



C.P.R. Liceo "La Paz"

Proyecto Fin de Ciclo

Desarrollo de Aplicaciones Multiplataforma

Autor: Laura Lodeiro Casas

Tutor: Jesús Ángel Pérez Roca

Resumen

StayRPE es una aplicación integral de **gestión y seguimiento de entrenamientos de fuerza**, diseñada para optimizar la **planificación, ejecución y análisis de rutinas deportivas**. Este proyecto surge de la necesidad de ofrecer una herramienta moderna que permita a deportistas y entusiastas del fitness gestionar sus entrenamientos de manera eficiente, incorporando conceptos avanzados de periodización y metodologías como **RPE** (Rate of Perceived Exertion) y **RIR** (Reps in Reserve).

La solución abarca desde la **planificación estratégica a largo plazo**, mediante un sistema de **macrociclos y microciclos**, hasta la **gestión detallada de rutinas** y la **personalización de ejercicios**. Los usuarios pueden crear y organizar rutinas, definir series, repeticiones, pesos y **métricas de intensidad**, así como adaptar entrenamientos concretos según sus necesidades, manteniendo la coherencia de la planificación global.

Una de las **funcionalidades clave** es el **historial completo de entrenamientos**, que registra cada sesión con métricas detalladas como **volumen total, porcentaje de completitud, duración y análisis de rendimiento**, facilitando así el seguimiento del progreso y la toma de decisiones fundamentadas sobre el entrenamiento futuro.

El sistema combina una interfaz móvil intuitiva desarrollada con **React Native** (compatible con Android y iOS), junto con un backend robusto basado en **Java Spring Boot**. Se garantiza la **integridad y seguridad de los datos** mediante autenticación **JWT** y control de acceso por roles. La **arquitectura REST** y la **base de datos relacional** estructurada aseguran **fiabilidad, escalabilidad y protección de la información personal**.

StayRPE destaca por su **enfoque en la periodización** y la **flexibilidad**: permite planificar, adaptar y analizar entrenamientos de forma profesional, tanto para atletas amateur como para usuarios avanzados. Es, en definitiva, una herramienta que une la **teoría del entrenamiento deportivo** con la **práctica diaria** gracias a la **tecnología** y a un **diseño centrado en el usuario**.

Abstract

StayRPE is a comprehensive application for the management and tracking of strength training, designed to optimize the planning, execution, and analysis of sports routines. This project arises from the need to provide a modern tool that allows athletes and fitness enthusiasts to efficiently manage their workouts, incorporating advanced periodization concepts and methodologies such as RPE (Rate of Perceived Exertion) and RIR (Reps in Reserve).

The solution encompasses everything from long-term strategic planning—through a system of macrocycles and microcycles—to detailed routine management and exercise customization. Users can create and organize routines, define sets, repetitions, weights, and intensity metrics, and adapt specific workouts to their needs while maintaining the coherence of the overall plan.

One of the key features is a complete training history, which records each session with detailed metrics such as total volume, completion percentage, duration, and performance analysis, thus facilitating progress tracking and informed decision-making regarding future training.

The system combines an intuitive mobile interface developed with React Native (compatible with Android and iOS) and a robust backend based on Java Spring Boot. Data integrity and security are guaranteed through JWT authentication and role-based access control. The REST architecture and a well-structured relational database ensure reliability, scalability, and protection of personal information.

StayRPE stands out for its focus on periodization and flexibility: it enables professional planning, adaptation, and analysis of workouts for both amateur athletes and advanced users. Ultimately, it is a tool that brings together the theory of sports training with daily practice, thanks to technology and user-centered design.

Palabras Clave

React Native: Framework multiplataforma para desarrollar aplicaciones móviles nativas en iOS y Android usando JavaScript y React, optimizando tiempos de desarrollo y manteniendo un alto rendimiento.

Expo: Plataforma y conjunto de herramientas que facilita el desarrollo y despliegue de apps React Native, proporcionando APIs nativas y servicios en la nube para pruebas y publicación.

Spring Boot: Framework de Java que simplifica la creación de aplicaciones empresariales, ofreciendo configuración automática y un ecosistema robusto para el backend.

Node.js: Entorno de ejecución para JavaScript en el servidor, que permite construir aplicaciones web escalables y de alto rendimiento gracias a su modelo asíncrono y no bloqueante.

Express.js: Framework web minimalista para Node.js, que facilita la creación de APIs RESTful y la gestión eficiente de rutas y peticiones HTTP.

TypeScript: Superset de JavaScript que añade tipado estático opcional, mejorando la detección temprana de errores y la mantenibilidad del código.

Lombok: Librería de Java que reduce el código repetitivo generando automáticamente métodos como getters y setters mediante anotaciones.

JWT (JSON Web Tokens): Estándar para la autenticación basada en tokens, que permite la comunicación segura entre cliente y servidor y el control de acceso a recursos protegidos.

Autenticación basada en roles: Sistema de seguridad que restringe el acceso a funcionalidades según el rol del usuario (por ejemplo, usuario o administrador).

BCrypt: Algoritmo de hashing utilizado para el cifrado seguro de contraseñas, protegiendo las credenciales almacenadas.

REST API / API RESTful: Arquitectura de servicios web que utiliza HTTP y métodos estándar (GET, POST, PUT, DELETE) para operaciones CRUD sobre los recursos del sistema.

JPA (Java Persistence API): Especificación de Java para el mapeo objeto-relacional, facilitando la interacción con bases de datos mediante entidades y repositorios.

DTO (Data Transfer Object): Patrón de diseño para transferir datos entre capas de la aplicación, separando la lógica interna de la interfaz pública de la API.

Base de datos relacional: Sistema que organiza los datos en tablas relacionadas, asegurando la integridad y consistencia de la información.

Transaccionalidad: Propiedad que garantiza la consistencia e integridad de los datos mediante el control de transacciones en la base de datos.

Arquitectura en capas: Patrón que separa la aplicación en controladores, servicios, repositorios y modelos para mejorar la mantenibilidad y escalabilidad.

RPE (Rate of Perceived Exertion): Escala subjetiva (1-10) que permite medir la intensidad de cada serie en función de la percepción personal del esfuerzo.

RIR (Repeticiones en Reserva / Reps in Reserve): Métrica que indica cuántas repeticiones adicionales podría realizar el atleta antes del fallo muscular, utilizada para ajustar la intensidad.

Macro ciclo: Estructura de planificación deportiva que agrupa el entrenamiento en periodos largos (semanas o meses), permitiendo planificar objetivos a largo plazo.

Micro ciclo: Unidad básica de planificación dentro del macro ciclo, generalmente de una semana, que organiza la distribución específica de entrenamientos y descansos.

Periodización: Metodología de planificación sistemática que organiza las cargas de trabajo en ciclos para maximizar adaptaciones y prevenir el sobreentrenamiento.

Autoregulación: Capacidad del usuario para adaptar el volumen y la intensidad del entrenamiento en función de su estado físico y mental diario.

Volumen de entrenamiento: Métrica obtenida multiplicando el peso por el número de repeticiones, que cuantifica la carga total realizada en una sesión.

Customización granular de entrenamientos: Posibilidad de modificar parámetros de ejercicios en días concretos dentro de un macro ciclo sin alterar la estructura original.

Gestión de rutinas: Sistema para crear, editar, organizar y ejecutar rutinas de entrenamiento con detalles de ejercicios, series, repeticiones y cargas.

Biblioteca de ejercicios: Catálogo personalizable de ejercicios clasificados por grupos musculares, incluyendo la opción de agregar ejercicios propios.

Seguimiento en tiempo real: Funcionalidad que permite registrar el progreso durante el entrenamiento, usando cronómetros, contadores y registros instantáneos de rendimiento.

Historial de entrenamientos: Registro completo de todas las sesiones realizadas, almacenando métricas para análisis de progreso y tendencias a lo largo del tiempo.

Validación de datos: Proceso de comprobación de la integridad y formato de los datos introducidos, implementado tanto en el backend como en la app móvil.

Responsive Design: Diseño adaptativo que garantiza una experiencia de usuario óptima en cualquier tamaño de pantalla o dispositivo.

Interfaz intuitiva: Diseño centrado en la facilidad de uso, con navegación clara y acciones accesibles para minimizar la curva de aprendizaje.

A mi familia, por confiar en mí y apoyarme, y a mis profesores, por su dedicación, por ayudarme siempre que lo he necesitado y por ser una fuente de inspiración.

Contenido

Resumen	3
Abstract.....	4
Palabras Clave	5
Motivación y Objetivos	10
Estado del arte.....	14
Caso de estudio.....	18
Diagramas.	24
Desarrollo del proyecto	39
Manual Administrador	45
Manual Usuario.....	50
Viabilidad tecno-económica.	65
Trabajo futuro.....	68
Conclusiones.	71
Biblioteca de recursos web y referencias.....	73

Motivación y Objetivos

Motivación Personal

La idea de desarrollar **StayRPE** surge tras observar, durante varios años de práctica en el ámbito del entrenamiento de fuerza, la evolución natural que experimenta cualquier atleta: desde los inicios como principiante hasta alcanzar un nivel avanzado en disciplinas como el powerlifting. Esta progresión permite identificar de primera mano las carencias que presentan las herramientas digitales actuales para la gestión del entrenamiento deportivo.

En las etapas iniciales en el gimnasio, es frecuente atravesar por fases de **sobreentrenamiento**, falta de **periodización** y desconocimiento sobre la correcta estructuración de rutinas. A medida que se profundiza en disciplinas como el **powerlifting**, la **periodización** del entrenamiento se revela no solo como una recomendación, sino como una necesidad fundamental para el progreso deportivo.

La planificación, especialmente en deportes de fuerza, requiere la creación de **macrociclos** y **microciclos** donde, dentro de una misma rutina, las repeticiones, el **RPE** (Rate of Perceived Exertion) y las cargas varían a lo largo de las semanas. Esta metodología exige una herramienta capaz de gestionar dicha complejidad, necesidad que no está cubierta por las aplicaciones existentes en el mercado.

Análisis del Problema

La búsqueda de herramientas digitales adecuadas para optimizar el entrenamiento llevó a analizar aplicaciones populares como **Hevy** y **Strong**. Aunque estas plataformas destacan por su facilidad de uso y completas estadísticas, comparten una limitación importante: están dirigidas a un público generalista y no contemplan las necesidades de atletas avanzados.

El principal inconveniente reside en la imposibilidad de realizar una **periodización real**. Estas aplicaciones permiten crear rutinas, ejecutarlas y finalizarlas, pero no ofrecen la posibilidad de planificar la evolución de una rutina durante varias semanas o meses. Para atletas que requieren planificar, por ejemplo, cinco semanas con variaciones progresivas en repeticiones, **RPE** y cargas, estas herramientas resultan insuficientes.

Otro aspecto crítico que estas aplicaciones omiten completamente son los valores de **RIR** (Reps in Reserve) y **RPE**, fundamentales en el entrenamiento moderno. Hoy en día, prácticamente

ningún entrenador cualificado prescribe rutinas sin utilizar estos parámetros de autoregulación, ya que son esenciales para optimizar la intensidad y prevenir el sobreentrenamiento.

Ante estas limitaciones, tanto atletas como entrenadores suelen recurrir a hojas de cálculo (**Excel**) para periodizar los macrociclos. Si bien esta solución es funcional, resulta poco práctica para su uso en tiempo real en el gimnasio y carece de la experiencia de usuario que se espera de una aplicación móvil moderna. Esta problemática afecta tanto a atletas autodidactas como a entrenadores profesionales, quienes deben recurrir a soluciones improvisadas ante la falta de herramientas digitales especializadas, dificultando el seguimiento y el feedback en tiempo real.

Justificación de la Elección

La decisión de desarrollar **StayRPE** como proyecto de fin de ciclo se basa en varios factores fundamentales:

- Necesidad real identificada: El proyecto responde a una carencia detectada en el sector, confirmada mediante investigación y experiencia directa, que afecta especialmente a la comunidad de atletas avanzados en deportes de fuerza.
- Mercado desatendido: Las aplicaciones existentes se enfocan en captar el mayor número de usuarios posible, lo que resulta en herramientas genéricas que no satisfacen las necesidades específicas de atletas serios que requieren **periodización** avanzada.
- Aplicación técnica integral: El desarrollo del proyecto permite demostrar competencias en diseño de **bases de datos** relacionales, así como en la implementación de interfaces de usuario orientadas al uso en tiempo real
- Relevancia tecnológica: La elección de React Native como tecnología responde a la necesidad de una app móvil práctica para el gimnasio, mientras que una arquitectura backend robusta asegura la integridad de los datos.

Objetivos del Proyecto

Objetivos Principales

- Desarrollar una herramienta de **periodización** completa que permita planificar **macrociclos** y **microciclos** con la precisión que requieren los deportes de fuerza actuales..

- Integrar metodologías de entrenamiento avanzadas, incorporando **RPE** y **RIR** como elementos fundamentales del sistema.
- Ofrecer una experiencia de usuario optimizada para el uso en tiempo real durante el entrenamiento, combinando la comodidad de las mejores aplicaciones existentes con funcionalidades avanzadas.

Objetivos Técnicos

- Implementar una **arquitectura escalable** que soporte la complejidad de datos requerida para la periodización mientras mantiene un rendimiento óptimo.
- Demostrar competencias de desarrollo full-stack creando tanto una **API robusta** como una interfaz móvil nativa de alta calidad.
- Aplicar principios de ingeniería de software en un proyecto real que resuelva un problema auténtico del mundo deportivo.

Objetivos de Impacto

- Demostrar la viabilidad técnica de una solución especializada para atletas avanzados de **powerlifting**, halterofilia, strongman y culturismo, evidenciando que es posible crear herramientas específicas que superen las limitaciones de las aplicaciones generalistas existentes.
- Aplicar y expandir conocimientos adquiridos durante el ciclo formativo mediante el aprendizaje autónomo de nuevas tecnologías como **Spring Boot**, integrándolas con conceptos fundamentales como **bases de datos relacionales (MySQL)**, desarrollo de **APIs RESTful** y arquitecturas cliente-servidor trabajados durante la formación.
- Demostrar capacidad de autoaprendizaje y adaptación tecnológica al implementar **Spring Boot** como framework backend, superando el desafío de dominar una tecnología empresarial no vista en el ciclo pero ampliamente utilizada en el desarrollo profesional.
- Crear un prototipo funcional que sirva como prueba de concepto para futuras iteraciones, estableciendo una base técnica sólida que demuestre la factibilidad de implementar **periodización** avanzada en aplicaciones móviles utilizando tecnologías modernas.

- Validar la integración exitosa de metodologías de entrenamiento modernas (**RPE/RIR**) en una plataforma digital, demostrando que es posible unificar la teoría del entrenamiento deportivo con la tecnología móvil actual.
- Desarrollar competencias profesionales avanzadas en **arquitectura de software**, diseño de **APIs RESTful**, gestión de bases de datos relacionales complejas y desarrollo de interfaces móviles optimizadas para uso en tiempo real, complementando la formación académica con experiencia práctica en tecnologías empresariales.

Estado del arte.

Análisis de Aplicaciones Existentes

Hevy - Análisis Detallado

Hevy destaca por su interfaz intuitiva y experiencia de usuario excepcional durante la ejecución de entrenamientos. La aplicación ofrece un sistema de seguimiento en tiempo real muy pulido, con cronómetros automáticos, registro fácil de series y repeticiones, y visualización clara del progreso. Su sistema de estadísticas es particularmente robusto, proporcionando gráficos detallados de volumen, progreso por ejercicio y análisis histórico que facilitan la comprensión del rendimiento a largo plazo.

La aplicación también sobresale en la gestión de la biblioteca de ejercicios, con una base de datos extensa, descripciones detalladas, y la capacidad de agregar ejercicios personalizados con facilidad. Su sistema de plantillas de rutinas permite crear y reutilizar entrenamientos de manera eficiente.

El principal déficit de Hevy radica en la ausencia total de capacidades de **periodización**. La aplicación está diseñada para ejecutar rutinas estáticas sin contemplar la evolución progresiva que caracteriza a los programas de entrenamiento avanzados. No permite planificar cómo van a cambiar las variables de entrenamiento (repeticiones, intensidad, volumen) a lo largo de semanas o meses.

Además, carece completamente de integración de **RPE** y **RIR**, métodos fundamentales en el entrenamiento moderno para la **autoregulación** de la intensidad. Esta omisión limita significativamente su utilidad para atletas que siguen programas basados en percepción del esfuerzo.

Strong – Comparativa

Strong comparte características similares a **Hevy** en términos de funcionalidad básica y experiencia de usuario. Ofrece un seguimiento eficaz de entrenamientos, estadísticas comprehensivas y una interfaz limpia y funcional.

Al igual que **Hevy**, **Strong** adolece de las mismas deficiencias fundamentales en **periodización** y ausencia de metodologías modernas como **RPE/RIR**. Su enfoque se centra en la ejecución de rutinas individuales sin considerar la planificación a largo plazo que requieren los atletas avanzados.

Gymverse y Aplicaciones Premium

Durante la investigación se identificó **Gymverse** como una aplicación premium que potencialmente podría incluir funcionalidades de **periodización** más avanzadas. Sin embargo, su modelo completamente de pago impidió una evaluación exhaustiva de sus capacidades reales.

Las aplicaciones de entrenamiento que ofrecen funcionalidades avanzadas tienden a adoptar modelos de suscripción costosos que no necesariamente garantizan que cubran las necesidades específicas de periodización. Esta barrera económica dificulta la evaluación comparativa y sugiere un mercado donde las herramientas especializadas son inaccesibles para muchos atletas.

Análisis Comparativo con StayRPE

StayRPE se posiciona como una solución específicamente diseñada para atletas avanzados que comprenden la importancia de la **periodización científica**. A diferencia de las aplicaciones existentes que intentan abarcar todo el espectro de usuarios (desde principiantes hasta avanzados), **StayRPE** se enfoca deliberadamente en las necesidades no cubiertas de los atletas serios.

Ventajas Competitivas:

- **Periodización nativa:** Capacidad integral de planificar y gestionar macrociclos y microciclos
- **Integración completa de RPE/RIR:** Estos valores son elementos centrales del sistema, no características añadidas
- **Customización granular:** Posibilidad de personalizar rutinas a nivel de día específico sin alterar plantillas base
- **Enfoque especializado:** Diseño centrado en deportes de fuerza con metodologías actuales

Análisis de Tecnologías Utilizadas

Arquitectura Frontend: React Native

La selección de **React Native** responde a la necesidad específica de crear una aplicación móvil multiplataforma que sea práctica de usar en tiempo real durante el entrenamiento. Esta

tecnología permite desarrollar tanto para **iOS** como **Android** manteniendo una base de código única, optimizando el tiempo de desarrollo mientras se garantiza un rendimiento nativo.

Alternativas Consideradas:

- **Flutter**: Aunque ofrece rendimiento excelente, la curva de aprendizaje de **Dart** y la menor familiaridad con el ecosistema React influyeron en la decisión.
- **Desarrollo Nativo (Swift/Kotlin)**: Descartado por la duplicación de esfuerzo requerida para mantener dos bases de código separadas.
- **Aplicación Web (PWA)**: Inadecuada para el uso intensivo en gimnasio que requiere acceso offline y rendimiento optimizado.

React Native proporciona el equilibrio óptimo entre experiencia de usuario nativa, eficiencia de desarrollo y mantenibilidad a largo plazo.

Arquitectura Backend: Spring Boot

La elección de **Spring Boot** sobre alternativas más comunes en el desarrollo móvil como **Firebase** representa una decisión consciente orientada al aprendizaje y aplicación de tecnologías empresariales. Aunque **Spring Boot** está tradicionalmente orientado al desarrollo web, su arquitectura robusta y capacidades de creación de **APIs RESTful** lo hacen perfectamente adecuado para este proyecto.

Ventajas de Spring Boot:

- **Ecosistema maduro**: Acceso a herramientas empresariales como Hibernate, **Spring Security** y **Spring Data JPA**.
- **Escalabilidad**: Arquitectura preparada para crecimiento y complejidad adicional.
- **Aplicación académica**: Permite demostrar competencias en tecnologías utilizadas en el desarrollo empresarial.
- **Flexibilidad**: Capacidad de integrar múltiples tecnologías y patrones de diseño.

Aunque Firebase habría proporcionado una solución más rápida con menor configuración inicial, se descartó porque limitaría las oportunidades de aprendizaje en áreas como diseño de **bases de datos relacionales**, **arquitectura de APIs REST**, y gestión de seguridad personalizada.

Gestión de Datos: MySQL + JPA/Hibernate

MySQL fue elegido como sistema de gestión de base de datos principal debido a su robustez para manejar relaciones complejas y su excelente integración con el ecosistema **Spring Boot**. La naturaleza relacional de los datos en **StayRPE** (macrociclos, rutinas, ejercicios, customizaciones) hace que una **base de datos SQL** sea la opción más apropiada.

La utilización de **Java Persistence API (JPA)** con **Hibernate** como implementación proporciona una capa de abstracción elegante sobre la base de datos, facilitando el mapeo objeto-relacional y simplificando las operaciones **CRUD** complejas requeridas para la gestión de periodización.

MongoDB y otras **bases de datos NoSQL** fueron descartadas debido a que la estructura relacional de los datos de entrenamiento se beneficia significativamente de las garantías **ACID** y las capacidades de **JOIN** que ofrecen las **bases de datos relacionales**.

Caso de estudio.

Introducción

El presente caso de estudio analiza las soluciones técnicas específicas implementadas en **StayRPE** para resolver las problemáticas identificadas en el mercado actual de aplicaciones de entrenamiento. Cada solución representa una innovación técnica diseñada para cubrir necesidades no atendidas por las herramientas existentes, demostrando la aplicación práctica de conocimientos de **ingeniería de software** en un contexto real.

Problemáticas Identificadas y Soluciones Identificadas

Problema 1: Ausencia de Periodización en Aplicaciones Existentes

Descripción del Problema

Las aplicaciones de fitness actuales están diseñadas para ejecutar rutinas estáticas sin contemplar la evolución temporal que caracteriza a los programas de entrenamiento científicos. En **powerlifting** y otros deportes de fuerza, es fundamental planificar cómo van a cambiar las variables de entrenamiento (repeticiones, intensidad, **RPE**) a lo largo de semanas y meses.

Solución Implementada: Sistema de Macroциclos y Microциclos

StayRPE implementa un sistema de gestión temporal jerárquico que permite:

- **Creación de Macroциclos:** Períodos de entrenamiento a largo plazo (4-16 semanas) con objetivos específicos
- **Gestión de Microциclos:** Subdivisiones semanales dentro de cada macroциclo con planificación diaria detallada
- **Activación Dinámica:** Sistema que permite tener múltiples **macroциclos** definidos pero solo uno activo en cada momento
- **Planificación Temporal:** Cada día del **macroциclo** puede tener rutinas específicas asignadas con variaciones programadas

Arquitectura Técnica

La solución se basa en una **estructura de datos relacional** donde cada macrociclo contiene múltiples días, y cada día puede tener rutinas asignadas con **customizaciones específicas**. Esta arquitectura permite la flexibilidad necesaria para adaptarse a cualquier metodología de periodización.

Problema 2: Falta de Integración de RPE y RIR

Descripción del Problema

Los métodos modernos de **autoregulación** del entrenamiento (**RPE - Rate of Perceived Exertion**, **RIR - Reps in Reserve**) son completamente ignorados por las aplicaciones existentes, a pesar de ser fundamentales en la prescripción de entrenamientos actuales.

Solución Implementada: Integración Nativa de Metodologías Modernas

StayRPE incorpora **RPE** y **RIR** como elementos centrales del sistema, no como características secundarias:

- **Configuración en Rutinas:** Cada serie puede tener valores objetivo de **RPE** o **RIR** definidos desde la planificación
- **Registro durante Ejecución:** Interface optimizada para capturar estos valores en tiempo real durante el entrenamiento
- **Validación Inteligente:** Sistema que previene la configuración simultánea de **RPE** y **RIR** en la misma serie, respetando las mejores prácticas
- **Análisis de Progreso:** Capacidad de analizar la evolución de la percepción del esfuerzo a lo largo del tiempo

Implementación Técnica

Los valores de **RPE** y **RIR** están integrados tanto en la **estructura de datos de planificación** como en la **captura de datos durante la ejecución**, permitiendo un flujo completo desde la prescripción hasta el análisis posterior.

Problema 3: Rigidez en la Personalización de Rutinas

Descripción del Problema

Las aplicaciones existentes requieren crear rutinas completamente nuevas para hacer pequeños ajustes, lo que resulta en una proliferación inmanejable de plantillas similares o en la pérdida de la estructura base de la rutina.

Solución Implementada: Sistema de Customización Granular

StayRPE desarrolla un sistema único de personalización que permite:

- **Customización por Día:** Modificar rutinas para días específicos sin alterar la plantilla original
- **Herencia Inteligente:** Las rutinas mantienen su estructura base mientras permiten variaciones puntuales
- **Gestión de Versiones:** Capacidad de revertir customizaciones y volver a la rutina base
- **Flexibilidad Total:** Posibilidad de ajustar cualquier parámetro (ejercicios, series, repeticiones, intensidad) para días específicos

Arquitectura de Datos

La solución implementa un **patrón de herencia** donde las customizaciones se almacenan como **capas superpuestas** sobre la rutina base, permitiendo flexibilidad máxima sin comprometer la integridad de las plantillas originales.

Problema 4: Desconexión entre Planificación y Ejecución

Descripción del Problema

Existe una brecha significativa entre las herramientas de planificación (**Excel**, documentos) y las aplicaciones de ejecución, resultando en workflows fragmentados y pérdida de eficiencia.

Solución Implementada: Flujo Integrado de Planificación a Ejecución

StayRPE unifica todo el proceso en una plataforma cohesiva:

- **Planificación Directa:** Creación de **macrociclos** y rutinas dentro de la misma aplicación
- **Transición Fluida:** Paso directo de la planificación a la ejecución sin cambios de contexto
- **Seguimiento en Tiempo Real:** Interface optimizada para uso durante el entrenamiento con cronómetros, contadores y registro inmediato
- **Feedback Continuo:** Los datos de ejecución se integran automáticamente en el análisis de progreso

Experiencia de Usuario

La solución elimina la necesidad de herramientas externas como **Excel**, proporcionando un **ecosistema completo** que cubre desde la planificación estratégica hasta la ejecución táctica diaria.

Innovaciones Técnicas Específicas

Sistema de Gestión Temporal Avanzado

Desafío Técnico

Implementar un sistema que pueda manejar la complejidad temporal de los **macrociclos** mientras mantiene la flexibilidad para adaptarse a diferentes metodologías de **periodización**.

Solución Desarrollada

- **Arquitectura Modular:** Separación clara entre la estructura temporal (**macrociclos/días**) y el contenido (rutinas/ejercicios)
- **Gestión de Estado:** Sistema que mantiene un solo **macrociclo** activo mientras permite la preparación y gestión de futuros ciclos
- **Flexibilidad de Asignación:** Capacidad de asignar diferentes rutinas a diferentes días sin restricciones predefinidas

Validación Inteligente de Datos de Entrenamiento

Desafío Técnico

Implementar validaciones que respeten las reglas del entrenamiento deportivo sin ser restrictivas en exceso.

Solución Desarrollada

- **Validaciones Contextuales:** Lógica que previene configuraciones inválidas (como **RPE** y **RIR** simultáneos) mientras permite flexibilidad en otros aspectos
- **Validación en Tiempo Real:** Feedback inmediato durante la configuración para prevenir errores
- **Reglas Específicas del Dominio:** Implementación de conocimiento especializado sobre entrenamiento en la lógica de validación

Arquitectura de Customización Escalable

Desafío Técnico

Crear un sistema que permita personalizaciones complejas sin comprometer el rendimiento o la integridad de los datos.

Solución Desarrollada

- **Patrón de Capas:** Las customizaciones se superponen sobre las rutinas base sin modificarlas
- **Optimización de Consultas:** Uso eficiente de **JPA/Hibernate** para manejar las relaciones complejas entre rutinas base y customizaciones
- **Gestión de Memoria:** Carga lazy de datos para optimizar el rendimiento en dispositivos móviles

Metodología De Desarrollo

Enfoque Orientado al Usuario Final

El desarrollo de **StayRPE** se basó en **experiencia real de usuario**, utilizando necesidades auténticas identificadas durante años de práctica deportiva. Esta aproximación garantiza que cada característica implementada resuelve problemas reales en lugar de agregar funcionalidad teórica.

Iteración Continua

El proyecto siguió un **proceso iterativo** donde cada funcionalidad se desarrolló, probó y refinó basándose en su uso práctico durante entrenamientos reales, asegurando que la aplicación sea genuinamente útil en el contexto para el que fue diseñada.

Escalabilidad desde el Diseño

Cada componente del sistema fue diseñado considerando futuras expansiones, particularmente la implementación planificada del **sistema de roles entrenador-atleta**, asegurando que la arquitectura actual pueda evolucionar sin requerir refactorización completa.

Impacto y Diferenciación

Solución Integral vs. Herramientas Fragmentadas

StayRPE elimina la necesidad de utilizar múltiples herramientas (**Excel** para planificación, apps básicas para ejecución) proporcionando una plataforma unificada que cubre todo el ciclo de vida del entrenamiento.

Especialización vs. Generalización

A diferencia de las aplicaciones existentes que intentan servir a todos los usuarios, **StayRPE** se especializa deliberadamente en las necesidades de atletas avanzados, resultando en una herramienta significativamente más útil para su público objetivo específico.

Base para Ecosistema Futuro

La **arquitectura sólida** implementada en esta primera versión establece los cimientos técnicos necesarios para evolucionar hacia un ecosistema completo de gestión del entrenamiento que incluya **comunicación entrenador-atleta, análisis avanzado de datos**, y integración con dispositivos de medición.

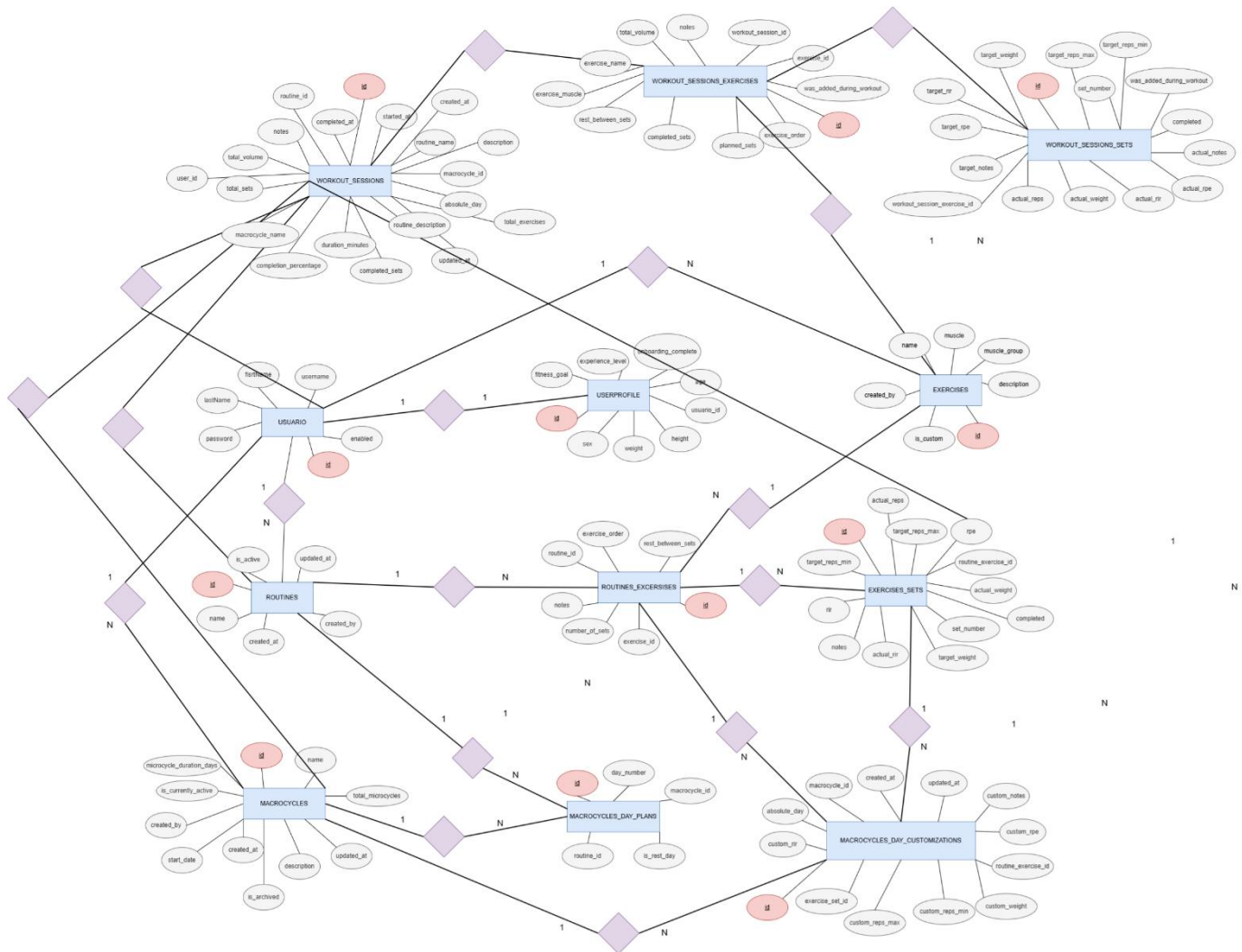
Conclusiones del Caso de Estudio

El desarrollo de **StayRPE** demuestra que es técnicamente viable crear **soluciones especializadas** que superen las limitaciones de las herramientas generalistas existentes. Las innovaciones implementadas no solo resuelven problemas específicos del entrenamiento deportivo, sino que también establecen **patrones arquitectónicos reutilizables** para futuros desarrollos en el ámbito de las aplicaciones deportivas especializadas.

La combinación de **tecnologías modernas (Spring Boot, React Native, MySQL)** con **metodologías de desarrollo centradas en el usuario** ha resultado en una solución que equilibra **funcionalidad avanzada** con **usabilidad práctica**, demostrando la aplicación exitosa de principios de **ingeniería de software** en un contexto real y especializado.

Diagramas.

Diagrama Entidad Relación



Estructura General

El diagrama E-R muestra **13 entidades** organizadas en tres bloques funcionales:

- **Superior:** Historial de entrenamientos (**WORKOUT_SESSIONS** y relacionadas)
- **Central:** Gestión de usuarios y ejercicios (**USUARIO**, **USERPROFILE**, **EXERCISES**)
- **Inferior:** Planificación (**ROUTINES**, **MACROCYCLES** y relacionadas)

Relaciones Principales

Relaciones 1:1

USUARIO ↔ USERPROFILE: Separación entre datos de acceso y perfil personal

Relaciones 1:N más relevantes

- **USUARIO > ROUTINES:** Un usuario puede crear múltiples rutinas
- **USUARIO > MACROCYCLES:** Gestión de varios planes de entrenamiento
- **ROUTINE > ROUTINE_EXERCISES:** Una rutina contiene varios ejercicios

Relaciones N:M con tablas intermedias

- **EXERCISES <> ROUTINES** mediante **ROUTINE_EXERCISES** (incluye order, numberOfSets)
- **ROUTINES <> MACROCYCLES** mediante **MACROCYCLE_DAY_PLANS** (incluye dayNumber, isRestDay)

Características del Diseño

El diseño mantiene separados los datos de planificación (**ROUTINES**, **EXERCISE_SETS**) de los de ejecución (**WORKOUT_SESSIONS**, **WORKOUT_SESSION_SETS**). Esto permite preservar el historial aunque se modifiquen las rutinas originales.

Sistema de personalización **MACROCYCLE_DAY_CUSTOMIZATIONS** permite ajustar parámetros específicos (peso, repeticiones, **RIR/RPE**) de series concretas en días determinados, sin alterar la rutina base.

Las **tablas intermedias** evitan redundancia y permiten atributos adicionales en las relaciones. Los ejercicios pueden reutilizarse en múltiples rutinas manteniendo sus configuraciones específicas.

Análisis Técnico

El diagrama presenta una **estructura compuesta** por **13 entidades interconectadas** que implementan un **modelo de datos normalizado**. La arquitectura permite la reutilización eficiente de ejercicios en diferentes contextos mediante las **tablas intermedias** correspondientes.

El sistema garantiza la **trazabilidad completa** del progreso del usuario manteniendo un **historial independiente** de las modificaciones en las rutinas base. La flexibilidad del diseño se evidencia en la capacidad de personalizar entrenamientos específicos sin comprometer la

integridad de los datos originales, aspecto fundamental para un **sistema de gestión deportiva profesional**.

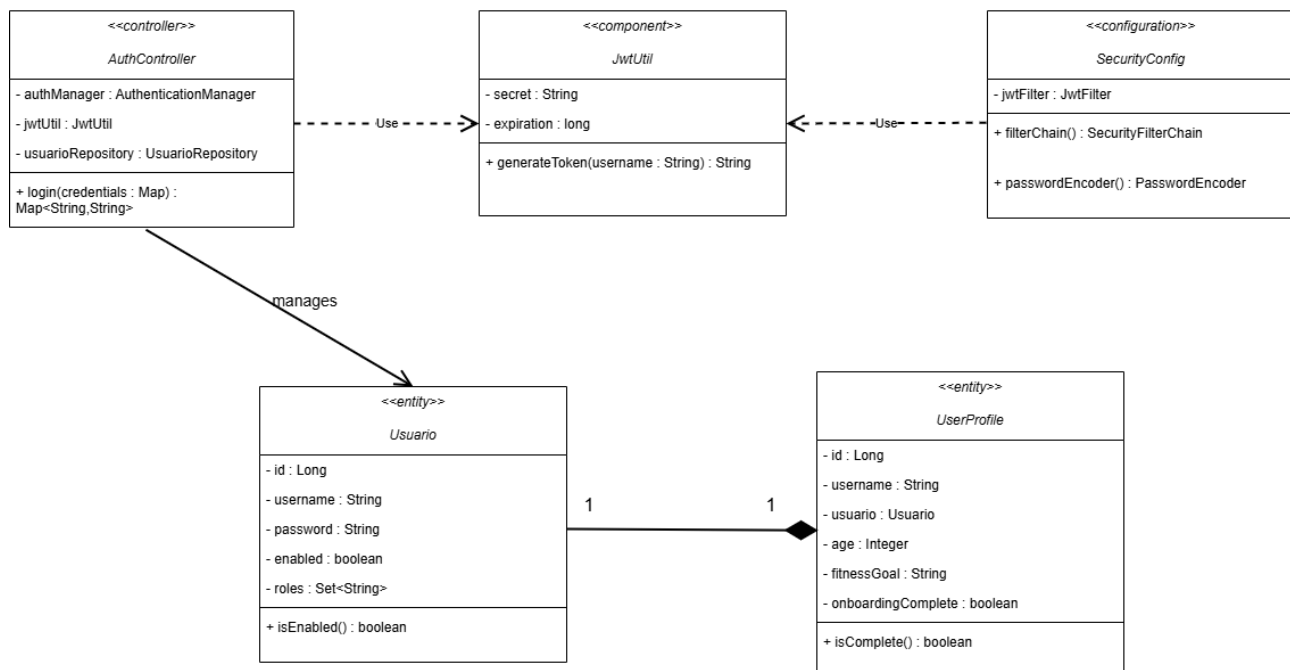
Diagrama De Clases

Los **diagramas de clases** se han organizado en cuatro **vistas modulares** para facilitar su comprensión y análisis. Esta organización visual agrupa las clases relacionadas funcionalmente, aunque la implementación sigue una **estructura tradicional de capas de Spring Boot** (controller, service, repository, model).

Esta división temática permite analizar por separado los diferentes aspectos del sistema: **autenticación, gestión de ejercicios, planificación de macrociclos e historial de entrenamientos**. Cada vista muestra las relaciones entre **controladores, servicios, repositorios y entidades** que colaboran para implementar una funcionalidad específica del sistema.

1. Módulo de Autenticación y Seguridad

Módulo de Autenticación y Seguridad



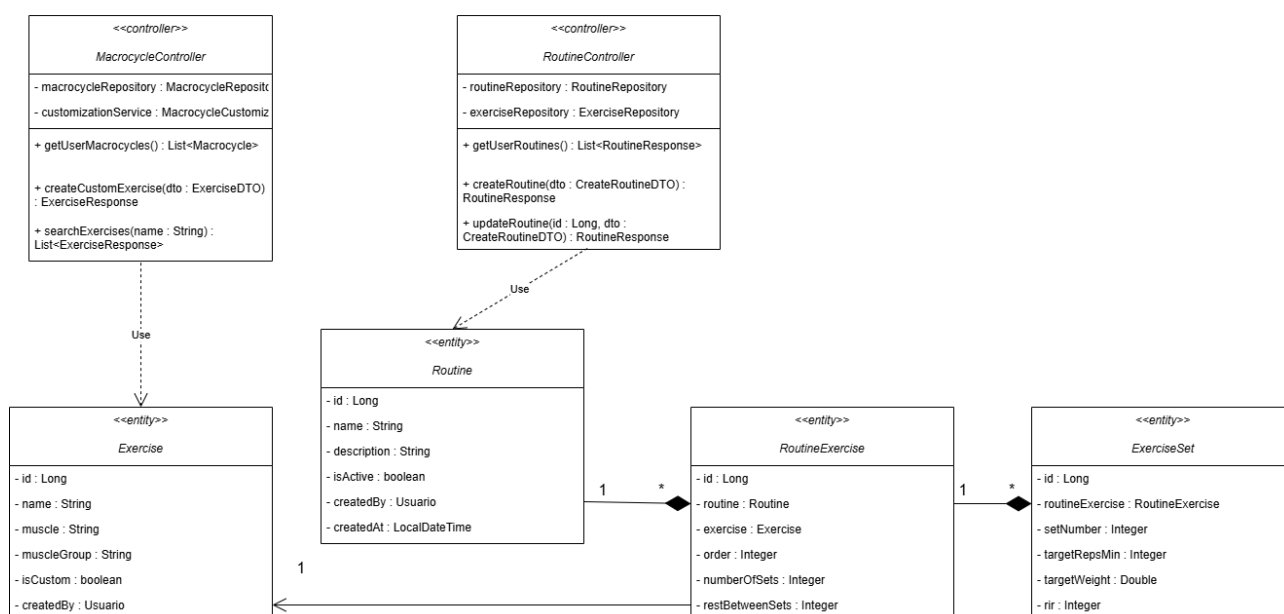
El diagrama muestra la implementación del **sistema de seguridad** basado en **JWT** (JSON Web Tokens). El controlador **AuthController** gestiona las operaciones de login mediante el

AuthenticationManager y utiliza **JwtUtil** para la generación de tokens. La configuración **SecurityConfig** establece la cadena de filtros de seguridad e integra el **JwtFilter** para la validación automática de tokens en cada petición.

La arquitectura separa claramente la gestión de usuarios (**Usuario** y **UserProfile** como entidades) de los **componentes de seguridad** (**JwtUtil**, **SecurityConfig**). Esta separación permite mantener la flexibilidad del sistema mientras garantiza la seguridad mediante encriptación de contraseñas con **PasswordEncoder** y validación de tokens.

2. Módulo de Ejercicios y Rutinas

Módulo de Ejercicios y Rutinas

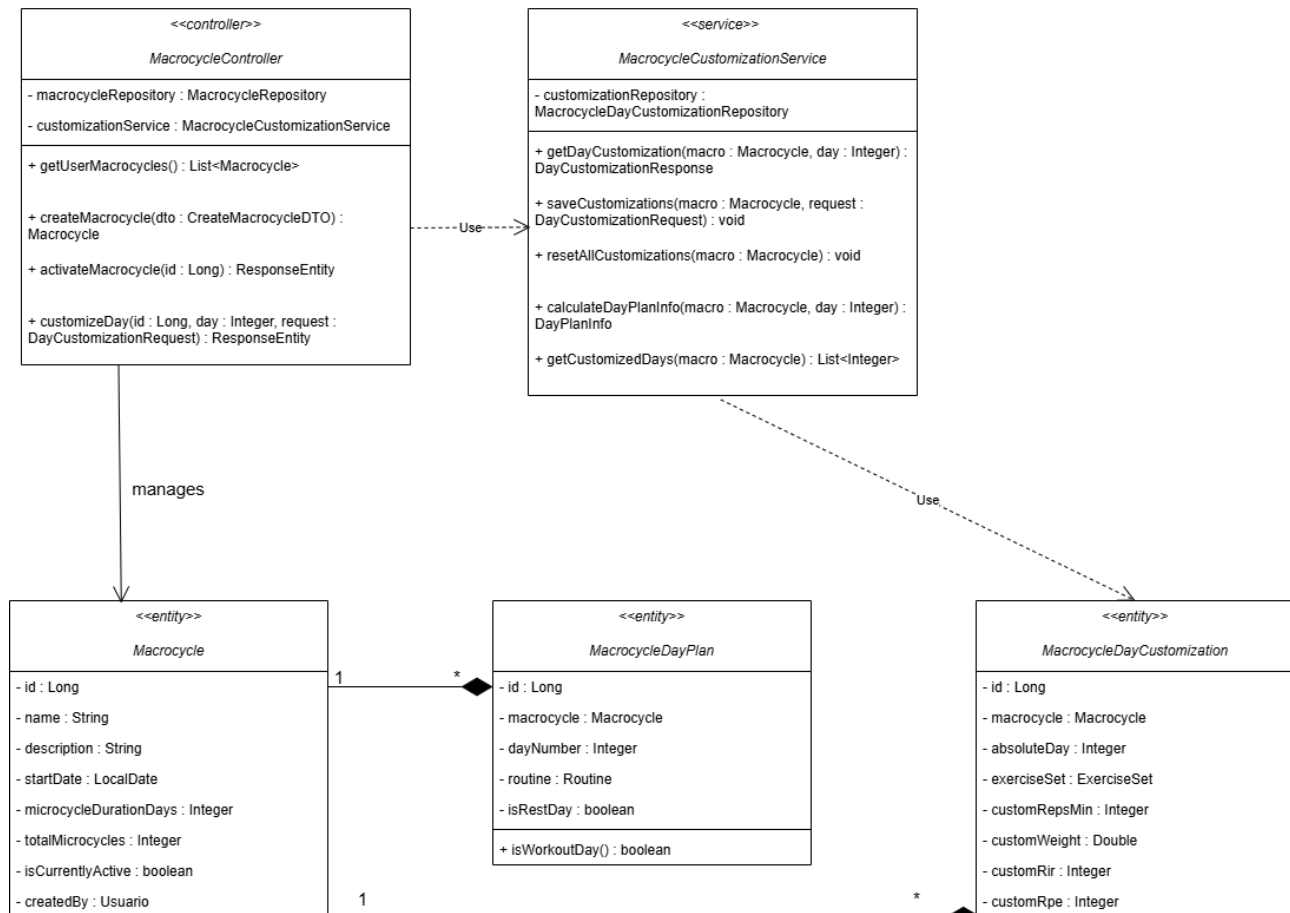


Este módulo implementa la gestión de ejercicios y rutinas mediante una **arquitectura de tres capas**. El **RoutineController** y **ExerciseController** gestionan las **peticiones HTTP**, mientras que los **repositorios** correspondientes manejan la **persistencia de datos**.

Las relaciones entre entidades muestran cómo una rutina (**Routine**) puede contener múltiples ejercicios (**Exercise**) a través de la **entidad intermedia RoutineExercise**, que incluye información específica como el orden y número de series. La entidad **ExerciseSet** detalla las especificaciones de cada serie individual, incluyendo objetivos de repeticiones, peso y **métricas de intensidad**.

3. Módulo de Macro Ciclos y Personalización

Módulo de Macro Ciclos y Personalización

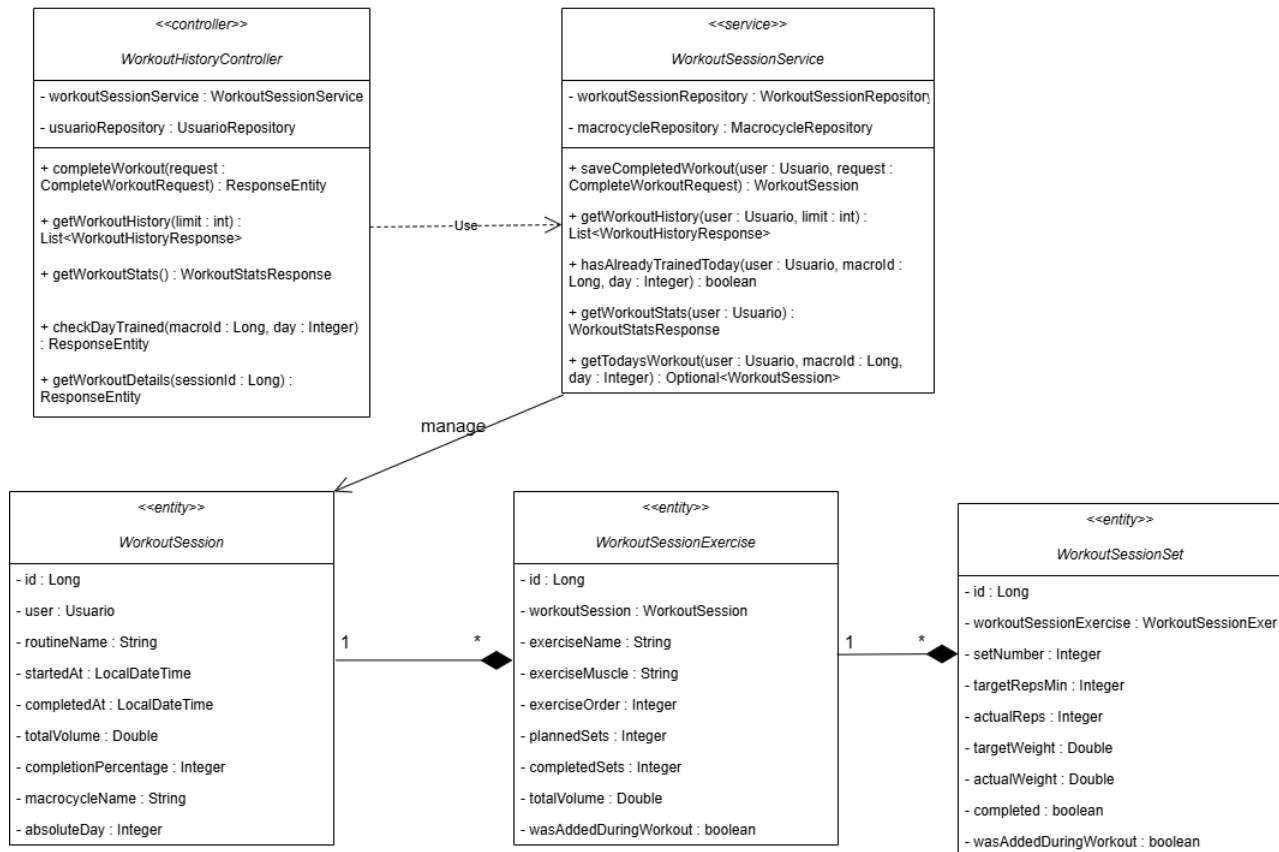


El módulo más complejo del sistema gestiona la **planificación a largo plazo** y las **personalizaciones específicas**. **MacrocycleController** coordina las operaciones principales mientras que **MacrocycleCustomizationService** maneja la **lógica de personalización avanzada**.

La entidad **Macrocycle** se relaciona con **MacrocycleDayPlan** para definir la **estructura diaria**, y con **MacrocycleDayCustomization** para permitir ajustes específicos sin modificar las rutinas base. Esta arquitectura permite implementar **periodización deportiva avanzada** manteniendo la **integridad de los datos** originales.

4. Módulo de Historial de Entrenamientos

Módulo de Historial de Entrenamientos



El **sistema de trazabilidad** se implementa mediante **WorkoutSessionService** que gestiona la **lógica de negocio** del historial. Las entidades **WorkoutSession**, **WorkoutSessionExercise** y **WorkoutSessionSet** replican la estructura de planificación pero almacenan **los datos reales de ejecución**.

WorkoutHistoryController proporciona **endpoints** para consultar estadísticas y progreso del usuario. La arquitectura permite analizar el **rendimiento histórico** independientemente de modificaciones en las rutinas originales, garantizando la **integridad del historial** de entrenamientos.

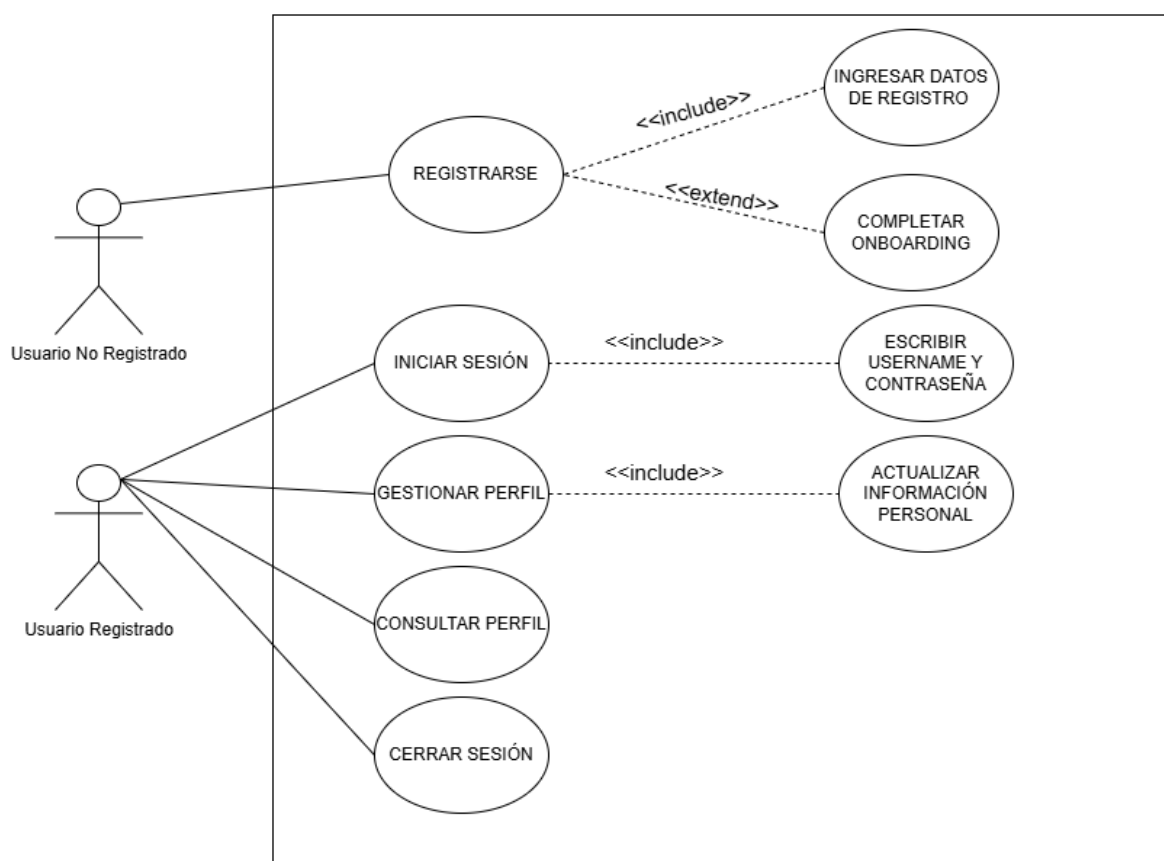
Diagramas de Casos de Uso

Los **diagramas de casos de uso** presentados ilustran la **funcionalidad completa** del sistema **StayRPE** desde la perspectiva del usuario. Esta documentación analiza cada caso de uso representado en los **cuatro diagramas modulares**, explicando las interacciones, dependencias y **flujos de trabajo** que definen el comportamiento del sistema.

La organización en **módulos temáticos** facilita la comprensión de las diferentes áreas funcionales: **autenticación y gestión de perfil**, **administración de ejercicios y rutinas**, **planificación de macrociclos**, y **seguimiento de entrenamientos**. Cada diagrama muestra las relaciones entre casos de uso mediante **asociaciones de inclusión y extensión**, proporcionando una vista arquitectural clara de los **requisitos funcionales** del sistema.

1. Autenticación y Perfil

Módulo Autenticación y Perfil



El diagrama de autenticación presenta dos **actores claramente diferenciados** que acceden a distintos conjuntos de funcionalidades. El **Usuario No Registrado** tiene acceso limitado a las operaciones de registro e inicio de sesión, mientras que el Usuario Registrado puede realizar la **gestión completa** de su perfil personal. La estructura del diagrama revela que el registro siempre requiere el ingreso de datos básicos, pero puede expandirse opcionalmente hacia el **proceso de onboarding** para completar la configuración inicial del perfil.

Una vez autenticado, el usuario accede a un conjunto ampliado de funcionalidades centradas en la **gestión de información personal**. El diagrama muestra que la actualización de información personal es un componente integral del caso de uso de **gestión de perfil**, mientras que la consulta de perfil opera como una funcionalidad independiente de solo lectura. La inclusión del caso de uso de **cierre de sesión** completa el ciclo de autenticación, proporcionando una terminación controlada del acceso al sistema.

2. Ejercicios y Rutinas

Módulo Ejercicios y Rutinas

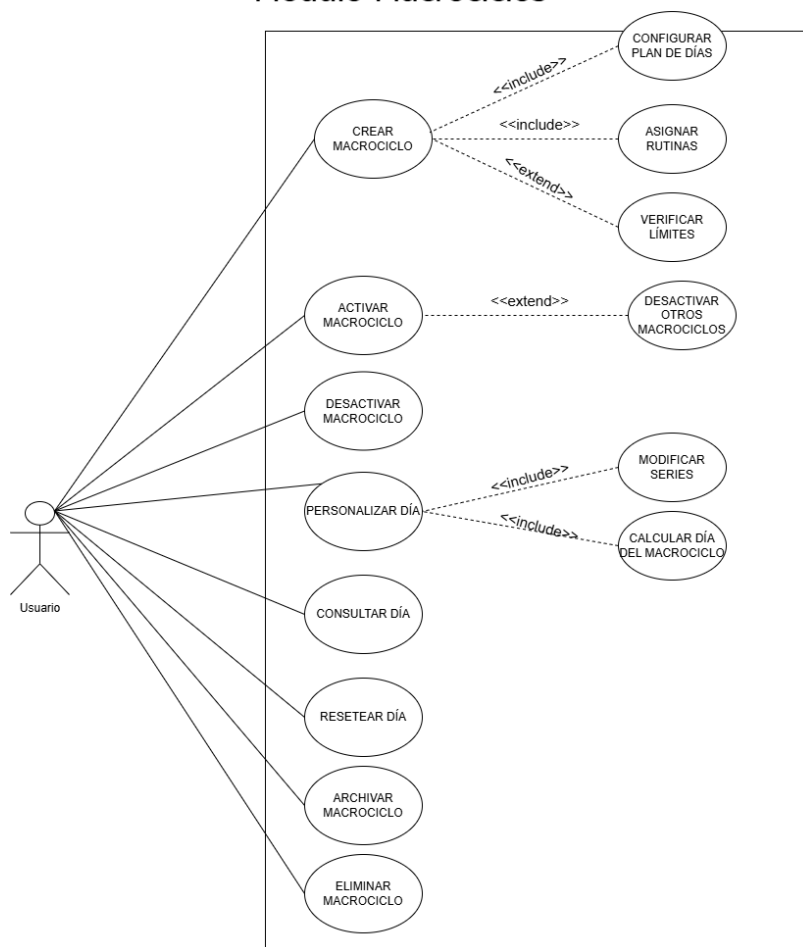


En este módulo, el actor **Usuario** interactúa con dos conjuntos de funcionalidades relacionadas pero distintas. Para la **gestión de ejercicios**, el usuario puede realizar "Buscar Ejercicios" para localizar movimientos específicos, "Crear Ejercicio Personalizado" para expandir la biblioteca con ejercicios propios, y "Eliminar Ejercicio" para remover ejercicios personalizados que ya no necesite.

Para el **diseño de rutinas**, el mismo actor puede ejecutar "Crear Rutina" para diseñar nuevos programas de entrenamiento, "Editar Rutina" para modificar rutinas existentes, "Eliminar Rutina" para remover rutinas obsoletas, y "Consultar Rutinas" para revisar todas las rutinas creadas. Esta distribución de casos de uso permite al usuario construir y gestionar completamente su **biblioteca de entrenamientos** desde los ejercicios individuales hasta rutinas complejas.

3. Macrociclos

Módulo Macrociclos

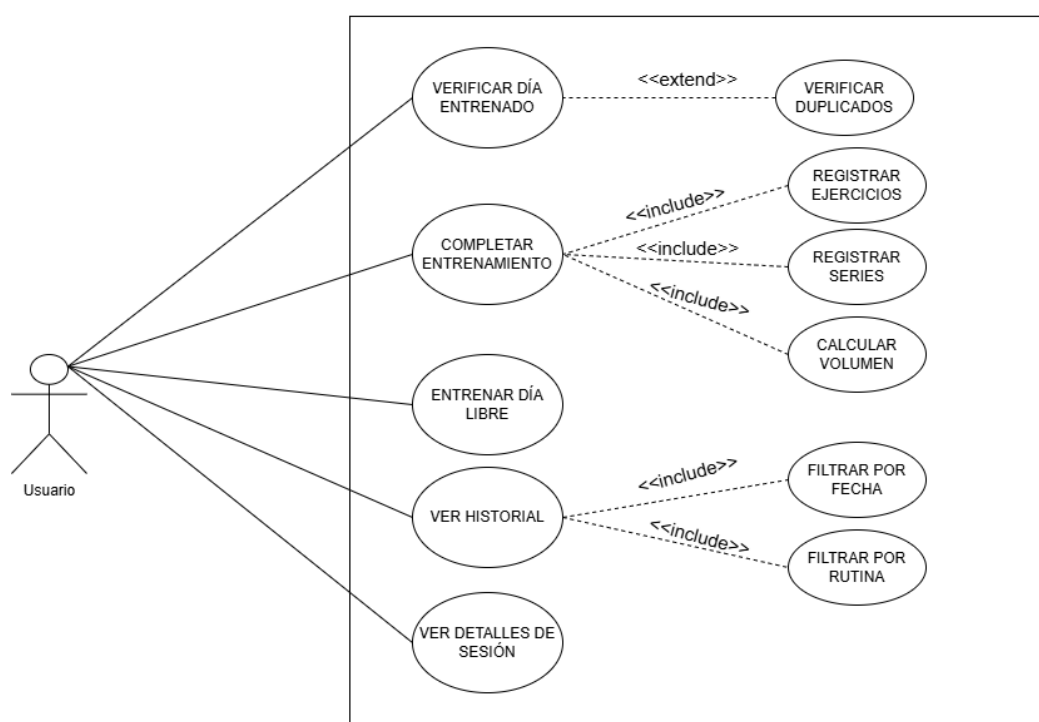


El actor **Usuario** en este módulo puede realizar **operaciones avanzadas de planificación periodizada** a través de múltiples casos de uso. Las operaciones principales incluyen "Crear Macrocielo" para diseñar **planes a largo plazo**, "Activar Macrocielo" para establecer un plan como activo, "Desactivar Macrocielo" para finalizar un plan activo, "Archivar Macrocielo" para preservar planes completados, y "Eliminar Macrocielo" para remover definitivamente planes no deseados.

Adicionalmente, el usuario puede realizar **operaciones específicas de personalización** a través de "Personalizar Día" para ajustar entrenamientos específicos, "Consultar Día" para revisar la planificación diaria, y "Resetear Día" para eliminar personalizaciones específicas. Esta variedad de casos de uso proporciona al usuario **control granular** sobre tanto la estructura general como los detalles específicos de su **planificación de entrenamiento**.

4. Entrenamientos e Historial

Módulo Entrenamientos e Historial



En este módulo final, el actor **Usuario** puede ejecutar casos de uso relacionados con la **ejecución real de entrenamientos** y el **análisis de su historial**. Para la ejecución, el usuario puede realizar "Verificar Día Entrenado" para comprobar si ya ha entrenado un día específico, "Completar Entrenamiento" para registrar una sesión ejecutada, y "Entrenar Día Libre" para registrar entrenamientos fuera de la estructura de **macrociclos**.

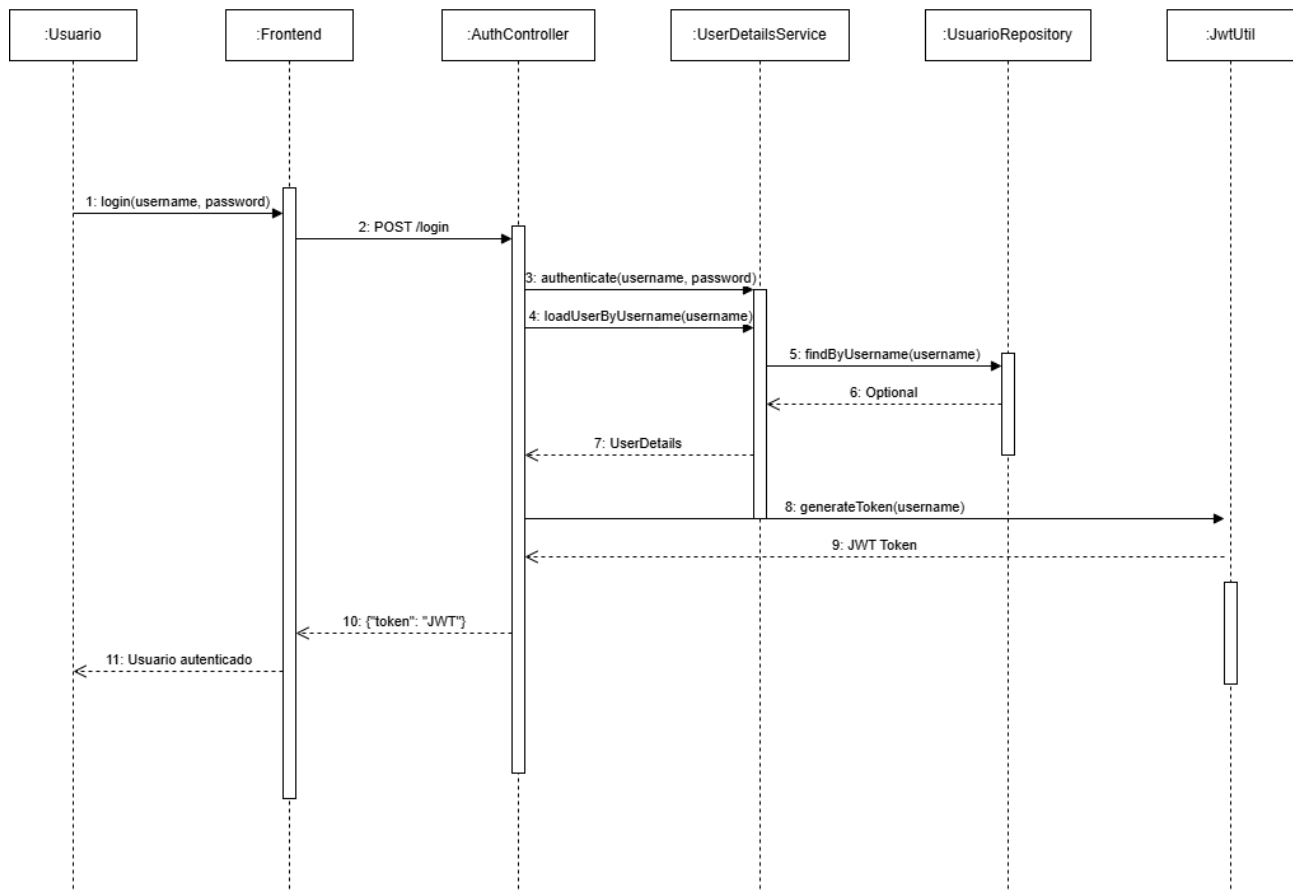
Para el **análisis histórico**, el usuario puede ejecutar "Ver Historial" para revisar sus entrenamientos pasados con **opciones de filtrado**, y "Ver Detalles de Sesión" para analizar en profundidad entrenamientos específicos. Estos casos de uso permiten al usuario tanto registrar su **actividad de entrenamiento** como analizarla posteriormente para tomar decisiones informadas sobre su programación futura.

Diagramas De Secuencia

Los diagramas de secuencia presentados ilustran los **flujos de comunicación** entre los diferentes componentes del sistema **StayRPE** para cinco **operaciones críticas**. Cada diagrama muestra la interacción temporal entre actores, controladores, servicios, repositorios y la base de datos, revelando la **arquitectura en capas** y los **patrones de comunicación** que caracterizan el sistema.

1. Autenticación de Usuario

Autenticación de Usuario

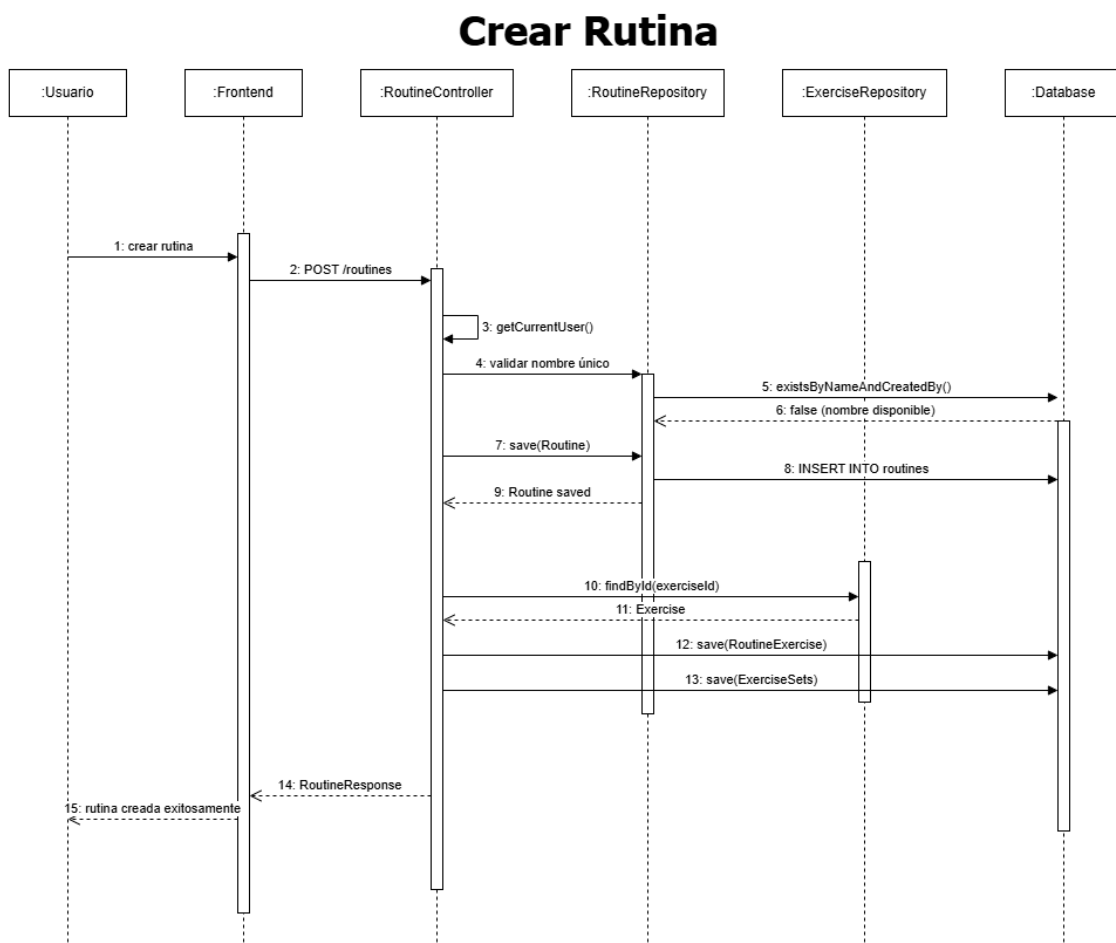


El proceso de autenticación inicia cuando el **Usuario** proporciona sus credenciales al **Frontend**, que inmediatamente transmite una petición **POST /login** al **AuthController**. El controlador ejecuta la autenticación mediante el **AuthenticationManager** y posteriormente

solicita los detalles del usuario al **UserDetailsService**. Este servicio consulta el **UsuarioRepository** con **findByUsername()**, que puede retornar información opcional del usuario desde la **base de datos**.

Una vez validadas las credenciales, el **AuthController** solicita la generación de un **token JWT** al **JwtUtil**, que retorna el **token firmado**. El controlador envía este token de vuelta al **Frontend**, que finalmente notifica al **Usuario** que la autenticación ha sido exitosa. El flujo demuestra el **patrón de seguridad basado en tokens** que caracteriza la arquitectura del sistema.

2. Crear Rutina

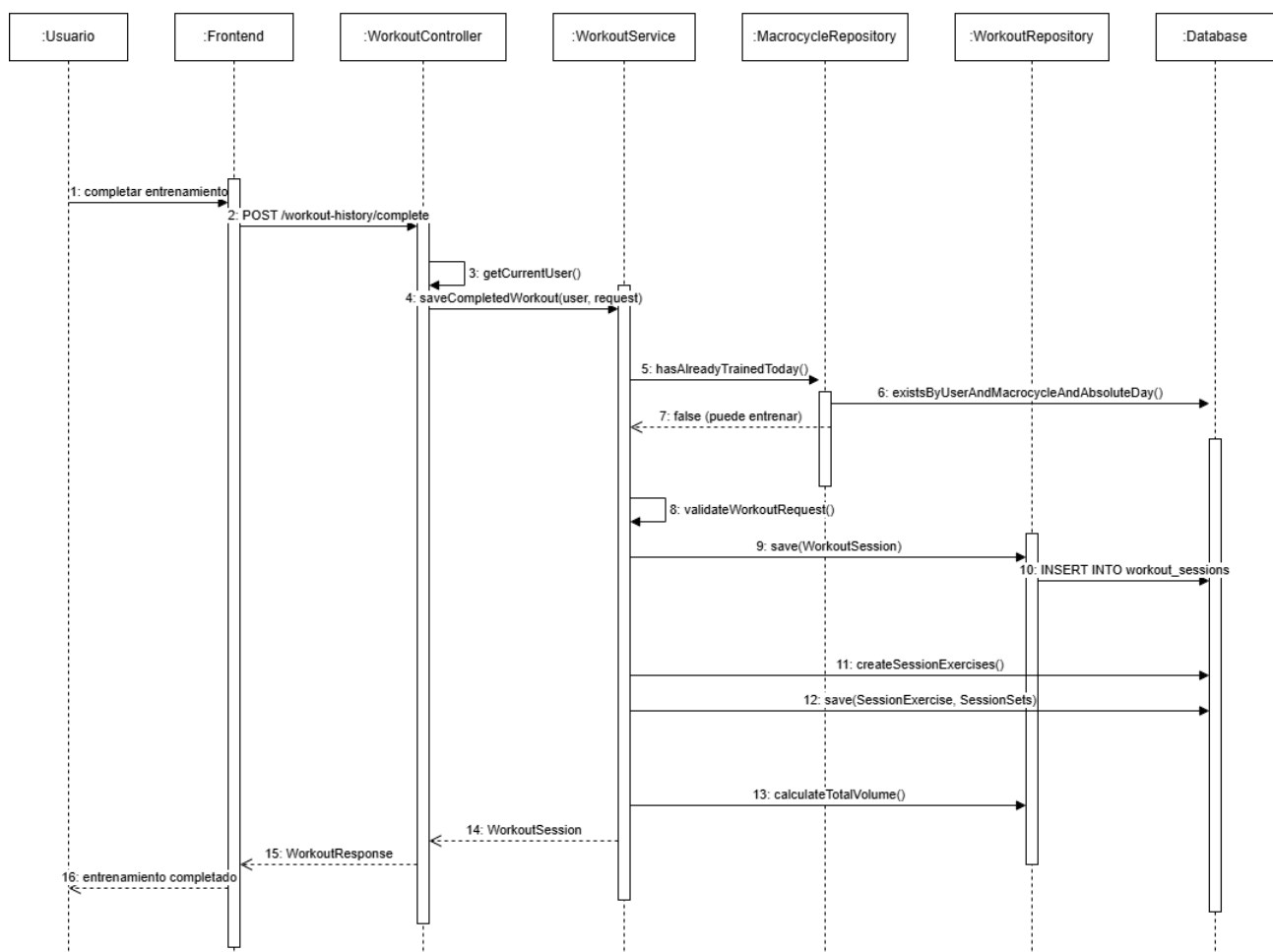


La creación de rutinas comienza cuando el **Usuario** solicita crear una nueva rutina al **Frontend**, que envía una petición **POST /routines** al **RoutineController**. El controlador obtiene primero la información del usuario actual y procede a validar que el nombre de la rutina sea único mediante consultas al **RoutineRepository**. Tras confirmar la disponibilidad del nombre, ejecuta **saveRoutine()** para persistir la rutina básica en la **base de datos**.

El proceso continúa con la búsqueda de ejercicios mediante **findByld()** en el **ExerciseRepository**, seguida por la creación de **RoutineExercise** y **ExerciseSet** asociados. Cada uno de estos componentes se persiste secuencialmente en la base de datos. Finalmente, el controlador retorna una **RoutineResponse** completa al **Frontend**, que confirma al Usuario que la rutina ha sido creada exitosamente. Este diagrama ilustra las **operaciones transaccionales complejas** que caracterizan la creación de **entidades compuestas**.

3. Completar Entrenamiento

Completar Entrenamiento

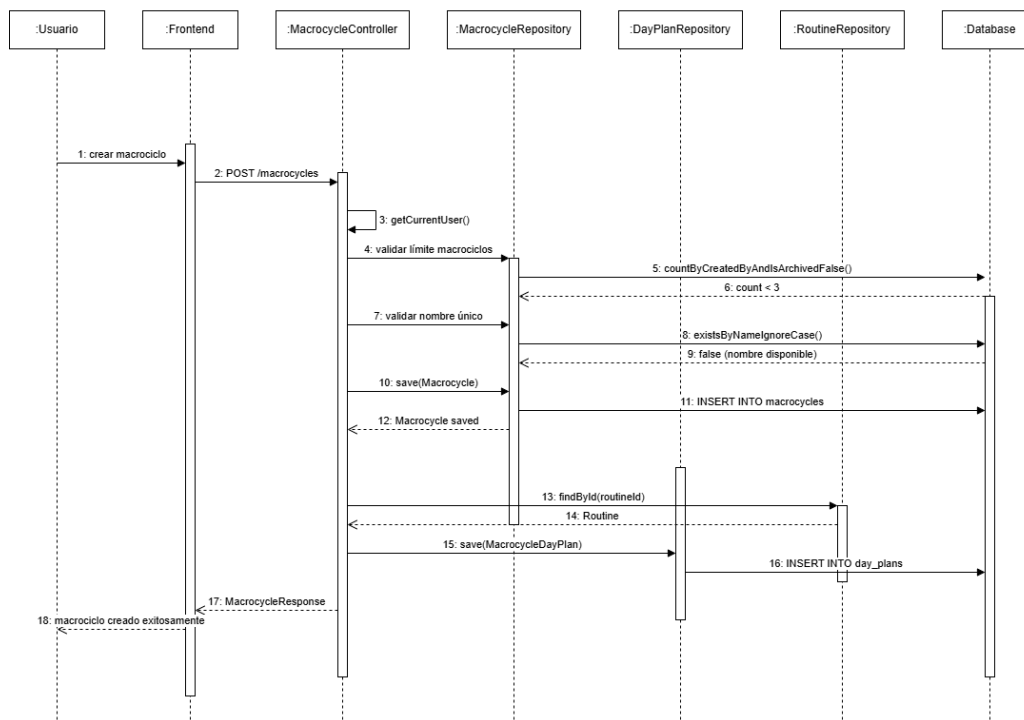


El registro de entrenamientos completados inicia cuando el **Usuario** solicita completar un entrenamiento al **Frontend**, que envía una petición **POST /workout-history/complete** al **WorkoutController**. El controlador obtiene el usuario actual y delega la operación principal al **WorkoutService** mediante **saveCompletedWorkout()**. El servicio ejecuta inmediatamente **hasAlreadyTrainedToday()** para verificar duplicados consultando el **MacrocycleRepository**.

Si no existen duplicados, el servicio procede con **validateWorkoutRequest()** y posteriormente **save()** para crear la **WorkoutSession** principal. El proceso continúa con **createSessionExercises()** que persiste tanto **SessionExercise** como **SessionSets** en la base de datos. Finalmente, **calculateTotalVolume()** actualiza las métricas de la sesión. El **WorkoutController** retorna una **WorkoutResponse** al **Frontend**, confirmando al Usuario que el entrenamiento ha sido registrado exitosamente.

4. Crear Macro ciclo

Crear Macro ciclo

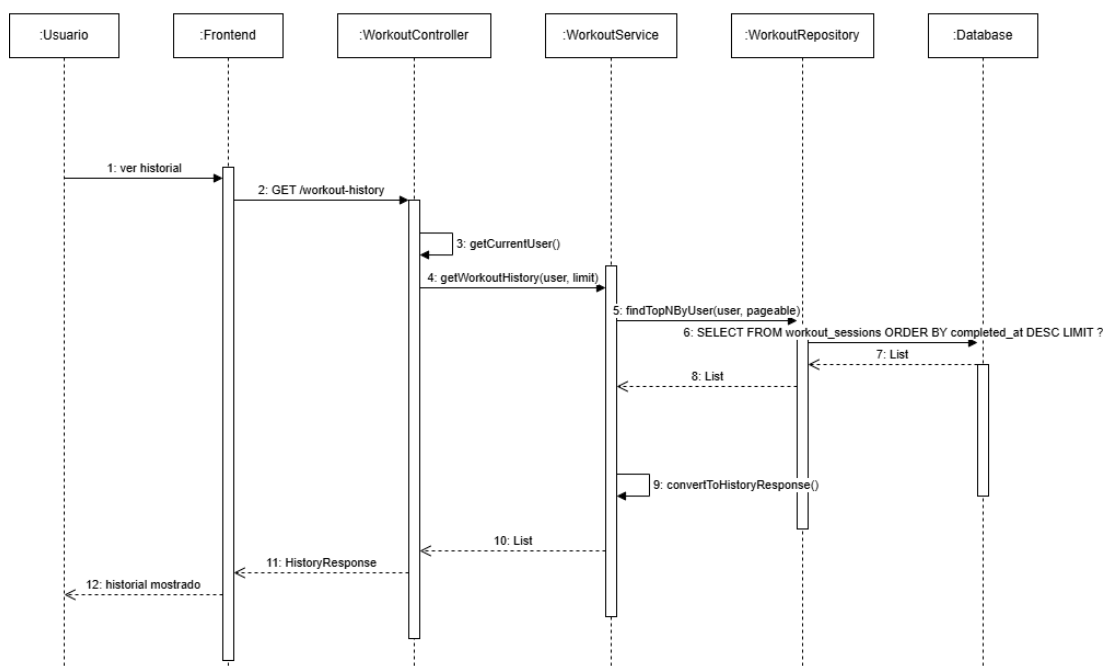


La creación de **macrociclos** comienza cuando el **Usuario** solicita crear un nuevo macrociclo al **Frontend**, que transmite una petición **POST /macrocycles** al **MacrocycleController**. El controlador obtiene el usuario actual y valida inmediatamente el límite de macrociclos activos mediante **countByCreatedByAndIsArchivedFalse()** en el **MacrocycleRepository**. Si el límite permite la creación, valida la unicidad del nombre con **existsByNameIgnoreCase()**.

Tras las validaciones exitosas, el controlador ejecuta **save()** para persistir el **macrociclo** base. Posteriormente, busca las rutinas asociadas mediante **findById()** en el **RoutineRepository** y procede a crear los **MacrocycleDayPlan** correspondientes, persistiéndolos en la base de datos. El controlador finaliza retornando una **MacrocycleResponse** completa al **Frontend**, que confirma al **Usuario** que el **macrociclo** ha sido creado exitosamente. Este proceso demuestra las **validaciones de negocio complejas** que caracterizan la gestión de **macrociclos**.

5. Obtener Historial de Entrenamientos

Obtener Historial de Entrenamientos



La consulta del historial inicia cuando el **Usuario** solicita ver su historial al **Frontend**, que envía una petición GET /workout-history al **WorkoutController**. El controlador obtiene el usuario actual y delega la operación al **WorkoutService** mediante **getWorkoutHistory()**. El servicio ejecuta **findTopNByUser()** con paginación en el **WorkoutRepository**, que retorna una **consulta SQL** ordenada por fecha de completitud descendente.

El repositorio retorna una lista de sesiones que el servicio procesa mediante **convertToHistoryResponse()** para transformar los datos en el **formato de respuesta** apropiado. El **WorkoutService** retorna esta lista procesada al controlador, que la envía al **Frontend** para mostrar al **Usuario** el **historial completo** de entrenamientos. Este diagrama ilustra el **patrón de consulta y transformación de datos** que caracteriza las **operaciones de lectura** del sistema.

Desarrollo del proyecto

Durante el desarrollo de **StayRPE**, he tomado múltiples **decisiones arquitectónicas** tanto en el **backend** como en el **frontend** que han sido fundamentales para crear una aplicación funcional y escalable. A continuación detallo las decisiones más importantes y la justificación detrás de cada una.

Arquitectura del Backend

Para el backend elegí **Spring Boot** siguiendo el **patrón MVC**, lo que me permitió una **separación clara de responsabilidades**. Los controladores como **ExerciseController**, **RoutineController** y **MacrocycleController** se encargan únicamente de manejar las **peticiones HTTP**, mientras que servicios especializados como **MacrocycleCustomizationService** y **WorkoutSessionService** contienen toda la **lógica de negocio compleja**. Esta separación facilita enormemente el **mantenimiento del código** y permite hacer modificaciones en una capa sin afectar las demás. Además, el uso de **DTOs** como **CreateRoutineDTO**, **DayCustomizationRequest** y **CompleteWorkoutRequest** asegura que solo se transfieren los datos necesarios entre **frontend** y **backend**, mejorando la **seguridad** y el **rendimiento**.

El sistema de autenticación se basa en **JWT** implementado en **JwtUtil**, que genera tokens con **30 días de duración** usando algoritmo **HMAC256**. El **JwtFilter** intercepta todas las peticiones para validar tokens antes del acceso a recursos protegidos, y las contraseñas se cifran con **BCrypt**. Esta configuración establece **rutas públicas** para login, registro y contenido público, mientras que las funcionalidades principales requieren **autenticación válida**.

Modelo de Datos y Relaciones

El diseño de la base de datos se centra en la entidad **Usuario**, de la que derivan todas las **relaciones principales**. Los ejercicios se modelaron con la entidad **Exercise**, permitiendo tanto ejercicios predefinidos del sistema como **ejercicios personalizados** mediante los campos **isCustom** y **createdBy**. Las rutinas siguen una estructura jerárquica donde **Routine** contiene **RoutineExercise**, que a su vez contienen **ExerciseSet**, definiendo series con parámetros como **targetRepsMin**, **targetRepsMax**, **targetWeight** y campos específicos para **RIR** y **RPE**.

Los **macrociclos** representan la funcionalidad más compleja del sistema. La entidad **Macrocycle** define **planificaciones a largo plazo** con **startDate**, **microcycleDurationDays** y **totalMicrocycles**. **MacrocycleDayPlan** especifica qué rutina realizar cada día del microciclo, mientras que **MacrocycleDayCustomization** permite modificar parámetros específicos por día absoluto sin alterar las rutinas originales. Esta **decisión arquitectónica** era crucial porque permite

a los usuarios personalizar entrenamientos específicos manteniendo la **integridad de sus rutinas base**.

Algoritmo de Cálculo de Días Absolutos

Una de las partes más complejas técnicamente fue desarrollar el **algoritmo** para convertir días de **microciclo a días absolutos** del **macrociclo**. El cálculo se implementa en el método `calculateDayPlanInfo()` donde el día absoluto se convierte en día del microciclo usando la fórmula: $\text{int dayOfMicrocycle} = ((\text{absoluteDay} - 1) \% \text{macrocycle.getMicrocycleDurationDays}()) + 1$. Por ejemplo, en un macrociclo con microciclos de 7 días, el día absoluto 15 correspondería al día 1 del tercer microciclo $((15-1) \% 7) + 1 = 1$. Este **algoritmo** es fundamental porque permite que un macrociclo de cualquier duración se repita cíclicamente, utilizando siempre el mismo patrón de rutinas definido en el primer **microciclo**.

Arquitectura del Frontend

Para el **frontend** elegí **React Native** con **Expo** como stack principal. Esta decisión se basó en que me permitía desarrollar una **aplicación multiplataforma** con un solo código base, algo crucial dado el tiempo limitado del proyecto. **Expo** facilitaba enormemente el proceso de desarrollo y testing, especialmente para funcionalidades como la **gestión del estado de autenticación** y el **almacenamiento local**. Aunque esto limitaba algunas funcionalidades nativas más avanzadas, para una aplicación de gestión de entrenamientos las ventajas superaban claramente las desventajas.

La **estructura de navegación** la implementé usando **Expo Router**, que proporciona navegación basada en el **sistema de archivos** similar a **Next.js**. Esto me facilitó mucho la organización del código y la comprensión de las rutas, especialmente al trabajar con **las pestañas principales** de la aplicación: **Calendario, Rutinas, Macrociclos y Perfil**.

Gestión del Estado y Datos

Una decisión importante fue cómo manejar el **estado de la aplicación**. Opté por usar **React hooks (useState, useEffect)** y **AsyncStorage** para la **persistencia local**, en lugar de librerías más complejas como **Redux**. Esta decisión se basó en que la aplicación, aunque tiene varias pantallas, no requería un **estado global** muy complejo. La mayoría de los datos se obtenían del backend cuando era necesario, y solo mantenía en el **estado local** información crítica como el **token de autenticación** y algunas preferencias del usuario.

Funcionalidades Específicas Implementadas

El **sistema de entrenamientos en vivo** fue probablemente la funcionalidad más compleja de implementar. En las pantallas **workout.tsx** y **exercise-detail.tsx** desarrollé un sistema que permite al usuario navegar entre ejercicios, completar series y manejar tiempos de descanso. Una característica que considero especialmente útil es el **timer automático de descanso** que aparece como overlay cuando completas una serie, sugiriendo cuándo es momento de la siguiente serie.

Una de las funcionalidades más innovadoras es el **sistema de personalización de días** en los **macrociclos**. Esta pantalla permite al usuario modificar series específicas de ejercicios para días particulares de su **macrociclo**, manteniendo la rutina base intacta pero adaptándola a necesidades específicas. La implementación requería manejar tres estados diferentes: valores originales, valores personalizados, y valores efectivos. El **modal de edición de series** fue especialmente complejo porque tenía que manejar diferentes tipos de intensidad (**RIR** vs **RPE**), validar inputs numéricos en tiempo real, y proporcionar una vista previa de cómo quedaría la serie modificada.

Validaciones y Manejo de Errores

Implementé **validaciones robustas** como verificación de rangos de fechas en **macrociclos**, límite de **3 macrociclos activos** por usuario, y validaciones de customización en las peticiones. Los controladores capturan excepciones y devuelven **respuestas HTTP descriptivas**. También implementé un **sistema consistente de manejo de errores** en toda la aplicación, mostrando mensajes específicos cuando ocurren errores de red y ofreciendo opciones para reintentar.

Optimizaciones y Decisiones Técnicas

Para mejorar el **rendimiento** implementé varias optimizaciones. Los repositorios utilizan consultas **específicas** como **findByMacrocycleAndAbsoluteDayWithDetails()** para traer solo los datos necesarios. Se implementó **paginación** en el historial de entrenamientos y las estadísticas utilizan **consultas agregadas** para optimizar el rendimiento con grandes volúmenes de datos.

El **diseño visual** se basó en una **paleta de colores coherente** centrada en el morado como color principal, transmitiendo profesionalidad y modernidad. Implementé un **sistema de componentes reutilizables** con estilos consistentes, y en lugar de usar listas simples, opté por **tarjetas expansibles** que muestran información básica pero permiten ver detalles cuando el usuario lo necesita.

La elección de **Spring Boot** se justifica por su **ecosistema maduro** y facilidad para crear **APIs REST**, mientras que **JPA/Hibernate** proporciona una **abstracción robusta** sobre la base de datos. El diseño permite futuras extensiones manteniendo la separación entre ejercicios del **sistema y personalizados**, y la **estructura de servicios** facilita añadir funcionalidad sin modificar código existente.

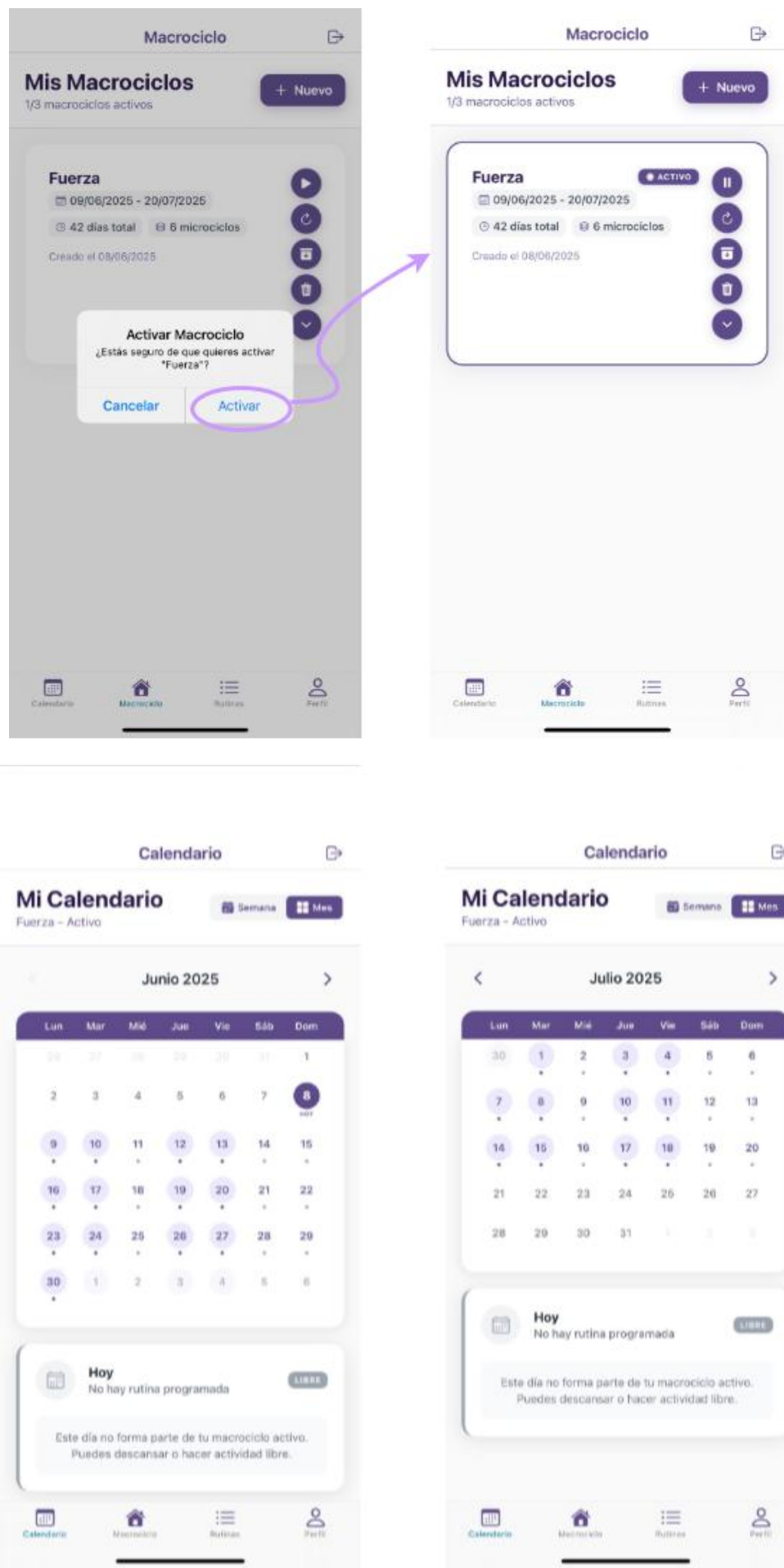
Problemas Encontrados, Soluciones Implementadas y Casos de Prueba

Durante el desarrollo encontré dos **problemas técnicos principales** que requerían soluciones específicas y validación exhaustiva mediante **casos de prueba**.

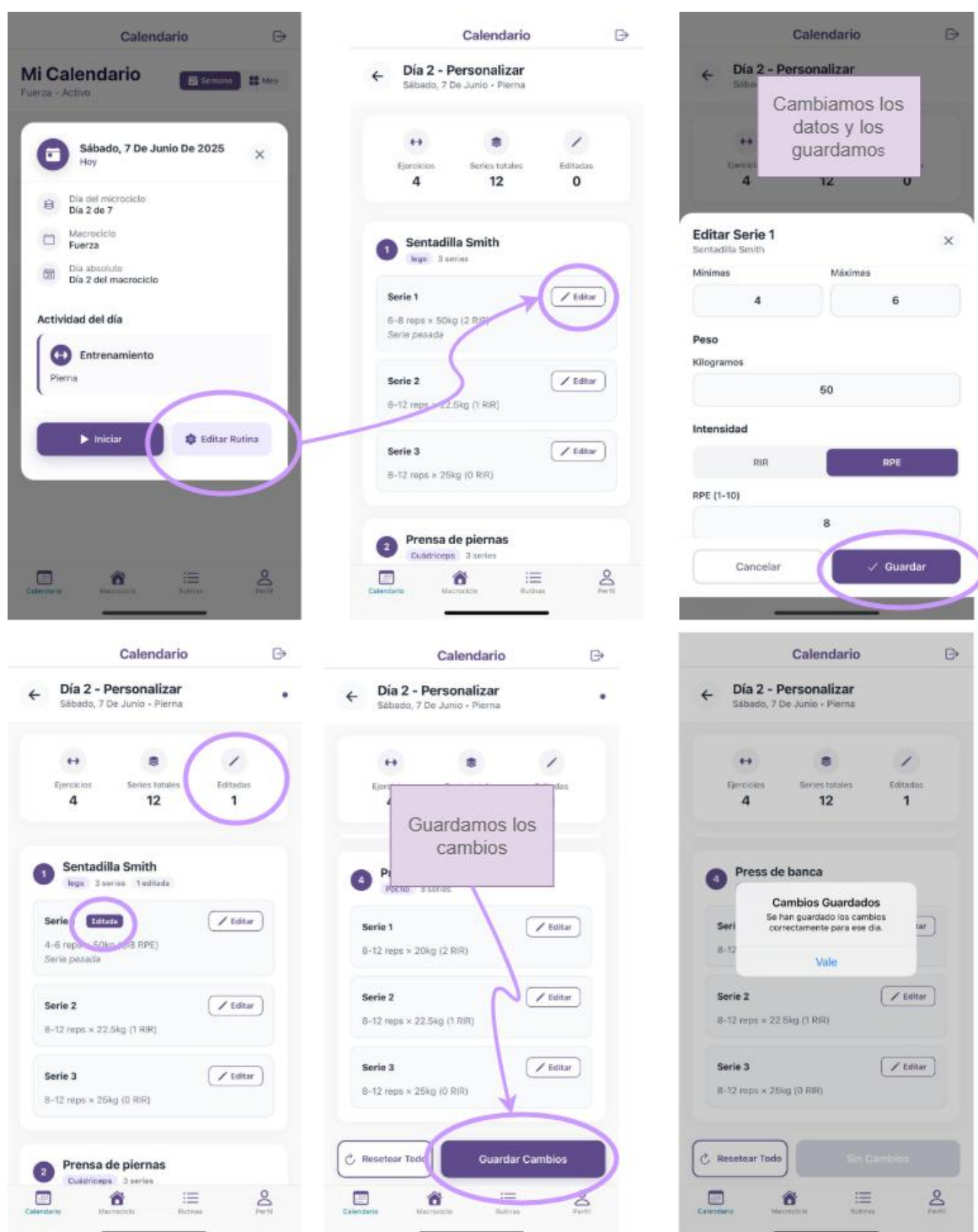
El primer problema era la posibilidad de **editar rutinas específicas** dentro de un **macrociclo** sin las validaciones adecuadas. Inicialmente, cuando un usuario quería modificar una rutina que ya estaba asignada a un macrociclo, no había control sobre si esos cambios debían afectar solo a días específicos o a toda la rutina en general. Esto podía generar **inconsistencias** donde el usuario pensaba que estaba personalizando un día concreto pero en realidad modificaba la rutina base para todos los días. Para solucionarlo, implementé el sistema de **MacrocycleDayCustomization** que permite modificar parámetros específicos por **día absoluto** sin alterar las rutinas originales. Esto se logra manteniendo tres estados: valores originales de la rutina, valores personalizados para días específicos, y valores efectivos que son los que realmente se usan durante el entrenamiento.

El segundo problema era la posibilidad de repetir **entrenamientos** del mismo día varias veces dentro de un **macrociclo**. Inicialmente no tenía control sobre la **unicidad de entrenamientos** por día absoluto de **macrociclo**, lo que permitía que un usuario pudiera entrenar múltiples veces el mismo día de su planificación, generando **datos duplicados** y confusión en el seguimiento. Para solucionarlo, implementé el método **hasAlreadyTrainedToday()** en **WorkoutSessionService** que verifica si ya existe un **WorkoutSession** para la combinación de usuario, **macrociclo** y día absoluto específico. Además, añadí **validaciones en el frontend** que consultan esta información antes de permitir iniciar un nuevo entrenamiento, mostrando un mensaje claro al usuario si ya ha entrenado ese día.

Para validar estas soluciones realicé múltiples **casos de prueba específicos**. Probé crear un **macrociclo** de 4 semanas con **microciclos** de 7 días, obteniendo como resultado 28 días totales calculados correctamente por el sistema. Testé la funcionalidad de **personalización** modificando el peso de sentadillas del día 15 de 100kg a 110kg, verificando que la rutina original se mantuviera intacta y solo el día 15 específico quedara modificado con el nuevo peso. También validé el **sistema de prevención** de entrenamientos duplicados intentando entrenar el mismo **día absoluto** múltiples veces confirmando que el sistema bloqueaba correctamente los intentos posteriores al primero.



Probé el **flujo completo** de crear una rutina, asignarla a un **macrociclo**, personalizar algunos días, y luego ejecutar entrenamientos tanto originales como personalizados. También probé **escenarios de error** como perder conexión durante un entrenamiento o cerrar la app accidentalmente mientras se está entrenando, implementando un **sistema de preservación de progreso** que guarda el estado actual y lo restaura cuando es necesario. Estas pruebas me permitieron integrar **capturas de pantalla** que muestran el comportamiento correcto del sistema en cada uno de estos casos, documentando tanto el **flujo normal** como la **gestión de errores**.



Manual Administrador

Para instalar y ejecutar **StayRPE** en su versión completa, es necesario cumplir unos **requisitos básicos** tanto para el **backend** como para el **frontend**.

Para la parte backend, necesitas tener instalado **Java 17** o superior, **Maven 3.6** o superior y **MySQL 8.0** o superior junto con **MySQL Workbench** para gestionar la **base de datos**. Se recomienda contar con al menos **4GB de RAM** para asegurar un funcionamiento fluido. Como **editor de código** puedes utilizar **IntelliJ IDEA**, **Visual Studio Code** o **Eclipse**, según tu preferencia.

En cuanto al **frontend**, debes tener **Node.js 18.0** o superior, un gestor de paquetes como **npm** o **yarn** y **Expo CLI** instalado globalmente. Es necesario tener **Git** para clonar el repositorio, y un **editor de código** (**Visual Studio Code** es la opción recomendada). Además, necesitarás un **dispositivo móvil** con la app **Expo Go** instalada, o bien un **emulador** configurado en **Android Studio**. Es imprescindible que el **backend StayRPE** esté funcionando y accesible, ya que toda la funcionalidad de la app depende de él.

Instalación Rápida

Backend

Primero instala **MySQL Workbench** desde la web oficial. Conéctate a tu **servidor MySQL local** y crea la **base de datos** ejecutando el comando SQL `CREATE DATABASE stayrpe;`. Descarga el proyecto **StayRPE backend** y sitúate en la carpeta raíz (stayrpe-backend). Puedes ejecutar el proyecto desde la terminal usando `mvn spring-boot:run`, o desde tu editor de código favorito ejecutando la clase **StayrpeApplication** directamente. **Hibernate** se encargará automáticamente de crear todas las tablas necesarias.

Frontend

Instala **Node.js** desde nodejs.org y asegúrate de tener **npm** o **yarn**. Instala **Expo CLI** globalmente ejecutando `npm install -g @expo/cli`. Clona el repositorio frontend y accede a la carpeta del proyecto con `git clone <url-del-repositorio>` y luego `cd stayrpe-frontend`. Instala las dependencias necesarias con `npm install` o `yarn install`. Configura la conexión al backend editando la variable `API_URL` en los archivos correspondientes del frontend, apuntando a tu **IP**, dominio o localhost según el entorno. Finalmente, inicia el servidor de desarrollo de Expo con `npx expo start` o `npm start`.

Configuración de la Base de Datos y Backend

Asegúrate de que el archivo **application.properties** dentro de `src/main/resources` contiene la configuración adecuada para tu base de datos **MySQL**, por ejemplo:

```
spring.datasource.url=jdbc:mysql://localhost:3306/stayrpe
spring.datasource.username=root
spring.datasource.password=tu_password_mysql
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
```

Puedes ejecutar el **backend** tanto desde la terminal (`mvn spring-boot:run`) como desde tu IDE (ejecutando la **clase principal**). El **backend** funcionará por defecto en el puerto 8080, pero puedes cambiarlo editando la propiedad `server.port`.

Configuración del Frontend y Conexión con el Backend

El **frontend** de **StayRPE** se conecta con el **backend** a través de la URL configurada en la variable `API_URL`. Debes revisar y actualizar este valor en todos los archivos donde se realicen peticiones a la API, tanto para entornos de **desarrollo local**, como para **red local** o despliegue en **servidor remoto**.

¿Cómo saber tu IP para conectar desde otros dispositivos de la red?

Si no sabes cuál es la **IP** de tu ordenador en la **red local** (por ejemplo, para conectar tu móvil al **backend** mientras ambos están en la misma WiFi), abre una terminal o símbolo del sistema en Windows y ejecuta:

Busca la **dirección IPv4** (normalmente algo como `192.168.1.xxx`). Esa es la IP que debes poner en `API_URL` para que otros dispositivos de la red puedan acceder a tu **backend**.

```
ipconfig
```

```
//En macOS o Linux el comando es:
```

```
ifconfig
```

Métodos de Ejecución

Puedes ejecutar el **frontend** de **StayRPE** de varias formas:

- **En dispositivo móvil:** Instala **Expo Go** y escanea el **QR** que te muestra el terminal tras ejecutar `npx expo start`.
- **En emulador Android:** Usa **Android Studio** y ejecuta `npx expo start --android`
- **En emulador iOS** (solo en Mac): Ejecuta `npx expo start --ios`.
- **En web:** Para pruebas rápidas, puedes usar `npx expo start --web`.

Estructura de Carpetas del Proyecto

El proyecto sigue la **estructura estándar** de aplicaciones modernas.

Frontend (stayrpe-frontend):

- **app/:** contiene las pantallas principales y los módulos (calendario, rutinas, perfil, onboarding, etc.)
- **components/:** componentes reutilizables de la app
- **package.json, app.json:** dependencias y configuración
- **README.md:** documentación general

Backend (stayrpe-backend):

- **src/main/java:** todo el **código fuente Java**, incluyendo la clase principal
- **src/main/resources:** archivos de **configuración** y **scripts SQL**
- **pom.xml:** dependencias **Maven**

Usuarios de Prueba y Datos Iniciales

Puedes **registrar nuevos usuarios** directamente desde la app **StayRPE** en la pantalla de registro. Para pruebas, se recomienda crear algunos usuarios con los siguientes datos:

- **Email:** admin@stayrpe.com | Contraseña: 123456 | Nombre: Admin | Apellidos: StayRPE
- **Email:** atleta@stayrpe.com | Contraseña: 123456 | Nombre: Atleta | Apellidos: Prueba

Desde el **backend**, también puedes crear usuarios usando herramientas como **Postman** (enviando una petición POST a /api/register), o insertarlos directamente en la base de datos con **MySQL Workbench**.

Carga opcional de ejercicios predefinidos:

No es obligatorio, pero si quieres tener una **base de datos inicial** con ejercicios clásicos organizados por grupo muscular, puedes crear un archivo llamado **data.sql** dentro de **src/main/resources** y pegar el siguiente script. Así, Spring Boot lo ejecutará automáticamente al iniciar el proyecto y tendrás los ejercicios cargados por defecto.

```
INSERT INTO exercises (name, muscle, muscle_group, description, is_custom)
VALUES
('Press de banca', 'Pecho', 'upper_body', 'Ejercicio básico para desarrollo del pectoral mayor', false),
('Press inclinado con mancuernas', 'Pecho', 'upper_body', 'Enfoque en la porción superior del pecho con mancuernas', false),

('Dominadas', 'Dorsales', 'upper_body', 'Ejercicio de tracción vertical fundamental', false),
('Remo con barra', 'Dorsales', 'upper_body', 'Ejercicio compuesto para la espalda', false),

('Peso muerto', 'Dorsales', 'full_body', 'Ejercicio compuesto fundamental para toda la cadena posterior', false),
('Remo con T-bar', 'Dorsales', 'upper_body', 'Variante de remo con agarre neutro', false),

('Sentadillas', 'Cuádriceps', 'lower_body', 'Ejercicio fundamental para piernas', false),
('Prensa de piernas', 'Cuádriceps', 'lower_body', 'Ejercicio de empuje para cuádriceps', false),

('Peso muerto rumano', 'Isquiotibiales', 'lower_body', 'Ejercicio fundamental para isquiotibiales', false),
('Curl femoral acostado', 'Isquiotibiales', 'lower_body', 'Ejercicio de aislamiento', false),

('Hip thrust', 'Glúteos', 'lower_body', 'Ejercicio específico para glúteos', false),
('Sentadillas sumo', 'Glúteos', 'lower_body', 'Variante de sentadilla para glúteos', false),

('Press militar', 'Hombros', 'upper_body', 'Ejercicio fundamental para hombros', false),
('Press con mancuernas sentado', 'Hombros', 'upper_body', 'Variante sentada del press', false),
```


Verificación de la Instalación y Funcionalidad

Para verificar que todo está correctamente configurado:

- Accede a la ruta **http://localhost:8080/api/publico** en tu navegador y asegúrate de recibir una **respuesta JSON**.
- Usa **MySQL Workbench** para comprobar que se han creado las **tablas necesarias** (usuarios, rutinas, ejercicios, etc.).
- En la **app StayRPE**, prueba a registrar un nuevo usuario. Si el registro funciona, la **conexión está bien configurada**.
- Revisa que puedes acceder, crear y modificar rutinas, **macrociclos** y ver tu perfil.

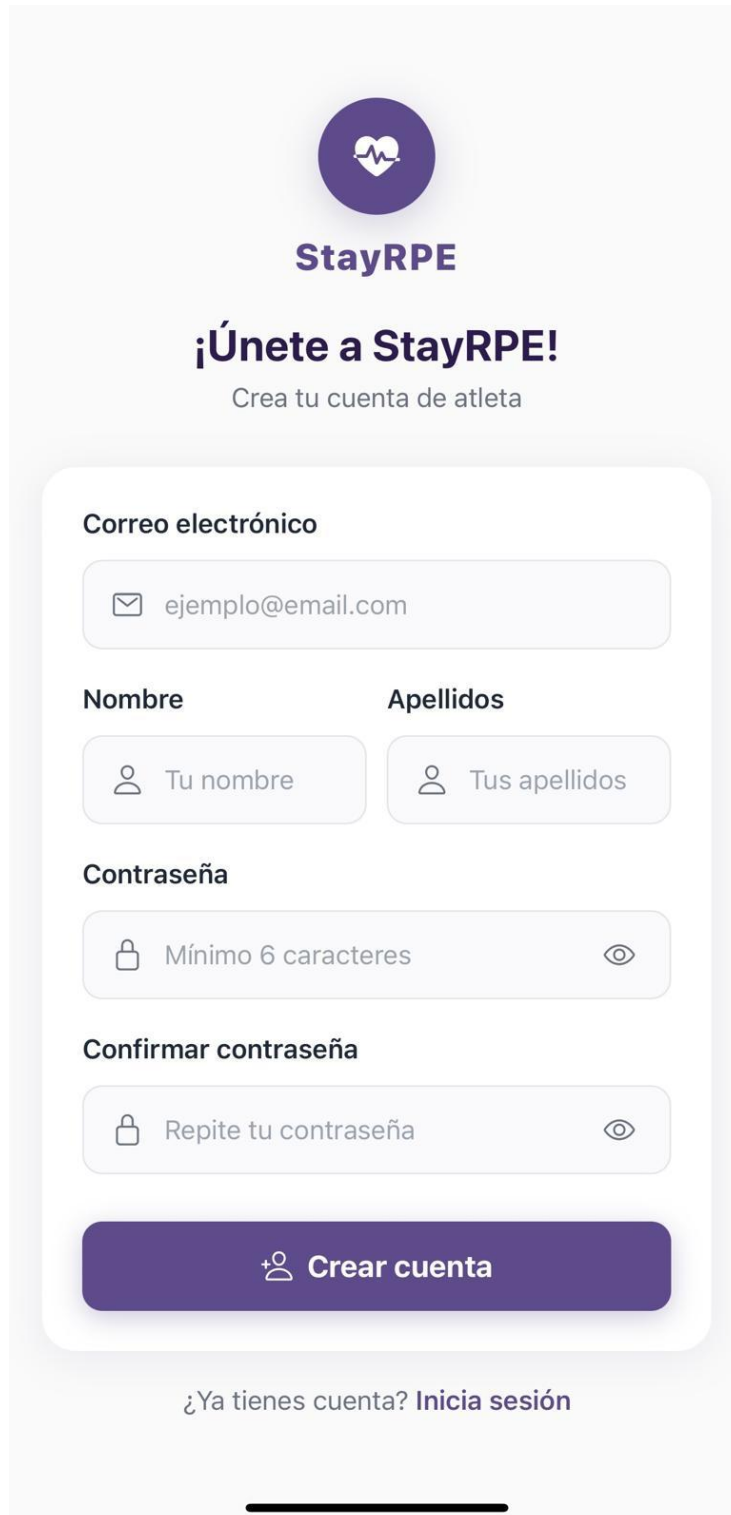
Logs y Debugging

Los **logs del backend** aparecen en la consola donde ejecutes el proyecto o en la ventana de tu **IDE**. Puedes ajustar el **nivel de detalle** en **application.properties**.

Para el **frontend**, los **logs** se ven en la terminal donde ejecutaste **expo start**, o en la consola de desarrollo de **Expo Go**.

Manual Usuario

Pantalla de Registro



The image shows a mobile app registration screen for 'StayRPE'. At the top, there is a purple circular logo with a white heart and pulse line. Below the logo, the text 'StayRPE' is displayed in a bold, dark purple font. Underneath, the heading '¡Únete a StayRPE!' is shown in a large, bold, dark purple font, followed by the subtext 'Crea tu cuenta de atleta' in a smaller, regular dark purple font. The registration form is contained within a white rounded rectangle with a subtle shadow. It features several input fields: 'Correo electrónico' with a placeholder 'ejemplo@email.com', 'Nombre' and 'Apellidos' each with a person icon and placeholder text, 'Contraseña' with a lock icon, placeholder 'Mínimo 6 caracteres', and an eye icon for toggling visibility. Below this is the 'Confirmar contraseña' section with a similar lock icon, placeholder 'Repite tu contraseña', and an eye icon. A large purple button with a white person icon and the text 'Crear cuenta' is positioned at the bottom of the form. Below the form, the text '¿Ya tienes cuenta? Inicia sesión' is displayed in a dark purple font. A black horizontal line at the very bottom represents the mobile home indicator bar.

StayRPE

¡Únete a StayRPE!

Crea tu cuenta de atleta

Correo electrónico

✉ ejemplo@email.com

Nombre **Apellidos**

👤 Tu nombre 👤 Tus apellidos

Contraseña

🔒 Mínimo 6 caracteres 👁

Confirmar contraseña

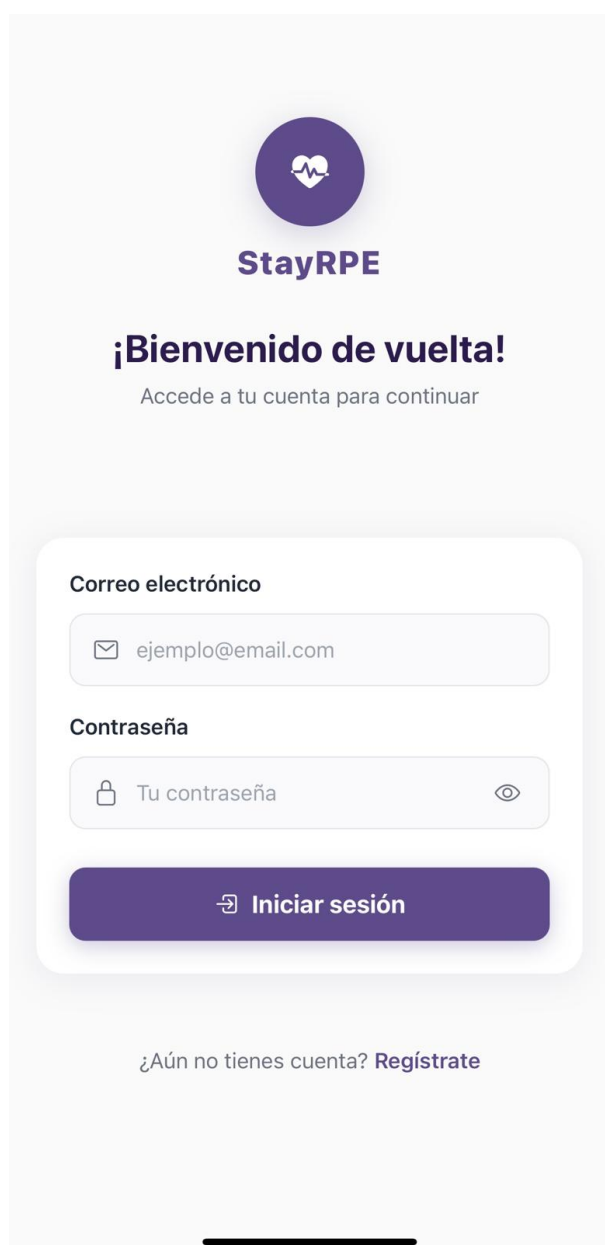
🔒 Repite tu contraseña 👁

Crear cuenta

¿Ya tienes cuenta? **Inicia sesión**

La **pantalla de registro** permite a los nuevos usuarios crear una cuenta en **StayRPE**. El formulario solicita **información básica** como correo electrónico, nombre, apellidos y contraseña. Todos los campos incluyen **validación en tiempo real** para asegurar que la información ingresada sea correcta. Una vez completado el registro, el usuario puede proceder al **proceso de configuración inicial** de su perfil.

Inicio de Sesión



StayRPE

¡Bienvenido de vuelta!

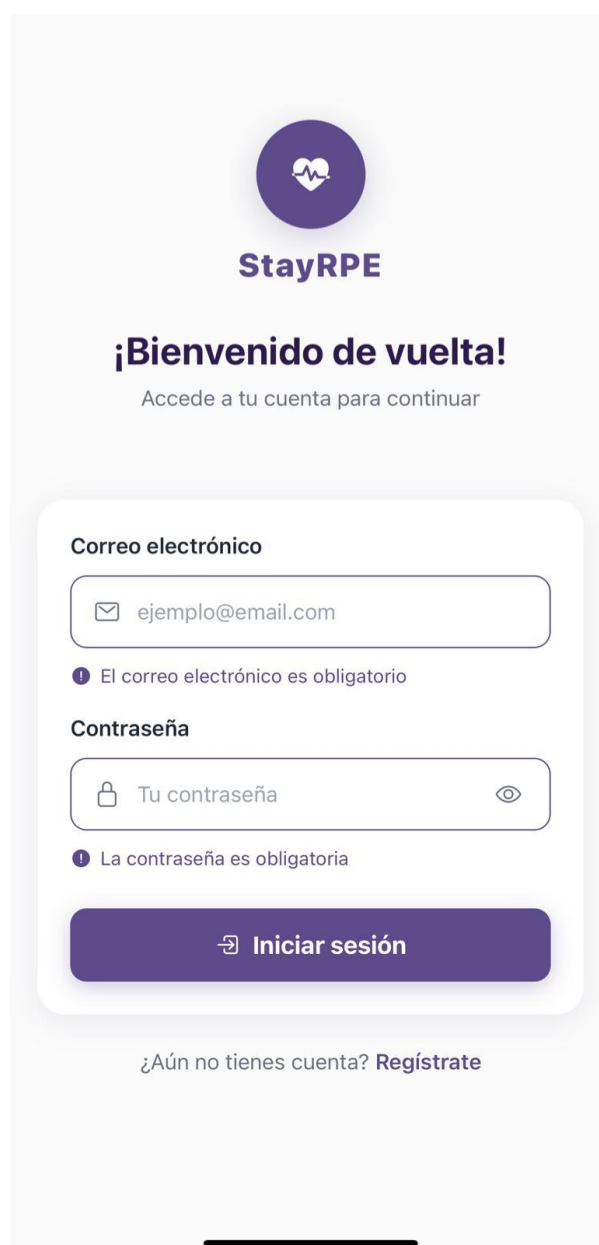
Accede a tu cuenta para continuar

Correo electrónico

Contraseña

[Iniciar sesión](#)

¿Aún no tienes cuenta? [Regístrate](#)



StayRPE

¡Bienvenido de vuelta!

Accede a tu cuenta para continuar

Correo electrónico

❗ El correo electrónico es obligatorio

Contraseña

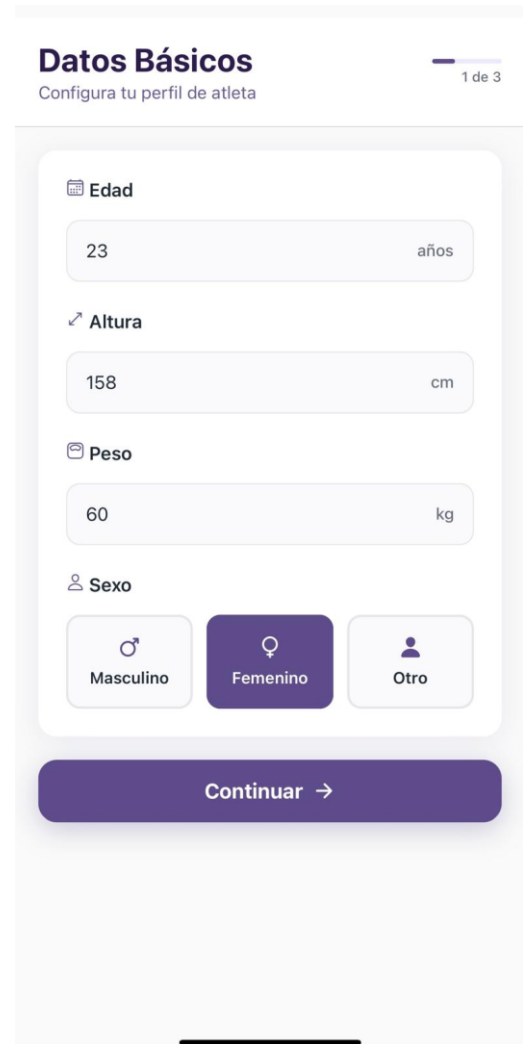
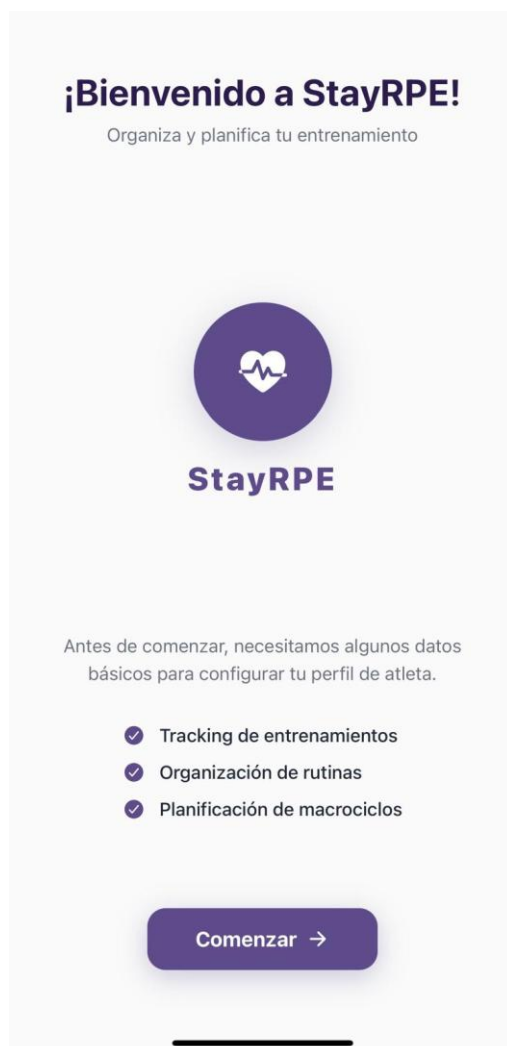
❗ La contraseña es obligatoria

[Iniciar sesión](#)

¿Aún no tienes cuenta? [Regístrate](#)

La **pantalla de login** permite a usuarios existentes acceder a sus cuentas mediante su **correo electrónico** y **contraseña**. La interfaz mantiene **consistencia visual** con la **pantalla de registro** y incluye opciones para **mostrar/ocultar la contraseña**. Si las credenciales son incorrectas, se muestran **mensajes de error** claros. Los usuarios nuevos pueden navegar fácilmente a la **pantalla de registro** desde un enlace en la parte inferior.

Proceso de Onboarding



Datos Básicos 1 de 3

Configura tu perfil de atleta

Edad

23 años

Altura

158 cm

Peso

60 kg

Sexo

☐ Masculino ☒ Femenino ☐ Otro

[Continuar →](#)

Experiencia

Define tu nivel de entrenamiento

3 de 3

Principiante

Menos de 6 meses de experiencia en entrenamiento estructurado

Intermedio

6 meses a 2 años de entrenamiento regular y estructurado

Avanzado

Más de 2 años de entrenamiento constante y especializado

Completar configuración ✓

Objetivo Principal

¿En qué te quieres enfocar?

2 de 3

Perder grasa

Enfoque en reducir grasa corporal y mejorar definición

Ganar músculo

Priorizar el crecimiento muscular y aumento de masa

Ganar fuerza

Incrementar la fuerza máxima y capacidad de carga

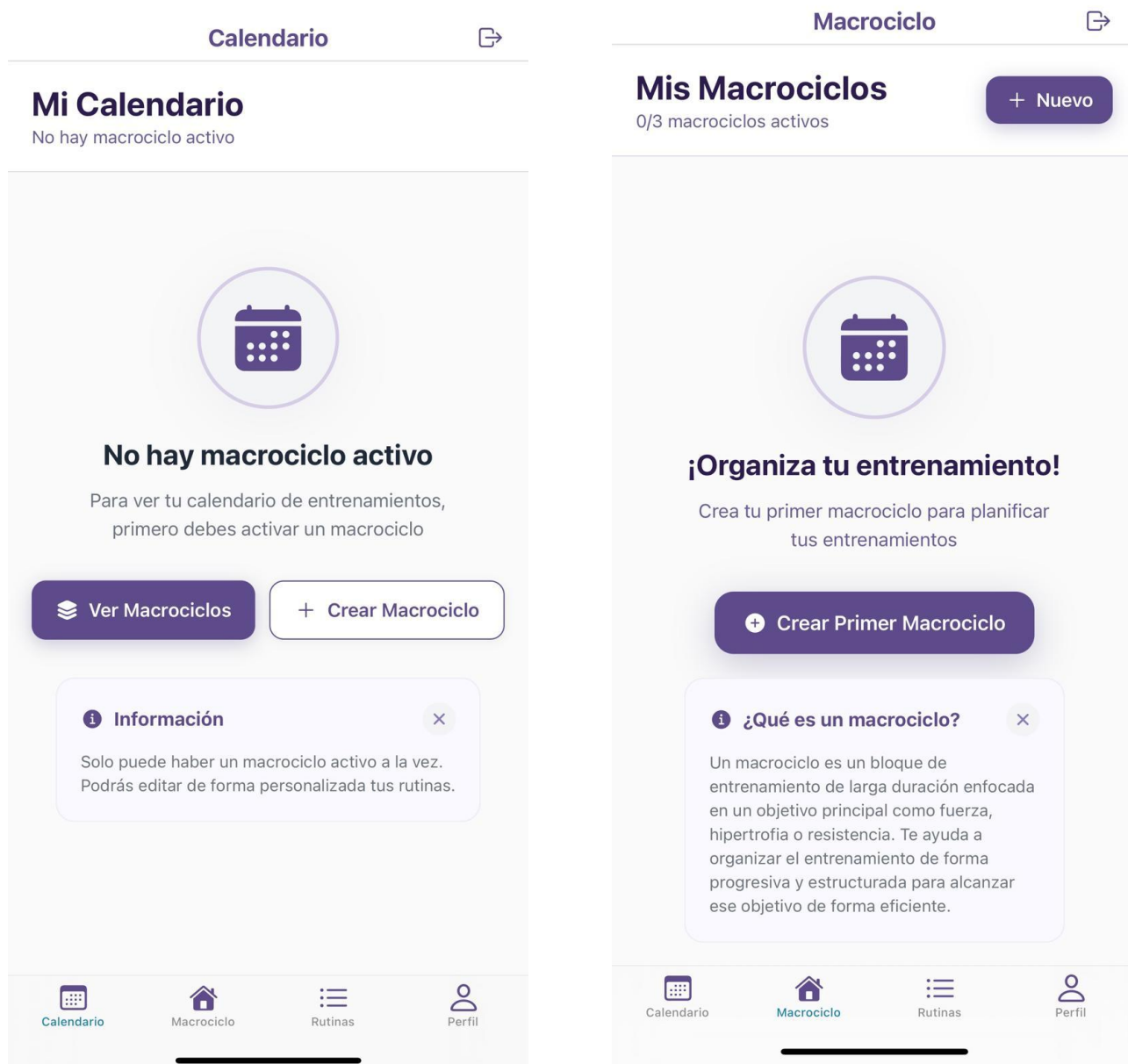
Mantener forma

Conservar el nivel actual de forma física y rendimiento

Continuar →

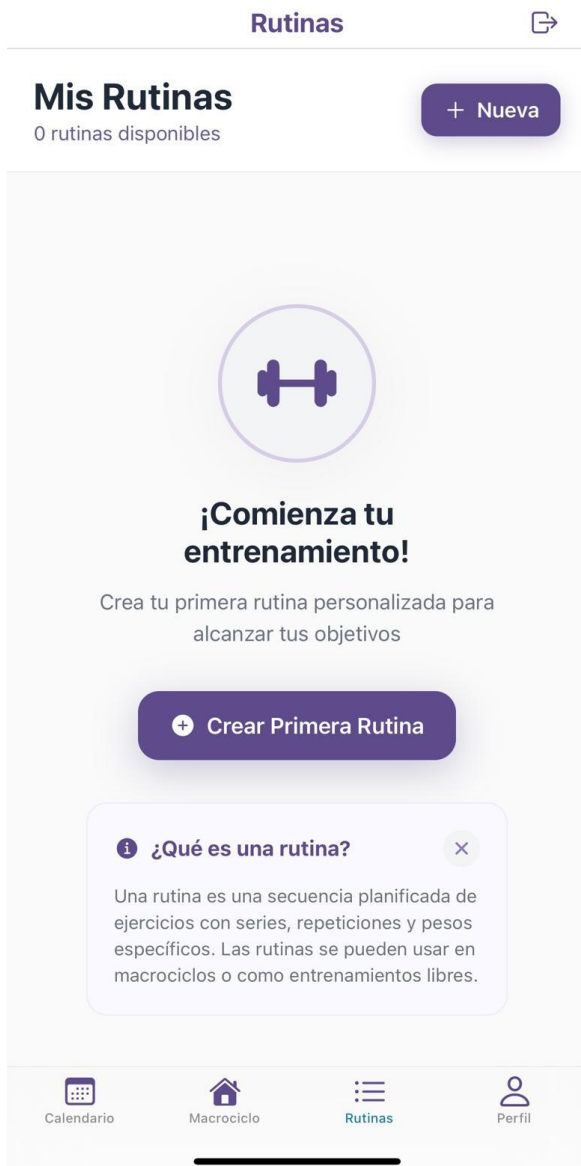
Una vez completado el registro, **StayRPE** guía al usuario a través de un **proceso de onboarding** que se ejecuta únicamente la primera vez que accede a la aplicación. Este proceso consta de **tres pasos esenciales** para personalizar la experiencia de entrenamiento: primero se recopilan **datos básicos** como edad, altura, peso y sexo; luego se selecciona el **objetivo principal** de fitness (perder grasa, ganar músculo, ganar fuerza o mantener forma); y finalmente se establece el **nivel de experiencia** del usuario (principiante, intermedio o avanzado). La aplicación muestra una barra de progreso visual para indicar el avance a través de estas etapas. Una vez completado este proceso, el usuario accede directamente a la **aplicación principal** y no necesita repetir esta configuración.

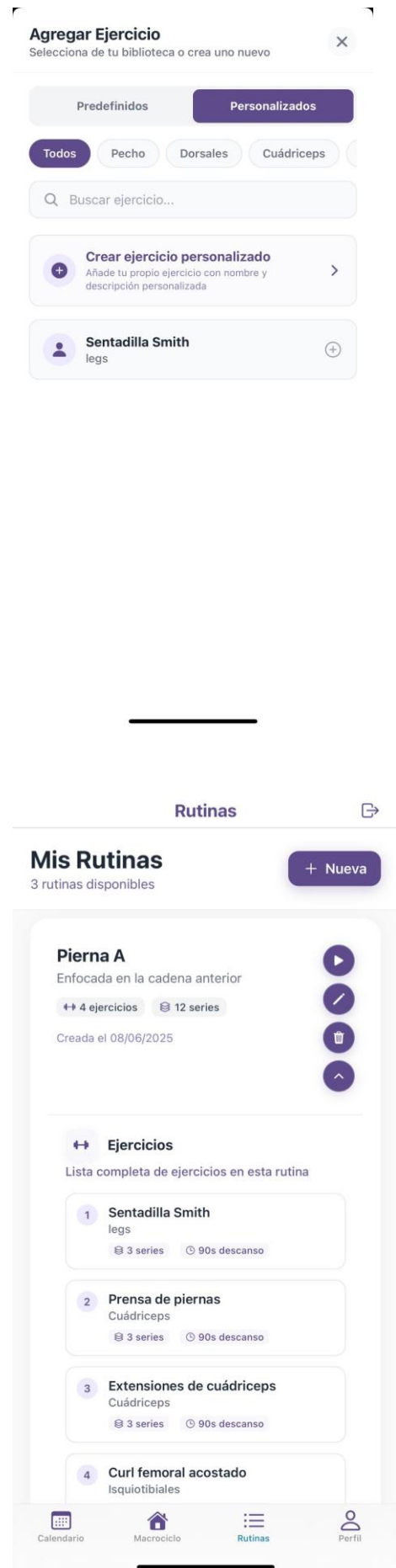
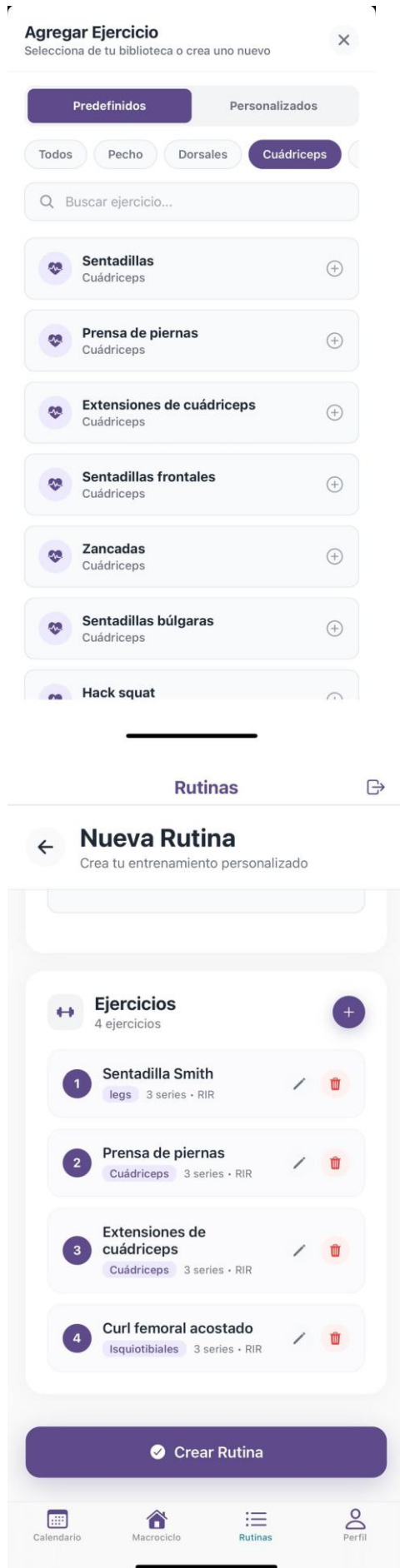
Tarjetas Informativas del Sistema



StayRPE incluye **tarjetas informativas** que aparecen la primera vez que el usuario visita cada sección, proporcionando **guías contextuales** y explicaciones de **funcionalidades específicas**. La aplicación utiliza **AsyncStorage** para recordar qué tarjetas han sido cerradas por cada usuario específico, creando **claves únicas** basadas en el nombre de usuario (por ejemplo, "calendar_info_hidden_username"). Una vez cerradas, el estado se guarda localmente en el dispositivo y no vuelven a mostrarse, manteniendo la **interfaz limpia** para usuarios experimentados mientras ofrecen **orientación inicial** a nuevos usuarios. El contenido de cada tarjeta está adaptado específicamente a la funcionalidad de la pantalla donde aparece.

Crear Rutinas







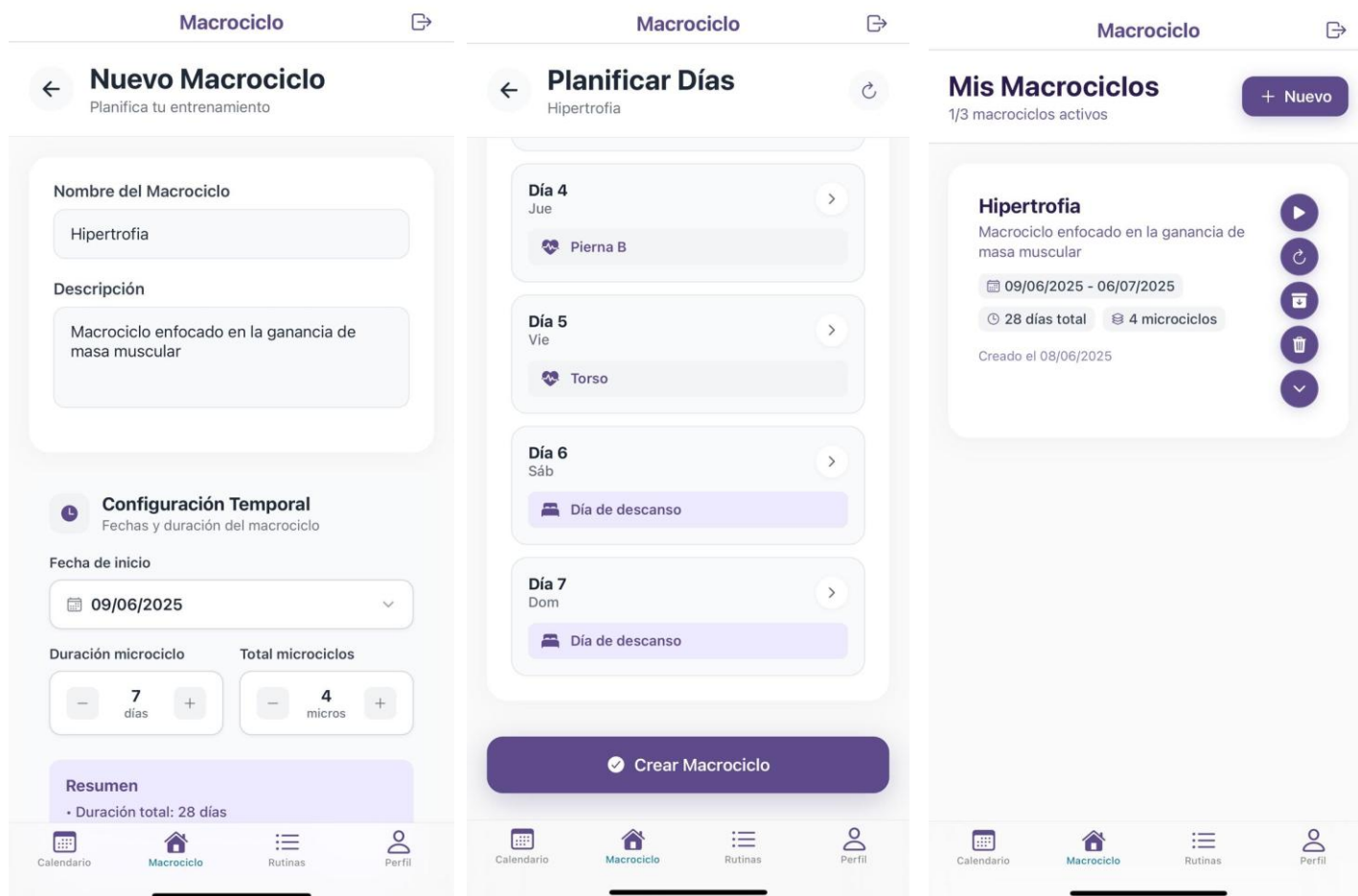
StayRPE facilita la **creación de rutinas** mediante un **flujo intuitivo** que comienza con una pantalla donde el usuario define el **nombre y descripción** de la rutina. Inicialmente, la sección de ejercicios muestra un **estado vacío** con una clara **llamada a la acción** para agregar el primer ejercicio. Al presionar el **botón "+"**, se despliega un **modal** que permite seleccionar **ejercicios existentes** de la **biblioteca personal** o crear nuevos **ejercicios personalizados**. Una vez añadido un ejercicio, la aplicación abre automáticamente una **pantalla de configuración detallada** donde se pueden definir series, repeticiones, pesos, intensidad (**RIR/RPE**) y **tiempos de descanso**, proporcionando **control completo** sobre cada aspecto del entrenamiento antes de guardar la rutina.

Una vez creada, cada rutina incluye opciones de **gestión completas**: **botón de inicio** (icono de play) para **entrenamientos libres** independientes de **macrociclos**, **edición** (lápiz) para modificar la configuración, **eliminación** con restricciones que previenen borrar **rutinas activas** en **macrociclos**, y una función de **despliegue** (flecha expandible) que muestra un **resumen detallado** de todos los ejercicios incluidos con sus respectivas configuraciones.

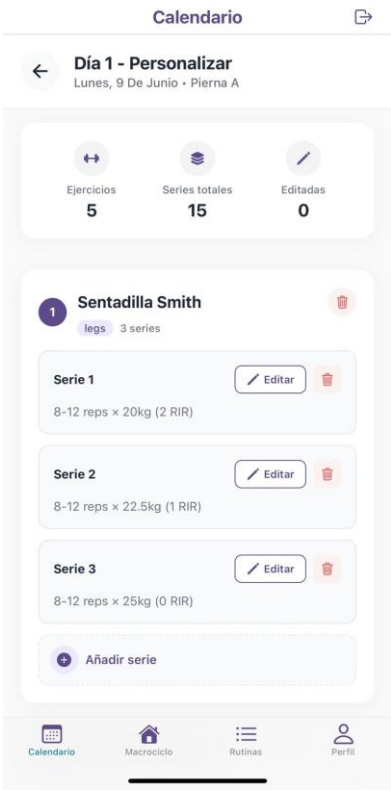
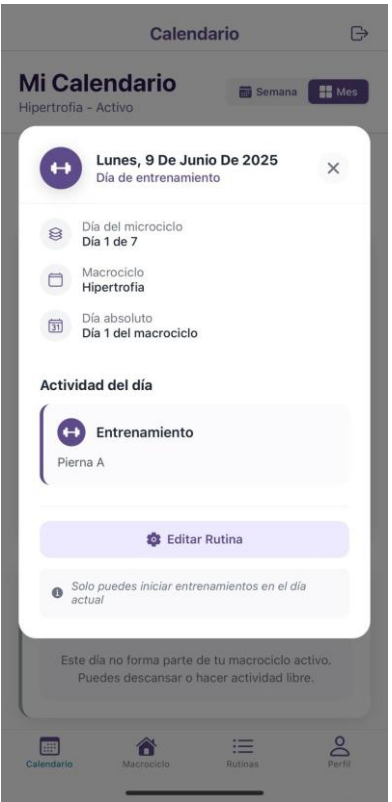
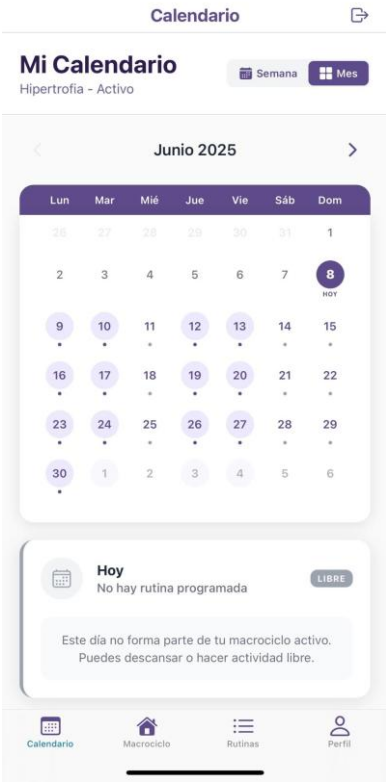
El sistema incorpora **protección automática** que detecta cuando una rutina está siendo utilizada en **macrociclos activos**. En estos casos, si el usuario intenta modificar la rutina, el sistema bloquea la **edición directa** para evitar conflictos en la **planificación existente**, pero ofrece automáticamente la opción de crear una **copia duplicada** de la rutina. Esta funcionalidad permite mantener la **integridad de los macrociclos activos** mientras proporciona flexibilidad para crear variaciones de rutinas. Las **validaciones y alertas contextuales** informan al usuario sobre estas limitaciones y las alternativas disponibles.

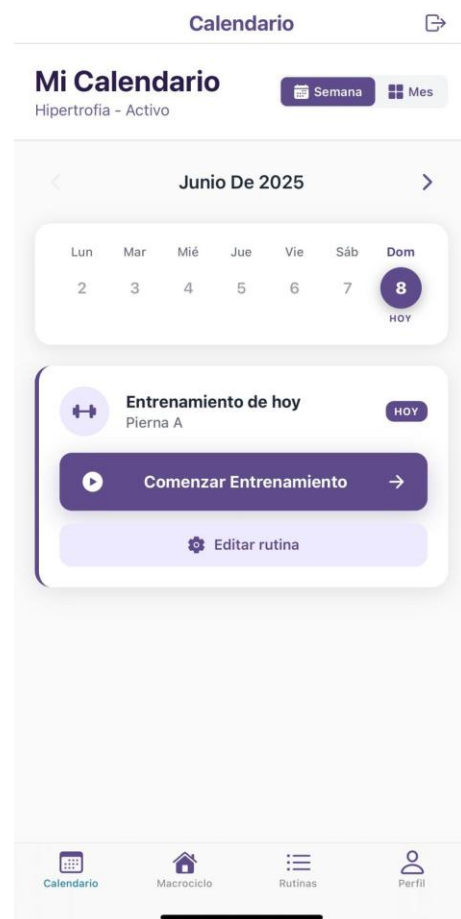
Creación De Macrociclos

La creación de **macrociclos** en **StayRPE** sigue un **proceso estructurado** que comienza con la definición de **parámetros básicos y temporales**. En la primera pantalla, el usuario establece el **nombre y descripción** del **macrociclo**, seguido de una sección de "**Configuración Temporal**" donde se define la **fecha de inicio** mediante un **selector de calendario**, la **duración de cada microciclo** (ajustable entre 1-14 días) y el **número total de microciclos**, con **controles intuitivos** de incremento/decremento. La aplicación calcula automáticamente la **duración total** y muestra un **resumen en tiempo real**. El siguiente paso dirige al usuario a la pantalla de "**Planificar Días**" donde debe asignar rutinas **específicas** o **días de descanso** a cada día del **microciclo**, mostrando claramente **el día de la semana** correspondiente y permitiendo seleccionar entre **rutinas existentes** o marcar como **día de descanso**. Una vez completada la **planificación diaria**, el macrociclo se crea y aparece en la **vista principal** con opciones de gestión completas: **activación/desactivación**, **reseteo de fechas**, **archivado**, **eliminación** y **expansión** para ver detalles, junto con restricciones inteligentes que previenen la eliminación de macrociclos activos.



Macrociclo Activo





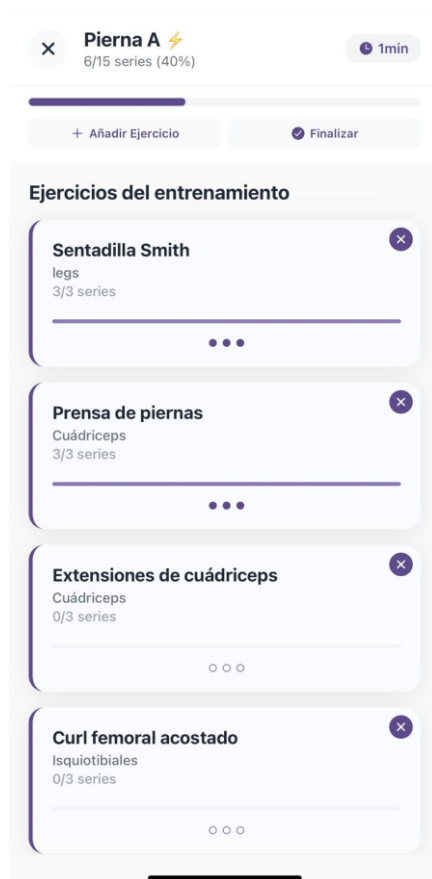
Una vez activado un **macrociclo** en **StayRPE**, este se integra completamente con el **sistema de calendario** para proporcionar una **experiencia de entrenamiento estructurada** y guiada. El calendario muestra automáticamente la **planificación diaria** con días marcados según la configuración del **macrociclo**, diferenciando visualmente entre **días de entrenamiento** (con rutinas específicas) y **días de descanso** mediante **códigos de color** y indicadores.

Al seleccionar un **día específico** del **macrociclo activo**, se despliega un **modal detallado** que muestra **información contextual** como el **día del microciclo** (Día 1 de 7), el **macrociclo al que pertenece** ("Hipertrofia"), el **día absoluto** del macrociclo, y la **actividad programada** (rutina "Pierna A" o descanso). Desde este modal, los usuarios pueden iniciar entrenamientos directamente o acceder a la funcionalidad **"Editar Rutina"** para personalizar el entrenamiento del día específico, manteniendo la flexibilidad dentro de la **estructura planificada del macrociclo**.

Además, el sistema ofrece **opciones avanzadas de gestión** para maximizar la **reutilización de macrociclos**. Los usuarios pueden **resetear un macrociclo completo**, lo que permite cambiar la **fecha de inicio** y volver a comenzar el programa desde el día 1, manteniendo toda la estructura de entrenamiento pero adaptándose a nuevas **circunstancias temporales**. Esta funcionalidad es especialmente útil para repetir **macrociclos exitosos** o reiniciar después de interrupciones por vacaciones o lesiones.

Es importante destacar que cuando se **desactiva un macrociclo**, se pierden permanentemente todas las **personalizaciones** y ediciones realizadas en días específicos. Esta medida asegura la **integridad del sistema** y evita **conflictos de datos**, pero requiere que los usuarios sean conscientes de que las **modificaciones personalizadas** no se conservan al desactivar. Por tanto, se recomienda completar un **macrociclo** antes de desactivarlo o estar seguro de que no se necesitarán las **personalizaciones futuras** antes de proceder con la desactivación.

Iniciar Entrenamiento



Pierna A ⚡
6/15 series (40%) 1min

+ Añadir Ejercicio Finalizar

Ejercicios del entrenamiento

- Sentadilla Smith**
legs
3/3 series
- Prensa de piernas**
Cuádriceps
3/3 series
- Extensiones de cuádriceps**
Cuádriceps
0/3 series
- Curl femoral acostado**
Isquiotibiales
0/3 series



Sentadilla Smith
0/3 series 0min

Series (3) Añadir

1 2 3 +

Serie 1 Eliminar

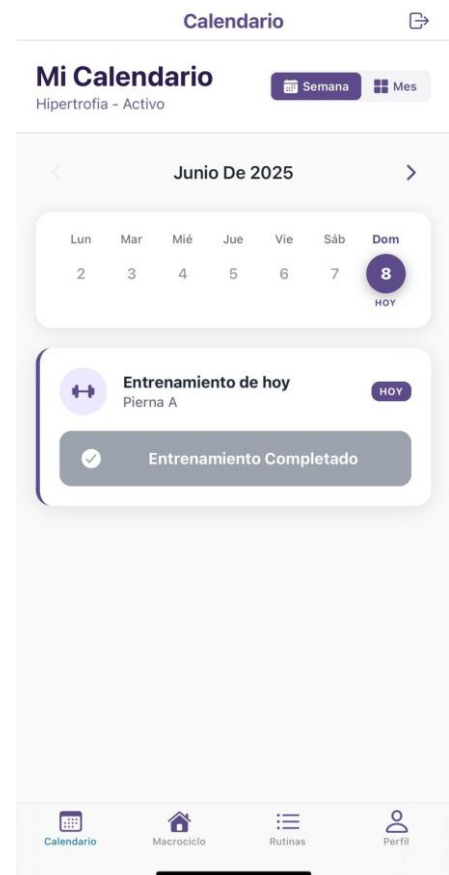
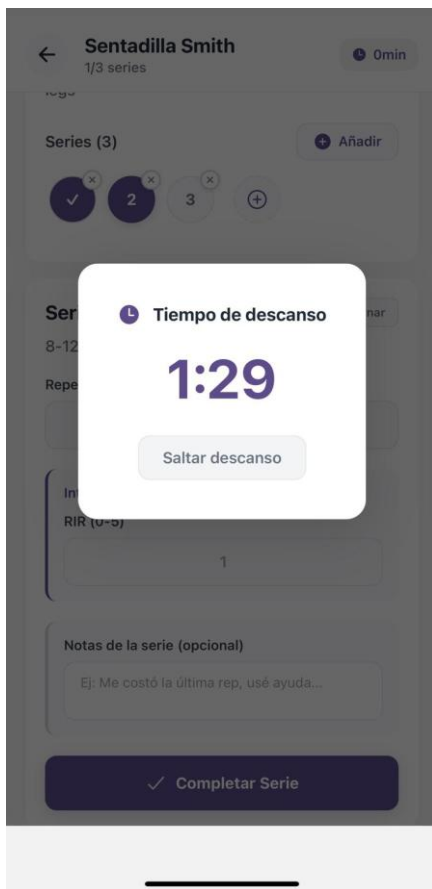
8-12 reps x 20kg (2 RIR)

Repeticiones	Peso (kg)
8	20

Intensidad percibida:
RIR (0-5)
2

Notas de la serie (opcional)
Subirle el peso el próximo día

✓ Completar Serie



El **sistema de inicio de entrenamientos** en **StayRPE** ofrece un **proceso fluido** e inteligente que adapta la experiencia según el contexto del usuario. Los entrenamientos pueden iniciarse de dos formas principales: como **entrenamientos libres** desde la vista de **rutinas individuales** o como parte de un macrociclo activo desde el calendario. Al iniciar un entrenamiento, la aplicación carga automáticamente la **rutina seleccionada** y presenta una **vista de ejercicios** donde cada ejercicio muestra su **progreso individual** mediante **indicadores visuales de series completadas** (puntos de progreso). La interfaz permite navegar fluidamente entre la **vista general del entrenamiento** y la **vista detallada** de cada ejercicio, mostrando información específica como **repeticiones objetivo**, **peso**, y valores de **RIR/RPE**. Durante la ejecución, el sistema incluye un **temporizador de descanso automático** que aparece como **modal** tras completar cada serie, permitiendo al usuario seguir los **tiempos recomendados** o saltarlos según su criterio. Los entrenamientos pueden **personalizarse en tiempo real** agregando nuevos ejercicios desde la **biblioteca de ejercicios**, eliminando series o ejercicios si es necesario, y registrando **notas específicas** tanto por serie como para el entrenamiento completo. Al finalizar, la aplicación presenta **estadísticas completas** de duración, **series completadas** y **porcentaje de finalización**, guardando automáticamente todos los datos en el **historial** con información detallada de cada ejercicio y serie realizada.

Perfil De Usuario

Perfil

Mi Perfil

Información personal y configuración

Prueba Styrpe

prueba-admin@gmail.com

Edad

23 años

Altura

158 cm

Peso

60 kg

Información

Tus objetivos y experiencia

Sexo

Femenino

Objetivo

Ganar músculo

Nivel de experiencia

Avanzado

Opciones

Configuración y datos

Calendario

Macro ciclo

Rutinas

Perfil

Perfil

Macro ciclos Archivados

1 macro ciclo archivado

Hipertrofia

ARCHIVADO

Macro ciclo enfocado en la ganancia de masa muscular

08/06/2025 - 05/07/2025

28 días total 4 micro ciclos

Archivado • Creado el 08/06/2025

Calendario

Macro ciclo

Rutinas

Perfil

Perfil

Historial de Entrenamientos

1 entrenamientos

Todos

Libres

Hipertrofia

Más reciente

Información

Toca cualquier entrenamiento para ver los detalles completos de ejercicios y series realizadas.

Pierna A

40%

Hipertrofia - Día 1

Hoy, 11:42

1m

6/15 series

5 ejercicios

1295 kg

Calendario

Macro ciclo

Rutinas

Perfil

Perfil

Editar Perfil

Actualiza tu información personal

Guardar

Información Personal

Datos básicos de tu perfil

Nombre

Prueba

Apellido

Styrpe

Edad

23

Altura (cm)

158

Peso (kg)

60

Sexo

Información biológica

Masculino

Femenino

Otro

Objetivo de Fitness

¿Cuáles son tus metas principales?

Calendario

Macro ciclo

Rutinas

Perfil

La **gestión del perfil** en **StayRPE** reúne toda la **información personal** del usuario y su **progreso de entrenamiento** en un espacio organizado y fácil de usar. La **pantalla principal del perfil** incluye los **datos básicos** (nombre, edad, altura, peso), la **selección de género**, y permite configurar **objetivos de fitness** según lo que cada persona quiera conseguir.

Una de las funciones más útiles es el "**Historial de Entrenamientos**", que guarda automáticamente cada **sesión realizada** con datos como cuánto tiempo duró, cuántas series se hicieron, qué ejercicios se completaron y el **peso total movido**, todo organizado con **filtros por tipo de entrenamiento** y ordenado por fecha. También se mantiene un archivo de los macrociclos anteriores con toda su información: cuándo se crearon, cuánto duraron y cuántos **microciclos** tenían, dando la opción de **reactivarlos o eliminarlos** si ya no se necesitan. Todo esto se presenta de forma visual con tarjetas que muestran **porcentajes de progreso** e indicadores claros, haciendo que sea súper sencillo revisar cómo va evolucionando el entrenamiento y mantener la motivación viendo el **progreso real** conseguido con el tiempo.

Viabilidad tecno-económica.

StayRPE es una aplicación móvil orientada a la **periodización** y el seguimiento del entrenamiento en deportes de fuerza como **powerlifting**, **halterofilia**, **culturismo** y **strongman**, basada en la **metodología RPE** (Rate of Perceived Exertion). Este análisis estudia la viabilidad económica y técnica de una futura puesta en producción, partiendo del prototipo desarrollado como proyecto de fin de ciclo.

Si **StayRPE** evoluciona hacia un **producto real**, podría diferenciarse claramente de otras apps de fitness tracking como **Hevy**, **Strong** o **Jefit** gracias a su **especialización en deportes de fuerza** y su enfoque avanzado en periodización y personalización. El proyecto se **inspira** en casos como **MyFitnessPal**, que partió como **indie** y demostró que la innovación bien enfocada puede triunfar incluso en mercados saturados.

A futuro, la **estrategia** más efectiva de lanzamiento pasaría por **contactar y apoyar a entrenadores y atletas del sector**, ofreciéndoles la app como herramienta real para su día a día, invitándoles a probarla y a participar activamente en su mejora, asegurando así una primera base de usuarios comprometidos y un producto ajustado a las necesidades reales del nicho.

Costes y Viabilidad del Frontend

El **coste monetario directo anual** del **frontend** sería muy bajo, ya que la inversión principal es en **horas de trabajo personal** y no en dinero. Los únicos **gastos reales** serían:

- **Publicación en App Store:** 99 €/año
- **Publicación en Google Play:** 25 € (pago único)
- **Herramientas de diseño como Figma:** gratis o hasta 144 €/año si se usa versión Pro
- **Servicios opcionales Pro para despliegues automáticos:** hasta 99 €/mes, aunque no son obligatorios en la fase inicial

Usando **tecnologías open source** y **planes gratuitos** siempre que sea posible, el rango total de **costes monetarios directos** estaría entre **124 € y 1.367 € al año**, según los servicios contratados.

Para **rentabilizar la inversión**, con un **modelo freemium** y **precios competitivos** (4,99 €/mes para usuarios premium, 39,99 €/año, y 9,99 €/mes para entrenadores), bastaría captar unos 824 usuarios premium para alcanzar el punto de equilibrio. Dado que la tasa de

conversión en este nicho suele estar en torno al 5%, sería suficiente llegar a 16.480 usuarios totales, un objetivo realista teniendo en cuenta **benchmarks** como **Hevy**, que supera los 100.000 usuarios activos.

El enfoque siempre sería permitir que los usuarios tengan acceso gratuito a la mayor parte de **la funcionalidad básica** de la app, y fomentar el paso a **premium** a través de **métodos suaves**, no restrictivos, como **mejoras opcionales, funciones avanzadas o personalización extra**, evitando limitar de forma agresiva la experiencia de quienes usan la versión gratuita.

Viabilidad y Costes del Backend

El **backend** puede mantenerse desde el primer día con una infraestructura en la nube optimizada y un **coste anual** muy bajo. Los principales **gastos reales** incluirían:

- **Servidor cloud dedicado:** entre 60 € y 120 € al año, suficiente para cubrir la carga inicial y un crecimiento moderado.
- **Almacenamiento y copias de seguridad:** aproximadamente 30 € a 50 € al año.
- **Servicio de correo electrónico transaccional:** unos 20 € al año.
- **Herramientas básicas de desarrollo, integración y monitorización:** gratuitas en sus versiones para proyectos pequeños o en fase inicial.
- **Otras utilidades como CDN y certificados SSL:** pueden obtenerse sin coste adicional en la mayoría de proveedores básicos.

En total, el coste **monetario real** del **backend** durante el primer año estaría en torno a **110 €-190 €**, cubriendo hasta los primeros mil usuarios sin necesidad de invertir más. Escalar hasta 10.000 usuarios supondría un incremento muy gradual y controlado de estos costes, sin que el presupuesto se dispare.

El **punto de equilibrio** del **backend** es muy bajo: solo tres usuarios premium ya cubrirían los costes anuales de la infraestructura. Con 60 usuarios totales (5% premium) se lograría una base rentable. El **beneficio anual potencial** varía entre 883 € (500 usuarios) y 9.398 € (5.000 usuarios), según el ritmo de adopción.

Factores de éxito y mitigación de riesgos

- **Enfoque especializado en deportes de fuerza y periodización real**, diferenciándose de apps más genéricas.
- **Crecimiento impulsado** por la **comunidad y feedback** de entrenadores y atletas.

- **Stack técnico robusto, escalable** y con **costes controlados**.
- **Riesgos mitigados** con backups automáticos y elección de infraestructuras que permiten escalar sin rediseñar.
- **El nicho fuerza** en el mercado español y latinoamericano está menos saturado, lo que facilita la captación de usuarios.

Conclusión

StayRPE es un proyecto indie altamente viable tanto en **frontend** como en **backend**, con una **inversión monetaria mínima** y **potencial de escalado realista**. El éxito dependerá de desarrollar un MVP sólido, implicar a la comunidad desde el inicio y apostar por la mejora continua basada en las **necesidades reales del sector**.

El **modelo freemium**, la **especialización** y la **estructura técnica** permiten competir desde el primer día con un presupuesto reducido, asegurando la **rentabilidad** a partir de una base pequeña de **usuarios premium**. Toda la inversión principal es en tiempo de trabajo personal y no en costes económicos directos, lo que hace aún más atractiva la viabilidad del proyecto para su posible paso a **producción**.

Este análisis es una **estimación orientativa** basada en escenarios futuros y serviría como base para una posible puesta en producción y escalado del proyecto **StayRPE**.

Trabajo futuro.

Aunque **StayRPE** ya ofrece una **base sólida** y funcional para la **planificación y seguimiento de entrenamientos de fuerza**, existen múltiples **mejoras y nuevas funcionalidades** que podrían implementarse en futuras versiones para aportar aún más valor a los usuarios y diferenciar la aplicación de otras **alternativas del mercado**. Algunas de las líneas de trabajo futuro más interesantes son:

- **Sistema entrenador-atleta con roles específicos:** Implementar un sistema de roles donde los usuarios puedan registrarse como "**Entrenador**" o "**Atleta**". Los entrenadores tendrían la capacidad de añadir clientes a través de **invitaciones por correo electrónico**, crear y asignar macrociclos **personalizados** a cada **atleta**, y hacer **seguimiento del progreso** de todos sus deportistas desde un panel centralizado. Los atletas podrían aceptar invitaciones de entrenadores y recibir automáticamente los **planes de entrenamiento asignados**.
- **Sistema de invitaciones y asignación de planes:** Desarrollar un **flujo completo** donde el **entrenador** pueda enviar **invitaciones por email** a potenciales clientes, gestionar una lista de atletas activos, asignar **macrociclos específicos** con un solo clic, y enviar actualizaciones o modificaciones de rutinas que se **sincronicen automáticamente** en el dispositivo del **atleta**.
- **Notificaciones y recordatorios automáticos:** Implementar un **sistema de avisos** que recuerde a los usuarios cuándo toca entrenar, cuándo hay que registrar un nuevo peso corporal o cuándo termina un **macrociclo**, tanto por **notificación push** como por correo electrónico.
- **Panel de estadísticas avanzadas:** Ampliar la sección de **estadísticas** con **gráficos personalizados, comparativas** entre diferentes macrociclos, análisis de **puntos fuertes y débiles**, y **recomendaciones automáticas** de ajuste de rutinas en función del rendimiento histórico.
- **Marketplace de rutinas y macrociclos:** Crear un espacio donde los usuarios puedan compartir, descargar o comprar **rutinas y planes de entrenamiento** creados por otros miembros de la **comunidad** o **por entrenadores certificados**.

Departamento de IA Generativa

Después de estar de prácticas durante estos meses en un **departamento de IA generativa**, me he podido plantear varias mejoras que, aunque serían a muy largo plazo ya que probablemente necesitarían un **equipo especializado en IA generativa** y poder costear los **gastos asociados**, representan una **evolución natural** muy interesante para StayRPE:

- **Chat generativo para programación:** Desarrollar un **asistente conversacional** inspirado en las **mejores prácticas de entrenamiento** que permita a los usuarios solicitar macrociclos mediante chat. Por ejemplo: "Crea un macrociclo de 8 semanas para ganar fuerza, entreno 3 días por semana" y que genere automáticamente un **plan completo** con rutinas específicas adaptadas a los intereses y nivel del usuario.
- **Generación automática de rutinas:** Implementar un sistema que, basándose en **objetivos simples del usuario** ("quiero una rutina de empuje", "necesito trabajar piernas"), **genere rutinas completas** con ejercicios, series y repeticiones apropiadas.
- **Generación de reportes en formato TXT:** Implementar un sistema que genere automáticamente **reportes y resúmenes** en formato de texto plano que el usuario pueda almacenar, compartir o imprimir. Por ejemplo, una especie de **resumen completo** del macrociclo que incluya la **programación semanal**, **progresiones planificadas**, **objetivos específicos** y notas del entrenador, todo en un formato fácil de leer y archivar.

Mejoras en Tracking y Cálculos

- **Tracking automático de RM:** Desarrollar **algoritmos** que calculen y actualicen automáticamente el **1RM estimado** de cada ejercicio basándose en las series completadas, el peso utilizado y el **RPE/RIR** reportado.
- **Calculadora bidireccional de cargas:** Implementar un sistema que funcione en ambas **direcciones**: por un lado, generar automáticamente un **RM estimado** basándose en el rendimiento histórico del usuario, y por otro lado, a partir de un RM conocido, calcular automáticamente cuántos kilogramos hay que levantar en un ejercicio específico para cumplir con un **RPE/RIR objetivo** determinado. Por ejemplo, si el usuario tiene un RM de 100kg en press banca y quiere entrenar a RPE 8 con 5 repeticiones, el sistema calcularía automáticamente que debería usar aproximadamente 85kg.

Flexibilidad Extrema en Macrociclos

- **Editor avanzado semana a semana:** Expandir el sistema actual de personalización para permitir que los **macrociclos** sean completamente editables, pudiendo modificar rutinas, ejercicios y **progresiones semana a semana** con total libertad.
- **Patrones de rutinas múltiples:** Permitir que un mismo **macrociclo** incluya diferentes **patrones de rutinas** que se alternen según la semana o fase del plan, ofreciendo mayor **variedad y especificidad** en la **programación**.

En definitiva, **StayRPE** es un proyecto pensado para seguir evolucionando y **especializándose** tanto para atletas como para entrenadores, convirtiéndose en una **herramienta verdaderamente útil** para este sector. Estas mejoras son ideas que podrían transformar la aplicación, aunque algunas de ellas serían simplemente a muy largo plazo. La mejora más próxima y prioritaria sería implementar el sistema atleta-entrenador, que sería el siguiente paso natural en la **evolución de la plataforma**.

Conclusiones.

Al finalizar el desarrollo de **StayRPE**, puedo decir que he cumplido la mayoría de los **objetivos** que me propuse al inicio. El proyecto nació con la intención de crear una **aplicación de entrenamiento** realmente útil para **deportes de fuerza**, y creo que **StayRPE** lo consigue: permite planificar **macrociclos** y **microciclos**, ajustar los entrenamientos con parámetros como RPE y RIR, y hacer un seguimiento detallado del progreso, algo que otras **aplicaciones actuales** no ofrecen de esta forma.

Lo que hace especial a **StayRPE** es su **enfoque específico** en los **deportes de fuerza** y su comprensión de conceptos como **RPE, RIR y periodización**, que son fundamentales para los atletas pero que las aplicaciones generalistas no abordan correctamente. Esta **especialización** permite un **control mucho más preciso del entrenamiento** y una progresión más efectiva, llenando un vacío real en el mercado actual de aplicaciones de fitness.

Durante este proceso, he aprendido mucho tanto de **programación** como de **organización y gestión de un proyecto**. En el **frontend** me he sentido cómoda trabajando con **React Native**, ya que es una tecnología que ya había utilizado en el ciclo **formativo**. Sin embargo, la mayor dificultad —y también el mayor aprendizaje— ha sido la parte del **backend**, ya que **Spring Boot** no se vio en clase y he tenido que aprender por mi cuenta cómo implementar toda la **API**, la **seguridad con JWT** y la **integración de la base de datos relacional**.

En el **aspecto técnico**, he adquirido **conocimientos sólidos** en **arquitectura de APIs REST**, manejo de **bases de datos relacionales complejas** con **JPA/Hibernate**, implementación de seguridad con **JWT**, y diseño de sistemas escalables. También he mejorado mis habilidades en la gestión de estado complejo en **React Native** y en la integración entre **frontend** y **backend**. El trabajo con modelos de datos complejos para representar **macrociclos**, rutinas y personalizaciones me ha enseñado la importancia de un **diseño de base de datos** bien pensado.

Además del **aprendizaje técnico**, este proyecto me ha enseñado la importancia de una **buena planificación y documentación**. He aprendido a dividir funcionalidades complejas en tareas manejables, a priorizar características según su **impacto**, y a mantener un **código limpio** y documentado que facilite **futuras mejoras**. La gestión de un proyecto de esta envergadura me ha dado una perspectiva valiosa sobre el **desarrollo de software profesional**.

Uno de los mayores retos ha sido encontrar el equilibrio entre **funcionalidad** y **usabilidad**. Crear una aplicación que sea lo suficientemente potente para **atletas** avanzados pero que también sea accesible para **usuarios menos experimentados** ha requerido muchas iteraciones en el **diseño de la interfaz** y la **experiencia de usuario**. También ha sido desafiante implementar la lógica compleja de **personalización de entrenamientos** manteniendo la consistencia de datos.

Por supuesto, han quedado cosas por hacer que me gustaría desarrollar en el futuro, como el **sistema de comunicación** entre **entrenador y atleta**, la integración con **dispositivos externos**, mejorar la **parte visual** con temas personalizables y **gráficos avanzados**, o añadir **notificaciones push** y recomendaciones automáticas de rutinas. Además, sería interesante seguir optimizando el **rendimiento de la aplicación** y crear herramientas de análisis aún más avanzadas para el usuario.

En definitiva, **StayRPE** no es solo un **proyecto académico** para mí, sino una **herramienta** que me gustaría poder seguir desarrollando y perfeccionando aunque sea de uso personal. Este proyecto me ha confirmado que la **programación** no es solo escribir código, sino resolver problemas reales y **crear valor para las personas**. Me ha supuesto un reto tanto a **nivel técnico como personal**, y me ha permitido desarrollar nuevas habilidades y descubrir hasta dónde soy capaz de llegar cuando tengo una meta clara.

Me siento preparada para afrontar **nuevos desafíos profesionales** con la confianza de que puedo aprender y adaptarme a cualquier **tecnología** que sea necesaria, y me quedo con muchas ganas de seguir mejorando **StayRPE** y también de afrontar **nuevos proyectos** en el futuro.

Biblioteca de recursos web y referencias.

Oracle Corporation, O. (2024): Spring Boot Reference Documentation, Spring Framework Documentation, pp. 15-45, disponible en: <https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/>

Oracle Corporation, O. (2024): Spring Security Reference, Spring Security Documentation, pp. 120-180, disponible en: <https://docs.spring.io/spring-security/reference/>

Oracle Corporation, O. (2024): Spring Data JPA Reference Documentation, Spring Data Documentation, pp. 30-85, disponible en: <https://docs.spring.io/spring-data/jpa/docs/current/reference/html/>

Auth0 Inc., A. (2024): Java JWT Library Documentation, JWT Implementation Guide, pp. 1-25, disponible en: <https://github.com/auth0/java-jwt>

Project Lombok Team, P. (2024): Project Lombok Documentation, Lombok Features Guide, pp. 1-20, disponible en: <https://projectlombok.org/features/all>

Meta Platforms, M. (2024): React Native Documentation, React Native Foundation, pp. 1-150, disponible en: <https://reactnative.dev/docs/getting-started>

Expo Team, E. (2024): Expo Documentation, Expo Inc., pp. 1-200, disponible en: <https://docs.expo.dev/>

React Native Community, R. (2024): AsyncStorage Documentation, React Native Community, pp. 1-30, disponible en: <https://react-native-async-storage.github.io/async-storage/>