

## MEMORIA ESCRITA DEL PROYECTO

CFGS Desarrollo de Aplicaciones Web

# MyWorkoutlog

**Autor:** Cristian Menjíbar López

**Tutor:** Alberto Castedo Espeso

**Fecha de entrega:** 04/12/2025

**Convocatoria:** 1s2526

**Documentos del proyecto:** [Enlace a la carpeta del proyecto](#)





## Índice de contenidos

### Contenido

1.	Introducción .....	2
1.1.	Motivación.....	2
1.2.	Abstract.....	3
1.3.	Objetivos propuestos (generales y específicos).....	4
2.	Estado del Arte.....	6
3.	Metodología usada.....	10
4.	Tecnologías y herramientas utilizadas en el proyecto.....	13
5.	Planificación, Diagnóstico y Contexto Laboral .....	15
6.	Análisis del proyecto .....	17
7.	Diseño del proyecto .....	32
8.	Despliegue y pruebas .....	41
9.	Conclusiones.....	47
10.	Vías futuras .....	48
11.	Bibliografía/Webgrafía .....	50
12.	Anexos .....	52
12.1.	Manual de Instalación .....	52
12.2.	Manual de usuario.....	56
12.3.	Glosario .....	63
12.4.	Índice de imágenes .....	64
12.5.	Imágenes y Diagramas .....	66



## 1. Introducción

MyWorkoutLog es una aplicación web que permite a los usuarios registrarse para poder progresar siguiendo patrones reales de entrenamiento y, al mismo tiempo, realizar un seguimiento detallado de sus rutinas. Esta aplicación ofrece la oportunidad de registrar las fechas de cada día de entrenamiento, los ejercicios realizados, así como los datos de cada serie (peso utilizado, número de series y repeticiones). Además, esta aplicación web incluye un sistema de métricas que permite evaluar el progreso del usuario y mejorar la planificación de sus entrenamientos.

### 1.1. Motivación

La idea de desarrollar MyWorkoutLog como proyecto final de este ciclo formativo de grado superior surgió de la necesidad personal de anotar los registros de mis ejercicios durante mi propia experiencia de entrenamiento en el gimnasio.

Hace más de dos años empecé a entrenar en el gimnasio municipal de la localidad donde vivo, y durante este tiempo he podido observar cómo muchos compañeros registran manualmente sus marcas en cuadernos o libretas. Esta práctica, en mi opinión, resulta poco práctica y es muy limitada, lo que me ha llevado a investigar sobre la posibilidad de digitalizar este proceso y llevar a cabo una aplicación que además de garantizar un control de nuestros registros, pueda enseñarnos como mejorar en nuestros entrenamientos de manera sencilla.

El desarrollo de una aplicación web orientada al seguimiento de los entrenamientos fomenta soluciones más sostenibles y económicas, al no tener que depender de materiales físicos como folios y bolígrafos. Además, también ahorra tiempo, ya que ofrece la oportunidad de registrar toda la información de nuestros ejercicios con tan solo pulsar la pantalla de nuestro teléfono.

Tras investigar el mercado, comprobé que la mayoría de las herramientas de este tipo estaban incluidas en aplicaciones propias de grandes cadenas de gimnasios o bajo modelos de suscripción, lo que las hace inaccesibles para una gran parte de usuarios de gimnasios más básicos o personas que entran al aire libre. Además, durante esta investigación encontré aplicaciones muy limitadas en cuanto funciones, la mayoría requerían un plan de



pago para acceder a su versión completa. En otros casos estas aplicaciones estaban en inglés, y en otros ofrecían “planes milagro” basados únicamente en cuestionarios de medidas y preferencias del usuario sin tener en cuenta los datos reales de entrenamiento. Por otro lado, no existen aplicaciones que ofrezcan una funcionalidad principal centrada exclusivamente en el progreso real basándose en los resultados y el trabajo de cada usuario, donde los datos sean los que definan la mejora.

A nivel personal, el entrenamiento forma parte esencial de mi rutina diaria, y considero que disponer de una herramienta como “MyWorkoutLog” facilita enormemente la organización, el progreso y la motivación de cualquier persona que desee mejorar su rendimiento. Este proyecto representa, por tanto, la unión entre mis intereses personales y los conocimientos adquiridos, dando lugar a una aplicación útil, práctica y desarrollada con la finalidad ayudar a los usuarios a gestionar su progreso deportivo de forma sencilla y eficaz.

## 1.2. Abstract

MyWorkoutLog is a web application designed to help users manage and monitor their workout routines in a simple way. The principal idea for this project comes from my own experience at the gym, where I noticed that many people still use notebooks or paper to write down their training sessions. This method is not only uncomfortable but also makes it difficult to keep a long-term record of progress. For this reason, I decided to create a digital solution that allows users to store and visualize all their training information online.

The application allows registered users to create their own personalized workouts, add exercises, and record key data such as repetitions, sets, weight, and rest time. Users can also include notes for each training session, making it easier to remember specific details about their performance. In addition, the platform includes a metrics section that offers visual statistics, such as the number of workouts per month or the maximum weight lifted for each exercise.

One of the most distinctive features of MyWorkoutLog is the training progress system, also called the “training laboratory”, where users can analyze their improvements based on real training patterns. This system allows users to detect when they reach their maximum performance in a certain exercise and suggests possible progressions to continue improving



safely. By using this function, the application not only stores data but also becomes a real assistant that helps users make their routines more effective.

This project combines my personal interest in fitness with my passion for web development. It demonstrates how technology can be used to create useful and practical tools that make everyday activities easier. MyWorkoutLog is intended to be a simple yet powerful application that helps people stay organized, motivated, and consistent in their training routines.

### 1.3. Objetivos propuestos (generales y específicos)

#### Objetivos generales:

El objetivo principal de este proyecto es desarrollar una aplicación web completa denominada MyWorkoutLog que incorpore un sistema CRUD. Esta aplicación estará orientada al seguimiento y análisis del rendimiento en entrenamientos de fuerza. La aplicación permitirá a los usuarios registrar sus entrenamientos de forma personalizada, gestionar los datos de cada sesión (peso, repeticiones y series) y consultar métricas de progreso que reflejen su evolución a lo largo del tiempo.

Además, este proyecto tiene como finalidad aplicar de manera práctica los conocimientos adquiridos durante el ciclo formativo de Desarrollo de Aplicaciones Web, integrando tecnologías de front-end y back-end en un entorno funcional y seguro. De este modo, se busca crear una herramienta útil tanto a nivel técnico como personal, que facilite la gestión digital del entrenamiento y fomente el aprendizaje en el desarrollo de aplicaciones dinámicas y orientadas al usuario.

#### Objetivos específicos:

- **Gestión de usuarios:** En nuestra aplicación tendremos un sistema de gestión de usuarios donde podremos administrar los diferentes perfiles dentro de la plataforma, desde el panel de administración, será posible habilitarlos o por el contrario restringir su acceso. Además, podremos ver todos los usuarios que hay afiliados a nuestra plataforma y añadir ejercicios a la base de datos



- **Sistema de Login:** Sistema con el que permitiremos a los usuarios iniciar sesión en nuestra aplicación, consiguiendo así que cada usuario tenga su propio panel independiente del resto. Con este sistema también podremos iniciar sesión como administradores para poder gestionar a los demás usuarios.
- **Gestión del catálogo de ejercicios:** Al iniciar sesión, los usuarios podrán insertar, actualizar y eliminar ejercicios de su propio catálogo, lo que hará la aplicación personalizable a cualquier rutina.
- **Registro de entrenamientos:** Cada usuario podrá crear sesiones de entrenamientos por fecha, asignándoles un nombre (Ejemplos: Entrenamiento en gimnasio, entrenamiento en la calle, Natación en piscina) y opcionalmente añadir notas.
- **Registro de series por ejercicio:** Dentro de cada entrenamiento el usuario podrá añadir series dentro de cada ejercicio, indicando el número de repeticiones realizadas, el peso utilizado y el tiempo de descanso.
- **Sección ‘Métricas’:** Los usuarios dispondrán de un apartado donde podrán filtrar sus entrenamientos de distintas formas y consultar distintos datos como: Peso máximo levantado por grupo muscular, número de entrenamientos de un grupo muscular específico y días entrenados mensualmente, todo esto en un rango de fechas seleccionado por el usuario.
- **Laboratorio de entrenamiento:** La aplicación integrara un módulo interno denominado Laboratorio de entrenamiento, destinado a analizar el rendimiento de cada usuario para ofrecer recomendaciones de progreso personalizadas. Este sistema permitirá sugerir incrementos de carga o decrementos en base a si se cumplen o no los objetivos establecidos durante el entrenamiento anterior.
- **Diseño responsive:** La aplicación se desarrollará siguiendo el enfoque mobile-first, en el que se prioriza en primer lugar la versión para móviles y después se adapta a pantallas de mayor tamaño, como tablets y ordenadores de escritorio. Este enfoque evita la pérdida de información al escalar la interfaz a resoluciones más grandes.
- **Seguridad y almacenamiento de datos:** Para asegurar un correcto almacenamiento de datos y mantener la información dentro de nuestra base de datos segura, cifraremos todas las contraseñas y datos sensibles de nuestros usuarios afiliados mediante un algoritmo de hashing que impida la recuperación directa de la información original, asegurando la confidencialidad de los datos almacenados.



## 2. Estado del Arte

La relación del ser humano con el ejercicio físico nace de una necesidad que los humanos hemos tenido desde el principio de nuestros días. Cazar, recolectar y desplazarse recorriendo largas distancias para sobrevivir formaban parte de la vida diaria. Con el tiempo esa necesidad también se transformó en juego y símbolo social. En la antigua Grecia, surgieron los primeros gimnasios y la creación de las Olimpiadas, que, compitiendo en honor a los dioses en pruebas atléticas, marcaron un hito en la medición del rendimiento. En Roma, la preparación física se orientó más al poder militar, sin perder de vista el espectáculo y la competición pública. Entre los siglos XIX y XX emerge la cultura física moderna, el deporte se institucionaliza con federaciones y “renacen” los Juegos Olímpicos modernos.



Ilustración 1: Atletas con llama olímpica en la antigua Grecia.  
Fuente: <https://www.bbc.com/mundo/noticias-57829651>

Desde mediados del siglo XX el culturismo y el fitness comercial consiguen dar acceso a la sociedad a pesas y máquinas, nacen cadenas de gimnasios y se profesionalizan instructores. De esta etapa también nace un nuevo hábito, llevar un registro del entrenamiento, primero en libretas y agendas, y después, con el nacimiento de la informática personal almacenando datos en hojas de cálculos donde se automatizan resultados y se dibujan las primeras gráficas. En paralelo, foros y blogs permiten compartir rutinas y dietas, y en las primeras redes sociales se comparan progresos y se comparten dudas.

En la última década, el seguimiento de los entrenamientos ha pasado de registros manuales en papel a aplicaciones móviles especializadas que permiten anotar estas marcas personales



de manera sencilla, gracias a los avances tecnológicos, podemos ver o mostrar nuestras métricas personales en cualquier momento con tan solo tocar una pantalla, este cambio ha mejorado la accesibilidad de este tipo de datos en tiempo real. Sin embargo, el ecosistema actual está fuertemente orientado a aplicaciones móviles, mientras que las experiencias completas en escritorio o en red siguen siendo minoritarias, ya que la mayoría de estas aplicaciones se centran más en la idea del entrenamiento en tiempo real que de la planificación y productividad de los días en el gimnasio, además existe una fuerte barrera idiomática, ya que abundan interfaces y catálogos de ejercicios en inglés, dejando fuera una parte importante del público hispanohablante.

En este contexto se enmarca MyWorkoutLog, la aplicación web que será la base de este proyecto. Tras una investigación del mercado, buscamos diferenciarla con tres pilares: experiencia completa en navegador (también en escritorio), orientación al público de habla castellana y modelo free en funciones especiales.

A continuación, se presentan las aplicaciones analizadas y los hallazgos principales:

- **Hevy:** Se trata de una red social con un tracker de entrenamiento bastante bueno, dando la posibilidad de compartir rutinas y entrenamientos. Si bien la app es gratuita y sin publicidad, la versión sin pagar tiene limitaciones importantes, solo permite guardar hasta 4 rutinas y el historial de progreso visible se limita a 3 meses, permite guardar hasta 7 ejercicios personalizados. Dispone de versión web/escritorio para consultar y editar las rutinas. Resulta accesible para usuarios de habla hispana, pero su enfoque principal es global
- **Strong:** Es una aplicación para registrar entrenamientos de musculación, en su versión gratuita impone restricciones, solo permite crear 3 rutinas personalizadas y las métricas son solo para versiones pro. Además, no cuenta con versión web, la mayoría de la aplicación está en inglés ya que la traducción al español es deficiente.



- **JEFIT:** La versión gratuita es funcional, pero incluye bastante publicidad, y muchas funciones avanzadas como la planificación de rutinas, está limitada a 5 días en su versión gratuita, además, su interfaz está muy recargada. Al contrario que la mayoría de las aplicaciones, JEFIT, si tiene una sincronización con su portal web. No obstante, la aplicación aún no está traducida de forma completa al español.
- **FitNotes:** Es una app de registro de entrenamientos con un enfoque minimalista, bastante limitada ya que carece de funcionalidades avanzadas como análisis detallados, no dispone de versión web ni de aplicación de escritorio. Además, está solo en inglés.
- **Fitia:** Aplicación basada en el control de la alimentación, tiene un módulo de entrenamiento, pero se limita a estimar calorías gastadas en el entrenamiento, no registra ni series ni cargas.



Ilustración 2:  
Heavy. Fuente:  
Google Play



Ilustración 3:  
Strong.  
Fuente:  
Google Play



Ilustración 4:  
JEFIT. Fuente:  
Google Play



Ilustración 5:  
Fitnotes.  
Fuente:  
Google Play



Ilustración 6:  
Fitia. Fuente:  
Google Play

## Conclusión:

Tras analizar las aplicaciones y buscar modelos web, el panorama es evidente, casi todo está pensado para móvil, la mayoría de funciones relevantes quedan solo para planes de pago, el idioma predominante suele ser el inglés, y en la mayoría predominan “planes milagro” basados en cuestionarios de medidas y preferencias. Además, apenas hay alternativas con una versión web/escritorios realmente potentes. En otras lo relativo al entrenamiento se reduce a módulos secundarios para estimar calorías en aplicaciones de nutrición, pero no permiten planificar ni trackear métricas personales de fuerza.



Nosotros, para diferenciarnos del resto, adoptamos una progresión lineal personalizable, el usuario define su esquema Serie×Repetición (por ejemplo, 4×12, 4×10, 5×10) y si completa todas las series alcanzando el objetivo de repeticiones en cada una, la siguiente sesión de entrenamiento incrementara la carga en +2,5 kg (paso configurable); si no lo completa, se repite el mismo peso. Este “laboratorio de entrenamiento” sencillo, basado en tus propios datos, permitirá recomendar un progreso efectivo y medirlo con métricas claras.



### 3. Metodología usada

Durante el desarrollo de este proyecto, podemos diferenciar dos ciclos claramente distintos. El primero corresponde a la fase de planteamiento, planificación y documentación de la propuesta, mientras que el segundo se centra en el desarrollo de la aplicación de forma paralela a la redacción de la memoria.

Es en esta segunda fase donde resulta necesario definir el modelo de trabajo que utilizaremos para desarrollar nuestra aplicación. Teniendo en cuenta que el equipo de trabajo está formado por una sola persona, que los requisitos iniciales no están completamente definidos y que será necesario documentar cada fase del proyecto, he decidido utilizar un modelo iterativo-incremental. Este modelo permite trabajar mediante ciclos o iteraciones, desarrollando pequeñas partes de la aplicación en cada iteración, además facilita volver a fases anteriores en caso de que surjan nuevas funcionalidades y se necesite documentarlas adecuadamente en la memoria.

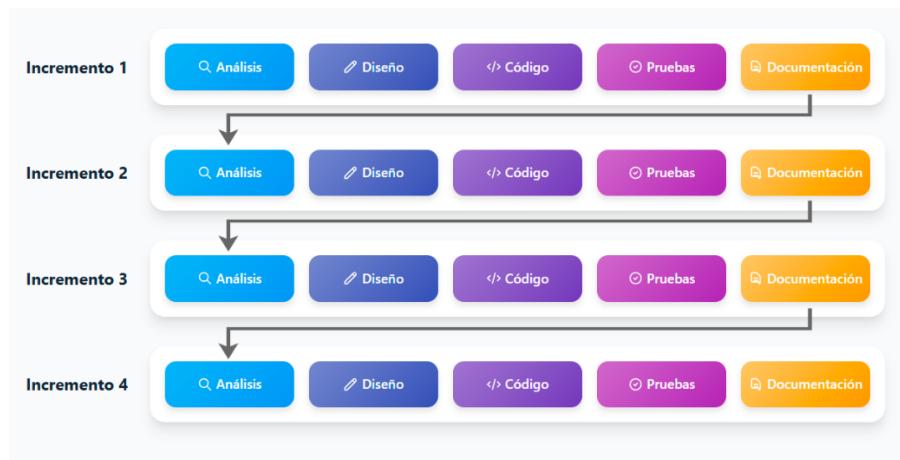


Ilustración 7 Metodología Iterativa-Incremental. Fuente: Elaboración Propia

Dado que se trata de un proyecto con finalidad formativa, no será necesario realizar una estimación de costes ni existe la preocupación de que el desarrollo se prolongue indefinidamente.



El objetivo principal será entregar un producto que cumpla con los requisitos de la normativa dentro del plazo establecido, mientras que todas las funcionalidades adicionales que puedan plantearse se incluirán en el apartado de vías futuras de esta misma memoria y solo se realizaran en caso de que nos sobre tiempo.

Cada fase o incremento tendrá entonces 5 fases distintas que a continuación explicaremos con más detalle:

**Análisis:** En esta fase del modelo reuniremos los principales requisitos funcionales y no funcionales, crearemos nuestro tablero Kanban, aprovecharemos el diagrama entidad relación que realizamos para definir la relación de tablas que entregamos en la propuesta y crearemos diagramas de casos de uso y diagramas de clases.

**Diseño:** Con los requisitos ya tomados y un buen análisis previo empezaremos prototipando nuestra aplicación utilizando la herramienta Figma, con la que podremos crear mockups de alta calidad tomando un enfoque mobile-first, después definiremos todas las secciones de nuestra aplicación, generaremos prototipos para pantallas más grandes y además predefiniremos la paleta de colores utilizada en nuestra aplicación.

**Codificación:** Con el diseño y el análisis ya definidos, comenzamos la fase de codificación, donde desarrollamos tanto el backend como el frontend de la aplicación. En primer lugar, se implementa una maquetación básica que servirá como estructura inicial para integrar un backend sólido y funcional. Una vez establecida la lógica del servidor y garantizado el correcto funcionamiento de la base de datos y las rutas, se procede a desarrollar el frontend siguiendo los prototipos diseñados previamente, aplicando el enfoque mobile-first y asegurando la coherencia visual con el resto de la aplicación.

**Pruebas:** Una vez finalizada la codificación, se realizan pruebas para comprobar que cada parte de la aplicación funciona correctamente. En esta fase se verifica que todas las funciones cumplan los requisitos establecidos y que no existan errores en el uso normal del sistema. Se comprueba el correcto funcionamiento del registro e inicio de sesión, la gestión de datos en la base de datos y la comunicación entre el frontend y el backend. En caso de detectar errores, estos se anotan en el tablero Kanban y se corregirán antes de continuar con el siguiente incremento. El objetivo de esta fase es asegurar que la aplicación sea estable y funcione tal y como se ha diseñado.



Como complemento al modelo iterativo-incremental utilizaremos la metodología ágil Kanban para gestionar las tareas de una manera visual y organizada, esta metodología permite tener una visión global y en tiempo real el estado del proyecto mediante un tablero dividido en columnas, en este tablero se reflejará cada iteración del modelo iterativo-incremental, de manera que desarrollaremos pequeños módulos o fragmentos funcionales de la aplicación, permitiendo probarlos y documentarlos antes de avanzar al siguiente.

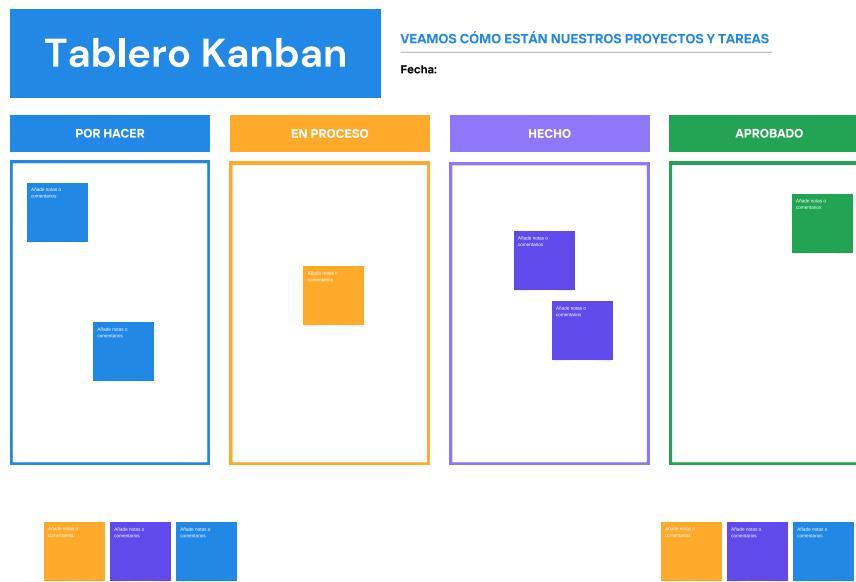


Ilustración 8: Representación de Metodología Kanban. Fuente: Elaboración propia: Canva

La herramienta seleccionada para implementar este sistema será Jira, una de las plataformas más utilizadas en entornos profesionales para la gestión ágil de proyectos.



## 4. Tecnologías y herramientas utilizadas en el proyecto

**HTML5:** Es el lenguaje de marcas utilizado para maquetar y crear la estructura de nuestra aplicación web. Se trata de la última versión definida por el consorcio W3C (World Wide Web Consortium) y representa una evolución con respecto a sus versiones anteriores, ya que incorpora nuevas etiquetas semánticas, soporte multimedia nativo y una mejor integración con CSS y JavaScript.

**CSS3:** (Cascading Style Sheets 3) es el lenguaje utilizado para definir la apariencia visual y el diseño de los elementos y componentes diseñados mediante HTML. Esta versión 3 introduce mejoras con respecto a sus predecesoras, como la posibilidad de aplicar animaciones, transiciones, transformaciones y el uso de media queries, que permiten crear diseños de manera responsive dependiendo del tamaño del navegador o del dispositivo.

**JavaScript:** Lenguaje de programación interpretado que se utiliza para darle un comportamiento dinámico a nuestra aplicación. Gracias a él, es posible manipular todos los elementos maquetados con HTML, conocidos como DOM (Document Object Model), podremos modificar su contenido, realizar comprobaciones previas al envío de datos mediante, abrir ventanas modales, mostrar mensajes dinámicos y mejorar la experiencia de usuario de forma general.

**PHP:** Lenguaje de programación interpretado que se ejecuta en el lado del servidor. Se utiliza principalmente para dar la lógica de servidor, procesar formularios y conectar nuestra aplicación con la base de datos. En este proyecto, PHP se encarga de recibir y almacenar los datos enviados desde los formularios, realizar las consultas a la base de datos y gestionar las sesiones de usuario. Además, nos permite establecer un controlador principal index.php que actúa como enrutador, cargando la vista correspondiente según la acción del usuario, esta estructura se basa en el patrón Modelo-Vista-Controlador (MVC), donde el modelo es el encargado de gestionar las operaciones con la base de datos, lo que mejora la claridad, el mantenimiento y la eficiencia del código.

**XAMPP:** Entorno de desarrollo gratuito que incluye un servidor web Apache para ejecutar nuestra aplicación y realizar pruebas de manera local, el lenguaje de programación PHP, y el sistema gestor de bases de datos MySQL/MariaDB junto con la herramienta de administración



phpMyAdmin, con la cual podemos almacenar la base de datos relacional y gestionarla gráficamente.

**SQL:** SQL (Structured Query Language) es un lenguaje de consulta estructurado utilizado para gestionar y manipular bases de datos relacionales. Permite crear, modificar y eliminar estructuras de datos, así como insertar, actualizar, consultar y borrar registros de forma eficiente. En este proyecto, SQL se utiliza para realizar las operaciones CRUD necesarias en el funcionamiento de la aplicación.

**Git:** Sistema de control de versiones que permite gestionar el historial de cambios en el código del proyecto. En este proyecto se utilizará para mantener copias de seguridad, así como volver a estados anteriores en caso de error.

**Visual Studio Code:** Entorno de desarrollo que utilizaremos para codificar nuestra aplicación. Gracias a sus extensiones, permite trabajar de forma cómoda con HTML, CSS, JavaScript PHP y SQL, además de integrar herramientas como el control de versiones con Git o la vista en directo desde el navegador.

**Jira:** Herramienta de gestión de proyectos desarrollada por Atlassian, permite planificar y controlar el progreso de las distintas fases del proyecto, será la herramienta encargada de crear nuestro tablero Kanban

**Lucidchart:** Herramienta online que utilizaremos para realizar los diagramas de casos de uso y entidad-relación de nuestro proyecto.

**Dbdiaagram:** Herramienta online que utilizaremos para ilustrar nuestro modelo relacional.

**Figma:** En este proyecto, Figma se ha utilizado para diseñar la interfaz visual de la aplicación, definiendo la disposición de los elementos, la paleta de colores, el diagrama de clases y la estructura general de las distintas pantallas. Gracias a su entorno intuitivo y a la posibilidad de previsualizar los prototipos, Figma facilita la planificación del diseño frontend y asegura una coherencia visual entre todas las secciones de la aplicación.

**Adobe Photoshop / illustrator:** Softwares de edición de imagen, que utilizaremos para el tratamiento de las imágenes de nuestra aplicación web.

**OBS:** Software que será utilizado para la grabación de nuestro video de defensa del proyecto.

**Microsoft Oficce:** Software de ofimática con el que elaboraremos tanto la presentación como la memoria del proyecto escrito.



## 5. Planificación, Diagnóstico y Contexto Laboral

Para la planificación del proyecto se tomó como punto de partida el 22 de septiembre de 2025, fecha de inicio de la asignatura, momento en el que comencé a elaborar una propuesta inicial. Una vez obtenido el apto de la propuesta, el siguiente paso fue diseñar un diagrama de Gantt que sirviese como guía para estimar el tiempo necesario para completar cada fase.

En dicho diagrama se incluyó la fase previa a la validación de la propuesta, ya que en ella se sentaron las bases para elaborar los apartados iniciales de la memoria. A partir de este punto, la planificación se ha estructurado conforme al modelo iterativo-incremental, lo que ha permitido desarrollar la aplicación de manera progresiva, integrando el trabajo de documentación y las pruebas en cada ciclo de desarrollo. Además, complementé esta metodología con un tablero Kanban. Gracias a esta herramienta, ha sido posible gestionar el tiempo de manera más eficiente, compatibilizando el desarrollo del proyecto con mi vida laboral y las demás asignaturas del semestre.

A continuación, mostraré el diagrama de Gantt estimado, pero en los '[Anexos](#)' de este documento encontraremos el diagrama de Gantt real, para así poder comparar los tiempos previstos con los realmente empleados y comentar de forma personal las principales diferencias encontradas durante el desarrollo.

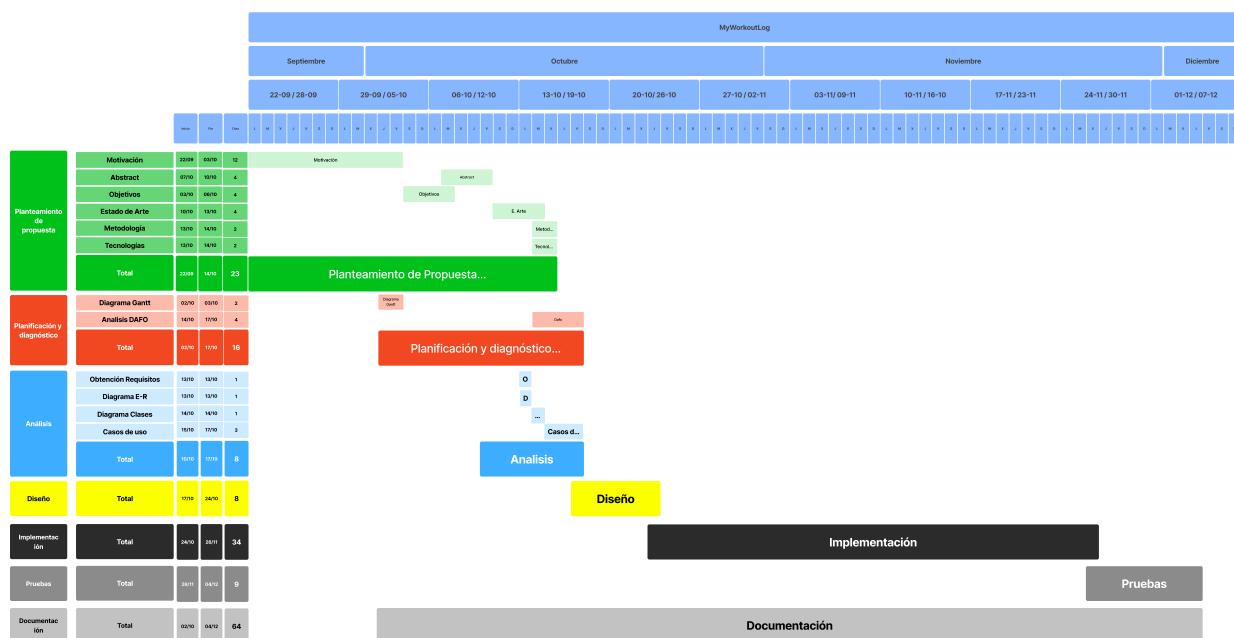


Ilustración 9: Diagrama de Gantt estimado. Fuente: Elaboración propia con Figma



Para analizar los puntos fuertes y débiles que podría tener nuestra aplicación y antes de pasar a la siguiente fase, fue necesario realizar un análisis **DAFO** para investigar cuales serían sus principales **Debilidades Amenazas Fortalezas y Oportunidades** con respecto a las demás aplicaciones del mercado. Este análisis es una de las herramientas de marketing más utilizadas hoy en día, ya que permite tener una visión global tanto del entorno interno del proyecto (sus capacidades y limitaciones) como del entorno externo (las condiciones del mercado y la competencia).

A continuación, se detallan los principales resultados del análisis DAFO:

**Fortalezas:**

- Interfaz sencilla, clara e intuitiva para registrar y consultar entrenamientos.
- Aplicación 100 % web, accesible desde cualquier dispositivo sin necesidad de instalación.
- Enfoque centrado en el progreso del usuario mediante el logro de objetivos reales.
- Métricas y personalización completa sin que esto signifique pagar la versión premium.

**Debilidades:**

- Proyecto desarrollado por un único integrante, sin experiencia, con tiempo y recursos limitados.
- Experiencia reducida en el uso de frameworks modernos.

**Oportunidades:**

- Creciente interés por el fitness y las herramientas de seguimiento digital.
- Escasez de aplicaciones deportivas totalmente en español.
- Posibilidad de integrar en el futuro un módulo de progreso asistido por inteligencia artificial.

**Amenazas:**

- Aparición de nuevas aplicaciones con integración avanzada de IA que puedan dejar obsoleta nuestra aplicación y nuestra idea principal de avance.
- Competencia de desarrolladores y empresas internacionales con mayor experiencia y recursos.



En conclusión, las debilidades mostradas al final se entienden como parte del proceso formativo, y podrán solventarse con la adquisición de experiencia y una futura especialización. Por otro lado, los puntos fuertes de nuestra aplicación, junto a la oportunidad futura de implementar un módulo de progreso por Inteligencia Artificial, abren paso a una hipotética oportunidad para después de haber consolidado un público objetivo, poder ofrecer esta funcionalidad como un servicio incluido en un modelo de suscripción.

## 6. Análisis del proyecto

Para empezar a analizar el proyecto, tomaremos como referencia los objetivos planteados anteriormente, donde podemos observar que en la aplicación que existirán dos roles de usuario diferenciados dentro de nuestro sistema.

Por un lado, existirán los usuarios administradores, encargados de la gestión de usuarios y del mantenimiento general de la plataforma, (Rol que en este caso será representado por mí, aunque en un entorno real con un equipo más amplio podría estar compuesto por varios miembros de la empresa) y por otro lado, se encontrarán los usuarios básicos, quienes representarán al público general y podrán utilizar la aplicación para registrar, consultar y analizar sus entrenamientos de manera personalizada.

Definidos los roles principales, pasamos a establecer los requisitos funcionales y no funcionales que conformarán el comportamiento general de la aplicación.

### Requisitos Funcionales:

- **RF1:** El sistema permitirá el registro de nuevos usuarios mediante formulario
- **RF2:** El administrador podrá gestionar usuarios para aprobarlos en el sistema o eliminarlos.
- **RF3:** Los usuarios activos podrán iniciar sesión en la aplicación y acceder a su panel personal.
- **RF4:** El usuario podrá insertar ejercicios a su catálogo personal.
- **RF5:** El usuario podrá eliminar ejercicios propios.
- **RF6:** El administrador podrá crear o eliminar ejercicios del catálogo general de la aplicación.
- **RF7:** El usuario podrá crear entrenamientos por nombre y añadir notas opcionales.
- **RF8:** El usuario podrá editar o eliminar entrenamientos existentes.



- **RF9:** El usuario podrá añadir ejercicios a un entrenamiento.
- **RF10:** En cada ejercicio del entrenamiento, el usuario podrá registrar series indicando repeticiones, peso y tiempo de descanso.
- **RF11:** El usuario podrá editar o eliminar series ya registradas.
- **RF12:** El usuario podrá consultar métricas de su progreso, visualizadas mediante resúmenes.
- **RF13:** El usuario podrá filtrar métricas por rango de fechas y/o por ejercicio.
- **RF14:** El sistema ofrecerá recomendaciones de progresión (incrementar, mantener o reducir carga) basadas en el histórico de series y el cumplimiento de objetivos.
- **RF15:** El usuario podrá entrar en el apartado perfil y modificar sus datos personales.
- **RF16:** El usuario podrá cerrar sesión.

#### Requisitos no funcionales:

- **RNF1:** La aplicación debe funcionar correctamente en los principales navegadores (Google Chrome, Edge, Firefox).
- **RNF2:** Las páginas deben cargarse de forma rápida y fluida.
- **RNF3:** La interfaz debe ser clara, sencilla e intuitiva.
- **RNF4:** Se aplicará una paleta de colores que reúna los requisitos de accesibilidad y usabilidad recomendados por el W3C.
- **RNF5:** El código debe estructurarse de forma modular para facilitar el mantenimiento y las actualizaciones.
- **RNF6:** Todas las contraseñas serán almacenadas bajo un algoritmo de hashing.

#### Diagrama Entidad-Relación:

Una vez definidos los requisitos de nuestra aplicación, pasamos a detallar el diagrama Entidad-Relación (E-R). Este diagrama fue elaborado en una fase inicial, incluso antes de enviar nuestra propuesta para el proyecto, ya que nos fue de gran ayuda realizarlo para poder definir cuales serían las tablas principales de la base de datos de nuestro sistema.

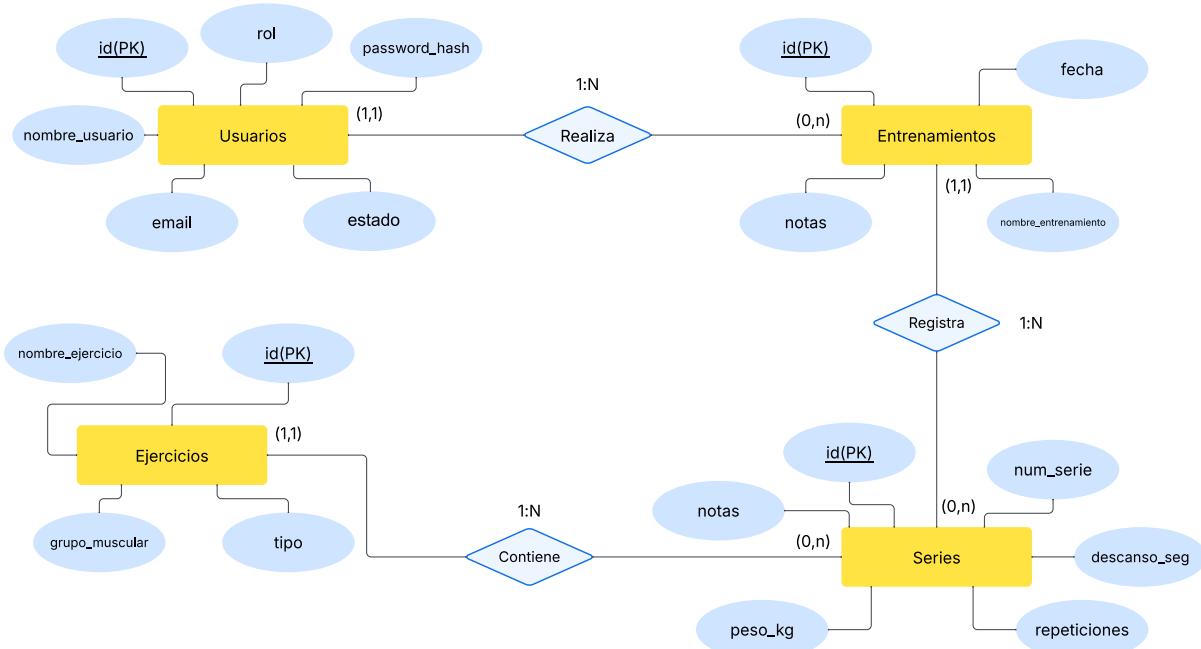


Ilustración 10: Diagrama Entidad-Relación (E-R). Fuente: Elaboración propia con Lucidchart

Este diagrama puede consultarse en el apartado de '[Anexos](#)' de este mismo documento en un formato más ampliado y con mayor resolución.

#### Descripción del diagrama:

En este diagrama podemos distinguir 4 entidades distintas: Usuarios, Entrenamientos, Ejercicios y Series. A continuación, pasaremos a explicarlas en detalle.

**Usuarios:** Almacena todos los datos de cada usuario registrado en la plataforma (nombre, email y contraseña). Sus atributos son los siguientes:

- Id (PK): Identificador único y autonumérico de cada usuario registrado en la plataforma.
- Nombre\_usuario: Almacenará el nombre del usuario tal y como lo introduce en su perfil.
- Email: Almacenará el correo electrónico del usuario, utilizado para el inicio de sesión y comunicaciones
- Password\_hash: Almacenará la contraseña del usuario de forma cifrada (hash), nunca en texto plano.
- Rol: Almacenará si el usuario es administrador o usuario básico de la plataforma.



- Estado: Almacenará la situación actual del usuario después de registrarse. Puede tener dos valores, activo o pendiente.

**Entrenamientos:** Contiene la información de las sesiones de entrenamiento realizadas por cada usuario, incluyendo la fecha y posibles notas.

- Id (PK): Identificador único y autonumérico de cada sesión de entrenamiento.
- Fecha: Almacenará la fecha en la que se realizó el entrenamiento.
- Nombre\_entrenamiento: Almacenará el nombre que el usuario le da al entrenamiento
- Notas: Almacenará observaciones sobre la sesión de entrenamiento (sensaciones, incidencias, etc.)

**Ejercicios:** Representa el catálogo de ejercicios disponible en la aplicación (global y/o personal).

- id (PK): Identificador único y autonumérico de cada ejercicio.
- Nombre\_ejercicio: Almacenará el nombre del ejercicio (“Press banca”, “Sentadilla”, “Jalón Polea” etc.)
- Grupo\_muscular: Almacenará el grupo muscular principal implicado en el ejercicio (“Pecho”, “Piernas”, “Espalda” etc.)
- Tipo: Almacenará el tipo de ejercicio (“Ejercicio de fuerza”, “Cardio” etc.)

**Series:** Entidad que actúa como tabla de unión entre los entrenamientos y los ejercicios, guardando el número de orden de la serie dentro del ejercicio, las repeticiones, el peso y las observaciones.

- Id (PK): Identificador único y autonumérico de cada serie registrada.
- Num\_serie: Almacenará el número de orden de la serie dentro del mismo ejercicio y entrenamiento (por ejemplo, 1, 2, 3...).
- Descanso\_seg: Almacenara el tiempo de descanso entre cada serie.
- Repeticiones: almacenará la cantidad de repeticiones realizadas en esa serie.
- Peso\_kg: almacenará el peso utilizado en esa serie.
- Notas: almacenará comentarios opcionales específicos de la serie (técnica, RPE, fallos, etc.).



### Modelo relacional:

A partir del diagrama Entidad–Relación, se ha desarrollado el modelo relacional, normalizándolo hasta su tercera forma normal (3FN), garantizando que no existan redundancias y que cada campo dependa únicamente de su clave principal obteniendo las siguientes tablas:

- Usuarios: **id(PK)**, nombre\_usuario, email, password\_hash, rol, estado
- Ejercicios: **id(PK)**, **id\_usuario(FK)**, nombre\_ejercicio, grupo\_muscular, tipo
- Entrenamientos: **id(PK)**, **id\_usuario(FK)**, fecha, nombre\_entrenamiento, notas
- Series: **id(PK)**, **id\_entrenamiento(FK)**, **id\_ejercicio(FK)**, num\_serie, descanso\_seg, repeticiones, peso\_kg, notas

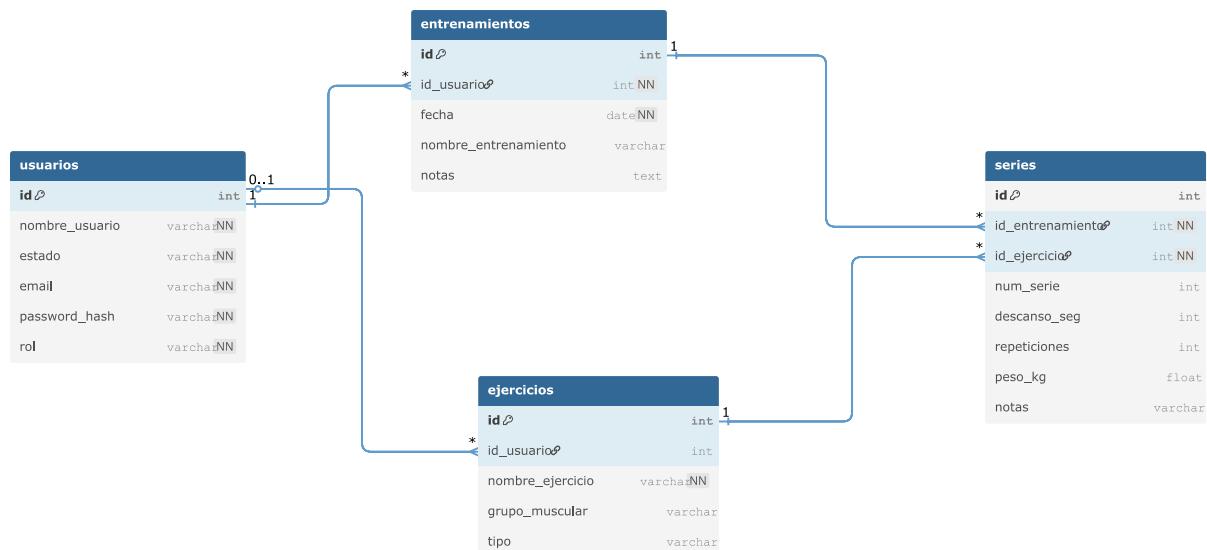


Ilustración 11: Modelo Relacional. Fuente: Elaboración propia con dbdiagram.

Este diagrama puede consultarse nuevamente en el apartado de '[Anexos](#)' de este documento.

Todas las relaciones de nuestro diagrama Entidad-Relación eran 1:N, lo que quiere decir que para pasar al modelo relacional hemos tenido que establecer las claves primarias de las entidades del lado 1 como claves foráneas en el lado N.

Además, la tabla Series tiene un papel central en la estructura del modelo, ya que actúa como nexo entre Entrenamientos y Ejercicios. Podríamos decir que entre estas dos entidades existe una relación N:M que se resuelve mediante la tabla series.



Por otro lado, la tabla Ejercicios admite un catálogo global o personal gracias a su diseño, si la clave foránea id\_usuario está NULL, el ejercicio es global, si tiene valor, es un ejercicio personal visible solo para su propietario.

### Diagrama de Clases:

Para definir la estructura del sistema, hemos creado un diagrama de clases que identifica las clases del modelo, sus atributos y sus métodos principales, proporcionando la visión orientada a objetos de nuestra aplicación. Este diagrama sirve como base para la posterior fase de Diseño, donde se detallará la arquitectura MVC (Modelo–Vista–Controlador) para separar la lógica de la aplicación de la presentación para el usuario.

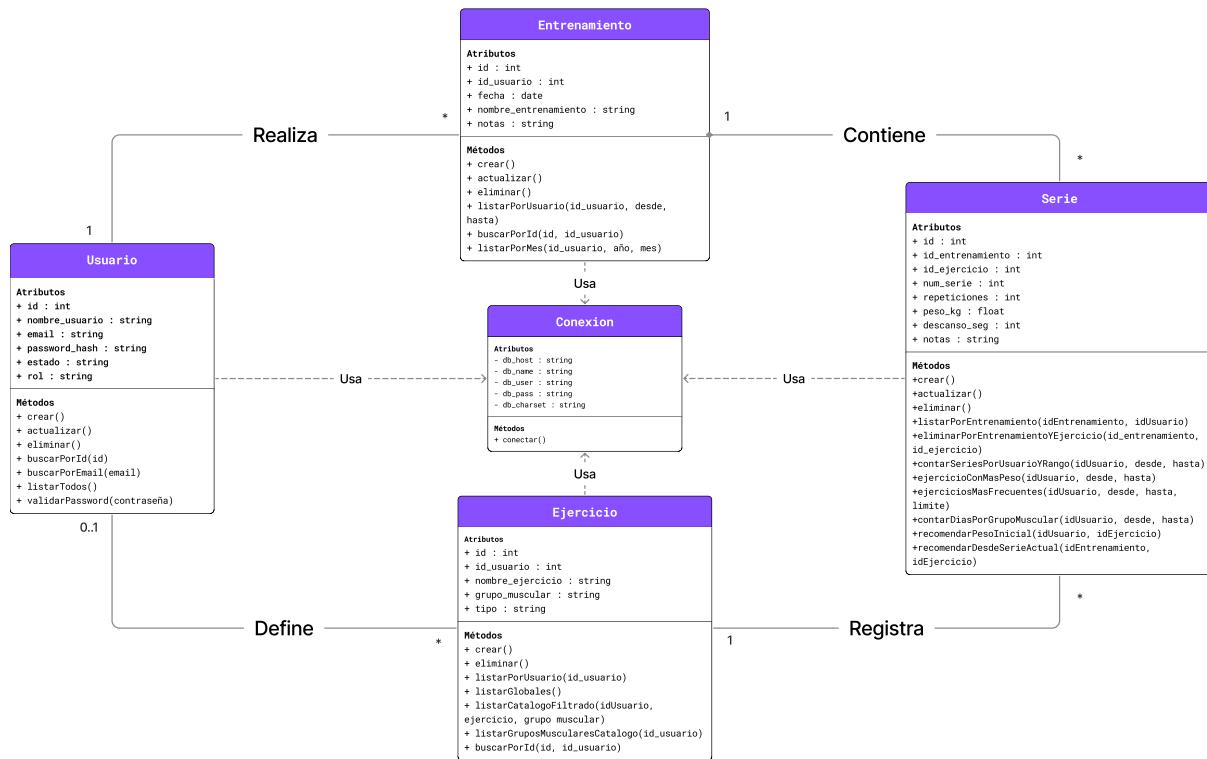


Ilustración 12: Diagrama de clases. Fuente: Elaboración propia con Figma.

Al igual que el modelo relacional y el diagrama entidad-relación, podremos analizar en profundidad este diagrama en '[Anexos](#)'.



## Relaciones:

- **Usuario — Entrenamiento (1 — \*).** Un usuario puede tener muchos entrenamientos, pero un entrenamiento solo puede tener un usuario
- **Entrenamiento — Serie (1 — \*).** Un entrenamiento tiene muchas series, pero cada serie pertenece a 1 entrenamiento. Esta relación se marca como composición (rombo negro) porque una serie no puede existir sin su entrenamiento y se elimina con él.
- **Ejercicio — Serie (1 — \*).** Un ejercicio aparece en muchas series, pero cada serie es solo de un ejercicio.
- **Usuario — Ejercicio (0..1 — \*) (propietario opcional / global).** Un usuario puede tener muchos ejercicios propios, cada ejercicio tiene 0 o 1 propietario:
  - 0 ⇒ es global (sin usuario, visible para todos).
  - 1 ⇒ es propio de ese usuario.
- **Conexión (Infraestructura):** se dibuja con línea discontinua (usa) y sin cardinalidades, porque es una clase con una relación de dependencia que todas las clases del modelo utilizan para acceder a la base de datos.

## Métodos:

- **Conexión:**
  - **conectar():** Abre una conexión PDO a MySQL.
- **Usuario:**
  - **crear():** Crea un nuevo usuario en la base de datos.
  - **actualizar():** Modifica los datos del usuario.
  - **eliminar():** Elimina el usuario actual en la base de datos.
  - **buscarPorId(id):** Busca usuario por ID.
  - **buscarPorEmail(email):** Busca usuario por email (login/registro).
  - **listarTodos():** Devuelve un listado de todos los usuarios
  - **validarPassword(contraseña):** Compara contraseña con el Hash guardado.
- **Ejercicio:**
  - **crear():** Crea un ejercicio nuevo. Puede ser un ejercicio global (si id\_usuario es null) o un ejercicio personal asociado a un usuario.
  - **eliminar():** Elimina un ejercicio de la base de datos..
  - **listarPorUsuario(id\_usuario):** Devuelve la lista de ejercicios personales creados por un usuario concreto.



- **listarGlobales():** Devuelve todos los ejercicios globales disponibles para todos los usuarios.
- **listarCatalogoFiltrado(id\_usuario, búsqueda, grupo):** Devuelve el catálogo de ejercicios combinando ejercicios globales y personales del usuario, permitiendo filtrar por texto (nombre del ejercicio) y por grupo muscular.
- **listarGruposMuscularesCatalogo(id\_usuario):** Obtiene los grupos musculares existentes en el catálogo de ejercicios del usuario (globales + personales) para llenar los filtros de la interfaz.
- **buscarPorId(id, id\_usuario\_actual)** Busca un ejercicio por su id y devuelve el ejercicio solo si el usuario actual tiene permiso para verlo (porque es suyo o porque es un ejercicio global).
- **Entrenamiento:**
  - **crear():** Crea entrenamiento asociado a un usuario.
  - **actualizar():** Actualiza datos de un entrenamiento.
  - **eliminar():** Elimina un entrenamiento, las series se borran gracias a las reglas de integridad referencial.
  - **listarPorUsuario(id\_usuario, desde, hasta):** Lista entrenamientos del usuario filtrados por fecha.
  - **buscarPorId(id, id\_usuario):** Obtiene un entrenamiento si es del usuario.
  - **listarPorMes(id\_usuario, año, mes):** Devuelve los entrenamientos de un usuario en un mes concreto.
- **Serie:**
  - **crear():** Crea una nueva serie asociada a un entrenamiento y a un ejercicio.
  - **actualizar():** Modifica los datos de una serie existente..
  - **eliminar():** Elimina una serie.
  - **listarPorEntrenamiento(id\_entrenamiento, id\_usuario):** Obtiene todas las series de un entrenamiento, comprobando que pertenece a un usuario.
  - **eliminarPorEntrenamientoYEjercicio(id\_entrenamiento, id\_ejercicio):** Elimina todas las series asociadas a un ejercicio concreto dentro de un entrenamiento
  - **contarSeriesPorUsuarioYRango(id\_usuario, desde, hasta):** Cuenta el número total de series realizadas por un usuario en un intervalo de fechas. Se utiliza en las métricas generales.



**ejercicioConMasPeso(id\_usuario, desde, hasta) (estático):** Obtiene el ejercicio en el que el usuario ha levantado más peso en el periodo indicado.

**ejerciciosMasFrecuentes(id\_usuario, desde, hasta, límite):** Devuelve los ejercicios que el usuario ha realizado con mayor frecuencia, ordenados por número de series.

**contarDiasPorGrupoMuscular(id\_usuario, desde, hasta):** Calcula cuántos días se ha trabajado cada grupo muscular en el intervalo de fechas seleccionado.

**recomendarPesoInicial(id\_usuario, id\_ejercicio):** Propone un peso inicial para un ejercicio concreto basándose en el histórico de series del usuario.

**recomendarDesdeSerieActual(id\_entrenamiento, id\_ejercicio):** Analiza la última serie realizada en ese entrenamiento y ejercicio y, en función de las repeticiones y de ciertas constantes internas, sugiere si se debe subir, mantener o bajar el peso en la siguiente serie.

#### Diagrama de casos de uso:

En este apartado se describen los casos de uso del sistema, los cuales representan las distintas interacciones que pueden realizar los actores con la aplicación.

Estos casos permiten identificar las funcionalidades principales del sistema y comprender cómo los distintos tipos de usuarios (administrador y usuario básico) se relacionan con los diferentes módulos de la aplicación.

A partir de ellos se desarrollan las especificaciones detalladas y se establecen las bases para el diseño posterior de la aplicación.

El diagrama general de casos de uso se encuentra incluido en el apartado de '[Anexos](#)', donde se puede consultar la representación gráfica completa de las relaciones entre actores y funcionalidades.



### Especificaciones de casos de uso:

CU-01	Registrarse		
Descripción	El usuario crea una cuenta en el sistema.		
Actores	Usuario básico		
Precondición	El usuario no debe tener una cuenta registrada.		
Secuencia normal	<b>Acción</b>		
	Paso	Sistema	Actor
	1		El usuario pulsa “Registrarse” desde la página inicial.
	2	El sistema muestra el formulario de registro.	
	3		El usuario completa los datos y envía la solicitud.
	4	El sistema valida los datos y guarda la cuenta con estado “pendiente”.	
	5	El sistema muestra mensaje indicando que la cuenta será verificada antes de poder iniciar sesión.	
Postcondición	La solicitud de registro queda almacenada en el sistema con estado “pendiente”.		
Excepciones	Paso	Acción	
	3	El correo ya está registrado → el sistema muestra un mensaje de error.	
	3	Campos vacíos o inválidos → el sistema solicita corrección.	
	4	Error al guardar la cuenta → el sistema informa de un fallo interno.	

CU-02	Gestionar usuarios		
Descripción	El administrador gestiona los usuarios registrados, pudiendo aprobar solicitudes pendientes o eliminar usuarios de la plataforma.		
Actores	Administrador		
Precondición	El administrador debe haber iniciado sesión.		
Secuencia normal	<b>Acción</b>		
	Paso	Sistema	Actor
	1		El administrador accede al panel de gestión de usuarios.
	2	El sistema muestra la lista de usuarios registrados con su estado (activo o pendiente)	
	3		El administrador selecciona un usuario y elige la acción deseada: aprobar o eliminar.
	4	El sistema solicita confirmación antes de aplicar los cambios.	
	5		El administrador confirma la acción.
	6	El sistema actualiza los datos del usuario o elimina el registro, y muestra un mensaje de confirmación.	
Postcondición	El usuario seleccionado queda actualizado o eliminado según la acción realizada.		
Excepciones	Paso	Acción	
	3	No se selecciona ningún usuario → el sistema muestra un aviso.	
	4	El administrador cancela la acción → no se realizan cambios.	
	6	Error en la base de datos → el sistema informa del fallo y no aplica los cambios.	



CU-03	Iniciar sesión		
Descripción	El usuario accede con sus credenciales.		
Actores	Usuario básico.		
Precondición	Estar registrado.		
Secuencia normal	Acción		
	Paso	Sistema	Actor
	1		El usuario pulsa "Iniciar sesión"
	2	El sistema muestra el formulario de login.	
	3		El usuario introduce email y contraseña.
Postcondición	4	El sistema valida credenciales y abre panel personal	
	Sesión iniciada.		
Excepciones	Paso	Acción	
	4	Credenciales inválidas → mensaje de error y permanece en login.	

CU-04	Crear ejercicio		
Descripción	Alta de un ejercicio en el catálogo personal.		
Actores	Usuario básico.		
Precondición	Sesión iniciada.		
Secuencia normal	Acción		
	Paso	Sistema	Actor
	1		El usuario accede a "Ejercicios" y pulsa "Nuevo ejercicio".
	2	El sistema muestra el formulario	
	3		El usuario completa y guarda.
Postcondición	4	El sistema valida y registra el ejercicio.	
	El ejercicio aparece en el listado personal.		
Excepciones	Paso	Acción	
	4	Datos inválidos o faltantes → el sistema informa y no guarda.	

CU-05	Eliminar ejercicio		
Descripción	Modificación o baja de un ejercicio propio.		
Actores	Usuario básico.		
Precondición	Sesión iniciada, existencia de ejercicios propios.		
Secuencia normal	Acción		
	Paso	Sistema	Actor
	1		El usuario abre "Ejercicios".
	2	Selecciona un ejercicio y pulsa "Eliminar".	
Postcondición	3		El sistema actualiza/elimina el registro y refresca el listado.
	Catálogo actualizado.		
Excepciones	Paso	Acción	
	3	Validación fallida en edición → se muestran errores y no se guarda.	



CU-06	Gestionar catálogo de ejercicios		
Descripción	Crear, editar o eliminar ejercicios del catálogo general de la aplicación.		
Actores	Administrador		
Precondición	El administrador debe haber iniciado sesión.		
Secuencia normal	Acción		
	Paso	Sistema	Actor
	1		Accede a "Gestionar ejercicios".
	2	Muestra el listado con opciones "Crear" y "Eliminar".	
	3		Selecciona una acción y completa el formulario o confirma la eliminación.
	4	Valida los datos y actualiza el catálogo.	
Postcondición	Catálogo actualizado.		
Excepciones	Paso	Acción	
	6	Nombre duplicado en catálogo → el sistema informa y no guarda.	
	6	Error en la base de datos → el sistema informa del fallo y no aplica los cambios.	

CU-07	Crear entrenamiento		
Descripción	Alta de un entrenamiento por fecha, con nombre y notas.		
Actores	Usuario básico.		
Precondición	Sesión iniciada		
Secuencia normal	Acción		
	Paso	Sistema	Actor
	1		El usuario va a "Entrenamientos" → "Nuevo entrenamiento".
	2	El sistema muestra el formulario para introducir el nombre del entrenamiento	
	3		El usuario introduce nombre y pulsa añadir ejercicio.
	4	El sistema registra el entrenamiento dándole la fecha del día en curso y abre su detalle.	
Postcondición	Entrenamiento creado		
Excepciones	Paso	Acción	
	4	Error en la base de datos → el sistema informa del fallo y no aplica los cambios.	

CU-08	Editar/Eliminar entrenamiento		
Descripción	Modificar o eliminar un entrenamiento.		
Actores	Usuario básico.		
Precondición	Sesión iniciada, entrenamiento existente.		
Secuencia normal	Acción		
	Paso	Sistema	Actor
	1		Desde el listado, seleccionar entrenamiento. Pulsar "Editar" (modificar campos) o "Eliminar".
Postcondición	2	El sistema guarda cambios o borra	
	Entrenamiento actualizado o eliminado.		
Excepciones	Paso	Acción	
	2	Validación fallida en edición → se muestran errores y no se guarda.	



CU-09	Añadir ejercicio a un entrenamiento		
Descripción	Incorporar un ejercicio del catálogo a un entrenamiento.		
Actores	Usuario básico.		
Precondición	Sesión iniciada, entrenamiento creado.		
Secuencia normal	Acción		
	Paso	Sistema	Actor
	1		El usuario pulsa añadir ejercicio
	2	El sistema muestra selector de ejercicios	
	3		El usuario elige uno y confirma.
Postcondición	4 El sistema lo vincula al entrenamiento.		
	El ejercicio aparece en el entrenamiento para registrar series.		
Excepciones	Paso	Acción	
	4	Error en la base de datos → el sistema informa del fallo y no aplica los cambios.	

CU-010	Registrar series en un ejercicio de un entrenamiento		
Descripción	Anotar series con repeticiones, peso y descanso.		
Actores	Usuario básico.		
Precondición	Sesión iniciada, CU-09 completado.		
Secuencia normal	Acción		
	Paso	Sistema	Actor
	1		El usuario pulsa "Añadir serie" e introduce repeticiones, peso y descanso.
	2	El sistema muestra la serie vinculada al ejercicio	
	Postcondición		
Series registradas en el entrenamiento			
Excepciones	Paso	Acción	
	1	Campos inválidos (vacíos) → error y no se guarda.	

CU-011	Editar/Eliminar series		
Descripción	Modificar o eliminar series registradas.		
Actores	Usuario básico.		
Precondición	Sesión iniciada, CU-10 completado.		
Secuencia normal	Acción		
	Paso	Sistema	Actor
	1		El usuario edita el formulario o pulsa en eliminar serie
	2	El sistema actualiza o elimina la serie	
	Postcondición		
Series actualizadas.			
Excepciones	Paso	Acción	
	2	Error en la base de datos → el sistema informa del fallo y no aplica los cambios.	



CU-012	Consultar métricas		
Descripción	Visualizar estadísticas		
Actores	Usuario básico.		
Precondición	Sesión iniciada, datos previos registrados.		
Secuencia normal	Acción		
	Paso	Sistema	Actor
	1		El usuario entra en "Métricas".
Postcondición	El sistema calcula y muestra gráficos/resúmenes.		
	El usuario visualiza su progreso.		
	Excepciones	Paso	Acción
Excepciones		2	Sin datos → el sistema muestra un estado vacío y sugiere registrar entrenamientos.

CU-013	Filtrar métricas		
Descripción	Filtrar métricas por rango de fechas y/o por ejercicio.		
Actores	Usuario básico.		
Precondición	Sesión iniciada, datos previos registrados.		
Secuencia normal	Acción		
	Paso	Sistema	Actor
	1		El usuario selecciona fechas y ejercicio (opcional).
Postcondición	El sistema recalcula y muestra resultados.		
	Métricas filtradas.		
	Excepciones	Paso	Acción
Excepciones		1	Sin datos → mensaje de "sin resultados".

CU-014	Recomendación de progresión		
Descripción	Ver sugerencias de incremento/mantenimiento/decremento según cumplimiento previo.		
Actores	Usuario básico.		
Precondición	Sesión iniciada, existencia de histórico de series para el ejercicio.		
Secuencia normal	Acción		
	Paso	Sistema	Actor
	1		El usuario selecciona ejercicio en un entrenamiento nuevo y solicita "ver recomendación"
Postcondición	El sistema analiza el histórico (objetivo de reps/series) y propone la progresión (p. ej., +2,5 kg si completó todas las series al objetivo).		
	Recomendación mostrada para la siguiente sesión.		
	Excepciones	Paso	Acción
Excepciones		1	Datos insuficientes → El sistema no muestra "ver recomendación"



CU-015	Modificar datos personales		
Descripción	El usuario accede a la sección "Perfil" y actualiza sus datos personales (nombre, email o información visible en su cuenta).		
Actores	Usuario básico.		
Precondición	Sesión iniciada		
Secuencia normal	<b>Acción</b>		
	Paso	Sistema	Actor
	1		El usuario accede al menú "Perfil".
	2	El sistema muestra un formulario con los datos personales actuales.	
	3		El usuario modifica los campos y pulsa en guardar cambios.
Postcondición	Los datos personales del usuario quedan actualizados en la base de datos.		
Excepciones	Paso	Acción	
	3a.	Algún campo no es válido (Email duplicado, campos vacíos). el sistema muestra un mensaje de error y no guarda los cambios.	
	3b.	Error inesperado en la base de datos: se muestra un mensaje de fallo y se indica al usuario que lo intente más tarde.	

CU-016	Cerrar sesión		
Descripción	El usuario finaliza manualmente su sesión en la aplicación.		
Actores	Usuario básico.		
Precondición	Sesión iniciada		
Secuencia normal	<b>Acción</b>		
	Paso	Sistema	Actor
	1		El usuario accede al menú "Perfil" y pulsa en cerrar sesión.
	2	El sistema destruye la sesión del usuario activo	
	3	El sistema redirige al usuario a la pagina de inicio de sesión	
Postcondición	La sesión del usuario se cierra correctamente y el sistema vuelve a un estado sin autenticación.		
Excepciones	Paso	Acción	
	2a.	No se puede destruir la sesión por un error del servidor → el sistema muestra un mensaje de error y permanece en la misma pantalla.	

Cada especificación de caso de uso está asociada al requisito funcional que le da origen. De este modo, el RF1 se relaciona con el CU-01, el RF2 con el CU-02, y así sucesivamente.



## 7. Diseño del proyecto

En la fase de diseño, el primer paso consistió en la creación de un logotipo para nuestra aplicación. Para ello, utilizamos la herramienta de inteligencia artificial de Canva, utilizando los conocimientos adquiridos en el módulo optativo de inteligencia artificial del ciclo formativo. Gracias a esta base teórica, fue posible elaborar un prompt que permitió generar las bases de nuestro logo.

Una vez seleccionada la idea principal del logotipo, se procedió a su posterior edición en Figma, donde ajustamos los vectores y creamos varias versiones de nuestro logo final.



Ilustración 13 Logotipo de MyWorkoutLog. Elaboración Propia: Canva AI

Tras conseguir el logotipo pasamos a definir la paleta cromática de nuestra aplicación, para ello se utilizó la herramienta web [Coolors.co](#), que permite generar combinaciones de colores equilibradas y accesibles. A partir de los colores seleccionados y con ayuda de Adobe Photoshop se crearon variaciones tonales destinadas a diferenciar estados interactivos como botones activos, deshabilitados o en hover. Además, se añadieron colores semánticos para representar estados del sistema como éxito, advertencia, error e información.



Ilustración 14: Gama cromática principal del sistema. Elaboración propia, Coolors, Adobe Illustrator.



Todas estas variaciones de color se guardaron como variables en Figma con la intención de guardarlas posteriormente también en CSS, con el objetivo de mantener una coherencia visual en todas las pantallas y facilitar la reutilización de la paleta durante la fase de implementación.

#### Mockups/Wireframes:



Ilustración 15: Algunos de los Mockups de MyWorkoutLog. Elaboración Propia: Figma.

Para comenzar el diseño de la aplicación, se tomó como punto de partida el conjunto de requisitos funcionales y los casos de uso definidos en la fase de análisis. Esto permitió orientar el diseño exclusivamente hacia las funcionalidades necesarias, evitando incorporar elementos que no aportaran valor al sistema. Para elaborar todas las pantallas se utilizó la herramienta Figma, que ofrece un entorno de trabajo vectorial, componentes reutilizables proporcionados por una amplia comunidad y un sistema de Auto Layout muy similar al comportamiento de flexbox en CSS. Gracias a estas características, fue posible crear un diseño escalable, coherente y fácilmente trasladable al desarrollo del front-end.

El proceso de diseño siguió la estrategia mobile-first, priorizando la creación de las pantallas para dispositivos móviles, ya que representan el uso principal previsto para la aplicación. Una vez completadas las versiones móviles, se adaptaron progresivamente a resoluciones mayores, garantizando así una correcta visualización en equipos de escritorio y evitando tener que simplificar posteriormente diseños demasiado complejos.

Todos los wireframes generados durante este proceso de diseño se encuentran disponibles en el apartado de Anexos, donde pueden consultarse en tamaño completo.



### Base de datos:

En primer lugar, para la creación de la base de datos se utilizó XAMPP, con el cual desde su panel de control pudimos iniciar los servicios de Apache y MySQL para acceder a la herramienta de administración phpMyAdmin desde nuestro localhost.

Desde la interfaz gráfica de phpMyAdmin se creó la base de datos “myworkoutlog” y a partir del modelo relacional diseñado previamente, se definieron las tablas mediante instrucciones SQL, incorporando en ellas la definición de claves primarias, claves foráneas y las restricciones necesarias para garantizar la integridad de los datos.

En estas claves foráneas se utilizaron reglas de integridad referencial mediante ON DELETE CASCADE para que, al eliminar un registro principal (por ejemplo, un usuario o un entrenamiento), se eliminases automáticamente los registros relacionados (como sus entrenamientos o series asociadas), evitando datos huérfanos y manteniendo la coherencia de la base de datos.

Además, para poblar la tabla ejercicios con un conjunto inicial de ejercicios globales de fuerza se partió de un recurso externo alojado en GitHub por el autor Dacaramo, publicado bajo licencia MIT, que incluía una base de datos de ejercicios en una de sus hojas de cálculo. A partir de dicha hoja se adaptó la estructura de los datos para ajustarla a la tabla ejercicios, utilizando únicamente la información relevante para la aplicación (nombre del ejercicio y grupo muscular). El proceso consistió en exportar la hoja de ejercicios a un fichero CSV y, posteriormente, importar dicho fichero a MySQL desde phpMyAdmin.

### Diseño de la arquitectura de la aplicación:

La aplicación se ha desarrollado siguiendo el patrón de arquitectura Modelo–Vista–Controlador (MVC), utilizando el archivo index.php, situado en la raíz del proyecto como único punto de entrada. Este archivo actúa como front controller y centraliza el enrutamiento de todas las peticiones. Los controladores actúan como intermediarios entre los modelos y las vistas, reciben las peticiones del usuario (datos enviados mediante formularios a través de \$\_GET o \$\_POST) aplican las reglas de la aplicación y llaman a los modelos, que son los encargados de acceder a la base de datos para consultar o almacenar la información. Una vez obtenidos y procesados los datos, el controlador selecciona la vista correspondiente y le pasa dicha información para que se muestre al usuario mediante las plantillas HTML de la aplicación.



### Front Controller:

- Comprueba el estado de la sesión mediante `session_status()` y, en caso de que todavía no exista una sesión iniciada, llama a `session_start()`. De esta forma, la aplicación puede trabajar con la información del usuario mediante la variable `$_SESSION`.
- A continuación, carga 3 archivos:
  - `config/Config.php`: Constantes necesarias para establecer la conexión a la base de datos (host, nombre de la base de datos, usuario y contraseña).
  - `config/Conexion.php`: Clase Conexión, responsable de crear el objeto PDO y realizar la conexión con la base de datos.
  - `controller/helpers.php`: incluye funciones globales de la aplicación, como por ejemplo `requireLogin()` (para proteger rutas privadas) o `esAdmin()` (para comprobar si un usuario es administrador). Define una lista blanca (Lista de controladores permitidos por seguridad) Los controladores que no estén en esta lista no estarán permitidos o no serán válidos.
- El front controller lee de la URL los parámetros controller y action mediante `$_GET`. Si no se proporcionan, se utilizan valores por defecto (auth como controlador y login como acción), de modo que, si el usuario accede a `index.php` sin parámetros, se le dirige a la pantalla de inicio de sesión. Después se normalizan estos valores usando expresiones regulares, con este proceso conseguimos limpiar lo que viene de la URL evitando así caracteres peligrosos o intentos de manipulación del enrutamiento.
- A partir del nombre del controlador se construye el nombre de la clase (por ejemplo, `AuthController`, `EntrenamientoController`) y la ruta al archivo donde debe estar definida (`controller/AuthController.php`, etc.). Se comprueba que el archivo existe, y si el archivo existe, se incluye con `require_once` y se crea una instancia de la clase del controlador.
- Por último, se verifica que el método correspondiente a la acción solicitada (por ejemplo, `login`, `crear`, etc.) existe dentro del controlador instanciado. Si existe, se llama a dicho método (`$controller->$action()`) gracias a este mecanismo, todas las peticiones siguen el mismo flujo:

(Front Controller) → Controlador → Modelos → Vista



### Capa Controlador:

AuthController: gestiona el inicio de sesión, el registro y el cierre de sesión de los usuarios.

DashboardController: muestra la pantalla de inicio con un resumen de la actividad reciente.

EntrenamientoController: se encarga del listado, creación, edición, eliminación y visualización de entrenamientos.

SerieController: gestiona las series asociadas a cada entrenamiento.

EjercicioController: controla el catálogo de ejercicios y los ejercicios personales del usuario.

MetricsController: prepara los datos necesarios para la sección de métricas y estadísticas.

PerfilController: permite al usuario gestionar su perfil y su foto.

AdminController: agrupa las funcionalidades del panel de administración (gestión de usuarios y de ejercicios globales).

El funcionamiento general de un controlador es el siguiente:

Recibe la petición que ha pasado previamente por el front controller (por ejemplo, “listar entrenamientos” o “crear una nueva serie”).

Lee y valida los datos de entrada provenientes de \$\_GET o \$\_POST.

Llama a los modelos correspondientes para consultar o modificar la información en la base de datos.

Prepara los datos que necesita la vista (listas de entrenamientos, series, métricas, mensajes de error o de confirmación, etc.).

Indica qué archivo de vista debe mostrarse asignando su ruta a la variable \$vistaContenido y, finalmente, carga el layout general de la aplicación.

Además, muchos controladores utilizan funciones auxiliares definidas en helpers, como requireLogin() (para asegurar que el usuario ha iniciado sesión antes de acceder a determinadas acciones), esAdmin() (para restringir el acceso a las secciones de administración) o normalizarTextoFormulario() para conseguir que todos los textos escritos en formularios empiecen por mayúsculas



### **Capa Modelo:**

#### **Modelo Usuario:**

Representa a los usuarios de la aplicación y gestiona la información relacionada con su cuenta (nombre, correo electrónico, contraseña cifrada, rol, estado, etc.). Incluye métodos para registrar nuevos usuarios, buscar usuarios por id o por email, actualizar sus datos y validar la contraseña utilizando funciones de hash. Este modelo es utilizado, por ejemplo, durante el proceso de inicio de sesión y en el panel de administración de usuarios.

#### **Modelo Entrenamiento:**

Representa cada entrenamiento realizado por un usuario. Almacena datos como la fecha, el nombre del entrenamiento y las notas asociadas. Proporciona métodos para crear, actualizar, eliminar y listar entrenamientos de un usuario, así como para buscar un entrenamiento concreto comprobando que pertenece al usuario autenticado. De esta forma se garantiza que cada usuario solo pueda acceder a sus propios entrenamientos.

#### **Modelo Ejercicio:**

Gestiona el catálogo de ejercicios disponibles en la aplicación. Permite trabajar tanto con ejercicios globales (visibles para todos los usuarios) como con ejercicios personales creados por un usuario concreto. Incluye métodos para crear y eliminar ejercicios, listar ejercicios filtrando por usuario, nombre o grupo muscular y obtener los grupos musculares existentes. Este modelo se utiliza en las pantallas de selección de ejercicio y en el perfil del usuario.

#### **Modelo Serie:**

Representa las series realizadas dentro de un entrenamiento (peso, repeticiones, descanso, notas, etc.). Además de las operaciones básicas de creación, actualización, borrado y listado de series, este modelo incorpora una parte de la lógica de negocio de la aplicación, como el cálculo del número total de series en un periodo, la obtención de los ejercicios más frecuentes, el ejercicio con mayor peso levantado o las recomendaciones para subir o bajar peso en función de las repeticiones realizadas. Estos métodos se utilizan sobre todo en la sección de métricas y en la pantalla de detalle de los entrenamientos.



### Capa Vista - Sistema de layout para la parte interna de MyWorkoutLog:

La capa Vista se encarga de la presentación de la información al usuario. Las vistas están organizadas en la carpeta view/, separadas por módulos (aplicación, autenticación, administración y vistas parciales reutilizables).

En las vistas internas de la aplicación (inicio, entrenamientos, métricas, perfil, panel de administración, etc.) se utiliza un layout común definido en view/layout.php. Este layout mediante require incluye los elementos que se repiten en todas las páginas del sistema:

- La cabecera interna (header2.php). Declara <!DOCTYPE html>, las etiquetas <html>, <head>, la ruta de nuestros estilos CSS, los metadatos y la apertura de <body>, además muestra la barra superior con el logotipo y la fotografía del usuario autenticado.
- La barra de navegación (nav.php) o el menú lateral (aside.php) mediante las funciones auxiliares definidas en helpers.php, muestra las opciones de navegación correspondientes según el rol del usuario. Además, según el tamaño de la pantalla, utiliza el menú lateral, o la barra de navegación.
- El pie de página (footer.php), donde se cargan los scripts JavaScript necesarios y se cierran las etiquetas principales del documento.

En la zona central de la página se define un contenedor <main class="main\_general"> donde se inserta dinámicamente el contenido específico de cada pantalla mediante la variable \$vistaContenido

Cada controlador, tras procesar la petición, asigna a \$vistaContenido la ruta de la vista que corresponde (por ejemplo, el listado de entrenamientos, la pantalla de métricas o el perfil de usuario) y a continuación carga el layout. De este modo, la aplicación reutiliza siempre la misma estructura general (cabecera, navegación/sidebar y pie) y únicamente cambia el contenido central, lo que facilita el mantenimiento del código y ofrece una velocidad de respuesta mayor, al solo tener que cargar la parte central de la aplicación.

```
myworkoutlog > view > layout.php
1  <?php require __DIR__ . '/partials/header2.php'; ?>
2  <?php require __DIR__ . '/partials/nav.php'; ?>
3  <?php require __DIR__ . '/partials/aside.php'; ?>
4  <main class="main_general">
5  |    <?php require $vistaContenido; ?>
6  </main>
7  <?php require __DIR__ . '/partials/footer.php'; ?>
```

Ilustración 16: Sistema de layout para MyWorkoutLog. Elaboración propia, captura de pantalla



### Capa vista - Login y registro:

En cambio, las vistas de autenticación (login y registro) no utilizan este layout interno. En estos casos, la propia vista de autenticación declara la variable del título de la página (por ejemplo, \$tituloPagina = 'Iniciar sesión';) y llama mediante require a un partial específico llamado header.php que genera el encabezado HTML básico y el title a partir de esa variable, es decir, se encarga de imprimir el <!DOCTYPE html>, las etiquetas <html>, <head> con sus metadatos y la apertura de <body>, sin incluir la barra superior con la foto del usuario ni los menús de navegación internos.

A continuación, dentro de la misma vista de autenticación se define únicamente el contenido del formulario (login o registro) y, finalmente, mediante require se incluye el mismo pie de página que para el resto de la aplicación.

### Recursos estáticos: estilos, imágenes y scripts:

La aplicación organiza sus recursos estáticos en la carpeta public/, que contiene los archivos accesibles directamente desde el navegador (hojas de estilo, imágenes y JavaScript).

En public/css/ se encuentra la hoja de estilos principal de la aplicación, donde se definen las reglas de maquetación y la paleta de colores previamente definida mediante variables CSS. Desde este archivo se da prioridad al diseño mobile-first planteado anteriormente, así como a la adaptación a resoluciones de escritorio mediante media queries, gestionando así la distribución del layout y de la visibilidad del nav o aside dependiendo del dispositivo.

La carpeta public/img/ agrupa los recursos gráficos utilizados en la interfaz (logotipos). Dentro de ella se ha creado, además, una subcarpeta llamada perfil/, destinada a almacenar las fotografías de perfil que suben los usuarios.

Cada imagen de perfil se guarda siguiendo una convención basada en el identificador del usuario, de modo que el archivo queda asociado directamente a su id en la base de datos. Esta organización facilita la recuperación de la imagen correspondiente desde las vistas, ya que basta con conocer el identificador del usuario autenticado para mostrar su fotografía en la cabecera y en la vista perfil de la aplicación.



En public/js/ se encuentran los scripts JavaScript utilizados por la aplicación. El uso de JavaScript en nuestra aplicación se ha limitado principalmente a utilidades básicas para así poder cumplir los plazos de entrega. Algunos de sus usos han sido:

- Abrir y cerrar ventanas modales (por ejemplo, formularios emergentes para editar notas o gestionar series).
- Gestionar determinados comportamientos asociados a eventos onchange en algunos formularios, mejorando la interacción sin necesidad de recargar la página completa.

#### Iconos con Lucide Icons

Para la iconografía de la interfaz se ha utilizado la biblioteca Lucide Icons, que proporciona un conjunto de iconos vectoriales ligeros y consistentes. La carga de la biblioteca se realiza mediante un script externo incluido en el layout de la aplicación, de modo que los iconos quedan disponibles en todas las vistas internas.



## 8. Despliegue y pruebas

### **Preparación del Entorno de Producción:**

Debido a los requisitos de la normativa del proyecto, el despliegue de la aplicación se ha llevado a cabo en local mediante la herramienta de desarrollo XAMPP, la cual integra un servidor web Apache, el lenguaje de programación PHP y el sistema gestor de bases de datos MySQL/MariaDB. Este entorno, configurado sobre Windows, nos ha permitido reproducir un escenario similar al que conseguiríamos si la aplicación estuviera alojada en un servidor de hosting y, además, unificar en una única herramienta la gestión del servidor y de la base de datos mediante la interfaz de phpMyAdmin accesible desde <http://localhost/phpmyadmin>.

Gracias a esta configuración y a la metodología iterativo-incremental, ha sido posible ir avanzando entre las distintas fases del proyecto y realizar pruebas a la vez que se desarrollaban nuevas funcionalidades. Al tratarse de una aplicación web desarrollada con lenguajes interpretados (PHP en el servidor y HTML/CSS/JavaScript en el cliente), no ha sido necesario un proceso de compilación previo: el código se ejecuta directamente en el servidor y el resultado se muestra de forma inmediata en el navegador que estemos utilizando.

### **Procedimiento de instalación y despliegue en otro equipo:**

1. Instalar XAMPP y activar los servicios Apache y Mysql en un equipo con sistema operativo Windows.
2. Copiar el proyecto en la carpeta htdocs de XAMPP (por ejemplo: C:\xampp\htdocs\myworkoutlog).
3. Importar la base de datos creada en MySQL mediante phpMyAdmin.
4. Importar los scripts SQL proporcionados con el proyecto:
5. Script de creación de la estructura de tablas.
6. Script de inserción de datos de prueba (usuarios y entrenamientos del usuario María).
7. Configurar el archivo de conexión a la base de datos (host, nombre de la BD, usuario y contraseña) en el fichero de configuración de la aplicación.
8. Iniciar los servicios de Apache y MySQL desde el panel de control de XAMPP.
9. Acceder a la aplicación desde el navegador mediante la URL <http://localhost/myworkoutlog> e iniciar sesión con alguno de los usuarios de prueba.



### Seguridad básica y copias de seguridad:

Aunque la aplicación se ejecuta en un entorno local y no está expuesta públicamente en Internet, se han tenido en cuenta una serie de medidas básicas de seguridad:

- Las contraseñas de los usuarios se almacenan mediante la función password\_hash, evitando guardar credenciales en texto plano.
- El acceso a las funcionalidades se controla mediante un sistema de roles, diferenciando entre usuario básico y administrador, lo que restringe el acceso a determinadas secciones de gestión.
- La sesión del usuario se valida en los controladores antes de permitir el acceso a las zonas privadas de la aplicación.
- En las operaciones con la base de datos se utilizan consultas preparadas para reducir el riesgo de inyecciones SQL.

En cuanto a las copias de seguridad, el proyecto se encuentra versionado mediante Git en un repositorio local, lo que permite recuperar estados anteriores del código en caso de incidencia. Además, la información almacenada en la base de datos puede ser exportada periódicamente desde phpMyAdmin a un archivo .sql, que actúa como copia de seguridad de los datos.

En un hipotético despliegue futuro en un servidor de hosting, sería necesario complementar estas medidas con la configuración de certificados SSL (HTTPS) para cifrar las comunicaciones.

### Usuarios y datos de prueba utilizados:

De cara a la realización de las pruebas de caja negra, se definió un conjunto controlado de usuarios y datos de prueba. En primer lugar, se crearon los siguientes usuarios mediante scripts SQL importados en phpMyAdmin:



Email	Rol	Contraseña
<a href="mailto:maria@myworkoutlog.com">maria@myworkoutlog.com</a>	Usuario	proyectodedaw
<a href="mailto:cristian@myworkoutlog.com">cristian@myworkoutlog.com</a>	Admin	proyectodedaw

El usuario María se ha utilizado como caso representativo de usuario estándar, mientras que la cuenta de administrador ha permitido verificar la correcta visualización de menús y zonas restringidas a este rol.

Además, se utilizaron herramientas de inteligencia artificial para generar de forma automatizada dos meses completos de entrenamientos asociados al usuario María, incluyendo series con ejercicios, fechas, series, repeticiones, descansos y pesos. Estos datos se volcaron en la base de datos mediante un script SQL importado desde phpMyAdmin, lo que ha permitido disponer de un volumen de información suficientemente amplio y realista para validar listados, filtros y cálculos de métricas dentro de la aplicación.



### Plan de pruebas de caja negra:

Las pruebas realizadas sobre la aplicación han seguido un enfoque de caja negra, tomando como referencia los requisitos funcionales de la misma. A continuación, se describen algunos de los casos de prueba más representativos, cubriendo todos los requisitos RF1–RF16.

RF Asociado/ID de prueba	Casos de prueba
RF01 / P-01	<b>Descripción de la prueba:</b> Registrar un nuevo usuario completando el formulario <b>Resultado esperado:</b> El sistema valida los datos, crea el usuario en estado pendiente/no activo y muestra un mensaje de confirmación del registro. <b>Resultado:</b> Correcto
RF02 / P-02	<b>Descripción de la prueba:</b> Acceder con el usuario administrador y aprobar un usuario recién registrado. <b>Resultado esperado:</b> El administrador ve el listado de usuarios pendientes, aprueba al usuario y este pasa a estado activo (puede iniciar sesión). <b>Resultado:</b> Correcto
RF02 / P-03	<b>Descripción de la prueba:</b> Acceder con el usuario administrador y eliminar un usuario existente. <b>Resultado esperado:</b> El sistema elimina el usuario seleccionado y deja de aparecer en el listado. No puede volver a iniciar sesión. <b>Resultado:</b> Correcto
RF03, RF16 / P-04	<b>Descripción de la prueba:</b> Iniciar sesión con un usuario activo (por ejemplo, María) y posteriormente cerrar sesión. <b>Resultado esperado:</b> El sistema permite el acceso al panel personal. Tras pulsar “Cerrar sesión”, la sesión se destruye y la aplicación redirige a la pantalla de inicio/login. <b>Resultado:</b> Correcto.



RF Asociado/ID de prueba	Casos de prueba
<b>RF04, RF05 / P-05</b>	<p><b>Descripción de la prueba:</b> Crear un ejercicio en el catálogo personal del usuario y, a continuación, eliminarlo.</p> <p><b>Resultado esperado:</b> El nuevo ejercicio aparece en el listado personal. Al eliminarlo, desaparece del catálogo del usuario y no puede utilizarse en nuevos entrenamientos.</p> <p><b>Resultado:</b> Correcto</p>
<b>RF06 / P-06</b>	<p><b>Descripción de la prueba:</b> Acceder con el usuario administrador y crear un ejercicio en el catálogo general; después eliminarlo.</p> <p><b>Resultado esperado:</b> El ejercicio se añade al catálogo general y está disponible para todos los usuarios. Al eliminarlo, deja de mostrarse en los listados de ejercicios.</p> <p><b>Resultado:</b> Correcto</p>
<b>RF07, RF08 / P-07</b>	<p><b>Descripción de la prueba:</b> Crear un entrenamiento con nombre y notas opcionales, editarlo y posteriormente eliminarlo.</p> <p><b>Resultado esperado:</b> El entrenamiento se guarda correctamente y aparece en el listado. Tras editarlo, se actualizan nombre/notas. Al eliminarlo, desaparece del listado.</p> <p><b>Resultado:</b> Correcto</p>
<b>RF09 / P-08</b>	<p><b>Descripción de la prueba:</b> Añadir varios ejercicios al entrenamiento creado para un día concreto.</p> <p><b>Resultado esperado:</b> Los ejercicios seleccionados quedan asociados al entrenamiento y se muestran en su detalle (pantalla de entrenamiento).</p> <p><b>Resultado:</b> Correcto.</p>



RF Asociado/ID de prueba	Casos de prueba
<b>RF10, RF11 / P-09</b>	<p><b>Descripción de la prueba:</b> Registrar varias series para un ejercicio (repeticiones, peso, descanso) y luego editar y eliminar alguna serie.</p> <p><b>Resultado esperado:</b> El sistema guarda las series y las muestra en la tabla correspondiente. Las series editadas se actualizan y las eliminadas desaparecen del listado.</p> <p><b>Resultado:</b> Correcto</p>
<b>RF12, RF13 / P-10</b>	<p><b>Descripción de la prueba:</b> Consultar las métricas generales del usuario y aplicar filtros por rango de fechas y por ejercicio.</p> <p><b>Resultado esperado:</b> Se muestran resúmenes de progreso (totales de series, pesos, ejercicios más frecuentes, etc.). Al aplicar filtros, las métricas se recalcularán correctamente.</p> <p><b>Resultado:</b> Correcto</p>
<b>RF14 / P-11</b>	<p><b>Descripción de la prueba:</b> Registrar varias sesiones de un mismo ejercicio con distinta carga y consultar las recomendaciones de progresión.</p> <p><b>Resultado esperado:</b> El sistema analiza el histórico de series y muestra recomendaciones coherentes (incrementar, mantener o reducir carga) según el rendimiento del usuario.</p> <p><b>Resultado:</b> Correcto</p>
<b>RF15 / P-12</b>	<p><b>Descripción de la prueba:</b> Entrar en el apartado Perfil, modificar datos personales (por ejemplo, nombre o email) y guardar cambios.</p> <p><b>Resultado esperado:</b> Los datos modificados se guardan en la base de datos y, al volver a acceder al perfil o recargar la página, se muestran los valores actualizados.</p> <p><b>Resultado:</b> Correcto.</p>



## 9. Conclusiones

Una vez finalizado este proyecto, y comparando mis diagramas de Gantt llegué a la conclusión de que, al principio, realicé una estimación poco realista de la fase de análisis y de diseño. Además, compatibilizar el desarrollo de este proyecto con un trabajo a jornada completa y otras tres asignaturas ha supuesto terminar dentro del plazo establecido un auténtico reto personal que ha exigido constancia, organización y esfuerzo constante.

Por otro lado, el desarrollo de MyWorkoutLog permitió alcanzar el objetivo principal del proyecto, elaborar una aplicación web que facilitara a los usuarios llevar un control de sus entrenamientos y de su progreso de forma sencilla. A partir de este objetivo general se definieron unos requisitos funcionales muy claros y, gracias al tiempo invertido en la fase de análisis (desde el diagrama Entidad–Relación hasta el diagrama de clases del modelo principal de la aplicación) y en la fase de diseño (con la elaboración de prototipos realistas de las pantallas), el tiempo de implementación se vio drásticamente reducido.

Además, el hecho de maquetar toda la aplicación de manera responsive siguiendo un enfoque mobile first permitió adaptar de forma mucho más sencilla la interfaz a pantallas de mayor tamaño.

A nivel personal y formativo, este proyecto supuso una oportunidad para consolidar mis conocimientos de desarrollo web full stack, abarcando desde el diseño de la base de datos y la lógica de negocio hasta la maquetación, los estilos y la usabilidad de la interfaz. El proceso no fue sencillo, ya que necesitaba documentarme de forma recurrente para realizar el proyecto de la manera más profesional y actualizada posible: seguí tutoriales para aprender a prototipar en Figma, comprendí cómo desarrollar una aplicación siguiendo el modelo MVC y me inicié en el uso de Git para llevar un control de versiones y contar con una copia de seguridad del código.

En definitiva, el desarrollo de MyWorkoutLog me ha permitido comprobar que casi cualquier objetivo resulta alcanzable si se afronta con constancia y planificación, y me ha ayudado a entender que todo lo aprendido en este ciclo formativo es útil a la hora de abordar un proyecto real, que no todo es empezar a escribir código sin planificar las cosas. Ahora solo espero que este proyecto se convierta en el punto de partida sobre el que seguir construyendo y mejorando como desarrollador.



## 10. Vías futuras

La versión actual de MyWorkoutLog cumple con los objetivos planteados de este proyecto, sin embargo, desde las primeras fases de análisis y diseño quedaron sobre la mesa ideas que no han podido implementarse por limitaciones de tiempo y que servirían como base para la evolución futura de la app.

Una de las ideas que se plantearon inicialmente, consistía en conseguir que el laboratorio de entrenamiento fuera completamente personalizable. La intención era permitir que, desde el apartado de entrenamientos, se pudiera modificar el rango de repeticiones que se iba a utilizar en cada ejercicio y, en función de ese rango, el sistema devolviera recomendaciones adaptadas. Sin embargo, esta aproximación incrementaba de forma significativa la complejidad del proyecto, por lo que finalmente se descartó y se optó por construir la lógica de las recomendaciones utilizando un rango fijo de 8 a 12 repeticiones. Esta opción podría reconsiderarse para implementaciones futuras del proyecto. A partir de esta base, también podría considerarse la opción de integrar un sistema más avanzado de recomendaciones apoyado en técnicas de IA, capaces de analizar el histórico de series y progresos del usuario para proponer ajustes de carga de forma todavía más precisa.

Desde el punto de vista de la interfaz y la capa de presentación, también existe margen de mejora. Por falta de tiempo no se ha podido profundizar en determinadas mejoras visuales ni en el uso avanzado de JavaScript. En la versión actual, las validaciones de formularios se realizan principalmente en el servidor y la interacción del lado del cliente es mínima. Tampoco se han incorporado efectos de hover, transiciones ni transformaciones que podrían aportar una experiencia de uso más pulida y moderna. En futuras iteraciones sería recomendable reforzar la capa de JavaScript para añadir validaciones en el lado del cliente.

Otra vía de evolución relevante sería la migración de MyWorkoutLog hacia una aplicación móvil. Para ello, en primer lugar, se aprovecharía la aplicación actual como backend exponiendo una API REST en PHP que devuelva los datos en formato JSON (usuarios, entrenamientos, series, métricas, etc. Sobre esta base, se desarrollarían clientes móviles en tecnologías como React Native o Flutter, que consumirían dicha API y ofrecerían una experiencia de uso optimizada para móviles.



Desde el punto de vista de la seguridad y la experiencia de registro, también existe posibilidad de mejora. En la versión actual, el alta de usuario es sencilla, pero en una versión futura, sería recomendable introducir un sistema de verificación por correo electrónico para que los administradores del sistema indiquen a los usuarios cuando están activos en la plataforma. Además, podría añadirse un sistema de CAPTCHA en el formulario de registro y en el de inicio de sesión con el fin de dificultar el uso de la plataforma por parte de bots.

Por último, se podrían explorar integraciones externas que aporten valor añadido sin modificar la esencia de la aplicación. Algunas posibilidades serían exportar los datos de entrenamientos y series a formatos estándar (CSV o Excel), conectar con aplicaciones de salud para cruzar métricas de fuerza con datos como pasos diarios o frecuencia cardiaca, o incluso integrar un sistema básico de comunidad donde los usuarios puedan compartir progresos de forma opcional.

En resumen, la versión actual de MyWorkoutLog debe entenderse como un primer paso sobre el que construir. Estas mejoras al fin y al cabo no solo aumentarían la utilidad de la aplicación si no también representarían para mi una oportunidad de crecimiento técnico y profesional en el futuro.



## 11. Bibliografía/Webgrafía

- PHP Group. (s. f.). PHP. Recuperado de <https://www.php.net/docs.php>
- Oracle Corporation. (s. f.). MySQL. Recuperado de <https://dev.mysql.com/doc/>
- Mozilla Foundation. (s. f.). MDN Web Docs – HTML, CSS y JavaScript. Recuperado de <https://developer.mozilla.org/>
- Apache Friends. (s. f.). XAMPP. Recuperado de <https://www.apachefriends.org/>
- Git Project. (s. f.). Git –. Recuperado de <https://git-scm.com/>
- Microsoft. (s. f.). Visual Studio Code. Recuperado de <https://code.visualstudio.com/>
- phpMyAdmin. (s. f.). phpMyAdmin. Recuperado de <https://www.phpmyadmin.net/>
- Atlassian.(s. f.). Jira Software. Recuperado de <https://www.atlassian.com/software/jira>
- Lucid Software. (s. f.). Lucidchart. Recuperado de <https://www.lucidchart.com/>
- dbdiagram.io. (s. f.). Dbdiagrams. Recuperado de <https://dbdiagram.io/>
- Figma. (s. f.). Figma – UI design tool. Recuperado de <https://www.figma.com/>
- Colorls. (s. f.). Colorls. Recuperado de <https://colorls.co/>
- Canva. (s. f.). Canva. Recuperado de <https://www.canva.com/>
- Adobe. (s. f.). Adobe Photoshop / Illustrator. Recuperado de <https://www.adobe.com/>
- Lucide. (s. f.). Lucide Icons. Recuperado de <https://lucide.dev/>
- Hevy Workout Tracker. (s. f.). Aplicación para registro de entrenamientos. Recuperado de <https://www.hevyapp.com/>
- Strong Workout Tracker. (s. f.). Aplicación para registro y planificación de rutinas. Recuperado de <https://www.strong.app/>
- JEFIT Workout Planner. (s. f.). Aplicación para planificación y seguimiento de entrenamientos. Recuperado de <https://www.jefit.com/>
- FitNotes. (s. f.). Aplicación para registro de rutinas de entrenamiento. Recuperado de <https://www.fitnotesapp.com/>
- Fitia. (s. f.). Aplicación para control de dieta y nutrición. Recuperado de <https://www.fitia.app/>



BBC News Mundo. (2021). Sin mujeres casadas y con un solo hotel: los fascinantes detalles de las Olimpiadas de la Grecia Antigua. Recuperado de <https://www.bbc.com/mundo/noticias-57829651>

Dacaramo. (s. f.). PlantillaEntrenamiento-Sudor-y-Sangre. GitHub. Recuperado de <https://github.com/Dacaramo/PlantillaEntrenamiento-Sudor-y-Sangre>

OBS Project. (s. f.). Open Broadcaster Software. Recuperado de <https://obsproject.com/>

Microsoft. (s. f.). Microsoft Office. Recuperado de <https://www.microsoft.com/>

Google LLC. (s. f.). Google Play Store: catálogo de aplicaciones móviles. Recuperado de <https://play.google.com/>



## 12. Anexos

### 12.1. Manual de Instalación

Este manual describe los pasos necesarios para instalar y poner en marcha la aplicación web MyWorkoutLog en un entorno local utilizando XAMPP.

El procedimiento está pensado para un entorno Windows 10/11, aunque podría adaptarse fácilmente a otros sistemas operativos que dispongan de un servidor web con PHP y un gestor de bases de datos MySQL/MariaDB.

#### **Contenido del paquete entregado:**

El proyecto se entrega en una carpeta con el siguiente contenido principal:

myworkoutlog/

    index.php → Front Controller de la aplicación (punto de entrada).

    config/

        Config.php → Configuración de conexión a la base de datos (host, nombre BD, usuario y contraseña).

        Conexion.php → Clase encargada de establecer la conexión PDO con MySQL.

    controller/ → Controladores PHP de la aplicación (lógica del lado servidor).

    model/ → Modelos de datos (Usuario, Ejercicio, Entrenamiento, Serie, etc.).

    view/ → Vistas y plantillas de la aplicación.

    public/

        css/css.css → Hoja de estilos principal.

        js/js.js → JavaScript de la parte cliente.

Además del código fuente, se adjunta dos archivos de respaldo para la base de datos :

myworkoutlog.sql

maria\_entrenamientos.sql



### Instalación de XAMPP:

- Acceder a la [página oficial de XAMPP](#) y descargar la versión para Windows.
- Ejecutar el instalador y seguir el asistente con la configuración por defecto.
- Una vez instalado, abrir el panel de control de XAMPP e iniciar los módulos, Apache y Mysql

**Nota:** si Apache no arranca por conflicto de puertos, será necesario cerrar otros servicios que estén usando el puerto 80/443.

### Despliegue del proyecto en XAMPP:

- Localizar la carpeta donde está instalado XAMPP (normalmente C:\xampp).
- Dentro de C:\xampp\htdocs\, crear la carpeta myworkoutlog y copiar el contenido de la carpeta DAW\_Codigo\_CristianMenjibar.zip dentro de ella:
- Resultado esperado: C:\xampp\htdocs\myworkoutlog\
- Comprobar que dentro de myworkoutlog se encuentran los archivos y carpetas indicados en el apartado 3.

### Creación de la base de datos:

- Abrir el navegador y acceder a phpMyAdmin:
- URL: <http://localhost/phpmyadmin>
- En el menú superior, pulsar en “Bases de datos”.
- En el campo Crear base de datos, escribir “myworkoutlog”
- Pulsar en Crear.



### Importar tablas:

- Una vez creada la base de datos myworkoutlog, seleccionarla en el panel izquierdo de phpMyAdmin.
- Ir a la pestaña “Importar”.
- Pulsar en “Seleccionar archivo”.
- Elegir el archivo myworkoutlog.sql incluido en la entrega del proyecto.
- Asegurarse de que el formato es SQL y mantener las opciones por defecto.
- Pulsar en “Importar” y esperar a que se complete la importación.
- Verificar que se han creado correctamente las tablas (usuarios, ejercicios, entrenamientos, series, etc.).
- Después importar el archivo maria\_entrenamientos.sql y seguir el mismo procedimiento.

### Configuración de la conexión a la base de datos:

La aplicación utiliza constantes definidas en config/Config.php para conectarse a la base de datos.

El contenido por defecto es el siguiente:

```
myworkoutlog > config > Config.php
1  <?php
2  const DB_HOST = 'localhost';
3  const DB_NAME = 'myworkoutlog';
4  const DB_USER = 'root';
5  const DB_PASS = '';
6 
```

Ilustración 17: Config.php. Fuente: Elaboración propia

Solo será necesario modificar este archivo si se ha utilizado un nombre de base de datos diferente, se ha utilizado un nombre de base de datos diferente, o se ha configurado un usuario y contraseña distintos a root sin contraseña.

En ese caso, editar Config.php con un editor de texto (por ejemplo, Visual Studio Code) y ajustar los valores correspondientes.



**Puesta en marcha de la aplicación:**

1. Asegurarse de que Apache y MySQL están en ejecución en el panel de control de XAMPP.
2. Abrir el navegador y acceder a la siguiente URL:
3. <http://localhost/myworkoutlog>
4. Si todo es correcto, deberá mostrarse la pantalla de login/registro de MyWorkoutLog.

**Acceso a la aplicación desde el teléfono móvil:**

Para poder visualizar la aplicación en un teléfono móvil conectado a la misma red local, se deben seguir estos pasos:

- En el ordenador, iniciar los servicios Apache y MySQL desde el panel de control de XAMPP.
- Abrir una terminal del sistema (Símbolo del sistema o PowerShell en Windows) y ejecutar el comando “ipconfig”.
- Localizar el apartado “Adaptador de LAN inalámbrica Wi-Fi” y anotar la dirección IPv4 que aparece (por ejemplo, 192.168.1.165).
- Comprobar que el ordenador y el teléfono móvil están conectados a la misma red Wi-Fi.
- En el navegador del teléfono móvil, escribir la siguiente dirección “DIRECCION\_IPV4/myworkoutlog”, por ejemplo “192.168.1.165/myworkoutlog”

Si la configuración es correcta, la aplicación MyWorkoutLog se mostrará en el navegador del teléfono móvil igual que en el ordenador.



## 12.2. Manual de usuario

### Acceso a la aplicación

Al acceder al sitio web, nos encontraremos con un formulario de login. Si disponemos de un usuario activo en la plataforma, podremos iniciar sesión, si no tendremos que proceder a registrarnos pulsando en “¿No tienes cuenta? Regístrate”, situado bajo el formulario de login.

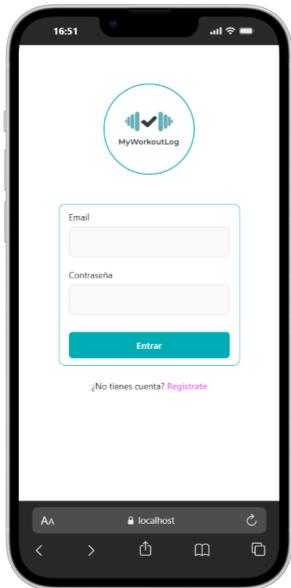


Ilustración 18: Pantalla de login. Fuente: Captura de pantalla

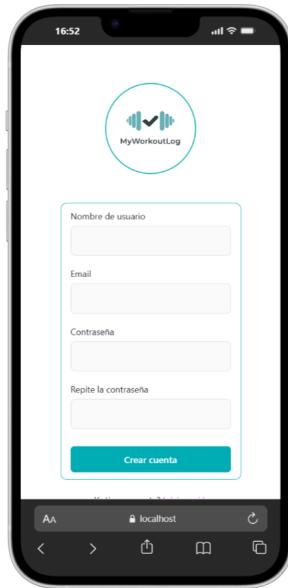


Ilustración 19: Pantalla de registro. Fuente: Captura de pantalla

Nota: Solo es posible acceder a la aplicación si la cuenta de usuario se encuentra en estado activo dentro del sistema. Esto puede conseguirse de dos formas:

- Cambiando manualmente el estado del usuario a “activo” desde phpMyAdmin (ruta habitual: <http://localhost/phpmyadmin>), modificando el campo correspondiente en la tabla de usuarios.
- Iniciando sesión con el usuario administrador del sistema e indicando que el usuario pase a estar activo.



Los datos del usuario administrador son:

Correo electrónico: cristian@myworkoutlog.com

Contraseña: proyectodedaw

Si el objetivo es simplemente probar la aplicación, se recomienda iniciar sesión con el usuario de pruebas, que ya dispone de entrenamientos y sesiones predefinidas:

Correo electrónico: maria@myworkoutlog.com

Contraseña: proyectodedaw

#### Pantalla de inicio (Dashboard):

En la parte inferior o parte lateral (Dependiendo de si la vista es en desktop o mobile) se encuentra la barra de navegación principal, desde la que se puede acceder a las distintas secciones de la aplicación:

- Inicio: acceso al dashboard general.
- Entrenar: gestión de entrenamientos y registro de nuevas sesiones.
- Métricas: visualización de estadísticas y evolución del usuario.
- Perfil: consulta y edición de los datos personales del usuario.

En el área central se muestra el panel principal, compuesto por:

- Un resumen con métricas destacadas del usuario en el último mes.
- Un listado con los últimos 7 entrenamientos con un botón que te permite acceder a cada uno de ellos

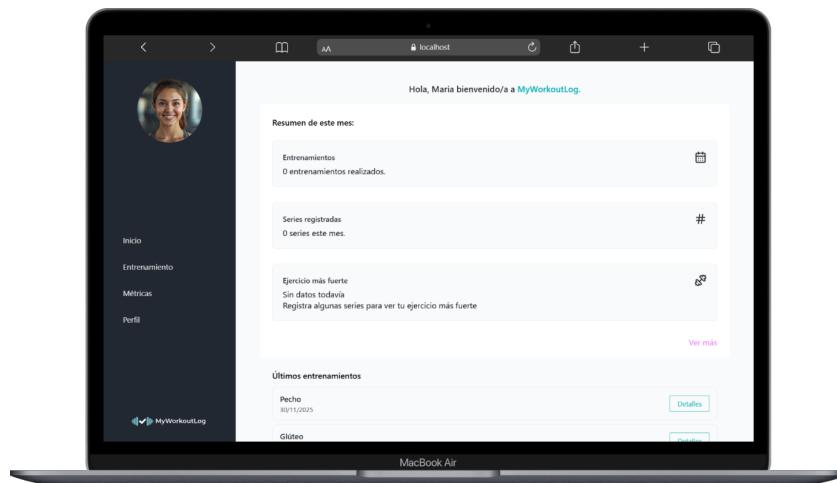


Ilustración 20: Dashboard MyWorkoutLog. Fuente: Captura de pantalla



### Sección entrenar:

La sección Entrenar es el punto de partida para crear y gestionar los entrenamientos del usuario.

Al hacer clic en la opción Entrenar del menú de navegación, se muestra una pantalla con un formulario para iniciar un nuevo entrenamiento, con los siguientes campos:

- Nombre del entrenamiento: campo de texto obligatorio donde se indica el nombre de la sesión (por ejemplo, “Pecho y tríceps”, “Full body”, etc.).
- Notas: campo de texto opcional, con un límite de caracteres, para añadir comentarios breves sobre la sesión (objetivo del día, sensaciones, etc.).

Al llenar este formulario se creará un entrenamiento al que se le asignará automáticamente la fecha actual, después el sistema nos redirigirá automáticamente a la pantalla de selección de ejercicios.

Además, en la parte inferior de esta pantalla aparece el bloque “Últimos entrenamientos” donde se visualiza el nombre del entrenamiento, la fecha y un botón llamado “Detalles”.

Al pulsar en este botón la aplicación abre de nuevo la pantalla de detalle del entrenamiento.

### Seleccionar ejercicios y añadir series:

Tras pulsar “Añadir ejercicio”, la aplicación muestra un listado de ejercicios disponibles y un buscador. Desde esta pantalla el usuario puede:

- Buscar un ejercicio escribiendo su nombre en el campo “Buscar ejercicio”.
- Filtrar por grupo muscular mediante un desplegable (por ejemplo: pecho, espalda, pierna, etc.).
- Crear un nuevo ejercicio pulsando el botón “Nuevo ejercicio”.

Si pulsamos en este último botón el sistema desplegará una ventana modal con un formulario para registrar nuevos ejercicios.

Una vez localizado el ejercicio deseado, el usuario pulsa el botón con el icono “+” para añadirlo al entrenamiento en curso.

Al hacerlo, la aplicación abre la pantalla de detalle del entrenamiento.



### Sección detalle de entrenamiento:

En la pantalla de detalle del entrenamiento, para cada ejercicio añadido se muestra una tarjeta con:

- El nombre del ejercicio.
- Una tabla con las series registradas, con las columnas: Serie, reps, kilos, descanso y notas.
- Una fila para añadir estos registros.
- Un apartado de recomendaciones

Además, cada serie dispone de opciones para:

- Editar notas mediante un botón tipo enlace que abre un pequeño cuadro de diálogo.
- Eliminar la serie más reciente mediante el botón «Eliminar serie».
- Eliminar el ejercicio completo del entrenamiento mediante el botón «Eliminar ejercicio».

### Recomendaciones del “Laboratorio de entrenamiento”:

Debajo de la tabla de series puede mostrarse un bloque de texto con una recomendación automática de peso para la siguiente serie. Estas sugerencias se basan en el historial del usuario con ese ejercicio (peso utilizado y repeticiones realizadas en entrenamientos anteriores).

Cuando todavía no hay historial suficiente, se muestra un mensaje indicando que las recomendaciones se activarán cuando se registren más series dentro de un rango de repeticiones determinado.



Debajo de las tarjetas de los ejercicios se muestran tres botones principales:

- Añadir ejercicio: redirige al selector de ejercicios.
- Ver resumen: despliega un modal con el resumen del entrenamiento.
- Eliminar entrenamiento: permite borrar por completo el entrenamiento actual

Ilustración 21: Modal Notas de la serie. Fuente: Captura de pantalla

Ilustración 22: Detalle de entrenamiento y registro de series. Fuente: Captura de pantalla



### Sección Métricas:

La sección Métricas permite consultar la evolución del usuario a lo largo del tiempo y obtener una visión global de sus entrenamientos.

En la parte superior de la pantalla se muestra un bloque con el texto «Rango» y un desplegable que permite seleccionar el periodo sobre el que se calculan las estadísticas:

- Últimos 30 días
- Mes anterior
- Desde siempre

Al cambiar el filtro, la página se recarga automáticamente y recalcula las métricas para ese rango de fechas.

Debajo de los filtros se muestra un bloque de resumen que incluye, los siguientes indicadores:

- Entrenamientos: número total de entrenamientos realizados en el periodo seleccionado.
- Series registradas: número total de series guardadas.
- Entrenamientos por grupo muscular, (por ejemplo, “Entrenamiento/s de pecho”, “Entrenamiento/s de pierna”, etc.).

A continuación, se muestra la tarjeta “Ejercicio con más fuerza”, donde MyWorkoutLog identifica el ejercicio en el que el usuario ha levantado más peso, en esta tarjeta se incluye:

- El nombre del ejercicio y el peso máximo registrado.
- Un botón para filtrar por grupo muscular (por ejemplo, ver solo el ejercicio más fuerte de “espalda” o de “pierna”).

Si aún no hay datos suficientes, se muestra un mensaje indicando que es necesario registrar más entrenamientos para poder calcular esta métrica.

Después se muestra la sección “Ejercicios más frecuentes” donde se muestra un listado con los ejercicios que el usuario ha utilizado en más entrenamientos dentro del rango seleccionado.



Y para finalizar en la parte inferior de la sección Métricas se muestra un calendario mensual donde se marcan los días en los que el usuario ha entrenado.

El calendario incluye:

- Navegación entre meses.
- Un buscado para filtrar por nombre de entrenamiento o ejercicio.
- Celdas resaltadas para los días con entrenamientos.

### **Sección Perfil:**

La sección Perfil permite al usuario gestionar sus datos personales, la contraseña, la foto de perfil y el catálogo de ejercicios personales.

En el bloque de foto de perfil se muestra:

- La imagen actual del usuario (si existe) o un avatar por defecto.
- Un botón “Añadir nueva foto de perfil”, que abre un selector de archivos para subir una imagen.
- Si ya existe una foto de perfil, también se muestra un botón “Eliminar foto de perfil” que permite borrarla.

En el apartado “Editar datos personales” el usuario puede modificar su nombre y su correo electrónico y su contraseña

### **Catálogo de ejercicios personales:**

En la parte inferior de la página de perfil se encuentra el «Catálogo de ejercicios personales», donde se listan los ejercicios creados por el propio usuario (por ejemplo, variantes específicas o ejercicios que no están en el catálogo global).

Junto a cada ejercicio aparece un botón para eliminarlo, que requiere confirmación previa. Estos ejercicios personales también pueden utilizarse posteriormente al crear entrenamientos desde la sección Entrenar.

Dentro de la página de perfil se incluye el enlace «Cerrar sesión», que permite salir de la aplicación de forma segura.



## 12.3. Glosario

---

### A

#### API REST

Conjunto de reglas que permiten que dos aplicaciones se comuniquen entre sí a través de peticiones HTTP (GET, POST, etc.), devolviendo normalmente datos en formato JSON. · 50

---

### B

#### Backend

Parte de la aplicación que se ejecuta en el servidor y se encarga de la lógica de negocio, el acceso a la base de datos y el procesamiento de las peticiones. · 13

---

### C

#### CRUD

Acrónimo de Create, Read, Update, Delete (Crear, Leer, Actualizar, Borrar). Hace referencia a las operaciones básicas que se realizan sobre los datos. · 6

---

### F

#### Front controller

Patrón de diseño en el que un único punto de entrada (en este caso, index.php) recibe todas las peticiones y decide qué controlador debe atenderlas. Permite centralizar la lógica de enrutamiento y la carga de recursos comunes. · 36

#### Frontend

parte de la aplicación que se ejecuta del lado del cliente · 13

---

### H

#### Helpers

Conjunto de funciones reutilizables que encapsulan tareas comunes y evitan repetir código. · 38

---

### L

#### Layout

Estructura de página reutilizable que define los elementos comunes de la interfaz (cabecera, menú de navegación, contenido principal y pie de página). · 38

---

### R

#### Responsive / Mobile first

Enfoque de diseño web que prioriza la versión para dispositivos móviles y, a partir de esta, adapta el diseño a pantallas más grandes mediante media queries. MyWorkoutLog se ha maquetado siguiendo una estrategia mobile-first. · 49

## 12.4. Índice de imágenes

Ilustración 1: Atletas con llama olímpica en la antigua Grecia. Fuente: <a href="https://www.bbc.com/mundo/noticias-57829651">https://www.bbc.com/mundo/noticias-57829651</a> .....	6
Ilustración 2: Heavy. Fuente Google Play .....	8
Ilustración 3: Strong. Fuente: Google Play.....	8
Ilustración 4: JEFIT. Fuente Google Play.....	8
Ilustración 5: Fitnotes. Fuente: Google Play.....	8
Ilustración 6: Fitia. Fuente: Google Play.....	8
Ilustración 7 Metodología Iterativa-Incremental. Fuente: Elaboración Propia .....	10
Ilustración 8: Representación de Metodología Kanban. Fuente: Elaboración propia: Canva.....	12
Ilustración 9: Diagrama de Gantt estimado. Fuente: Elaboración propia con Figma	15
Ilustración 10: Diagrama Entidad-Relación (E-R). Fuente: Elaboración propia con Lucidchart.....	19
Ilustración 11: Modelo Relacional. Fuente: Elaboración propia con dbdiagram.....	21
Ilustración 12: Diagrama de clases. Fuente: Elaboración propia con Figma.....	22
Ilustración 13 Logotipo de MyWorkoutLog. Elaboración Propia: Canva AI .....	32
Ilustración 14: Gama cromática principal del sistema. Elaboración propia, Coolors, Adobe Illustrator.....	32
Ilustración 15: Algunos de los Mockups de MyWorkoutLog. Elaboración Propia: Figma.....	33
Ilustración 16: Sistema de layout para MyWorkoutLog. Elaboración propia, captura de pantalla.....	38
Ilustración 17: Config.php. Fuente: Elaboración propia.....	54



Ilustración 18: Pantalla de login. Fuente: Captura de pantalla .....	56
Ilustración 19: Pantalla de registro. Fuente: Captura de pantalla.....	56
Ilustración 20: Dashboard MyWorkoutLog. Fuente: Captura de pantalla .....	57
Ilustración 21 Modal notas. Fuente: Captura de pantalla .....	60
Ilustración 22: Detalle de entrenamiento y registro de series. Fuente: Captura de pantalla.....	60



## 12.5. Imágenes y Diagramas

Diagrama de Gantt estimado:

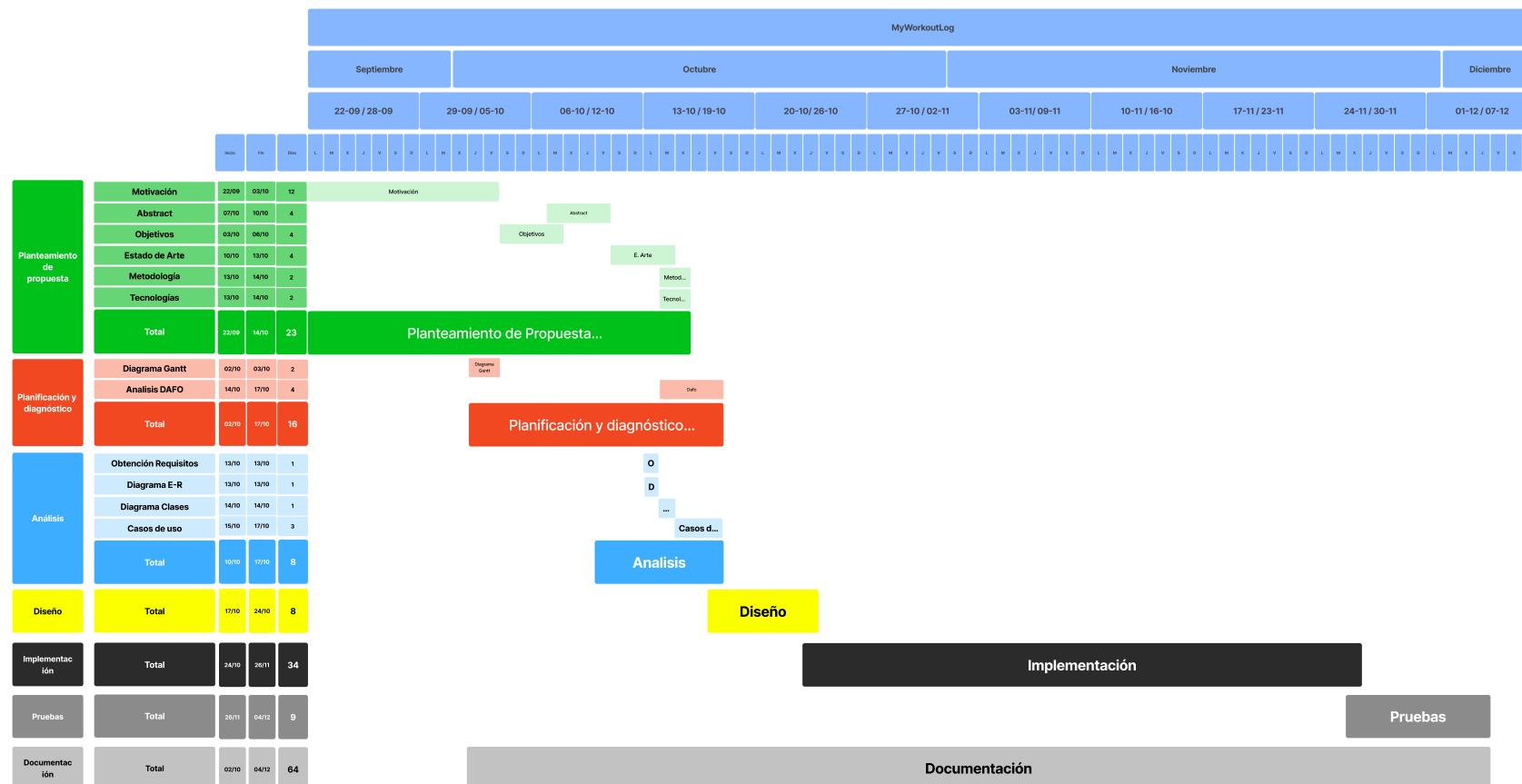




Diagrama de Gantt real:

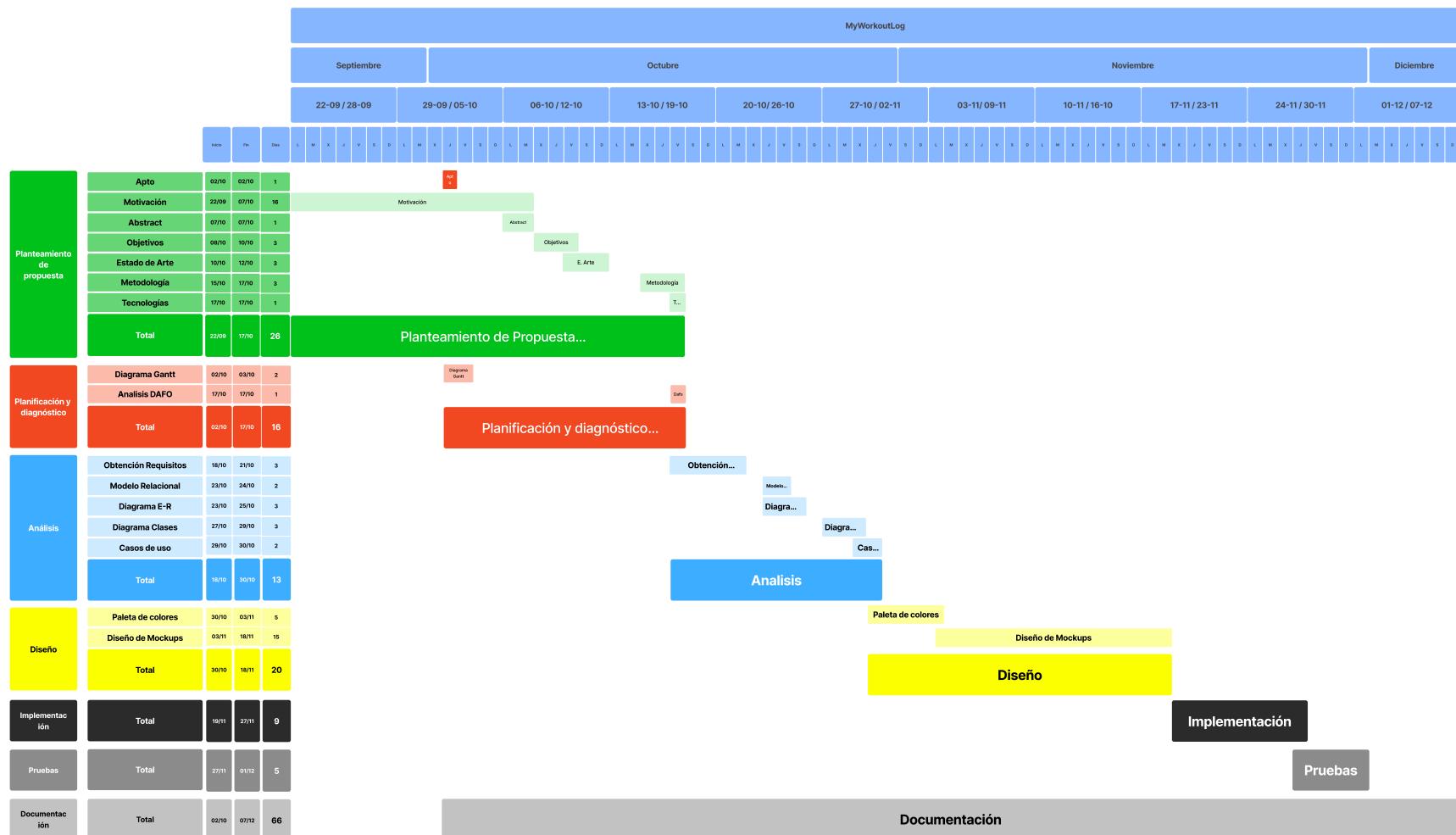
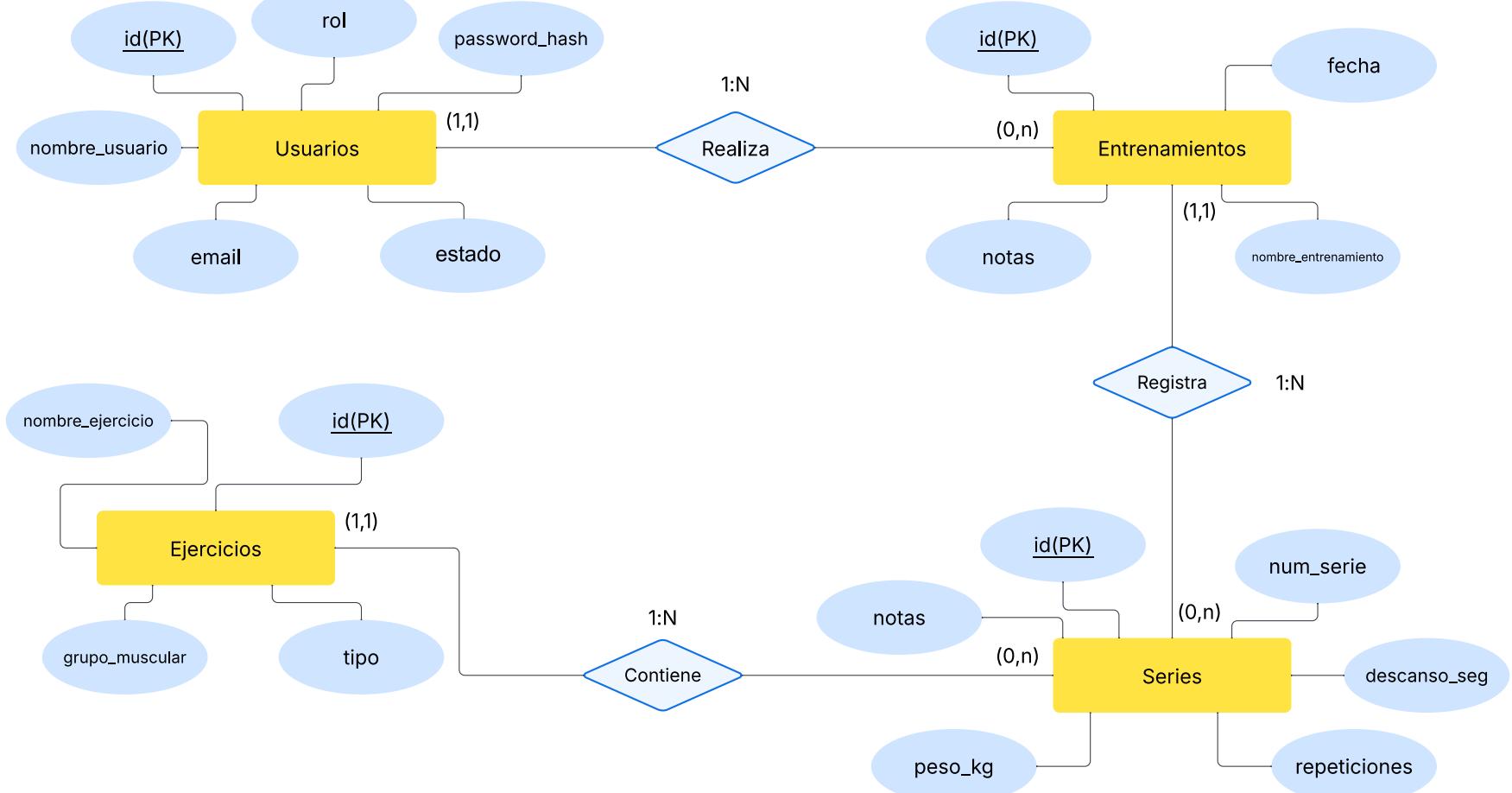




Diagrama entidad-relación:





Modelo relacional:

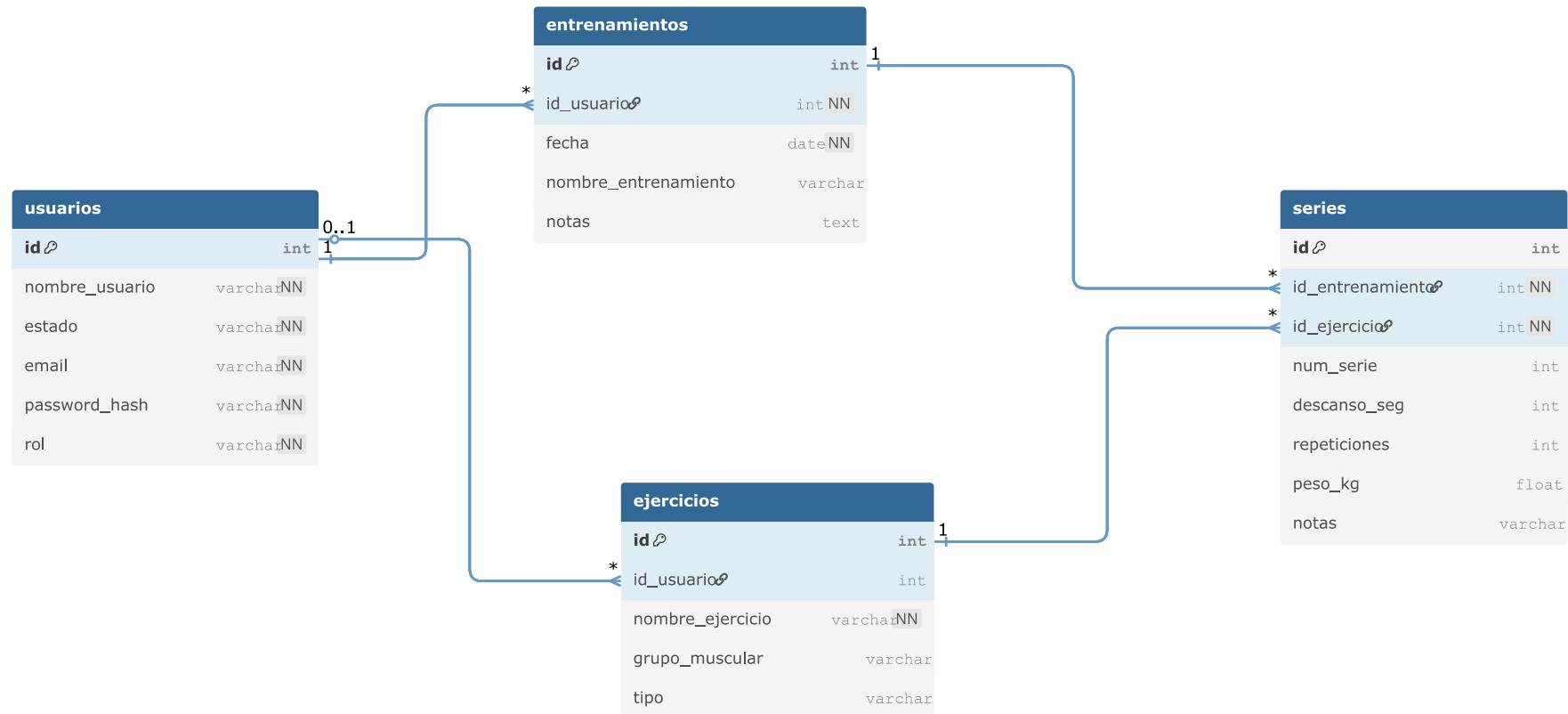




Diagrama de clases:

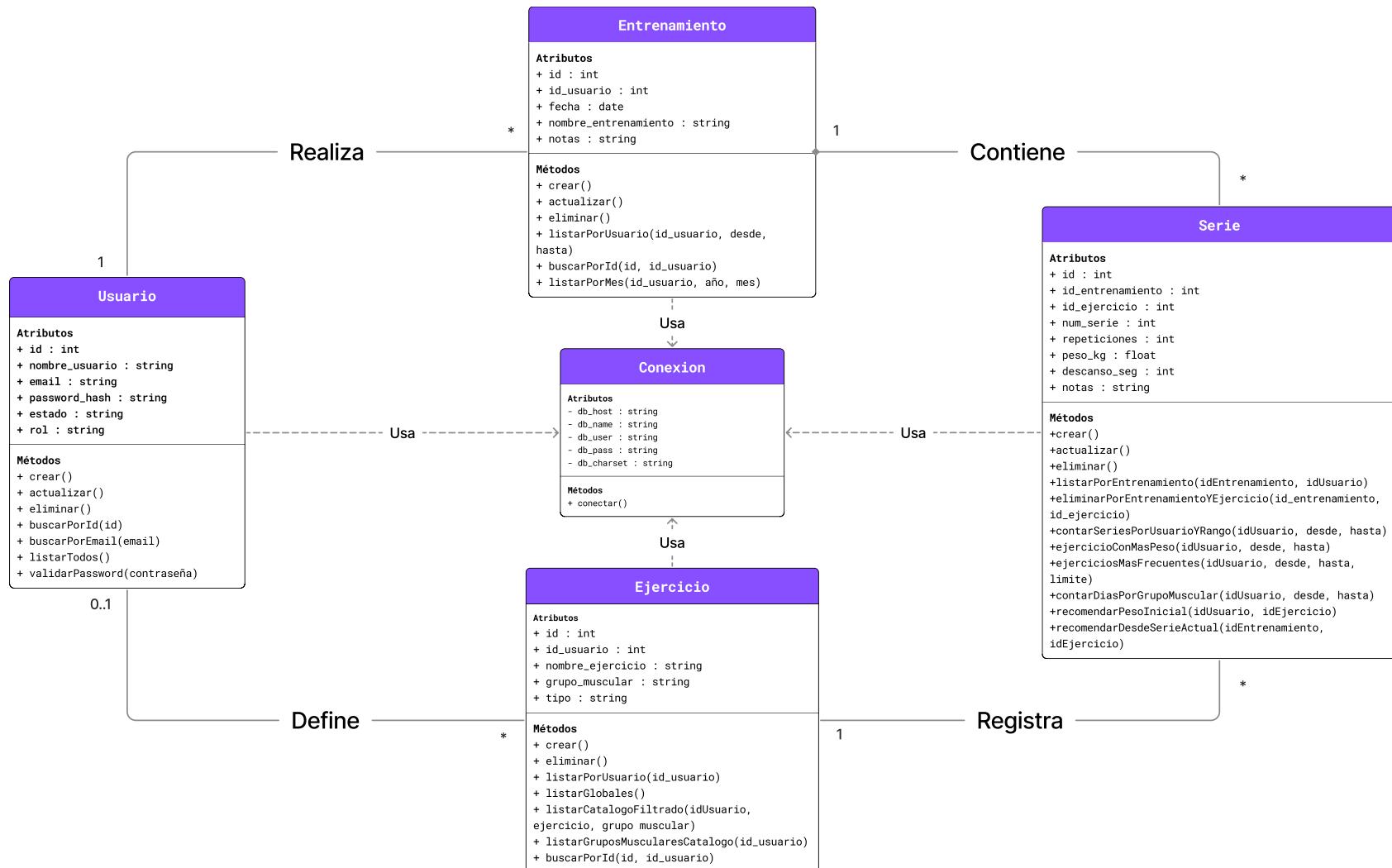
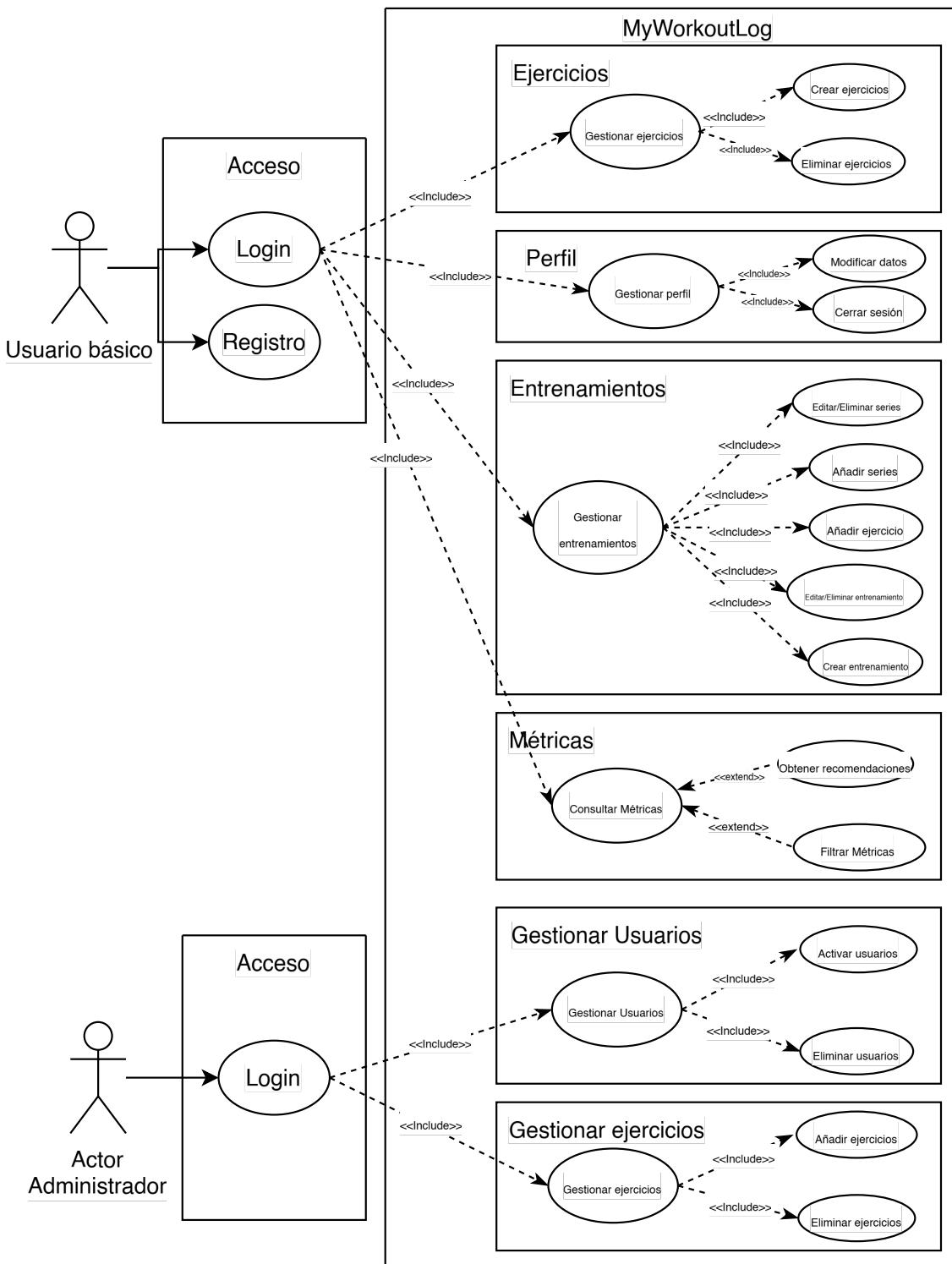


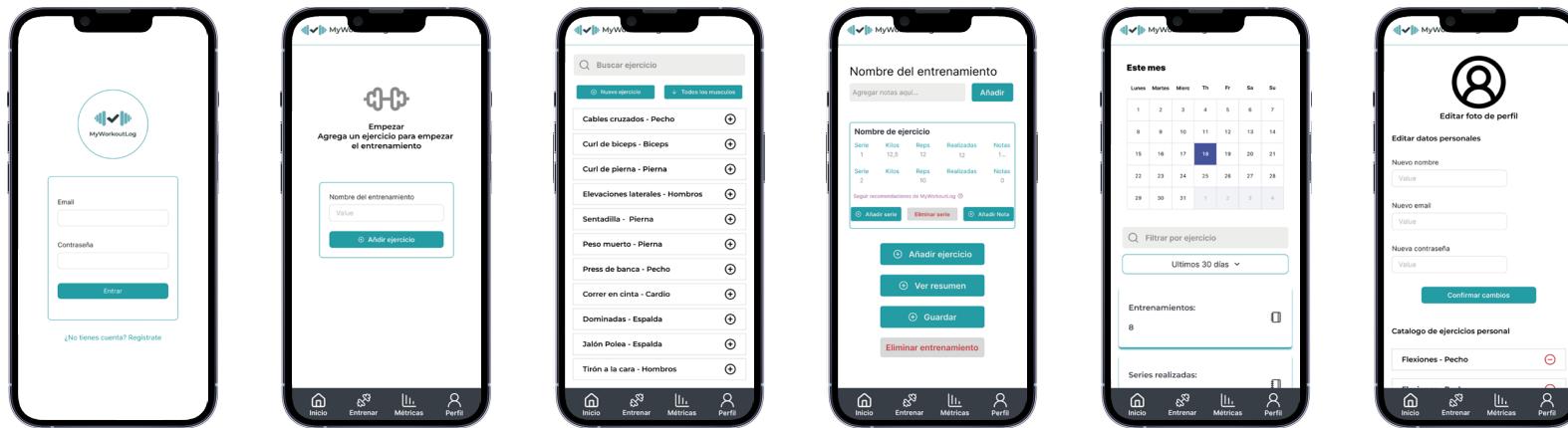


Diagrama de casos de uso:



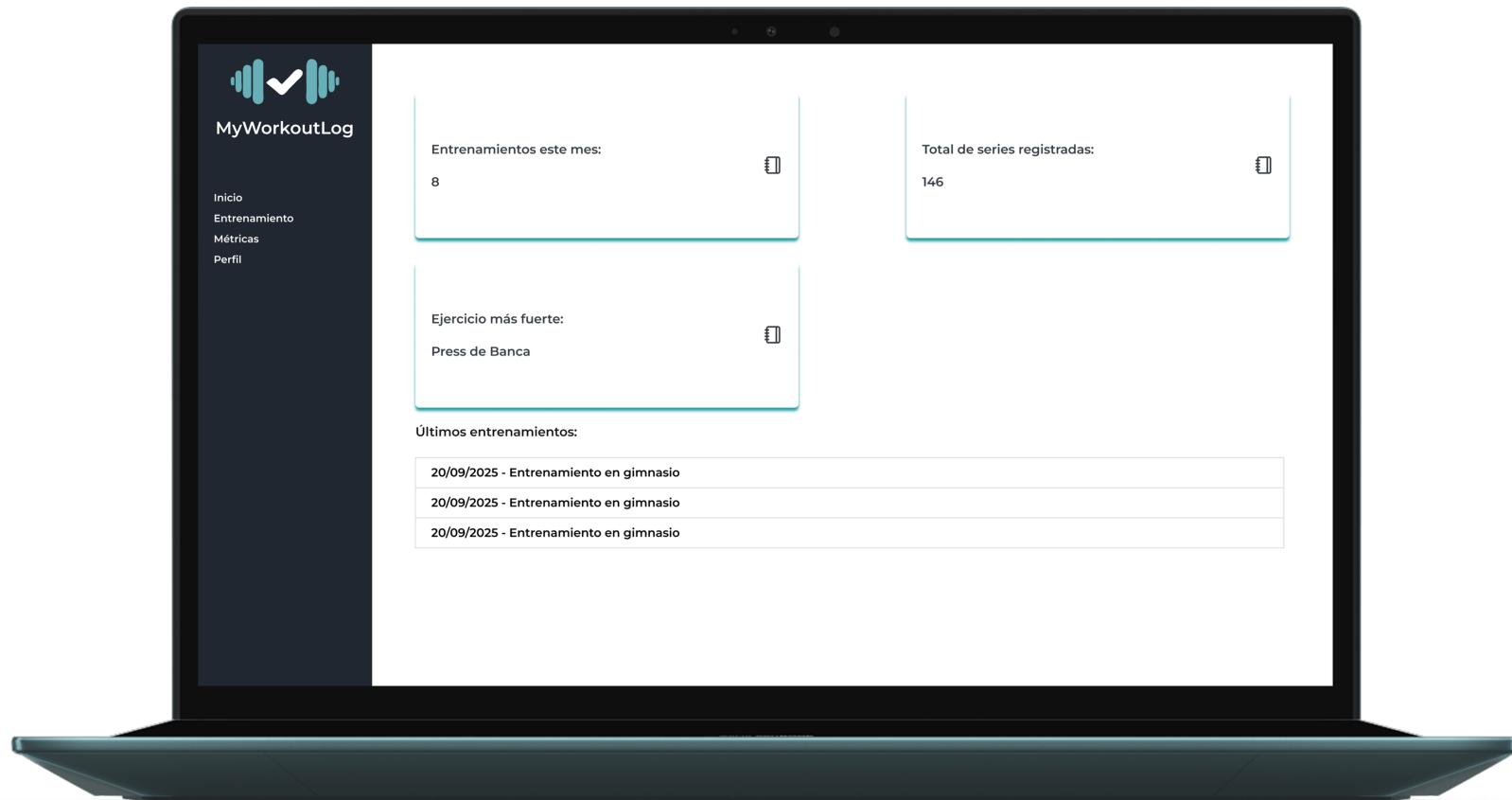


Mockups – Versión Mobile:





Mockups – Versión Desktop (Solo cambia el menú de navegación):





Cristian Menjíbar López  
MyWorkoutLog

ILERNA.

