-end* y se enviará posteriormente al *front-end*. Este *token* se genera utilizando la librería "jsonwebtoken" de "Node.js" y tiene la información necesaria para el manejo del usuario en el lado del cliente. Tendrá el correo electrónico, datos del usuario como nombre de perfil, tiempo de expiración de la sesión... El *token* "JWT" es de gran utilidad para asegurarse que no pueda ser falsificada la

información. Una vez recibido el *token* en el *front-end* se descifra y se utilizan los datos del usuario en la aplicación mientras la sesión siga activa.

6.1.3 Registro de intentos de acceso a la aplicación

Cada intento de acceso al sistema ya sea exitoso o fallido, se registra en la base de datos. Se guarda el correo de usuario y la *IP* con la que se ha intentado acceder y si el intento ha sido exitoso. Estos datos son importantes para identificar patrones de acceso sospechosos y también útiles para obtener estadísticas de actividad en la aplicación.

Si desde una *IP* determinada se acumulan varios intentos fallidos de acceso (un número definido en las variables de entorno), se bloquea el acceso del usuario con esa *IP*. Esto significa que cualquier solicitud adicional desde esa *IP* será automáticamente rechazada por el sistema durante un periodo especificado, incluso si las credenciales son correctas. Este mecanismo es esencial para proteger contra ataques de fuerza bruta, donde un atacante intenta adivinar contraseñas mediante el envío repetitivo de solicitudes de inicio de sesión.

Esta información se manda desde el *back-end* al *front-end* y permite visualizarse si se tiene el rol de administrador (**Figura 60**).

El mecanismo de bloqueo de *IP* es especialmente efectivo contra ataques de fuerza bruta, creando disuasión para continuos ataques, reduciendo el daño potencial y pudiendo detectar de forma temprana tendencias anómalas.

6.1.4 Almacenamiento de imágenes

La gestión de imágenes es una parte importante del flujo de trabajo y se ha llevado a cabo con "Cloudinary", una plataforma de gestión de activos digitales. Algunas de las ventajas de usar un servicio como "Cloudinary" son la **seguridad**, al reducir el riesgo de que las imágenes sean manipuladas o accedidas de forma inadecuada y la **escalabilidad**, al no tener que preocuparse por el almacenamiento de un servidor propio, por otra parte, establece un límite de almacenamiento gratis, por lo que lo que en caso de que el proyecto acabase escalando y necesitando más espacio, habría que pagar una mensualidad, por lo que dependiendo del proyecto, a la larga podría ser interesante realizar la inversión para tener un servidor propio. Pero para proyectos con un número limitado de imágenes como proyectos de comercio electrónico o "startups" puede ser la herramienta perfecta. En el proyecto se ha gestionado la gestión de imágenes con esta plataforma tanto en el *back-end* como en el *front-end*.

En el ***back-end*** se tiene un configurador que es un archivo con métodos para usar en los controladores. Se accede a "Cloudinary" accediendo a su API, utilizando la librería "cloudinary" de "Node.js". Las funciones de este configurador que se llaman desde los controladores son principalmente para subir y borrar imágenes.

Cuando se sube una imagen nueva (por ejemplo, durante la creación o edición de una plantilla o al asignarse a un usuario), primero se verifica si ya existe una imagen para esa plantilla o usuario. Si existe, se obtiene y la elimina de "Cloudinary" y luego se inserta la nueva imagen a Cloudinary que devuelve una *URL* pública que permite visualizar la imagen y se guarda en la base de datos. Además, también se podrán borrar de "Cloudinary" si desde el *front-end* se solicita hacerlo (por ejemplo, al quitar la imagen asociada a un usuario). Es útil ir borrando imágenes no utilizadas para usar el menor almacenamiento posible.

En ***front-end*** para mostrar las imágenes se hace una solicitud *HTTP* a la *URL* pública de la imagen (guardada en la base de datos). Para guardar las imágenes, primero se seleccionan utilizando un componente llamado "ImagePicker" que permite al usuario elegir una imagen de su galería, por último, al terminar el formulario se envía junto con los demás datos para que sea gestionada en el lado del servidor.

6.1.5 Notificaciones

A la hora de crear una notificación, tenemos tres parámetros que se pasarán al controlador del *back-end* para crear e insertar una notificación en la base de datos. Estos parámetros son el tipo, texto, *id* de usuario. El texto es el contenido de la notificación y el *id* de usuario será un parámetro que sirva como identificador para el usuario destinatario de la notificación. Por otra parte, el tipo es un atributo que sirve como resumen de la notificación y puede ser útil para realizar filtros. Habrá un tipo de notificación por cada situación en la que se pueda enviar; en la aplicación hay definidos los siguientes tipos de notificaciones:

- "UPDATE_SESSION": Las notificaciones de este tipo indican que una sesión de entrenamiento de una plantilla de entrenamiento activa para un usuario ha sido editada por su creador.
- "NEW_REVIEW": "Notificación que llegará al usuario creador de una plantilla de entrenamiento que esté pública al recibir una reseña.
- "LIKE_COMMENT": Esta notificación se enviará al usuario con un comentario en una reseña, si este comentario recibe un "Me gusta".

- "LIKE_REVIEW": Esta notificación se envía si un usuario escribe una reseña a una plantilla de entrenamiento y algún usuario le da "Me gusta".
- "REVIEW_RESPONSE": Esta notificación se envía si un usuario escribe una reseña a una plantilla de entrenamiento y algún usuario le responde.

El controlador para crear una notificación se integra con otras funciones del sistema y es llamado por otros controladores, por ejemplo, a la hora de editar una sesión, el sistema comprobará si existen usuarios que tengan esa plantilla activa y dinámicamente creará una notificación para ellos con un mensaje personalizado que indique sesión de entrenamiento ha sido editada y a que plantilla de entrenamiento pertenece. En otras palabras, desde el *front-end* nunca se llamará al controlador para crear una notificación, si no que los demás controladores cuando necesiten generar una notificación señalizarán al controlador para que la cree.

El controlador tras crear la notificación la inserta en la base de datos con dos parámetros extra, los cuales son la fecha de creación y otro parámetro que indique si la notificación ha sido leída. Esto da pie al siguiente controlador que permite cambiar una notificación de "no leída" a leída. Esto ayuda a los usuarios a distinguir entre las notificaciones nuevas y consultadas.

Por último, en el *back-end* se establece la lógica necesaria para poder consultar y enviar los datos al *front-end* dado un *id* de usuario destinatario y se devolverán ordenadas por fecha de creación. Consecuentemente en el *front-end*, al entrar en la vista de notificaciones se pedirán al *back-end* las notificaciones utilizando esta llamada y se indicarán de forma visual aquellas notificaciones leídas y no leídas (**Figura 47)**. Las notificaciones leídas mostrarán la fecha de creación de la notificación, un icono del contorno de un sobre y el texto de la notificación. En cambio, las notificaciones no leídas se mostrarán con el texto en negrita y el icono del sobre estará resaltado.

Nada más entrar en la pantalla de notificaciones, el *front-end* enviará la solicitud al *back-end* para indicar que las notificaciones han sido leídas y por tanto aparecerán como leídas al volver a entrar en la pantalla (actualizando la página web o en dispositivos móviles deslizando hacia arriba).

6.1.6 Eliminación dinámica de plantillas de entrenamiento

El manejo de datos en proyectos de software requiere decisiones estratégicas para balancear la trazabilidad y la eficiencia del almacenamiento. En este proyecto, donde los usuarios tienen la libertad de crear y eliminar sesiones de entrenamiento y plantillas, se plantea un reto particular:

1. **Diferenciación de datos**: No todos los datos eliminados son considerados "basura". Aquellos que se utilizan activamente o tienen registros de rendimiento asociados son valiosos y deben ser preservados.
2. **Implementación del atributo "hidden":** Para manejar este desafío, se utiliza un valor lógico llamado "hidden". Este atributo se establece como verdadero para ocultar, en lugar de eliminar, las plantillas de entrenamiento y las sesiones que contienen datos valiosos, evitando así su eliminación completa.

3. **Protección de la actividad del usuario**: Al marcar una sesión de entrenamiento como oculta, se garantiza que la información relevante para el usuario activo se preserve, aunque la sesión no sea visible para otros usuarios.

4. **Gestión de modificacione**s: Es crucial conservar la versión original de una sesión de entrenamiento cuando se realiza una edición, especialmente si existen registros de rendimiento asociados. Esto asegura que el historial refleje adecuadamente el desempeño pasado, a pesar de las modificaciones en los ejercicios de la sesión.

Este enfoque dual asegura que la base de datos no acumule datos innecesarios, facilitando la gestión del almacenamiento sin comprometer la integridad y la trazabilidad de los datos esenciales. La utilización del atributo "hidden" permite una solución flexible que se ajusta a las necesidades dinámicas de los usuarios y a la integridad de la información en el sistema.

Por tanto, el sistema al modificar o eliminar datos realiza la siguiente iteración: En primer lugar, comprueba si hay algún usuario que pueda requerir esos datos, en caso de encontrarlo no elimina el dato de la base de datos, lo esconde ("hidden" se establece como verdadero). En cualquier otro caso, lo elimina totalmente de la base de datos.

6.1.7 Edición de una sesión de entrenamiento

Al crear una sesión de entrenamiento, se crea vacía y nos redirige a una pantalla que permite editar la sesión de entrenamiento. Esta pantalla también sirve para editar sesiones de entrenamientos ya existentes. La pantalla se inicializa cargando los detalles de la sesión de entrenamiento (estarán vacíos si es una nueva sesión de entrenamiento). Esto incluye obtener el nombre de la sesión, las instrucciones asociadas y las agrupaciones de ejercicios previamente configuradas. Durante este proceso, también se cargan los tipos de registros disponibles desde la base de datos que pueden aplicarse a los ejercicios. La carga inicial es crucial porque establece el contexto en el que el usuario interactuará con los datos de la sesión.

Es importante entender que una sesión de entrenamiento tiene "agrupaciones de ejercicios detallados" asociados. Una agrupación de ejercicios detallados es una entidad que representa uno o más ejercicios detallados. Un ejercicio detallado es un ejercicio de la base al que se le añaden instrucciones dependiendo del tipo de registro seleccionado en base de datos. Están agrupados debido a que pueden existir "superseries", en otras palabras, ejercicios pensados para realizarse uno detrás del otro sin descanso.

La pantalla permite asignar o editar un nombre y una descripción. Además, utiliza el *widget* "ExerciseCard" para mostrar cada grupo de ejercicios detallados. Este componente es vital para facilitar la visualización y modificación de cada ejercicio individualmente. Los usuarios pueden:

- Reordenar ejercicios: Esto se maneja a través de una funcionalidad que permite ajustar el orden en que se muestran los ejercicios.
- Editar notas: Cada ejercicio puede incluir notas específicas, y los usuarios pueden editar estas notas directamente desde la interfaz.
- Agregar series a ejercicios: Si el usuario necesita agregar más series a un ejercicio, puede hacerlo fácilmente.
- Eliminar ejercicios o series: Esta es una operación crítica y se gestiona con cuidado para asegurar que no afecte inapropiadamente el orden o la integridad de los datos restantes.
- Asignar instrucciones a cada serie de cada ejercicio detallado.

Existe un validador que comprueba que antes de guardar una sesión de entrenamiento, la sesión tenga al menos un nombre y un ejercicio con una serie configurada (la descripción es opcional).

6.1.8 Registrar el rendimiento de una sesión

De forma similar a crear una sesión de entrenamiento, se posee un *widget* "RegisterCard" se encarga de manejar la interacción del usuario con cada grupo de ejercicios detallados de una sesión de entrenamiento. Este *widget* permite al usuario añadir series de ejercicios, actualizar los detalles de las series existentes y eliminarlos si es necesario, proporcionando una interfaz intuitiva y eficiente.

Cuando se inicia la pantalla encargada de registrar el rendimiento de una sesión de entrenamiento, se realiza una llamada inicial para cargar los datos de la sesión de entrenamiento, incluyendo todos los ejercicios que el usuario debe realizar. Este proceso de inicialización es crucial porque establece el contexto en el que el "RegisterCard" operará.

El *widget* "RegisterCard" tiene la capacidad de interactuar con cada ejercicio detallado con instrucciones. Los usuarios pueden ver información específica del ejercicio, como el nombre, la descripción y, opcionalmente, un video explicativo, accediendo a través de un icono de información. Esta funcionalidad es vital para asegurar que el usuario tenga claridad sobre cómo realizar cada ejercicio correctamente.

Pero su principal uso del *widget* "RegisterCard" es la de apuntar y añadir series. Cuando un usuario decide añadir una nueva serie a un ejercicio, la aplicación no solo registra esta nueva serie en la base de datos, sino que también permite al usuario ingresar o actualizar detalles como el número de repeticiones, el peso levantado o el tiempo empleado, según el tipo de registro que el ejercicio requiera. Este proceso es interactivo y está diseñado para capturar la entrada del usuario en tiempo real, lo cual es crucial para mantener la precisión del registro de entrenamiento.

6.1.9 Diseño de formularios

En el diseño de los formularios se ha priorizado la modularidad y la reutilización de componentes, lo que facilita la escalabilidad y el mantenimiento del código. Cada elemento del formulario se encapsula en *widgets* específicos, lo que permite una mayor claridad, legibilidad y una separación lógica del comportamiento y la presentación de la interfaz de usuario.

Para mejorar la experiencia del usuario y hacer los formularios más interactivos y atractivos, se utilizan componentes como el "*Stepper*". Este componente guía al usuario a través de una serie de pasos, haciendo que el proceso de registro o creación sea más claro y menos abrumador. Cada paso se define claramente y se pueden realizar validaciones específicas antes de pasar al siguiente, mejorando así la calidad de la entrada de datos y la interacción del usuario.

Además, se hace uso de *widgets* como "*TextField*", "*ImagePicker*", y "*DatePicker*", entre otros, para recoger de manera efectiva la información necesaria de los usuarios. Estos componentes se configuran con validaciones para asegurar que los datos recopilados cumplen con los requisitos específicos del sistema. Además del uso de componentes ya predefinidos por "Flutter", también se han creado *widgets* propios en caso de necesario un mayor control sobre su aspecto o funcionalidad.

La disposición de los elementos en pantalla se gestiona mediante contenedores flexibles que se adaptan a diferentes tamaños de pantalla. Esto se logra utilizando *widgets* como "*ScrollView*" y "*SafeArea*", que aseguran que la aplicación sea usable en una amplia variedad de dispositivos, manteniendo una presentación coherente y estética.

Este enfoque modular no solo facilita una experiencia de usuario cohesiva y adaptativa, sino que también optimiza el desarrollo y la futura escalabilidad de la aplicación. Con estas prácticas, se establece un sólido fundamento para una aplicación robusta, adaptable y fácil de usar.

6.1.10 Estadísticas

La aplicación será capaz de mostrar el progreso en el tiempo tanto del peso o medidas corporales como de cada ejercicio realizado durante una sesión de entrenamiento. Para las sesiones de entrenamiento, se incluirá un *widget* en forma de botón desplegable que permitirá seleccionar el ejercicio específico cuyas estadísticas se desean consultar. En el caso de las medidas corporales, este mismo botón desplegable presentará todas las opciones disponibles para el registro de medidas, facilitando así la visualización y seguimiento del progreso.

Además, se da la posibilidad de mostrar la información de dos formas alternativas: mediante una **gráfica** o mediante una **lista**.

Para implementar el gráfico se utiliza el paquete "syncfusion_flutter_charts" para proporcionar una representación gráfica detallada y dinámica de los datos. Este paquete permite una gran flexibilidad y personalización, facilitando la creación de gráficos interactivos que mejoran la experiencia del usuario al permitirle explorar en profundidad las tendencias y detalles de las medidas registradas. Algunas características de la implementación de la gráfica son:

- Interactividad: El componente "ZoomPanBehavior" se integra en el gráfico, permitiendo al usuario hacer zum y desplazarse por el gráfico. Esto es especialmente útil cuando se trata de grandes conjuntos de datos o periodos extendidos de tiempo, ya que el usuario puede enfocarse en segmentos específicos del gráfico para un análisis más detallado.
- "*Tooltip*": proporciona al usuario detalles adicionales al seleccionar o pasar el cursor sobre puntos específicos del gráfico, como los valores exactos de las medidas en fechas particulares o el valor registrado en el ejercicio de la sesión de entrenamiento para esa fecha.
- Eje X y Eje Y: El eje X se configura para mostrar fechas con formatos apropiados que cambian dinámicamente basados en el rango de fechas de los datos (por días o por meses), mientras que el eje Y muestra las medidas con la unidad apropiada, mejorando así la claridad y la relevancia de la información visualizada.

Como alternativa al gráfico, los datos también pueden ser representados en una lista. Esta visualización es beneficiosa para usuarios que prefieren una vista más tradicional o detallada de los datos punto por punto sin necesidad de interpretar un gráfico. Para cada entrada en la lista se muestra la fecha y el valor exacto, se tendrá en cuenta si es una unidad de medida o de peso y la preferencia del usuario entre el sistema métrico y el imperial.

6.2. Pruebas

El desarrollo del *software* del proyecto no consiste solo en la implementación de nuevas funcionalidades, sino que también es un proceso iterativo de depuración y corrección de errores.

Además de la continua depuración, se han proporcionado pruebas unitarias para las funciones de encriptación mediante SHA-256. Estas funciones son críticas ya que están relacionadas con la seguridad en el manejo de contraseñas, estas pruebas unitarias se han realizado utilizando el *framework* "Jest" [24] para ejecutar y evaluar las pruebas.

La primera función de la que sea realizado una prueba se encarga de tomar una contraseña como entrada y devuelve una versión cifrada (*hashed*) de esta contraseña. El propósito de esta función es asegurar que las contraseñas no se almacenan en texto plano en bases de datos, lo cual es una práctica de seguridad esencial.

La prueba unitaria creada para esta función tiene el objetivo de verificar que esta función devuelve un *hash* consistente para la misma entrada. Es decir, si aplicas la función a una misma contraseña en diferentes momentos, el resultado debería ser siempre el mismo, suponiendo que el secreto no cambia. Para implementarlo se establece un valor constante como secreto *hash* y se prueba que el *hash* de una contraseña específica es siempre el mismo cada vez que se aplica la función.

La siguiente función verifica si una contraseña proporcionada por el usuario, cuando se cifra, coincide con una contraseña almacenada de la base de datos. Esta es una operación crítica durante el proceso de autenticación de usuarios al iniciar sesión.

La prueba unitaria de esta segunda función se divide en un caso positivo y negativo, es decir, para que se considere la prueba un éxito, la función debe retornar el valor booleano "*true*" cuando la contraseña de entrada coincida con el *hash* almacenado y retornar "*false*" en otro caso. Para implementar el caso positivo de esta prueba primero se cifra una contraseña y luego se verifica que la función devuelve "*true*" cuando se compara esta misma contraseña con su hash. El caso negativo se implementa de forma similar, pero esta vez se utiliza una contraseña correcta y una incorrecta, se cifra la correcta, y luego se verifica que la función devuelve "*false*" al comparar la contraseña incorrecta con el hash de la correcta.

Las pruebas unitarias son fundamentales para garantizar que cada componente individual (en este caso, funciones) de un programa funcione correctamente de manera aislada. Permiten identificar errores en las etapas tempranas del desarrollo, facilitando las correcciones antes de que el código se integre con otras partes del sistema. En el contexto de un *back-end*, las pruebas unitarias son más críticas para las funciones y métodos que manejan la lógica del negocio y las operaciones de datos, más que para los controladores en sí. Los controladores a menudo dependen de la interacción con modelos de datos o servicios externos y pueden ser más adecuados para pruebas de integración o funcionales. Por esta razón, aunque es importante asegurar la calidad y el funcionamiento de los controladores, las pruebas unitarias no son siempre la herramienta prioritaria para ellos. Es más beneficioso centrarse en asegurar que la lógica del negocio y las operaciones de datos funcionen correctamente bajo condiciones controladas y predecibles, lo cual es el objetivo de las pruebas unitarias.

7

Conclusiones y Trabajos Futuros

Para terminar, se presentará una evaluación sobre las metas alcanzadas y las lecciones aprendidas a lo largo del desarrollo del proyecto. Posteriormente, se discutirán las ampliaciones posibles que podrían considerarse en para expandir la funcionalidad de la aplicación.

7.1 Objetivos cumplidos

Como se vio al principio del documento, se busca promover el ejercicio físico creando una herramienta que facilite el seguimiento de la progresión del deporte realizado. Es por ello por lo que el foco del proyecto se pone en las prácticas y técnicas de desarrollo del *software* para crear una herramienta multiplataforma con una interfaz de usuario amigable y que permita crear y compartir sesiones de entrenamiento, realizar su seguimiento mostrando estadísticas, permitir retroalimentación de otros usuarios y por supuesto asegurar la seguridad de datos y un rendimiento eficaz.

Como se ha podido ver a lo largo del documento estos objetivos se han cumplido en gran medida al igual que todos los casos de uso y requisitos para ambos roles de usuario y de administrador.

7.2 Dificultades encontradas

El desarrollo del *software* es un proceso largo y tedioso que requiere de paciencia, perseverancia, creatividad y estudio de las tecnologías utilizadas. Para el desarrollo de la aplicación, se ha aprendido desde cero a utilizar el lenguaje de programación "Dart" y "Flutter" mediante la lectura de documentación y guías. La programación en "Dart" es bastante desafiante al principio para cualquier usuario que provenga de diseñar interfaces utilizando lenguajes de marcado como "HTML", por lo que gran tiempo invertido al comienzo del desarrollo de la aplicación va dirigido a aprender a utilizar esta tecnología. Por otra parte, gran parte del tiempo del desarrollo se ha invertido en la depuración y eliminación de errores que surgen al no dominar el lenguaje de programación por completo.

El diseño de interfaces atractivas para distintos tamaños de pantalla ha resultado un gran desafío. Se han creado las pantallas de forma que se vean bien para dispositivos móviles de pantalla estrechas, tabletas y para ordenadores. Se ha logrado conseguirlo cambiando la disposición de los componentes según la amplitud de la pantalla. Esto es una tarea complicada para cualquier desarrollador acostumbrado a solo diseñar páginas web.

Había mucha lógica en el proyecto que en un principio pudiera parecer trivial, pero lleva más trabajo de lo que parece. Algunos ejemplos son:

- El uso de *tags* o etiquetas para filtrar las plantillas de entrenamiento
- Todo el formulario a la hora de registrar una sesión de entrenamiento y a la hora de crear una plantilla de entrenamiento.
- Encontrar y aprender a utilizar una librería con gráficas para "Dart".
- Permitir que un usuario pueda crear ejercicios de forma que sean únicos para él y que un administrador pueda hacerlo de forma global.
- La gestión de imágenes con "Cloudinary" y videos de "Youtube".
- Gestionar la eliminación dinámica y edición de plantillas de entrenamiento.

La magnitud del proyecto ha excedido las expectativas. Para completar todos los requisitos, casos de uso y objetivos se ha hecho un esfuerzo enorme, resultando en un gran número de pantallas, lógica en el lado del servidor extensa y un modelado de la base de datos modelo muy complejo con un gran número de tablas. Se ha intentado ser perfeccionista para cada componente y apartado lo que ha causado una inversión de tiempo más grande de lo esperada.

7.3 Posibles ampliaciones

La naturaleza de esta aplicación hace que siempre haya cosas por ampliar o mejorar. Un proyecto como este podría ser fácilmente relegado a un equipo de desarrollo en vez de a un solo individuo debido a su magnitud. Una de las primeras cosas que se podrían ampliar es la posibilidad de añadir un apartado para gestionar las dietas, idea que quería desarrollarse desde el principio, pero acabó en un plano secundario al comprender la magnitud real del proyecto. Este apartado podría ayudar a los usuarios a gestionar y compartir dietas con información de los macronutrientes y utilizar alguna *API* que permita comprobar los valores nutricionales o calorías de cada alimento.

Otra posible ampliación podría ser una mayor interacción entre usuarios. Un usuario podría tener un mural donde se puedan compartir los entrenos realizados y puedan ser vistos por otros usuarios. Se añadiría lógica para facilitar esta interacción como que los usuarios puedan seguir otros perfiles o que un usuario pueda poner su perfil público o privado. Este cambio transformaría la aplicación de una herramienta a una red social, por lo que sería una ampliación muy costosa pero posiblemente muy cómoda para los usuarios.

Se podrían integrar opciones de inicio de sesión a través de plataformas populares como "Google" puede aumentar significativamente la accesibilidad y la comodidad, facilitando a los usuarios el proceso de registro e ingreso, lo que podría mejorar la retención de usuarios.

La localización de la aplicación en varios idiomas según la región geográfica es otra ampliación valiosa que puede abrir mercados internacionales y aumentar la base de usuarios.

Estas son solo algunas de las posibles mejoras que se pueden implementar en futuras versiones de la aplicación, cada una de las cuales contribuiría a su evolución y comodidad para los usuarios a la hora de entrenar.

Referencias

[1] Organización mundial de la salud, Actividad física. URL: https://www.who.int/es/news-room/fact-sheets/detail/physical-activity

[2] Keelo. Keelo - Strength High Intensity HIIT Workouts. URL: https://keelo.com.

[3] TrainingPeaks. TrainingPeaks | Streamline Your Training. URL: https://www.trainingpeaks.com.

[4] Virtuagym. Virtuagym | El software de fitness Nº 1 para empresas y consumidores. URL: https://virtuagym.com/es/.

[5] Trifecta Nutrition. Trifecta | The #1 Fitness Meal Delivery Service | As Seen On Netflix. URL: https://www.trifectanutrition.com.

[6] SimpleDesign. SIMPLE DESIGN - Millions of Users" Choice. URL: https://simpledesign.ltd.

[7] Node.js Foundation. Node.js. URL: https://nodejs.org.

[8] Orsini, Lauren (2013). What You Need To Know About Node.js. URL: https://readwrite.com/what-you-need-to-know-about-nodejs/.

[9] TypeScript. TypeScript: JavaScript With Syntax For Types. URL: https://www.typescriptlang.org/.

[10] JavaScript | MDN. URL: https://developer.mozilla.org/es/docs/Web/JavaScript.

[11] npm, Inc. npm. URL: https://www.npmjs.com.

[12] Soper, Taylor (2020). Microsoft-owned GitHub to acquire JavaScript package manager Npm. URL: https://www.geekwire.com/2020/microsoft-owned-github-acquire-javascript-package-manager-npm/.

[13] Prisma. Prisma ORM. URL: https://www.prisma.io.

[14] Flutter - Beautiful native apps in record time. URL: https://flutter.dev.

[15] Dart - A Language for Professional Web Apps. URL: https://dart.dev.

[16] Microsoft. Visual Studio Code. URL: https://code.visualstudio.com.

[17] Chacon, Scott. Pro-Git. URL: https://git-scm.com.

[18] GitHub, Inc. GitHub. URL: https://github.com.

[19] Oracle. MySQL. URL: https://www.mysql.com.

[20] Becker, Ansgar. HeidiSQL. URL: https://www.heidisql.com.

[21] Cloudinary Ltd. Cloudinary. URL: https://cloudinary.com.

[22] JSON Web Tokens - jwt.io. URL: https://jwt.io.

[23] Fielding, Roy Thomas (2000). Architectural Styles and the Design of Network-based Software Architectures. URL: https://ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation.pdf

[24] JEST. “Jest · Delightful JavaScript Testing” URL: https://jestjs.io/es-ES/

[25] GhostTogether. How To Run Existing Flutter Project In VSCode. URL: https://semicolon.dev/flutter/how-to-run-existing-flutter-app-in-vscode

Apéndice A

Manual de Instalación

Requisitos	Comentarios
"MySQL"	
"HeidiSQL", "MySQL Workbench", ...	Puede facilitar la edición de datos.
"Visual Studio Code", "InteliJ", "Eclipse", ...	Cualquier editor de texto es válido, pero se recomienda "Visual Studio Code", se explicará la instalación suponiendo que "Visual Studio Code" es el editor de texto.
"Node.js"	Requisito esencial para ejecutar el entorno de servidor y el *framework* "Express". Obtener la versión más reciente.
"npm"	Gestor de paquetes necesario para instalar y administrar las librerías y dependencias de Node.js. Se recomienda tener como variable de entorno para facilitar su uso con "Visual Studio Code"
"Flutter"	Framework de UI utilizado para el desarrollo de interfaces de usuario en aplicaciones móviles y web. Obtener la versión más reciente. Se recomienda tener como variable de entorno para facilitar su uso con "Visual Studio Code"

"Git"	Se necesita tener instalado "Git", se recomienda tener como variable de entorno para facilitar su uso con "Visual Studio Code"
Cuenta de "Cloundinary"	Se necesita una cuenta de "Cloudinary" y obtener el "API Key", "API secret" y "Cloud name"
Cuenta de correo electrónico	Se necesita una cuenta de correo electrónico con una "App Password"

Back-end

1. Abrir "Visual Studio Code".
2. Clonar repositorio de github. Esto se puede hace de muchas formas y con muchas herramientas, de forma rápida con "Visual Studio Code" se pueden realizar los siguientes pasos:
   a. Abrir una carpeta vacía pulsando "Ctrl+K, Ctrl+O" o en el apartado "Open Folder".
   b. En el apartado "Terminal" que se encuentra en la barra superior pulsar en "New Terminal" o pulsar "Ctrl+shift+ñ"
   c. En el terminal recién abierto escribir el siguiente comando para clonar el repositorio del *back-end* del proyecto:

      git clone https://github.com/cberdejo/FitMatch_backend

3. Dentro de la carpeta generada "FitMatch_backend" obtener el *Script* que genera la estructura de la base de datos llamado "fitmatch.sql" y ejecutarlo en una conexión.
4. Hay que asegurarse que el terminal está ubicado dentro de la nueva carpeta generada. Se ejecuta el comando para acceder a la carpeta:

   cd .\FitMatch_backend

5. Dentro de la carpeta "FitMarch_backend" encontraremos un archivo llamado "package.json". Ahí estarán las dependencias del proyecto, se utilizará "npm" para instalarlas. Se escribe en el terminal el siguiente comando:

   npm install

   Si se ha realizado correctamente, debería haberse creado la carpeta "node_modules"

6. Dentro de la carpeta generada “FitMatch_backend” habrá que crear un archivo llamado “.env” y escribir las siguientes variables de entorno:

Variable de entorno	Instrucciones
PORT	Hace referencia al puerto en el que se abre el *back-end* para que sea accesible. Se recomienda usar el 3000
JWT_SECRET	Es una contraseña utilizada para cifrar los *tokens* “JWT”. Se recomienda una contraseña fuerte y compacta, se puede utilizar un generador de contraseñas online
DATABASE_URL	Esta *URL* permite conectarse con la base de datos debe tener la siguiente estructura: **"mysql://usuario:contraseña@localhost:puerto/fitmatch"** El usuario y contraseña son de la conexión para acceder a la base de datos. Se escribe “localhost” si se tiene la base de datos en local junto con su puerto, que suele asignarse 3306.
HASH_SECRET	Una contraseña utilizada para cifrar los datos con “SHA-256”. Se recomienda una contraseña fuerte y compacta, se puede utilizar un generador de contraseñas online
EMAIL	Se debe poner un correo que es el que se encarga de enviar correos electrónicos a los usuarios.
EMAIL_PASSWORD	Se debe obtener la “App password” del correo. Suele encontrarse en el apartado referente a la seguridad en un correo.
FRONTEND_URL	Se debe indicar la URL del *front-end*, lo más probable es que si está en local sea “http://localhost:4200/”
MINUTOS_EXPIRACION_TOKEN_VERIFICACION	Esta variable se encarga de definir cuánto dura la sesión de un usuario, se recomienda una hora de sesión, que debe ponerse en minutos, es decir 60.

CLOUDINARY_API_KEY	Se obtiene al iniciar sesión en “Cloudinary”
CLOUDINARY_CLOUD_NAME	Se obtiene al iniciar sesión en “Cloudinary”
CLOUDINARY_API_SECRET	Se obtiene al iniciar sesión en “Cloudinary”
MINUTOS_BLOQUEO	Al bloquearse una cuenta por repetidos intentos de acceso fallido, indicar cuántos minutos se bloquea la cuenta. Se recomienda poner entre cinco minutos y una hora.
INTENTOS BLOQUEO	El número de intentos de acceso fallidos que debe tener un usuario para ser bloqueado. Se recomiendan entre cinco o diez.
MINUTOS_CONTAR_BLOQUEO	Los minutos en los que lleva la cuenta de los intentos de acceso fallido, si pasan estos minutos, el contador empieza de nuevo a contar. Se recomienda menos de diez minutos.

7. Por último, para ejecutarlo escribir en el terminal:

```
npm run dev
```

8. En caso de querer compilar las pruebas utilizar el comando:

```
npm test
```

Front-end

1. Abrir “Visual Studio Code”.
2. Clonar repositorio de github. Esto se puede hace de muchas formas y con muchas herramientas, de forma rápida con “Visual Studio Code” se pueden realizar los siguientes pasos:
   a. Abrir una carpeta vacía pulsando “Ctrl+K, Ctrl+O” o en el apartado “Open Folder”.
   b. En el apartado “Terminal” que se encuentra en la barra superior pulsar en “New Terminal” o pulsar “Ctrl+shift+ñ”
   c. En el terminal recién abierto escribir el siguiente comando para clonar el repositorio del *back-end* del proyecto:

```
git clone https://github.com/cberdejo/FitMatch_frontend.git
```

3. Hay que asegurarse que el terminal está dentro de la nueva carpeta generada. Se ejecuta el comando para acceder a la carpeta.

```
cd .\FitMatch_frontend\
```

4. Se deben instalar las dependencias con el siguiente comando:

```
flutter pub get
```

5. Por último, se puede ejecutar el proyecto escribiendo:

```
flutter run
```

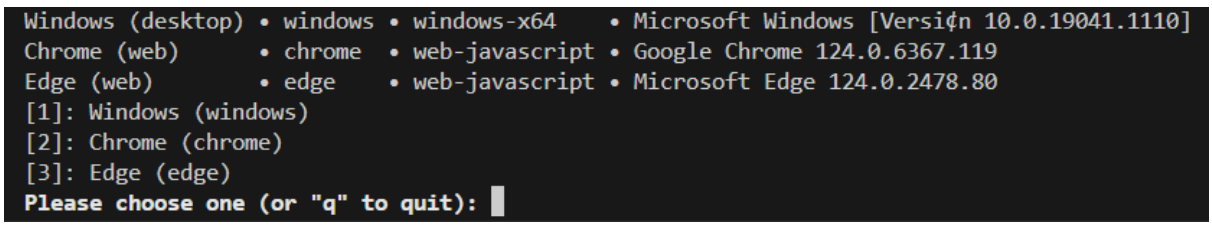

Figura 32 Respuesta del terminal tras hacer "flutter run"

6. Se debe escoger el navegador donde se despliega, escribiendo el número en el terminal.

Para la vista en un emulador de móvil, debe hacerse lo siguiente [25]:

1. **Instalación de extensiones**: Primero, asegúrate de tener las extensiones de "Dart" y "Flutter" instaladas en "VSCode". Puedes hacerlo desde la pestaña de extensiones de "VSCode".
2. **Configuración del emulador en Android Studio**: Antes de usar el emulador en VSCode, necesitas configurar un dispositivo virtual en Android Studio usando el Android Virtual Device (AVD) Manager. Esto creará un emulador que puedes seleccionar posteriormente en VSCode.

3. **Lanzamiento del emulador desde VSCode**:
    a. Abre la paleta de comandos con Ctrl+Shift+P (o Cmd+Shift+P en Mac).
    b. Escribe "Flutter: Select Device" y selecciona el emulador que configuraste en Android Studio.
    c. Para iniciar el emulador, puedes buscar "Flutter: Launch Emulator" en la misma paleta de comandos.
4. **Ejecución del proyecto Flutter**: Con el emulador en funcionamiento, puedes ejecutar tu proyecto Flutter utilizando el comando mencionado previamente, es decir:

```
flutter run
```

Apéndice B

Manual de Usuario

Para simplificar la presentación y facilitar la comprensión del diseño y la interfaz, se optará por explicar las pantallas de la versión móvil ya que el resto de las versiones (tableta digital y web) son similares (**Figura 33**). En el caso de que alguna de las interfaces sea muy distinta en alguna versión del proyecto se añadirá también en el manual. Esta decisión permite centrarse en detalles específicos y asegura una demostración visual coherente y directa, sin perder de vista la adaptabilidad y la experiencia del usuario en diferentes dispositivos.

Las interfaces se han diseñado con un enfoque minimalista y con una paleta de colores oscura y otra clara. Primero se mostrará la paleta de colores oscura con su explicación y luego se mostrarán las pantallas principales con paleta de color. Las pantallas y funcionalidades se pueden categorizar en dos roles distintos: el del cliente y el del administrador.

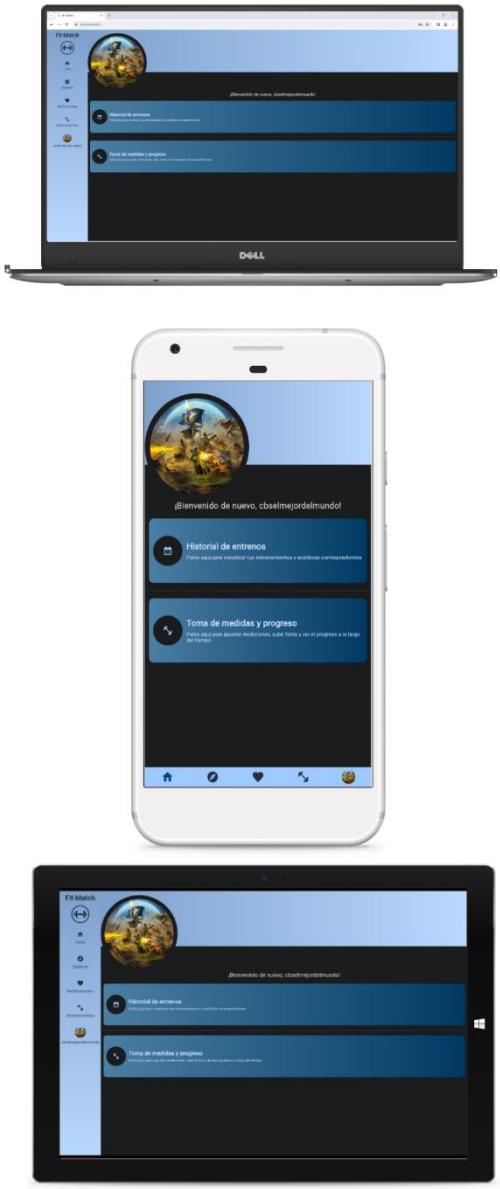

Figura 33 Aplicación en distintos dispositivos.

Acceso a la aplicación

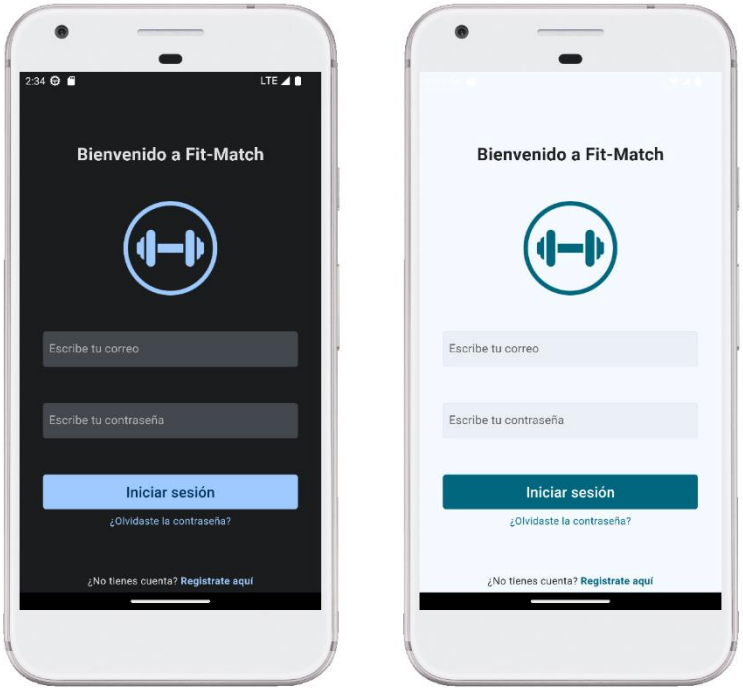

Figura 34 Pantalla de inicio de sesión, paleta de colores oscura y clara.

La **Figura 34** muestra la interfaz de inicio de sesión para la aplicación. Se ha optado utilizar un icono de una mancuerna ya que es simple, elegante y representativo de las funciones de "Fit-Match". La interfaz de inicio de sesión funciona como es de esperar: si un usuario tiene una cuenta, escribe sus credenciales y entra a la aplicación. Se pueden observar ambas paletas de colores en la **Figura 34**.

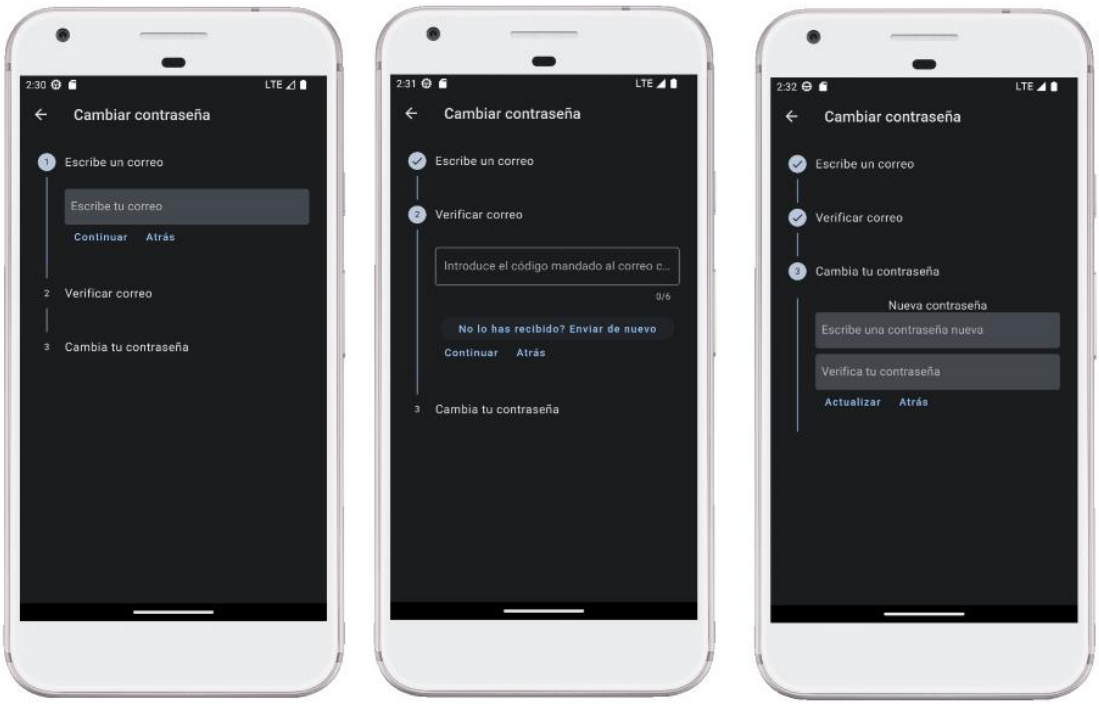

Figura 35 Interfaz destinada a recuperar la contraseña olvidada (paleta de colores oscura)

En caso de que un usuario olvide su contraseña, puede pulsar el botón "¿Olvidaste la contraseña?" bajo "Iniciar sesión**.** Al hacerlo, la aplicación lo redirige a una pantalla de tres pasos (véase la **Figura 35**). El primer paso requiere que el usuario ingrese su correo electrónico. En el segundo paso, se muestra un campo para introducir un código de seis dígitos, el cual no permite más dígitos y ofrece la opción de reenviar el código. Este código, una contraseña de un solo uso (OTP), verifica que el titular del correo sea quien intenta cambiar la contraseña. En el último paso, se presentan dos campos para ingresar y verificar una nueva contraseña; ambas deben coincidir para completar el cambio.

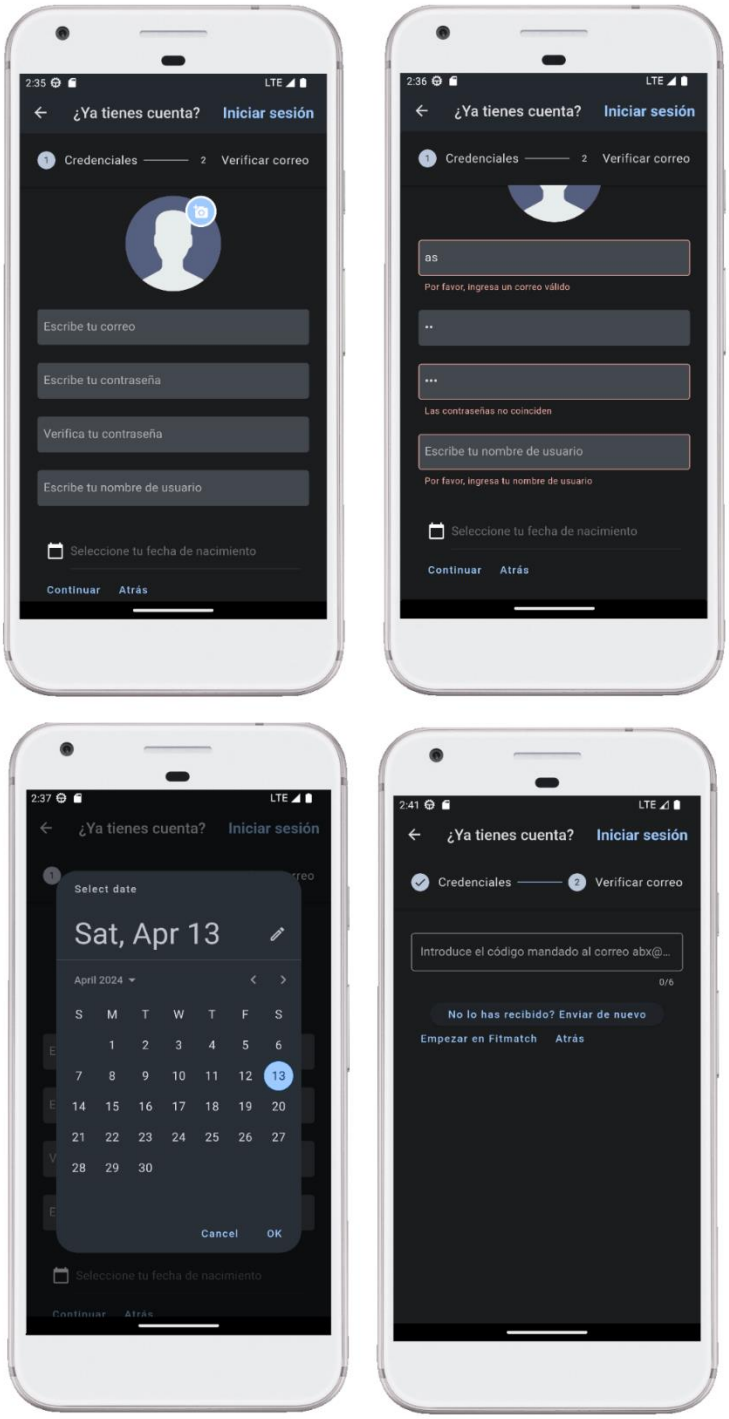

Figura 36 Pantalla de registro de usuario (paleta de colores oscura)

Por otro lado, si el usuario opta por seleccionar el botón “Regístrate Aquí” situado en la parte inferior de la interfaz de inicio de sesión (**Figura 34)**, la aplicación lo redirecciona a la pantalla destinada al registro de nuevos usuarios. Esta sección se estructura en dos etapas. La primera etapa se centra en la recolección de datos personales y credenciales del usuario; aquí tiene la posibilidad de subir una foto de perfil, además de ingresar su correo electrónico y contraseña, ambos esenciales para el acceso futuro a la cuenta.

Posteriormente, puede añadir un nombre de usuario y su fecha de nacimiento. Tal como muestra la **figura 36**, se implementan validaciones que confirman la validez del correo electrónico, verifican que no esté previamente registrado en la base de datos, aseguran que las contraseñas ingresadas coincidan y que el nombre de usuario no esté vacío. En la segunda etapa se envía una contraseña de un solo uso (OTP) que, si el usuario escribe correctamente, puede comenzar a utilizar la aplicación.

Figura 37 Captura del mensaje en correo con código de un solo uso.

En la **Figura 37** se muestra un ejemplo de un mensaje enviado a un usuario con un código de un solo uso. Este se enviará para recuperar la contraseña en caso de olvido y para crearse una cuenta.

Rol de cliente

Estas son las pantallas del rol de cliente divididas en secciones, una por cada sección principal en el menú de la aplicación: “Inicio”, “Descubrir”, “Notificaciones”, “Entrenamiento” y “Perfil”

Inicio

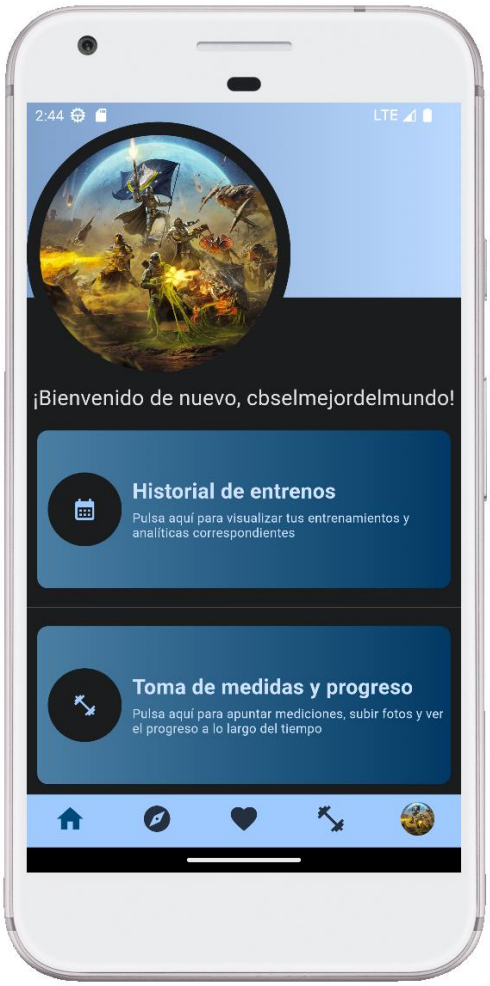

Figura 38 Interfaz inicial de la aplicación (paleta de colores oscura).

En la **Figura 38** se muestra la página inicial de la aplicación una vez el usuario de rol cliente ha iniciado sesión. Se puede apreciar un menú en forma de barra en la parte inferior que permite navegar entre las secciones de la aplicación. La que se muestra es la página “Inicio” y se muestra la foto de perfil en grande junto al nombre de usuario y varias opciones a subsecciones.

La sección “Inicio” está destinada a visualización de progreso en medidas corporales y sesiones de entrenamiento.

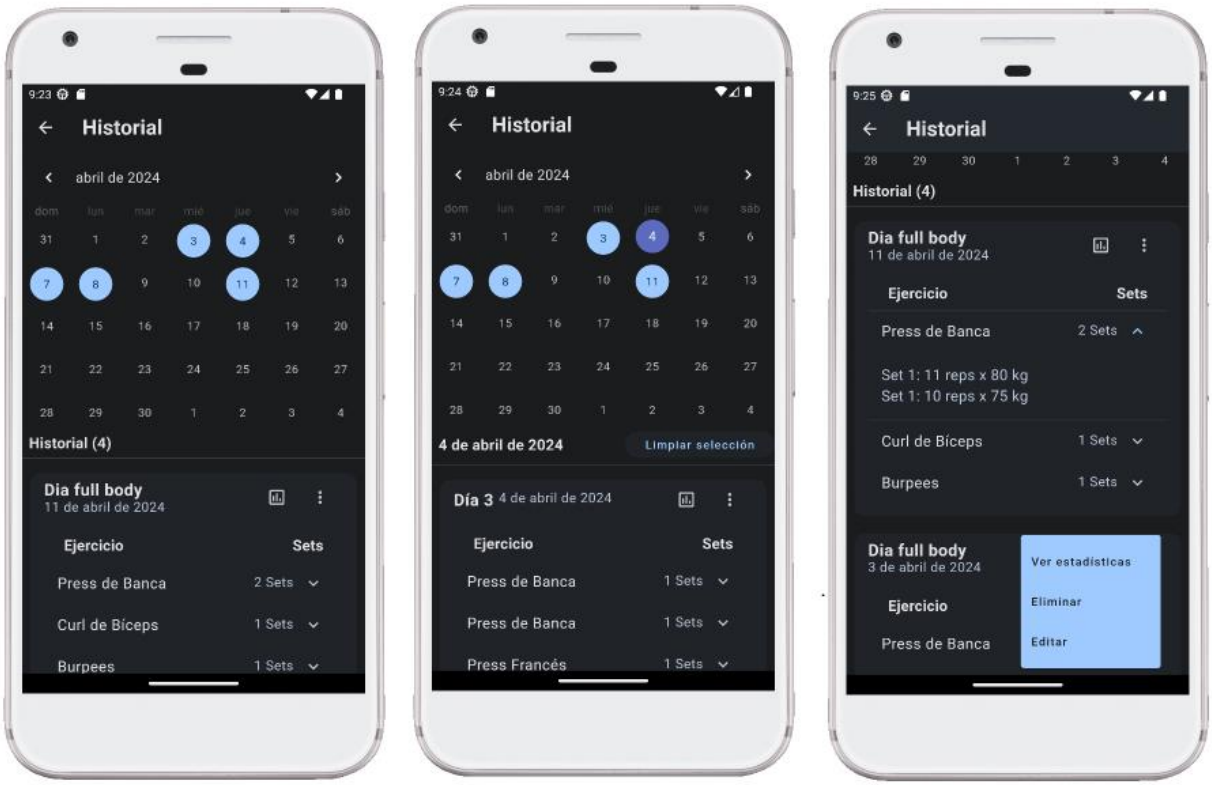

Figura 39 Interfaz de historial de sesiones de entrenamiento (paleta de colores oscura).

Desde la pantalla de inicio es posible navegar al historial donde se muestra un calendario que permite filtrar por día los registros a las sesiones de entrenamiento. Como se puede ver en la segunda captura de la **Figura 39**, en el que se ha seleccionado el 4 de abril. Esta pantalla permite editar y eliminar estos registros y visualizarlos. Además, permite navegar a una interfaz para ver estadísticas de la sesión de entrenamiento.

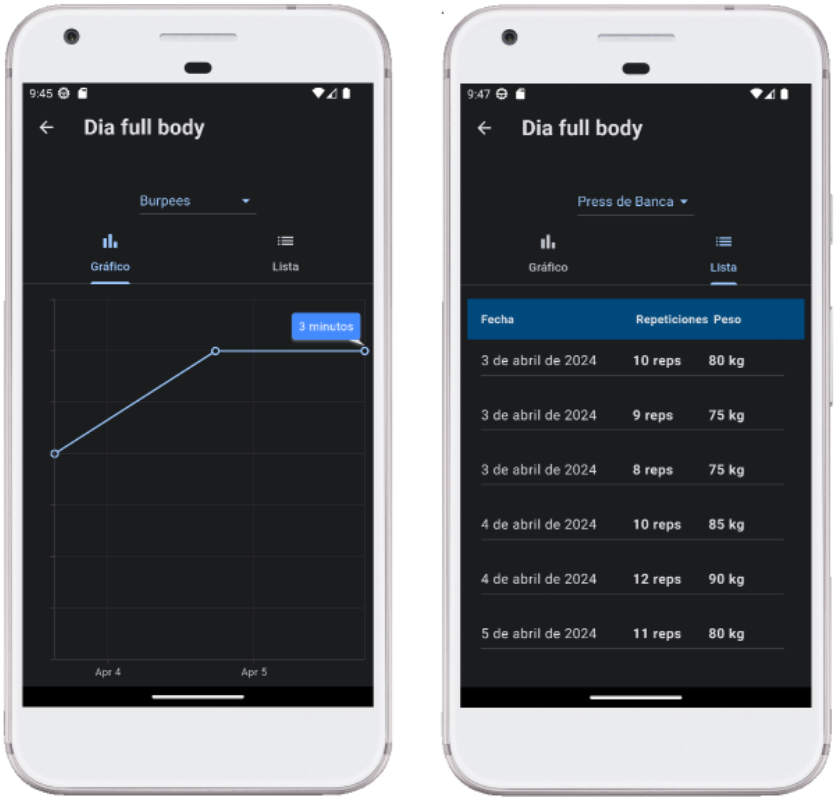

Figura 40 Interfaz de estadísticas de registro de progreso de un ejercicio (paleta de colores oscura).

En la pantalla de la **Figura 40** se puede ver el progreso de los registros para una sesión de entrenamiento. Es posible seleccionar el ejercicio o movimiento para el que quieres ver la progresión en el tiempo con un selector en la parte superior de la pantalla. Se puede ver una vista de con una gráfica o lista. La vista de gráfica al tocar con el dedo en cualquier punto de la gráfica (o poner el ratón sobre este en la versión web) muestra información del registro en la sesión. Además, permite hacer zum para gráficas con muchos datos.

Las unidades serán las correctas según la preferencia del sistema de medida del usuario (métrico o imperial) y el tipo de registro rango de tiempo, tiempo, “amrap”, número de repeticiones, rango de repeticiones y repeticiones máximas).

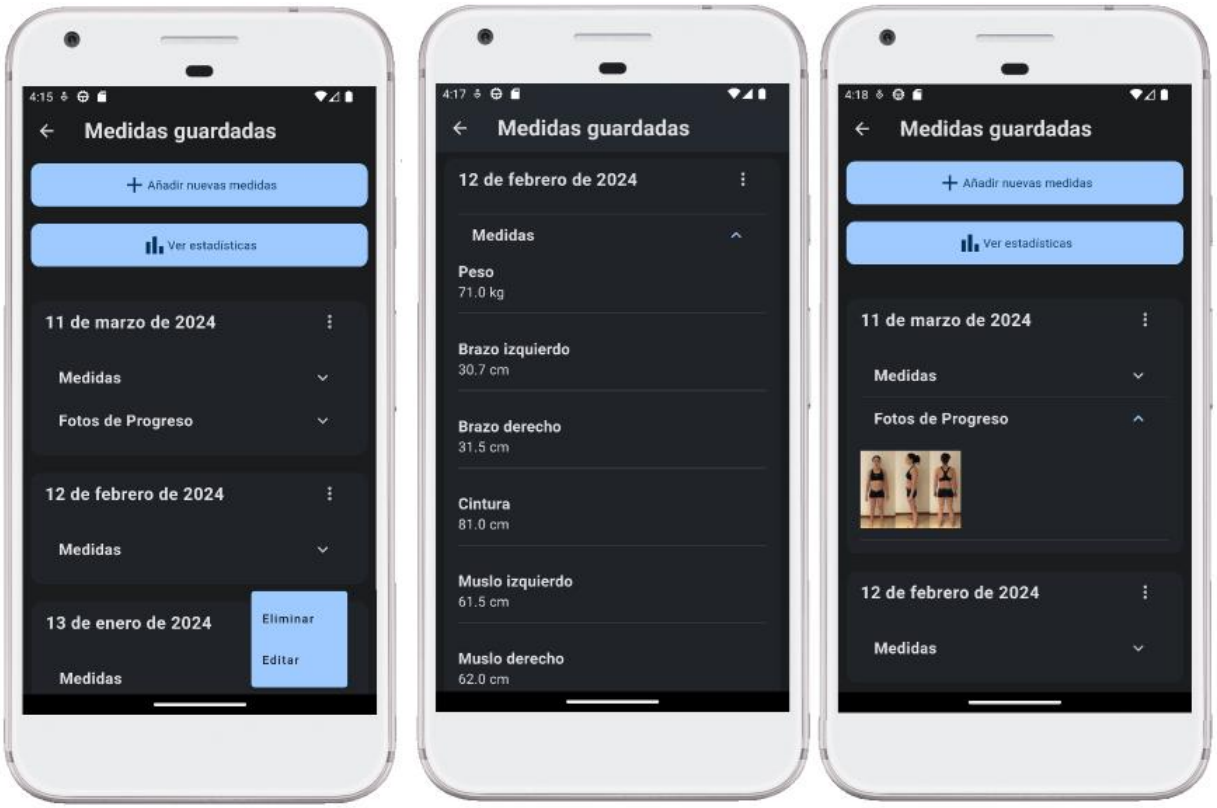

Figura 41 Interfaz destinada mostrar un historial de las medidas corporales (paleta de colores oscura).

En la **Figura 41** se muestra el historial de medidas corporales, esta interfaz da la posiblidad de visualizar las medidas y las fotos de progreso sacadas mediante desplegables. Además, permite al usuario editar mediciones existentes para una fecha en específico o eliminar la toma de medidas. En la parte superior de la interfaz hay dos botones. El primer botón navega a la pantalla para añadir nuevas medidas y el segundo muestro estadísticas de las medidas.

Figura 42 Pantalla para añadir nuevas mediciones (paleta de colores oscura).

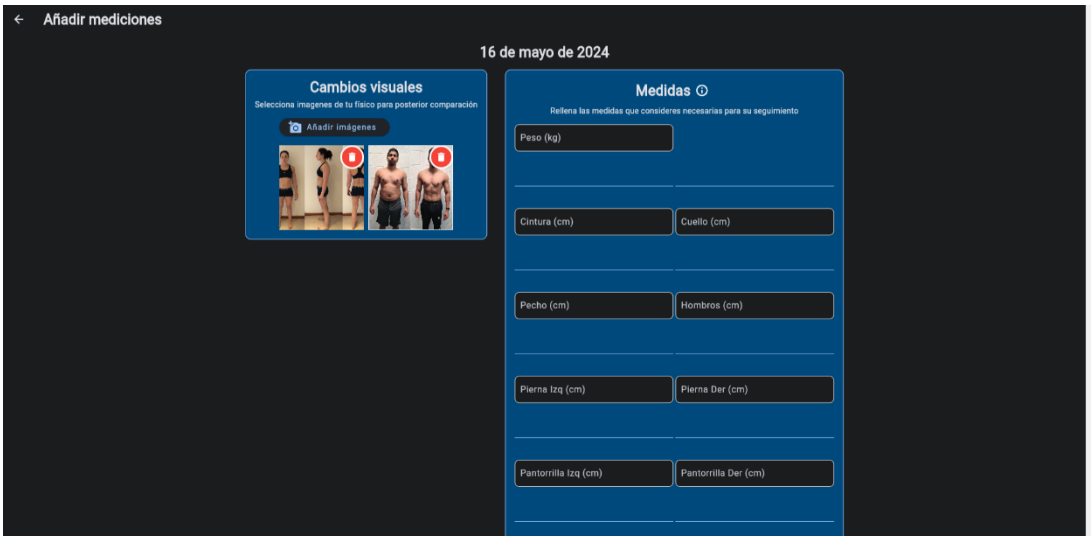

Figura 43 Versión web para la pantalla para añadir nuevas mediciones (paleta de colores oscura).

La **Figura 42 y 43** representan un formulario que permite para una fecha añadir fotos que representan el progreso físico de un usuario y medidas corporales. Las medidas incluyen brazos, hombros, cuello, pecho, cintura, piernas y pantorrillas, así como peso y otras métricas relevantes. Para poder guardar las mediciones debe existir al menos una medida o foto añadida.

Figura 44 Ventana emergente con consejos (paleta de colores oscura).

En caso de tener alguna duda con como tomar alguna medición de un grupo muscular, se puede pulsar al icono de información en el título de Medidas y emerge una ventana con consejos de como tomar las medidas de forma correcta.

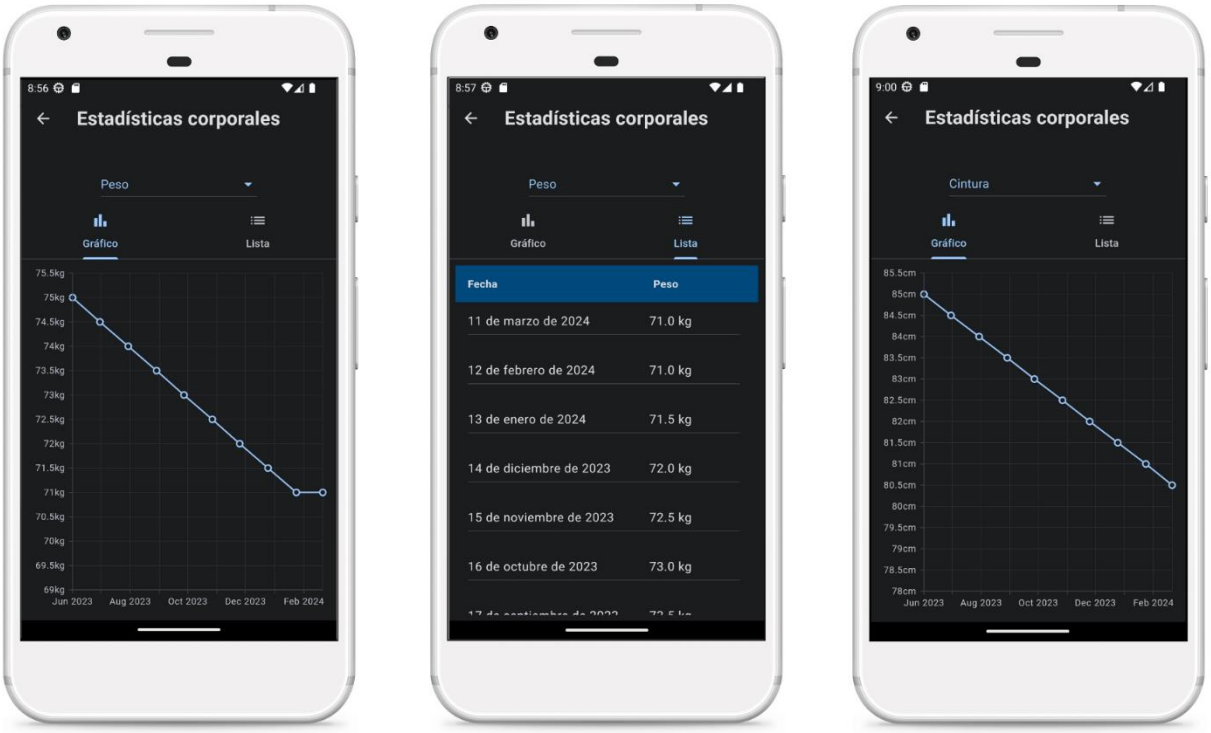

Figura 45 Pantalla de estadísticas de medidas corporales (paleta de colores oscura).

De forma análoga a la **Figura 40**, se muestran estadísticas esta vez para medidas corporales. El peso y las medidas mostrarán las unidades correspondientes en base a la preferencia del usuario y hay un selector que permite escoger estadística corporal en la que centrarse. Es importante visualizar de nuevo la aparición de la vista de lista y gráfica.

Las etiquetas de los ejes de las gráficas, tanto en las estadísticas de los registros de entrenamiento como en esta gráfica son dinámicas. Esto puede verse con las etiquetas de las fechas, que al estar más repartidas muestran una etiqueta por mes. Al hacer Zum, podríamos ver como las etiquetas de la fecha cambiarían a un sistema más preciso.

Descubrir

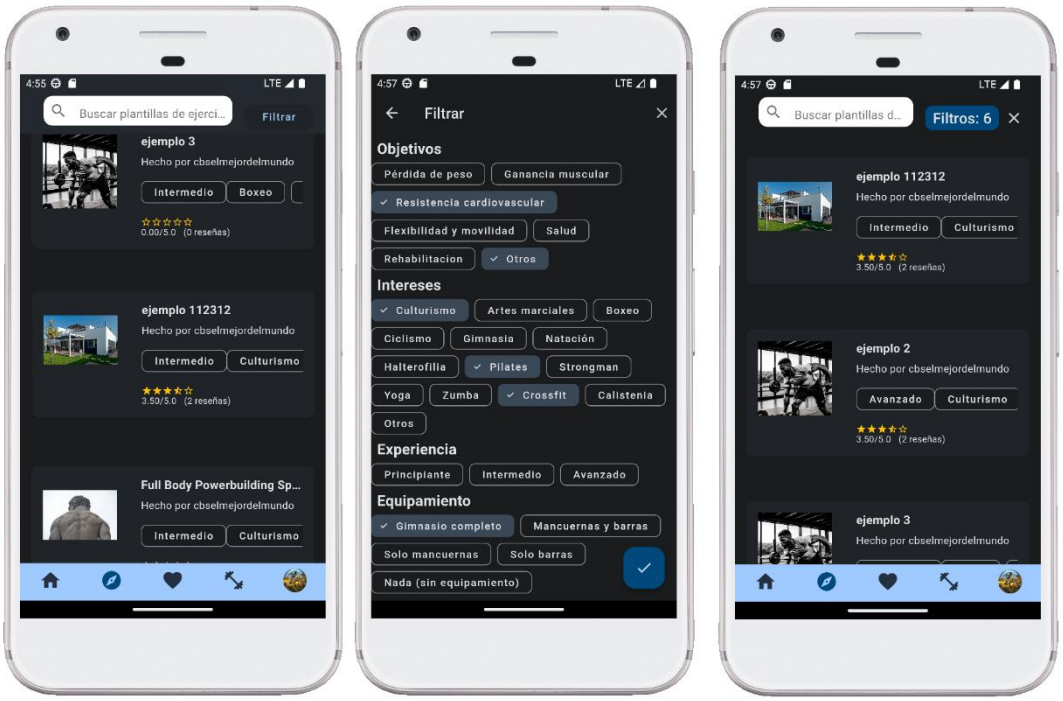

Figura 46 Interfaz para explorar plantillas de entrenamiento (paleta de colores oscura).

Las interfaces de la **Figura 46** facilitan la búsqueda de plantillas de entrenamientos que sean públicas. Se puede buscar por un título y añadiendo filtros que indiquen tus intereses en la búsqueda. A estos intereses se les llama "*tags*" o "etiquetas". Las etiquetas que permiten para el filtrado son nivel de experiencia, intereses en disciplinas deportivas, objetivos deportivos y equipamiento disponible.

Las tarjetas de la plantilla de entrenamiento tienen una foto o miniatura, un título, el autor y los *tags* o etiquetas asociadas a la plantilla de entrenamiento. Aunque en la imagen no se aprecia, es posible deslizar horizontalmente para visualizar todos los *tags*. Por último, se muestran la media de reseñas con un sistema de estrellas del 0 al 5 y el número de reseñas.

Al pulsar en una tarjeta nos redirige a una pantalla que nos muestra más información y nos permite interactuar con la plantilla de entrenamiento (**Figura 47 y 48**).

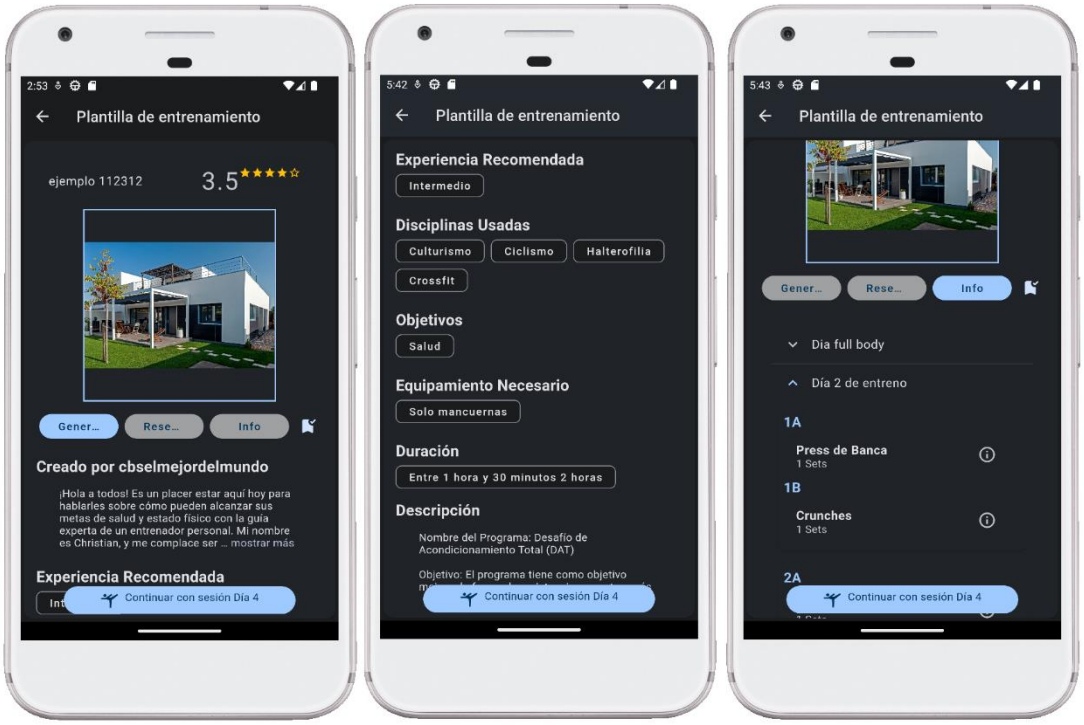

Figura 47 Tarjeta de plantilla de entrenamiento apartados "General" e Información" (paleta de colores oscura).

En la **Figura 47** se muestra información de la plantilla de entrenamiento. Arriba de la interfaz se puede ver el nombre de la plantilla de entrenamiento y su calificación. Justo debajo se aprecia la foto o miniatura de la plantilla de entrenamiento. Se puede ver que cada plantilla de entrenamiento tiene 3 apartados principales:

- General: Hay información del creador, una descripción y las etiquetas seleccionadas.
- Reseñas: Retroalimentación de los usuarios (véase la **Figura 48 y 49**).
- Info: Salen listadas las sesiones de entrenamiento de la plantilla con la opción de desplegarlas para ver los ejercicios de cada sesión de entrenamiento. Además, permite ver información de cada ejercicio pulsando el ícono de información ubicado a la derecha del nombre del ejercicio (véase la **Figura 54**).

Justo a la derecha de los botones para elegir entre apartados de la plantilla hay un botón en forma de marcador que proporciona la habilidad de guardar la plantilla de entrenamiento para consultarla más tarde (véase la **Figura 51**).

Por último, un botón flotante debajo de la interfaz que por defecto pondrá "Comenzar" Y el nombre de la primera sesión de entrenamiento. En caso de tener una sesión de entrenamiento a mitad sin haberla terminado el botón mostrará "Continuar" y el nombre de la sesión que tocaría. Según se vayan terminando las sesiones de entrenamiento, el

botón irá indicando a la siguiente sesión de entrenamiento descrita en la plantilla. Al darle al botón la aplicación redirige a una vista que permite introducir apuntes sobre el rendimiento de la sesión de entrenamiento.

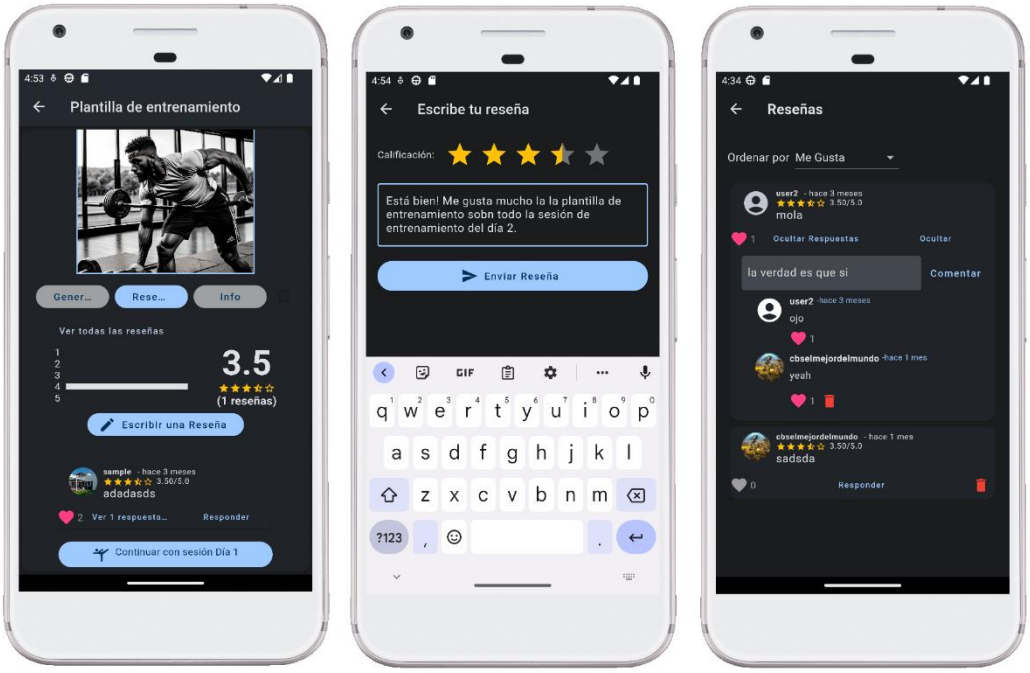

Figura 48 Pantallas de reseñas (paleta de colores oscura).

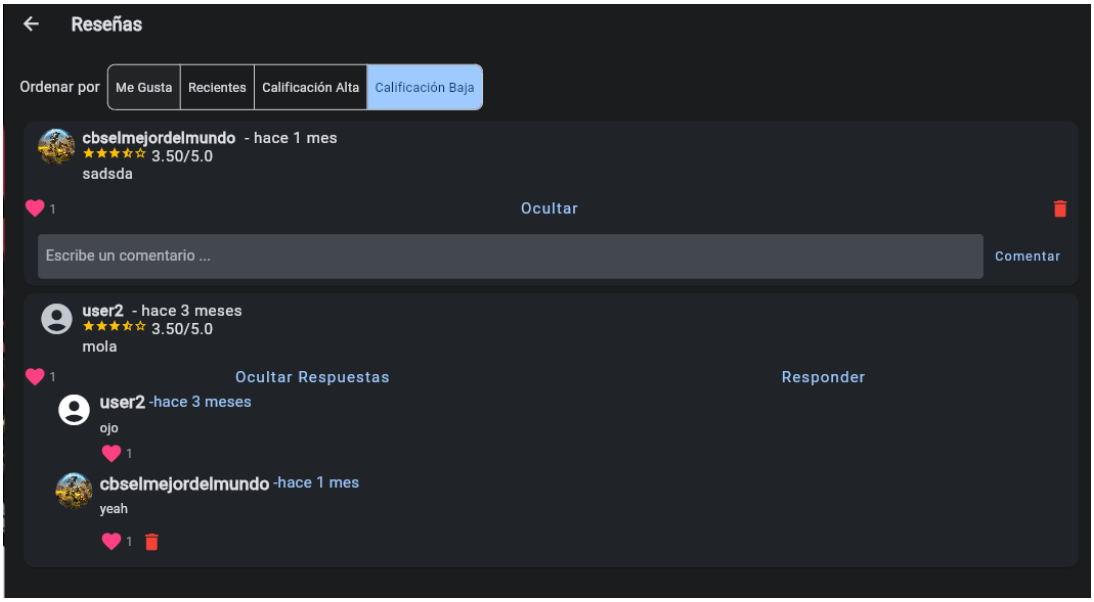

Figura 49 Interfaz web para pantallas de reseñas (paleta de colores oscura).

Este apartado permite ver reseñas de la plantilla de entrenamiento. Muestra un resumen general con un número de reseñas, puntuación y debajo comentarios. Es posible darle al botón de “Escribir una Reseña” que redirige a una interfaz que permite seleccionar una calificación deslizando el dedo en las estrellas y escribir un comentario.

Al pulsar “Ver todas las reseñas”, se pueden ver todas las reseñas. Cada reseña está compuesta por el nombre de usuario junto con su foto de perfil, hace cuando escribió la reseña y su calificación. Se puede dar “Me gusta” pulsando el botón con forma de corazón, responder a la reseña con un comentario y ver otras respuestas a la reseña. Por último, se puede filtrar por “Me gusta”, “Más recientes”, “Calificación más alta” y “Calificación más baja”.

Notificación

Figura 50 Pantalla de notificaciones (paleta de colores oscura).

En esta sección del menú hay una sola pantalla que muestra las notificaciones. Las notificaciones no leídas se resaltan y además indican las fechas. En la parte de implementación se explica todo acerca de los tipos de notificaciones posibles.

Entrenamiento

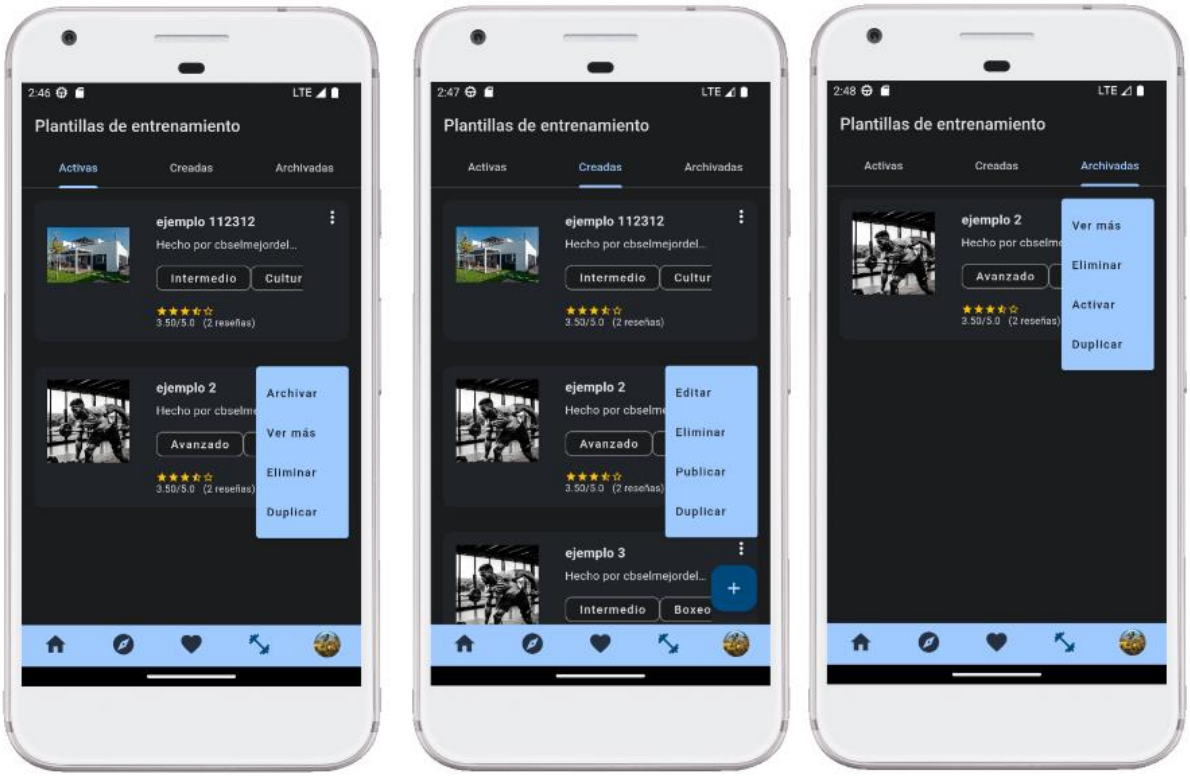

Figura 51 Pantallas de gestión de plantillas de entrenamiento (paleta de colores oscura).

La página principal en esta sección está dividida entre plantillas activas, creadas y archivadas.

Las plantillas activas son aquellas que se han guardado para ver más tarde de la sección "Descubrir" de la aplicación. Al estar activas, se entiende que son plantillas de entrenamiento cuyas sesiones de entrenamiento se están realizando. Es posible interactuar con ellas yendo a la vista de la plantilla de entrenamiento (**Figura 47**) al pulsar en ella o en "Ver más". Por lo que es una forma rápida de acceder a ella y empezar las sesiones de entreno que contienen. Además, se puede eliminar las plantillas si no se desea consultar más o si no se desea tener la plantilla como activa. Por último, se puede archivar una plantilla, por si se desea tener guardada para consultarla, pero no activa.

Las plantillas creadas son todas las plantillas del usuario que ha creado, puede entrar a la plantilla para hacer los ejercicios y ponerla como activa. Además, podrá eliminarla, editarla y publicarla para ser encontrada por los demás usuarios o hacerla privada si es

pública y desea que no siga siéndolo. Abajo a la derecha aparecerá un botón flotante que permite crear nuevas plantillas de entrenamiento.

Las plantillas archivadas pueden reactivarse o eliminarse de archivados, además de visualizarse. Todas las plantillas en estos paneles tienen la opción de duplicarse. Esto es útil por si queremos tener variaciones muy parecidas de plantillas de entrenamientos, pero con sutiles diferencias o para tener distintas versiones de un programa. También sería útil en caso de necesitar editar una plantilla ajena, se podría crear una copia a partir de una plantilla de otro usuario y editarla con necesidades particulares.

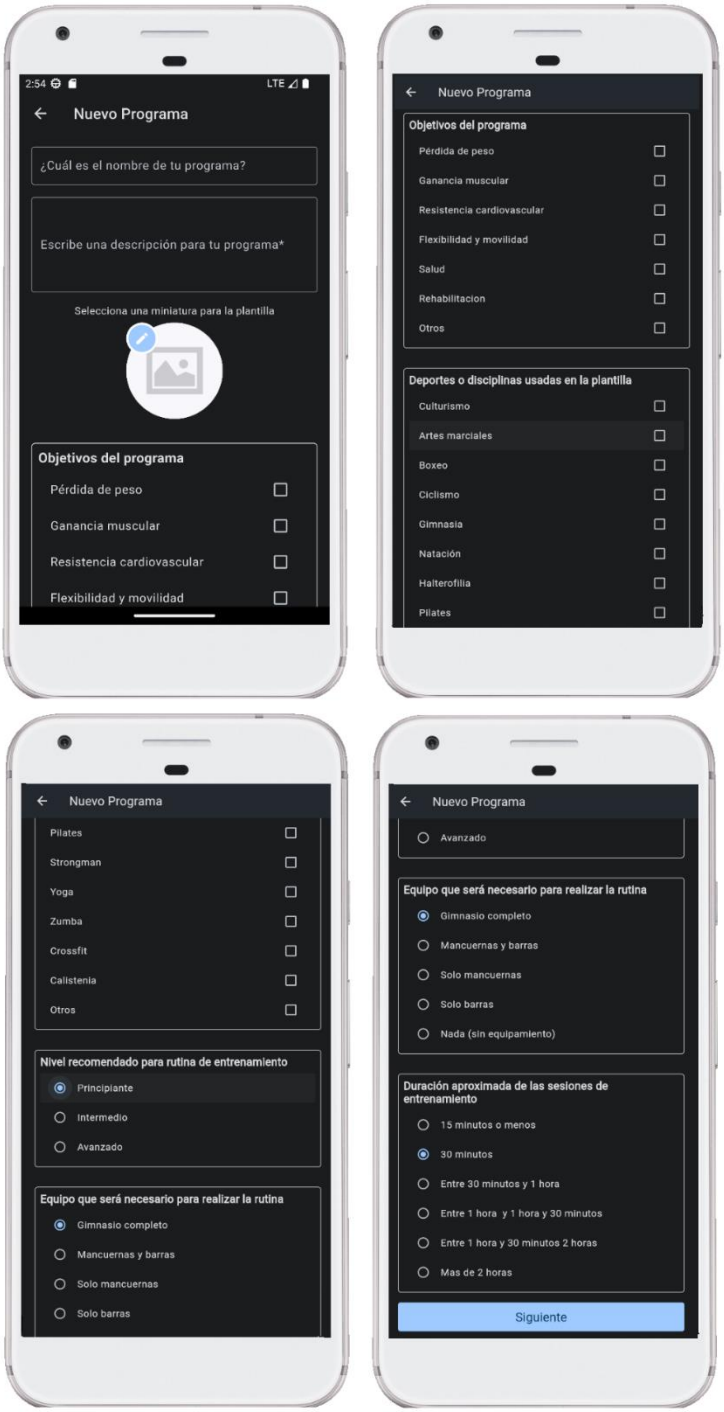

Figura 52 Pantalla de creación o edición de plantilla de entrenamiento (paleta de colores oscura).

La **Figura 51** muestra la interfaz para crear o editar una plantilla de entrenamiento, se escogen *tags* o etiquetas para facilitar la búsqueda de otros usuarios, se le añade una descripción, una foto de una miniatura y finalmente si no hay campos vacíos se pasa al siguiente paso (**Figura 50**).

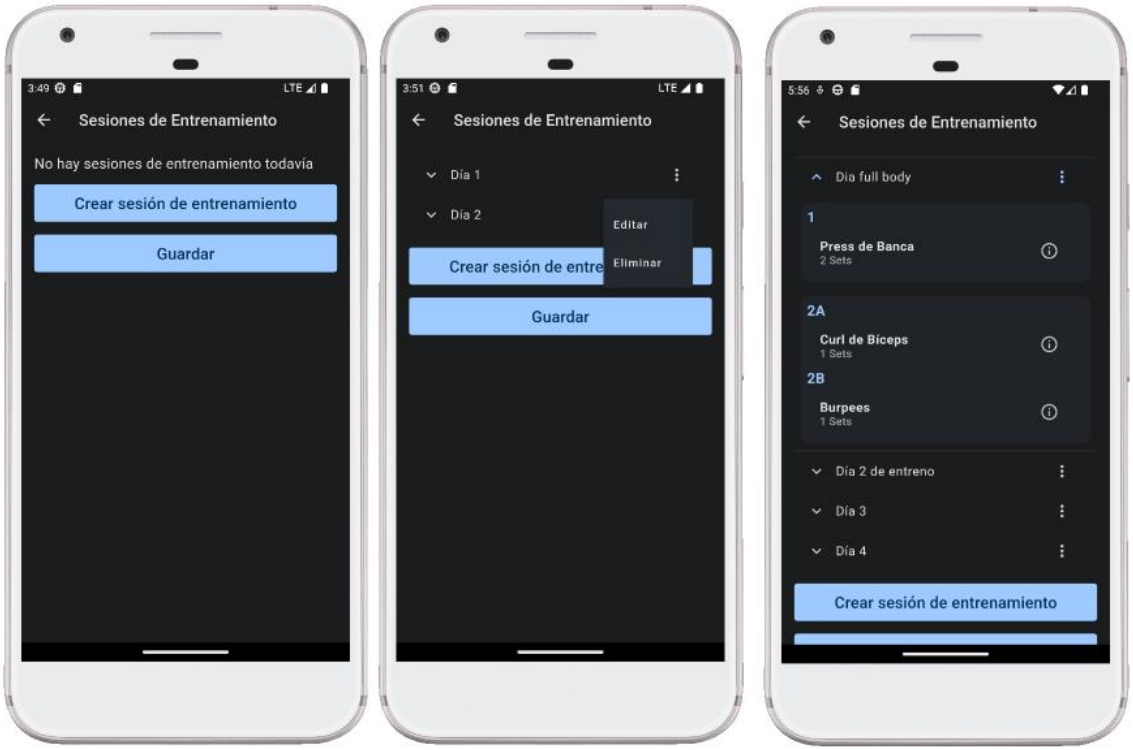

Figura 53 Pantalla de gestiones de sesión de entrenamiento (paleta de colores oscura).

Tras crear una plantilla de entrenamiento se pueden ir creando sesiones de entrenamientos. En cada sesión de entrenamiento se puede editar y eliminar. Además, cada sesión se puede desplegar y mostrar los detalles de la sesión de forma análoga a la ya vista en las figuras. Se puede ver el resultado en la **Figura 53.**

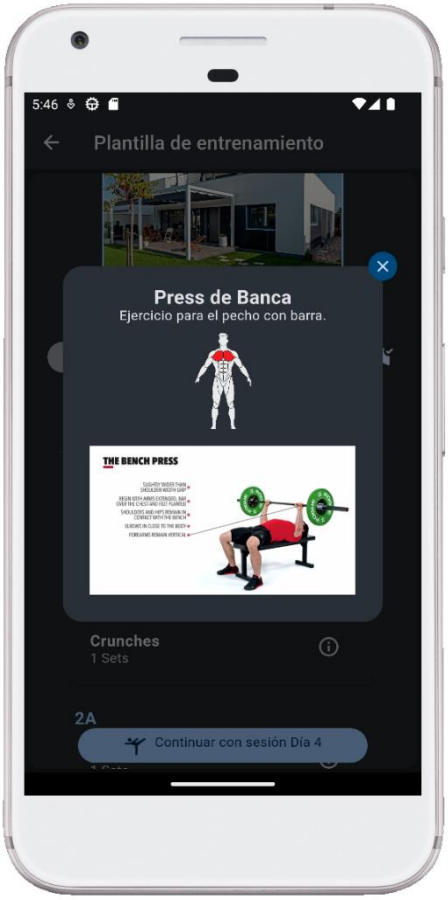

Figura 54 Información de un ejercicio (paleta de colores oscura).

Al darle al icono de información al lado del ejercicio nos saldrá una ventana emergente que muestre, el nombre, descripción de ejercicio, imagen de grupo muscular y un video (**Figura 54**).

Figura 55 Pantallas de creación o edición de sesión de entrenamiento (paleta de colores oscura).

Las pantallas de la **Figura 55** muestra la creación o edición de las sesiones de entrenamiento. El formulario de la pantalla da la opción de escoger un nombre, una descripción y añadir ejercicios (**Figura 55**). Los ejercicios se muestran en orden y pueden ir agrupados o en solitario. Al darle al icono de información saldrá la ventana de la **Figura 54** que dará más información del ejercicio escogido.

A cada ejercicio se le puede especificar un tipo de registro (“rango de repeticiones”, “objetivo de repeticiones”, “repeticiones máximas”, “amrap”, “tiempo” o “rango de tiempo”) y apuntar instrucciones según el tipo de registro. Se podrá elegir las series a realizar y especificar instrucciones para cada serie según el tipo de registro. Es posible también reordenar las agrupaciones de los ejercicios (última pantalla de la **Figura 55**).

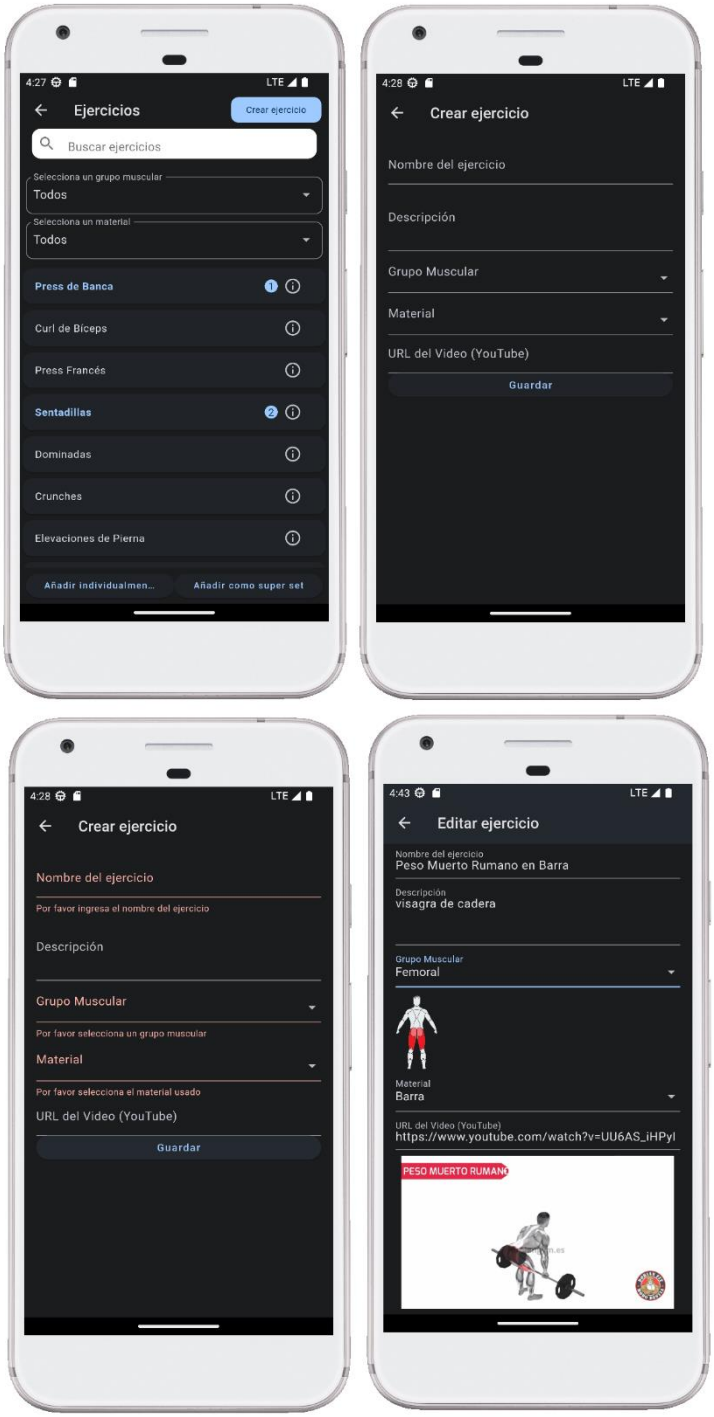

Figura 56 Pantallas para seleccionar y crear ejercicios (paleta de colores oscura).

La **Figura 56** se muestran cuatro pantallas dedicadas a la gestión de ejercicios. La primera en la esquina superior izquierda se accede desde la de crear o editar una sesión de entrenamiento al presionar la opción de añadir un ejercicio. Desde aquí se pueden añadir varios ejercicios y al seleccionar con un número el orden de estos. Al seleccionar un ejercicio la aplicación da la opción de añadirlo individualmente y al seleccionar varios da la opción de añadirlos como superserie o individualmente. Al presionar añadirlos como superserie, se añadirán agrupados, dando a entender que deben realizarse uno detrás de otro sin descanso, en otro caso se añadirán por separado. Se puede observar que el sistema facilita un filtrado por nombre, grupo muscular y material utilizado del ejercicio.

En caso de no tener un ejercicio específico, se puede crear uno, al que se le puede indicar un nombre, descripción, grupo muscular, material utilizado y una *URL* de un video de "Youtube". Los ejercicios que se crean con el rol de cliente solo serán visibles en las listas personales de cada usuario con rol de cliente, además podrán verlo los usuarios que utilicen la plantilla. Hay validadores para asegurarse que no haya campos importantes vacíos (la *URL* y la descripción son opcionales).

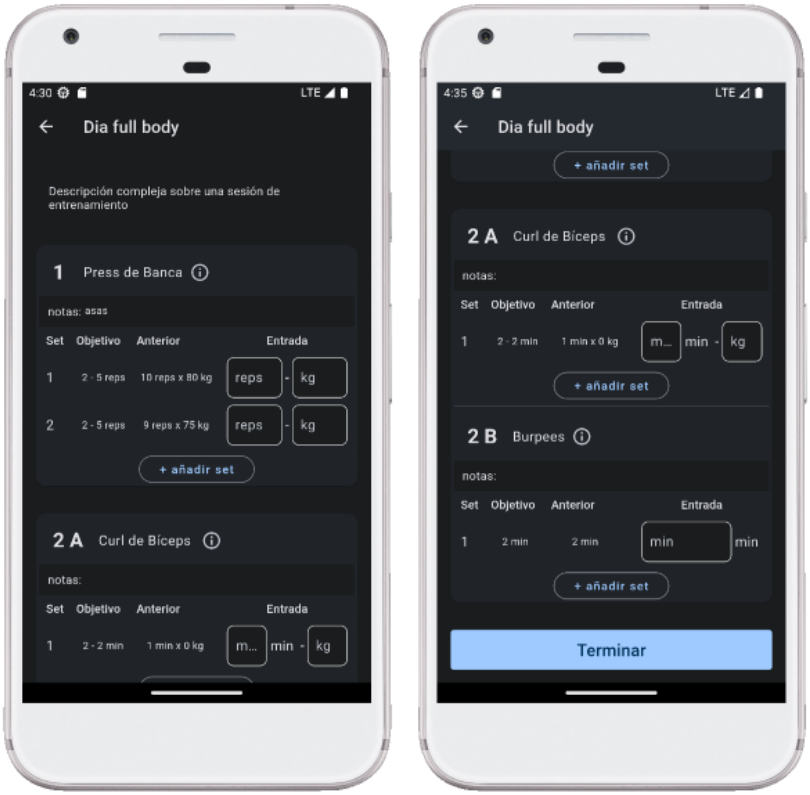

Figura 57 Pantalla para registrar una sesión de entrenamiento (paleta de colores oscura).

La pantalla de la **Figura 57** muestra cómo se inserta una sesión, tiene el título de la sesión y la descripción de la sesión de entrenamiento. Se pueden ver todas las agrupaciones

de los ejercicios y las series que se recomienda hacer, además la opción de hacer más series (o menos ya que no es obligatorio rellenar los campos). Para cada ejercicio de cada agrupación, se muestra el nombre del ejercicio con la opción de ver instrucciones (**Figura 54**), las notas apuntadas para el ejercicio y las instrucciones para cada serie. Además, se muestra el rendimiento de la sesión anterior, útil para un usuario que busque mejorar. Es posible salir de la pantalla sin terminar la sesión de entrenamiento, en caso de querer dejarla a mitad. Al darle al botón de finalizar se guardará su fecha de finalización. para luego poder consultarlo en el historial.

Dependiendo del tipo de registro habrá distintas formas de insertar el rendimiento y siempre con las medidas adecuadas dependiendo del tipo de registro y la preferencia del sistema de medida.

Perfil

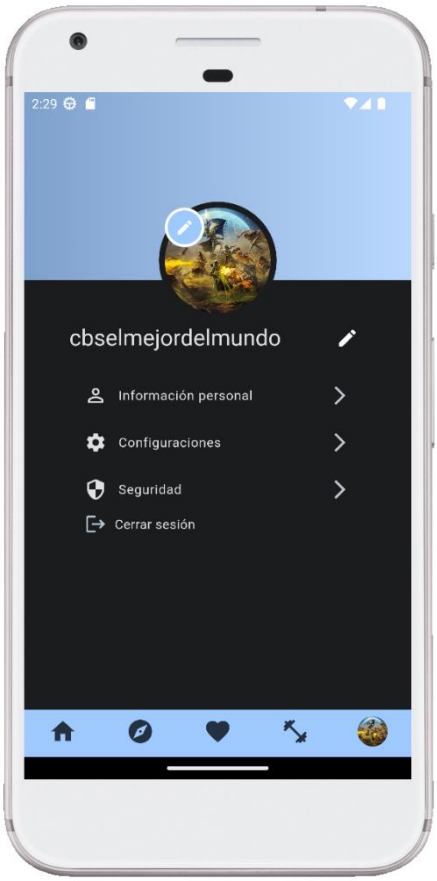

Figura 58 Pantalla de sección de perfil

La pantalla de la **Figura 58** muestra la interfaz que permite gestionar los ajustes y preferencias del perfil. Esta interfaz da la opción de cambiar la foto de perfil, el nombre de usuario. Los ajustes disponibles para gestionar se dividen en las categorías:

"información personal", "configuraciones" y "seguridad". Estas categorías se desglosan en la **Figura 58**. Desde esta pantalla también se lleva a cabo el cierre de sesión.

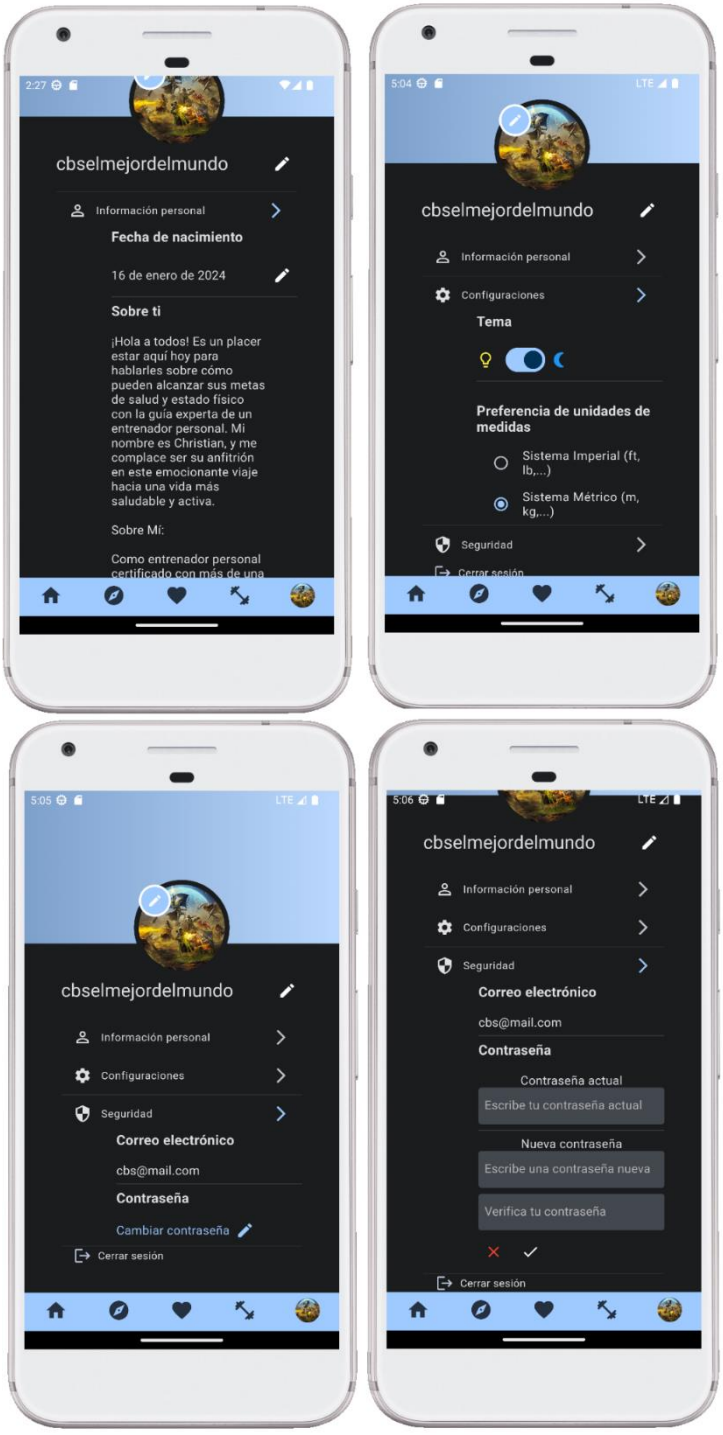

Figura 59 Interfaz de perfil (paleta de colores oscura).

En la **Figura 59** se pueden ver los ajustes y preferencias disponibles para que el usuario edite. En el primer apartado de información personal, se da la biografía y la fecha de nacimiento. La biografía será visible en las plantillas públicas del usuario.

En el apartado de configuraciones se da la opción de cambiar entre la paleta de colores oscura y clara y se permite la preferencia del sistema de medidas entre el sistema métrico y el imperial.

Por último, en el apartado de seguridad se puede cambiar la contraseña por una nueva.

Rol de administrador

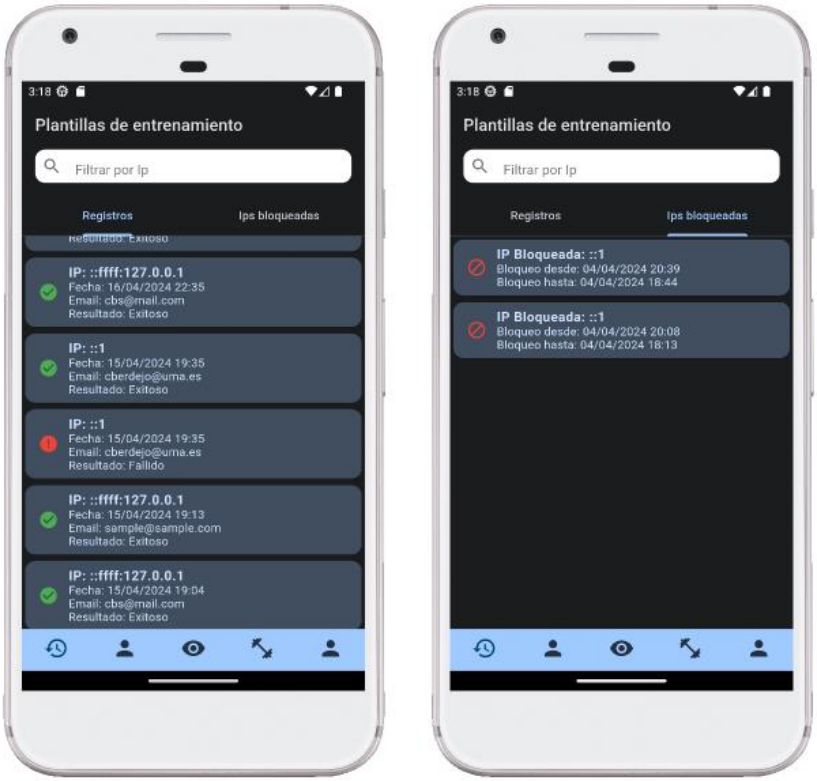

Figura 60 Pantalla de control de acceso (paleta de colores oscura).

La **Figura 60** muestra la pantalla accesible desde el rol de administrador permite ver todos los intentos de registro de acceso y las *IPs bloqueadas*. Para cada de registro de acceso se enseña la *IP*, la fecha del intento, con el correo que se ha intentado y si el resultado ha sido exitoso o no. En la pestaña de *IP*s bloqueadas se muestra la *IP* y el periodo en el que ha estado bloqueado. Una *IP* se podía bloquear si posee muchos intentos de acceso fallidos seguidos.

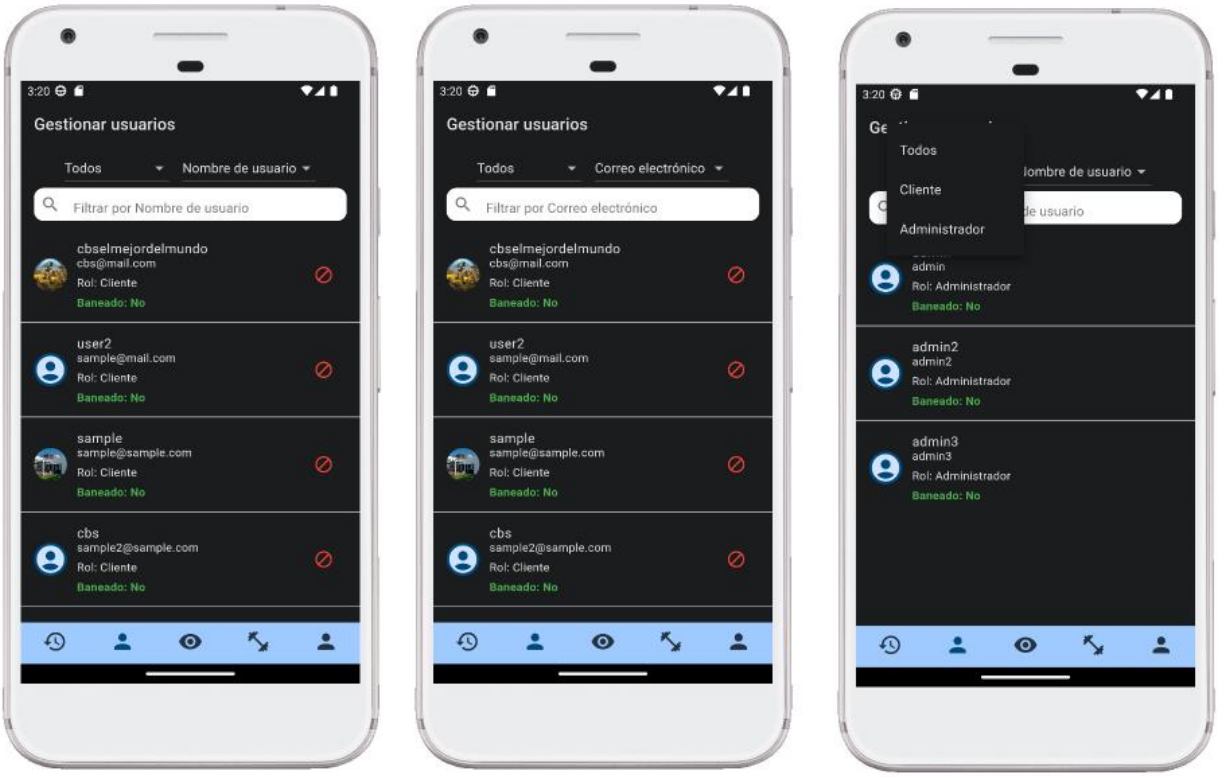

Figura 61 Pantalla de gestión de usuarios (paleta de colores oscura).

En la **Figura 61** se muestran varias imágenes de la pantalla que permite gestionar los usuarios. Es posible filtrarlos por roles y por nombre de usuario o correo electrónico. La aplicación muestra una lista de todos los usuarios y da la opción al administrador de restringirles el acceso a la aplicación.

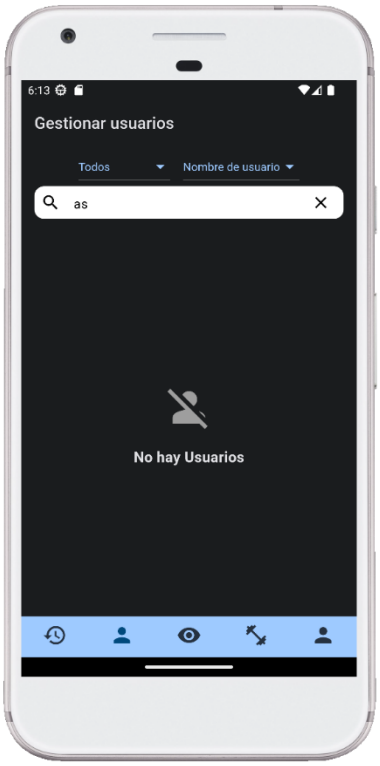

Figura 62 Pantalla de gestión de usuarios**,** usuarios no encontrados (paleta de colores oscura).

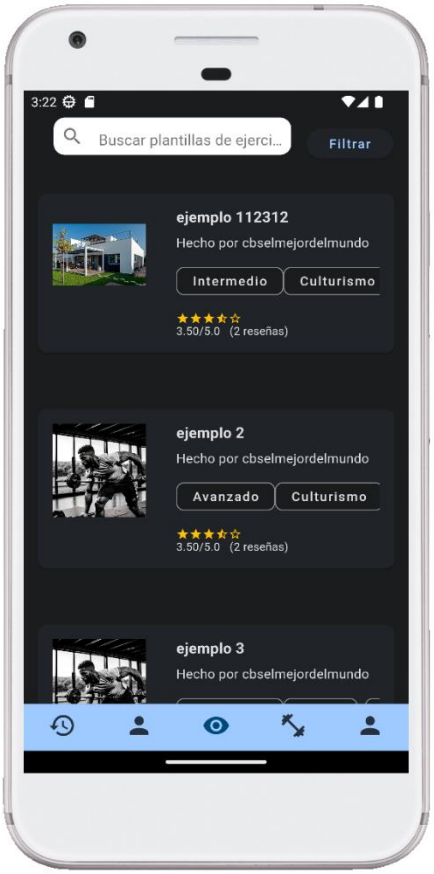

Figura 63 Visualización de plantillas de rol de administrador

Un usuario rol de administrador de forma similar que el rol de cliente puede visualizar las plantillas de entrenamiento.

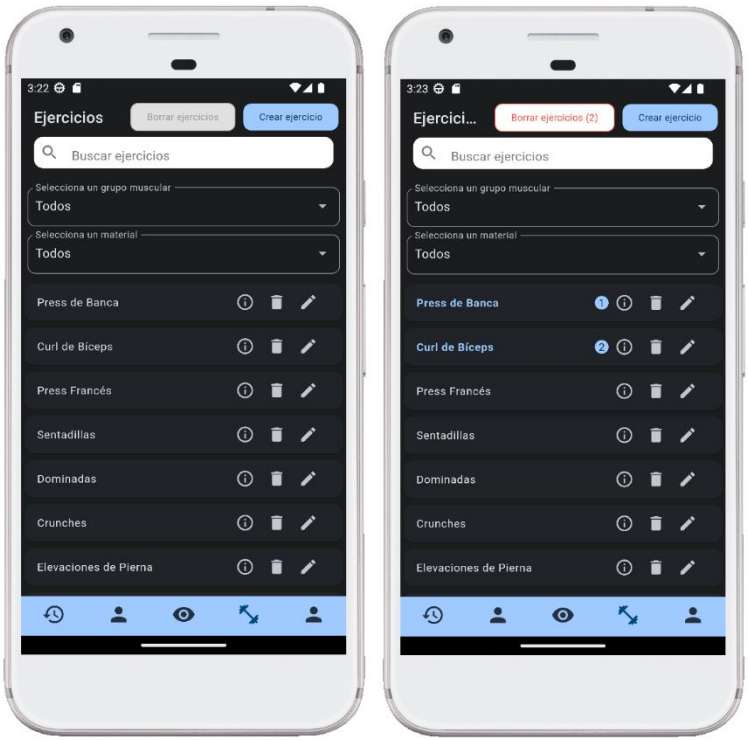

Figura 64 Gestión de ejercicios desde rol de administrador

Esta vista permite editar, eliminar y borrar ejercicios de forma global para todos los usuarios. La vista de creación y edición de ejercicios es igual a la ya mostrada en la **Figura 56.**

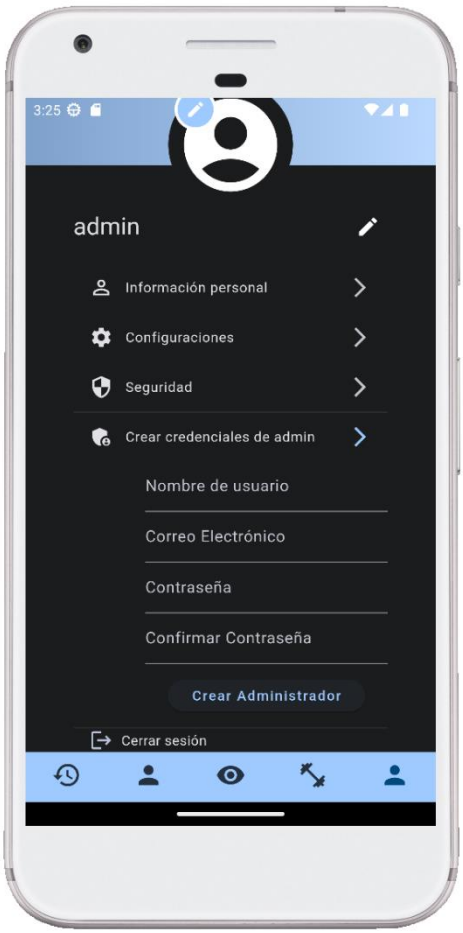

Figura 65 Vista de perfil de rol de administrador

Es idéntica a la vista de perfil de rol de cliente, pero añade la posibilidad de crear credenciales a otros administradores.

Estilo claro

Aunque para la mayoría de las imágenes se ha optado por mostrar la paleta oscura de colores, Aquí se muestran algunos ejemplos más de la tonalidad clara de la aplicación desde la **Figura 66 a 78.**

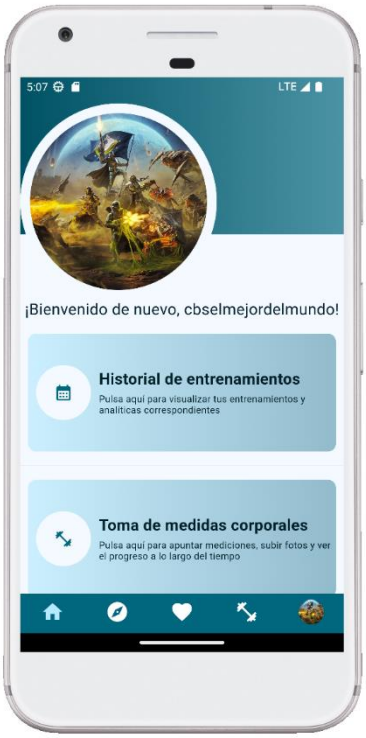

Figura 66 Página inicio (paleta de colores clara).

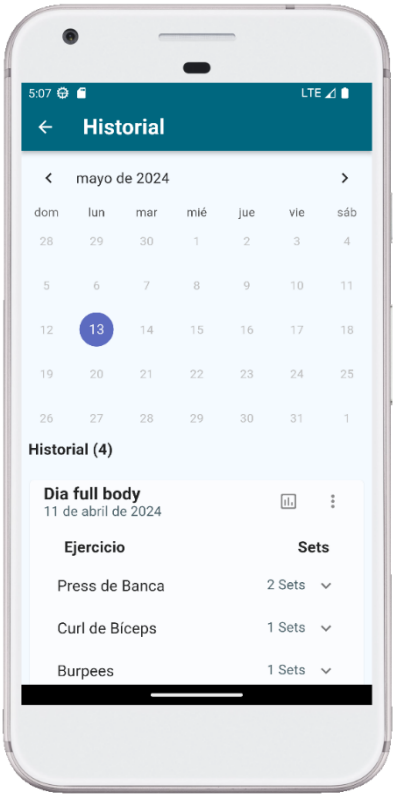

Figura 67 Interfaz de historial de sesiones de entrenamiento (paleta de colores clara).

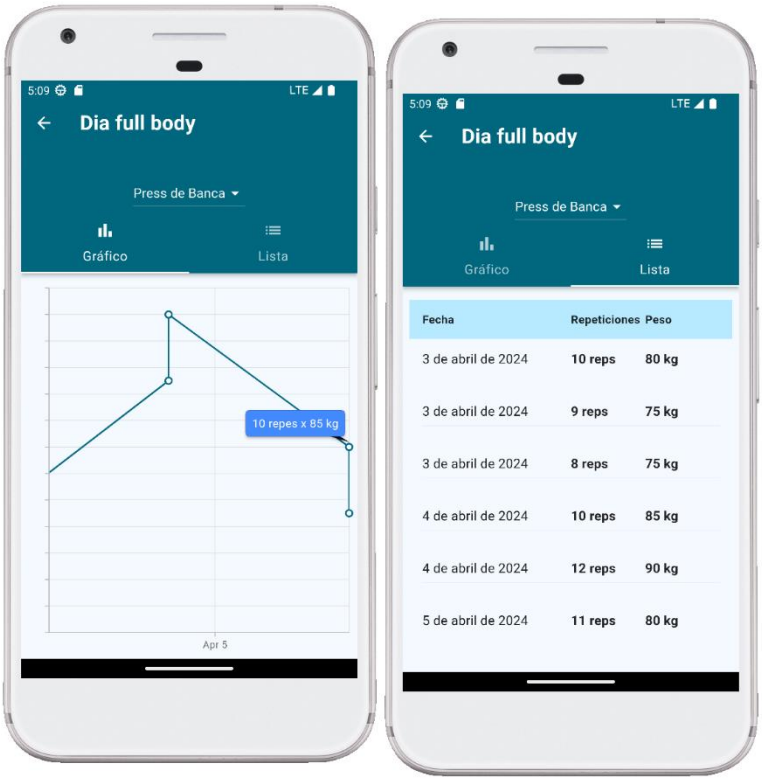

Figura 68 Interfaz de estadísticas de sesión de entrenamiento (paleta de colores clara).

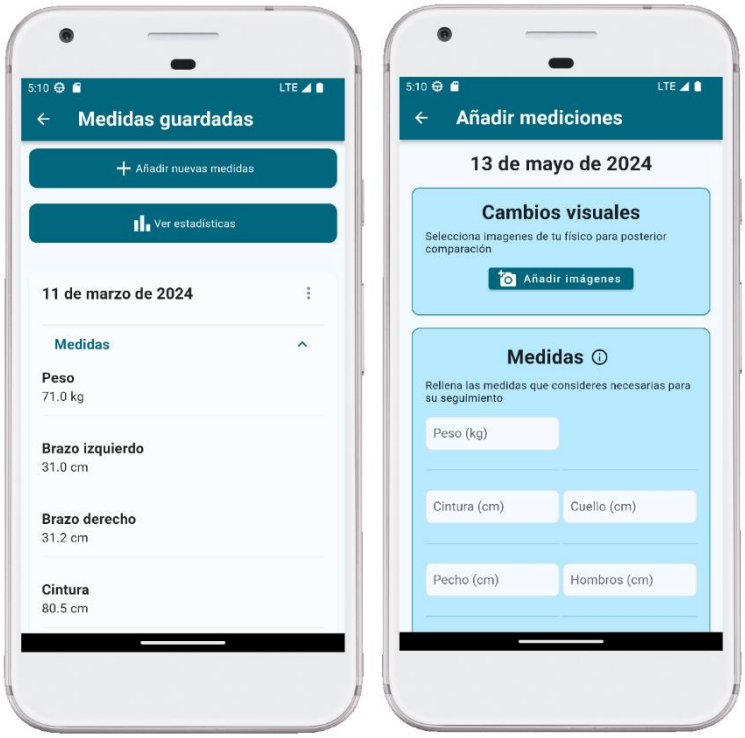

Figura 68 Pantallas para visualizar y añadir medidas corporales (paleta de colores clara).

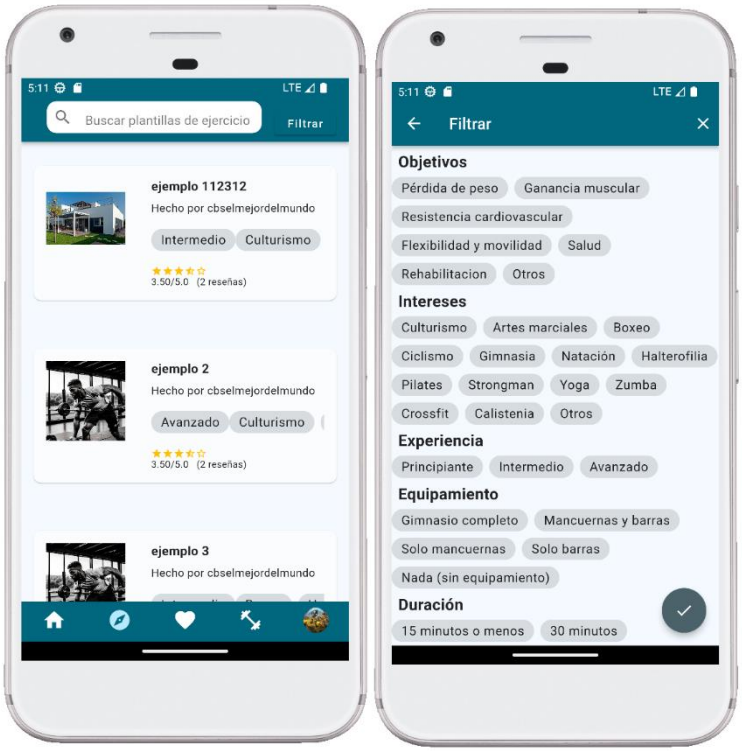

Figura 70 Pantallas de sección de descubrir con filtros (paleta de colores clara).

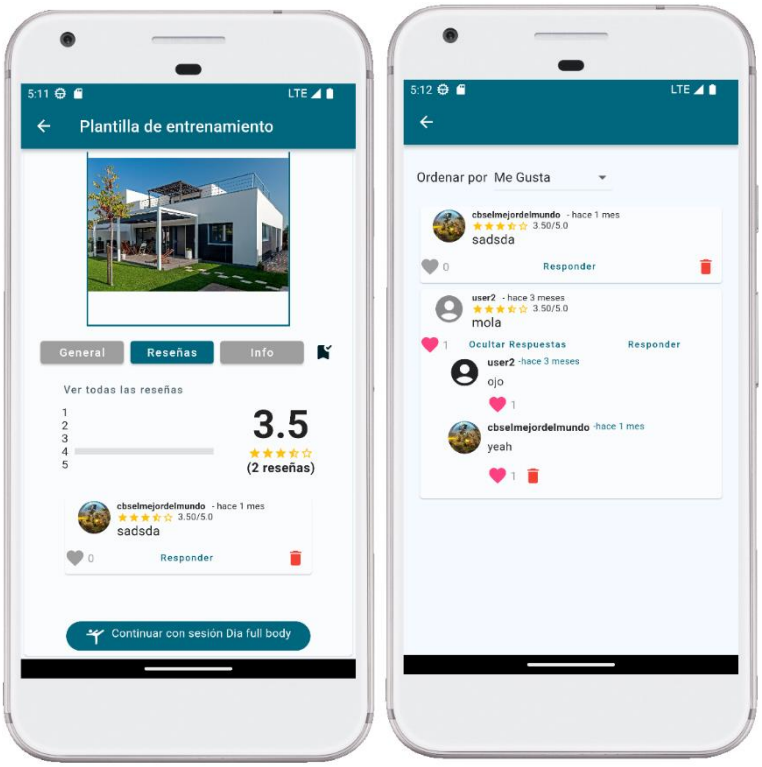

Figura 71 Pantallas de reseñas (paleta de colores clara).

Figura 72 Pantalla de notificaciones (paleta de colores clara).

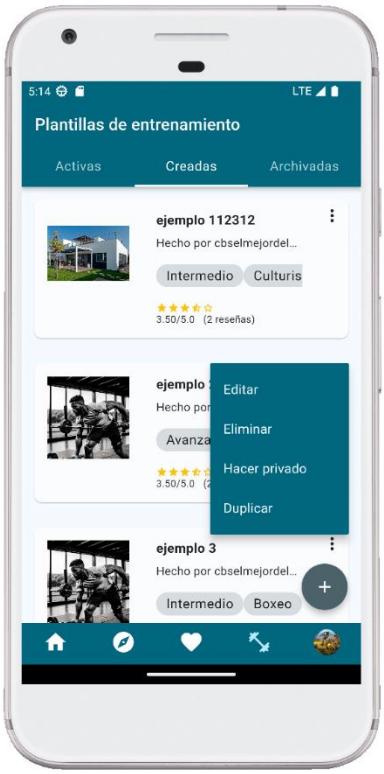

Figura 73 Pantalla de sección de entrenamiento (paleta de colores clara).

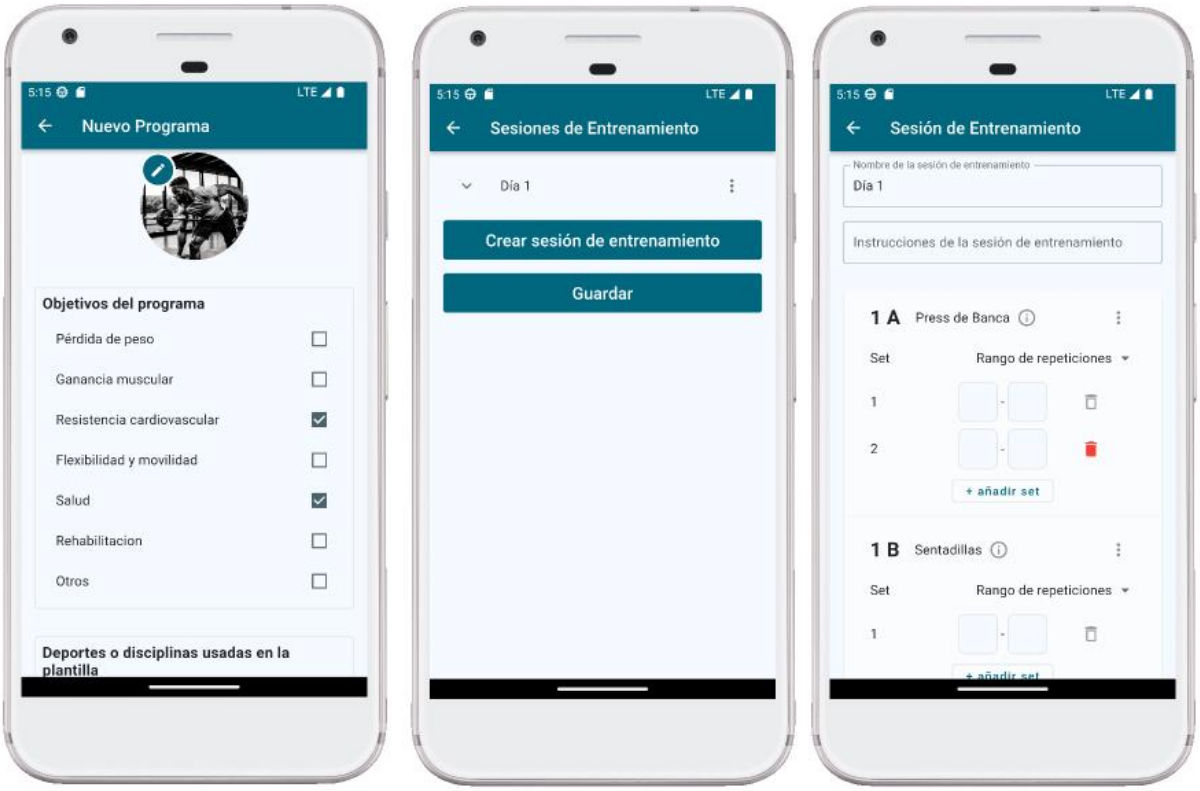

Figura 74 Creación de plantilla de entrenamiento y sesión de entrenamiento (paleta de colores clara).

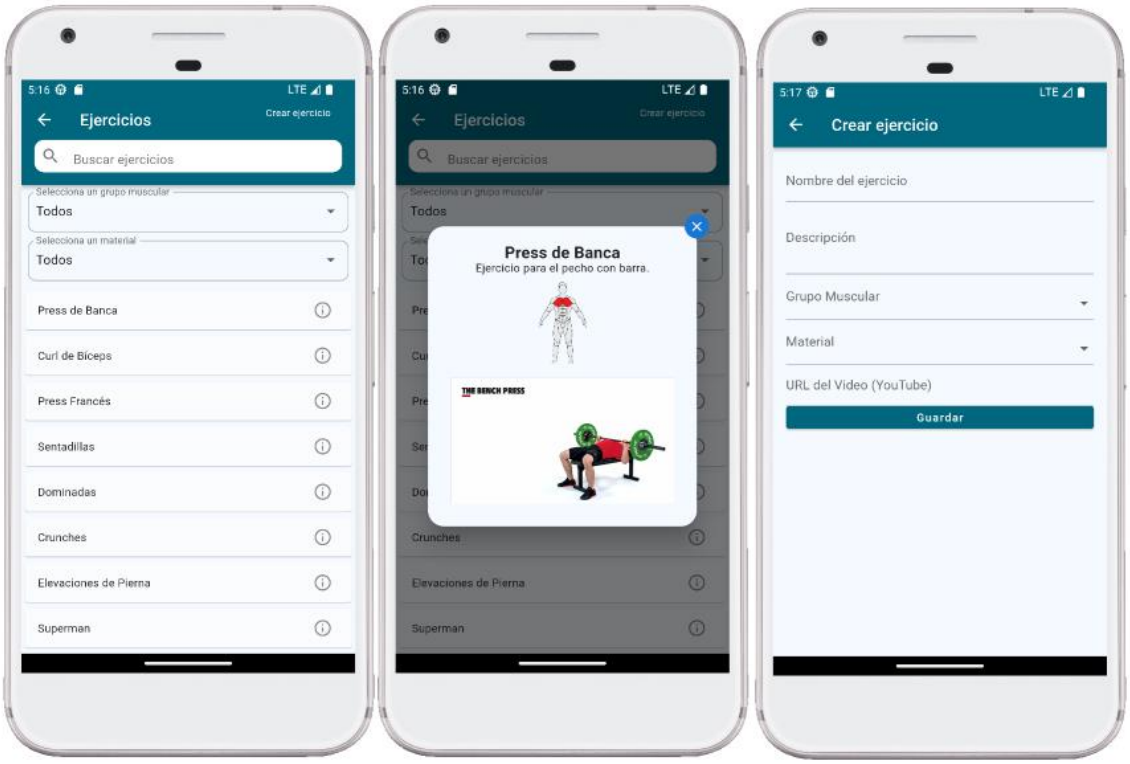

Figura 75 Selección y creación de ejercicios (paleta de colores clara).

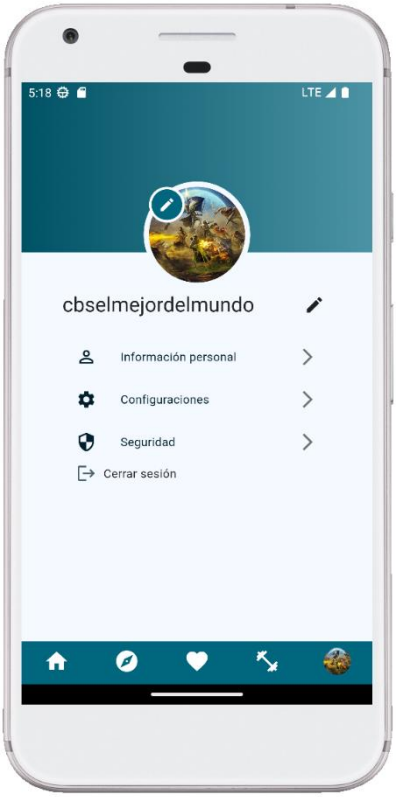

Figura 76 Pantalla de Perfil (paleta de colores clara).

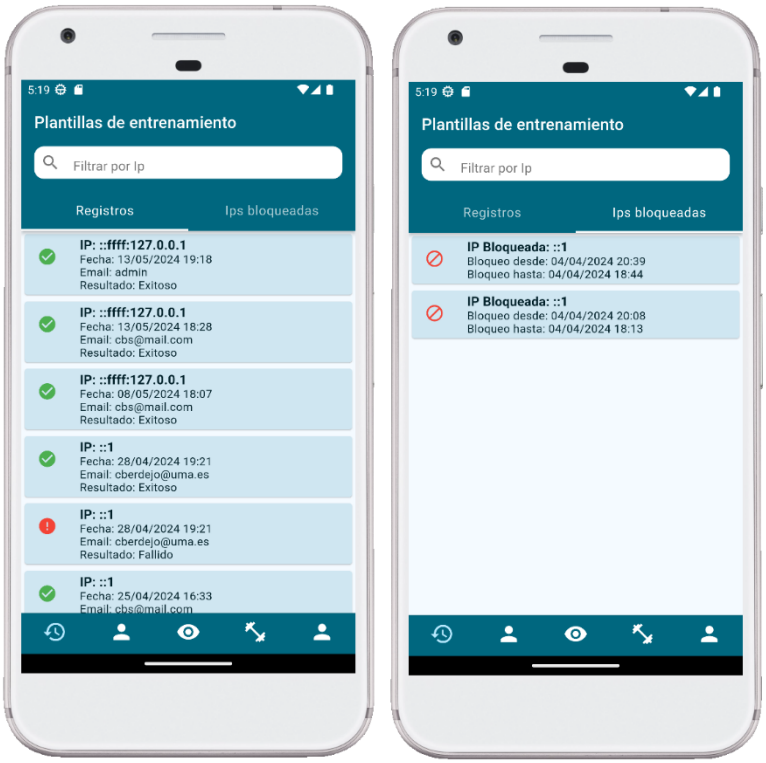

Figura 77 Pantalla de registros de acceso (paleta de colores clara).

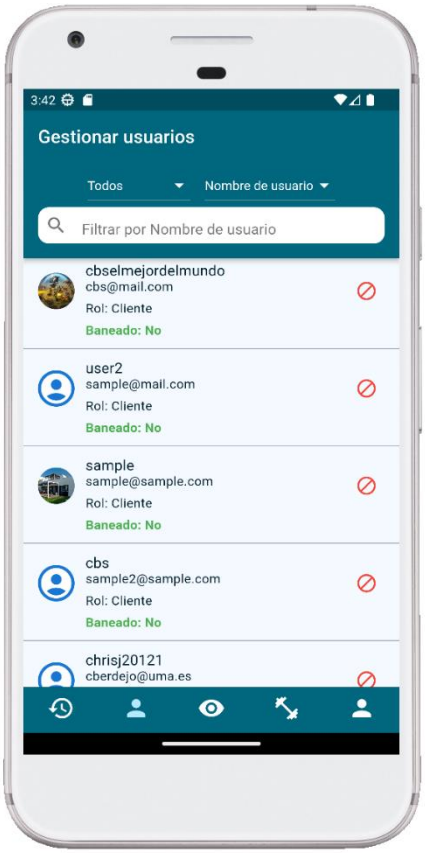

Figura 78 Interfaz de visualización de usuarios en el rol de administrador

UNIVERSIDAD DE MÁLAGA | uma.es

E.T.S. DE INGENIERÍA

E.T.S de Ingeniería Informática
Bulevar Louis Pasteur, 35
Campus de Teatinos
29071 Málaga