**UNIVERSIDAD CONTINENTAL**

**FACULTAD DE INGENIERÍA**

**ESCUELA ACADÉMICO PROFESIONAL DE INGENIERÍA DE SISTEMAS E INFORMÁTICA**



**PROYECTO**

**IMPLEMENTACIÓN DE UNA APLICACIÓN DE ENTRENAMIENTO PERSONAL PARA UNIVERSITARIOS DE LA CONTINENTAL CON RUTINAS CASERAS**

# PORTADA

**PRESENTADO POR:**

| **APELLIDOS Y NOMBRES** | **CÓDIGO** |
| --- | --- |
| **Aquino Villegas Katheriny Nicol** | **76750754** |
| **Cuellar Pampas Junior Jhosep** | **74304981** |
| **Lizaraso Taipe David** | **75176074** |
| **Huaman Huatuco Emilio** | **75243881** |
| **Marquez Barrera Marcel** | **72317809** |
| **Leiva Leiva Juan Alexis** | **77496923** |
| **Lopez Aquino Erik Angel** | **71000885** |
| **Soto Egoavil Sergio Aldair** | **74174615** |

**ASESOR:**

**HUANCAYO – PERÚ**

**2023**

# LISTA DE CONTENIDO

[PORTADA 1](#_gjdgxs)

[LISTA DE CONTENIDO 2](#_30j0zll)

[LISTA DE TABLAS 7](#_1fob9te)

[LISTA DE FIGURAS 8](#_3znysh7)

[CAPÍTULO 1 9](#_2et92p0)

[PLANTEAMIENTO DEL ESTUDIO 9](#_tyjcwt)

[1.1.](#_3dy6vkm) Aspectos Generales de la Empresa 9

[1.1.1.](#_1t3h5sf) Organigrama 9

[1.1.2.](#_4d34og8) Misión y visión 9

[1.2.](#_2s8eyo1) Diagnóstico del Problema 9

[1.3.](#_17dp8vu) Procesos de la Empresa 9

[1.4.](#_3rdcrjn) Oportunidad Encontrada 9

[1.5.](#_26in1rg) Detalles del Proyecto 9

[CAPÍTULO 2 10](#_lnxbz9)

[ESTUDIO DE FACTIBILIDAD 10](#_35nkun2)

[2.1.](#_1ksv4uv) Alternativas de Solución 10

[2.2.](#_44sinio) Factibilidad Técnica 10

[2.2.1.](#_2jxsxqh) … 10

[2.2.2.](#_z337ya) … 10

[2.3.](#_3j2qqm3) Factibilidad Económica 10

[2.3.1.](#_1y810tw) Gastos generales 10

[2.4.](#_4i7ojhp) Factibilidad Operacional 10

[2.4.1.](#_2xcytpi) Sistemas de ventas 10

[CAPÍTULO 3 11](#_1ci93xb)

[ANÁLISIS DE REQUERIMIENTOS 11](#_3whwml4)

[3.1.](#_2bn6wsx) Metas del Sistema de Información 11

[3.2.](#_qsh70q) Requisitos del Sistema 11

[3.2.1.](#_3as4poj) Requerimientos funcionales 11

[3.2.2.](#_1pxezwc) Requerimientos no funcionales 11

[3.3.](#_49x2ik5) Identificación de Actores del Sistema 11

[3.3.1.](#_2p2csry) … 11

[CAPÍTULO 4 12](#_147n2zr)

[PLANIFICACIÓN DEL PROYECTO 12](#_3o7alnk)

[4.1.](#_23ckvvd) Definición de Roles de Trabajo 12

[4.1.1.](#_ihv636) Product owner 12

[4.1.2.](#_32hioqz) Scrum master 12

[4.1.3.](#_1hmsyys) Team member 12

[4.1.4.](#_41mghml) Tester 12

[4.2.](#_2grqrue) Product Backlog 12

[4.3.](#_vx1227) Sprint Backlog 12

[4.3.1.](#_3fwokq0) Sprint 1 12

[4.3.2.](#_1v1yuxt) Sprint 2 12

[4.3.3.](#_4f1mdlm) Sprint 3 12

[4.3.4.](#_2u6wntf) Sprint 4 12

[4.3.5.](#_19c6y18) Sprint 5 13

[4.4.](#_3tbugp1) Planificación de Sprints 13

[4.4.1.](#_28h4qwu) Historias de usuario 13

[4.4.2.](#_nmf14n) Priorización de historias de usuario 13

[4.5.](#_37m2jsg) Cronograma de Actividades 13

[4.6.](#_1mrcu09) Gestión de Riesgos 13

[CAPÍTULO 5 14](#_46r0co2)

[DISEÑO DEL SISTEMA DE INFORMACIÓN 14](#_2lwamvv)

[5.1.](#_111kx3o) Diseño de Diagramas UML 14

[5.1.1.](#_3l18frh) Diagramas de casos de uso 14

[5.1.2.](#_206ipza) Diagramas de secuencia 14

[5.1.3.](#_4k668n3) Diagramas de colaboración 14

[5.1.4.](#_2zbgiuw) Diagramas de clases 14

[5.2.](#_1egqt2p) Diseño de Base de Datos 14

[5.2.1.](#_3ygebqi) Diseño conceptual (E/R) 14

[5.2.2.](#_2dlolyb) Diseño lógico 14

[5.2.3.](#_sqyw64) Diseño físico 14

[5.2.4.](#_3cqmetx) Modelado de base de datos 14

[5.3.](#_1rvwp1q) Diseño de Interfaces Básicas 14

[5.3.1.](#_4bvk7pj) Acceso login 15

[5.3.2.](#_2r0uhxc) Interfaz … 15

[CAPÍTULO 6 16](#_1664s55)

[CODIFICACIÓN DEL SOFTWARE 16](#_3q5sasy)

[6.1.](#_25b2l0r) Desarrollo del Sprint 1 16

[6.1.1.](#_kgcv8k) Sprint planning 16

[6.1.2.](#_34g0dwd) Sprint backlog 16

[6.1.3.](#_1jlao46) Historias de usuarios 16

[6.1.4.](#_43ky6rz) Taskboard 16

[6.1.5.](#_2iq8gzs) Daily scrum 16

[6.1.6.](#_xvir7l) Sprint review 16

[6.1.7.](#_3hv69ve) Criterios de aceptación 16

[6.1.8.](#_1x0gk37) Resultados del sprint 16

[6.1.8.1.](#_4h042r0) Evidencias. 16

[6.1.8.2.](#_2w5ecyt) Prueba de desarrollo. 16

[6.1.8.3.](#_1baon6m) …. 17

[6.1.9.](#_3vac5uf) Sprint retrospective 17

[6.2.](#_2afmg28) Desarrollo del Sprint 2 17

[6.2.1.](#_pkwqa1) Sprint planning 17

[6.2.2.](#_39kk8xu) Sprint backlog 17

[6.2.3.](#_1opuj5n) Historias de usuarios 17

[6.2.4.](#_48pi1tg) Taskboard 17

[6.2.5.](#_2nusc19) Daily scrum 17

[6.2.6.](#_1302m92) Sprint review 17

[6.2.7.](#_3mzq4wv) Criterios de aceptación 17

[6.2.8.](#_2250f4o) Resultados del sprint 17

[6.2.8.1.](#_haapch) Evidencias. 17

[6.2.8.2.](#_319y80a) Prueba de desarrollo. 18

[6.2.8.3.](#_1gf8i83) …. 18

[6.2.9.](#_40ew0vw) Sprint retrospective 18

[6.3.](#_2fk6b3p) Desarrollo del Sprint 3 18

[6.3.1.](#_upglbi) Sprint planning 18

[6.3.2.](#_3ep43zb) Sprint backlog 18

[6.3.3.](#_1tuee74) Historias de usuarios 18

[6.3.4.](#_4du1wux) Taskboard 18

[6.3.5.](#_2szc72q) Daily scrum 18

[6.3.6.](#_184mhaj) Sprint review 18

[6.3.7.](#_3s49zyc) Criterios de aceptación 18

[6.3.8.](#_279ka65) Resultados del sprint 18

[6.3.8.1.](#_meukdy) Evidencias. 19

[6.3.8.2.](#_36ei31r) Prueba de desarrollo. 19

[6.3.8.3.](#_1ljsd9k) …. 19

[6.3.9.](#_45jfvxd) Sprint retrospective 19

[6.4.](#_2koq656) Desarrollo del Sprint 4 19

[6.4.1.](#_zu0gcz) Sprint planning 19

[6.4.2.](#_3jtnz0s) Sprint backlog 19

[6.4.3.](#_1yyy98l) Historias de usuarios 19

[6.4.4.](#_4iylrwe) Taskboard 19

[6.4.5.](#_2y3w247) Daily scrum 19

[6.4.6.](#_1d96cc0) Sprint review 19

[6.4.7.](#_3x8tuzt) Criterios de aceptación 19

[6.4.8.](#_2ce457m) Resultados del sprint 20

[6.4.8.1.](#_rjefff) Evidencias. 20

[6.4.8.2.](#_3bj1y38) Prueba de desarrollo. 20

[6.4.8.3.](#_1qoc8b1) …. 20

[6.4.9.](#_4anzqyu) Sprint retrospective 20

[CAPÍTULO 7 21](#_2pta16n)

[PRUEBAS DE SOFTWARE 21](#_14ykbeg)

[7.1.](#_3oy7u29) Plan de Pruebas 21

[CONCLUSIONES 22](#_243i4a2)

[RECOMENDACIONES 23](#_j8sehv)

[ANEXOS 24](#_1idq7dh)

[Anexo 01. Manual Técnico 25](#_42ddq1a)

[Anexo 02. Manual de Usuario 26](#_2hio093)

# LISTA DE TABLAS

**No se encuentran elementos de tabla de ilustraciones.**

# LISTA DE FIGURAS

**No se encuentran elementos de tabla de ilustraciones.**

# CAPÍTULO 1

# PLANTEAMIENTO DEL ESTUDIO

## Aspectos Generales de la Empresa

La empresa en estudio es parte del sector tecnológico y se enfoca en desarrollar soluciones innovadoras que integren la tecnología en la vida cotidiana de sus usuarios. En este caso, el proyecto UniFit tiene como objetivo ofrecer una aplicación de bienestar físico diseñada para estudiantes universitarios de la Universidad Continental. UniFit permite realizar rutinas de ejercicio personalizadas y adaptadas a las necesidades de los estudiantes, contribuyendo a su bienestar físico y mental.

### Organigrama

El organigrama del proyecto UniFit se organiza de la siguiente manera:

**Directorio General:** Supervisa la implementación de cada fase, toma decisiones estratégicas y asegura la disponibilidad de recursos.

**Gerencia de Proyecto:** Responsable de la planificación, control de calidad, y cumplimiento de objetivos de cada etapa de desarrollo.

**Equipo de Desarrollo:** Compuesto por especialistas en el stack MERN (MongoDB, Express.js, React.js, Node.js), encargados de la lógica del backend, la interfaz de usuario, y la integración de la base de datos.

**Equipo de Marketing:** Encargado de diseñar y ejecutar la estrategia de promoción y lanzamiento para alcanzar a la población universitaria.

**Equipo de Atención al Cliente y Feedback:** Gestiona la interacción con los usuarios, recoge sus comentarios, y canaliza la información hacia el equipo técnico para implementar mejoras.

### Misión y visión

**Misión**: Ofrecer una plataforma que promueva la actividad física y el bienestar en estudiantes universitarios, integrando rutinas accesibles y personalizadas en su estilo de vida académico.

**Visión**: Ser una aplicación referente en el sector educativo y de bienestar, conocida por su enfoque adaptativo y compromiso con la salud integral de los estudiantes.

## Diagnóstico del Problema

Los estudiantes universitarios suelen tener dificultades para mantener una vida activa y saludable debido a sus obligaciones académicas y personales. Factores como la falta de tiempo, el difícil acceso a instalaciones deportivas, y la ausencia de una plataforma que considere su estilo de vida particular generan problemas como el estrés y el sedentarismo. UniFit se plantea como una solución a esta problemática, permitiendo a los estudiantes acceder a rutinas de ejercicio personalizadas sin afectar sus horarios de estudio y sin necesidad de equipo especial.

## Procesos de la Empresa

UniFit adopta procesos enfocados en la innovación y mejora continua:

1. **Investigación y Desarrollo (I+D)**: Se analiza la viabilidad de tecnologías y metodologías ágiles que faciliten el ajuste rápido a los requerimientos de los estudiantes.
2. **Desarrollo del Producto**: Incluye el diseño de la arquitectura de software utilizando el stack MERN, la creación de bases de datos en MongoDB, y el desarrollo de una interfaz intuitiva en React.js.
3. **Pruebas y Control de Calidad**: Durante la fase de pruebas beta, estudiantes de la Universidad Continental utilizan la aplicación y proporcionan feedback para garantizar que se ajuste a sus necesidades.
4. **Marketing y Promoción**: Estrategias de marketing digital enfocadas en captar el interés de la comunidad universitaria y resaltar los beneficios de UniFit.
5. **Feedback y Mejoras Continuas**: Las opiniones de los usuarios se recopilan para realizar ajustes constantes y mejorar la experiencia de uso.

## Oportunidad Encontrada

La creciente demanda de servicios de salud y bienestar adaptables presenta una oportunidad ideal para crear una aplicación de fitness específicamente diseñada para estudiantes universitarios. UniFit surge en respuesta a esta necesidad, ofreciendo una solución accesible y adaptada a los estudiantes de la Universidad Continental. A través de funcionalidades como recordatorios, seguimiento de progreso y una comunidad de apoyo, UniFit no solo mejora la salud física de los usuarios, sino que también fortalece el sentido de comunidad universitaria.

## Detalles del Proyecto

El proyecto UniFit tiene como objetivo el desarrollo de una aplicación de entrenamiento personalizada, diseñada para adaptarse al estilo de vida de los estudiantes universitarios. Los detalles clave del proyecto incluyen:

* **Nombre del Proyecto**: UniFit
* **Propósito**: Facilitar el acceso a rutinas de ejercicio personalizadas que se puedan realizar en cualquier lugar y horario conveniente para los estudiantes.
* **Características Principales**:
  1. **Perfil de Usuario**: Personalización de rutinas en función de las necesidades y objetivos del usuario.
  2. **Rutinas de Ejercicio**: Ejercicios y rutinas personalizadas basadas en el nivel físico y preferencias del usuario.
  3. **Recordatorios y Notificaciones**: Para mantener consistencia en las rutinas.
  4. **Sistema de Seguimiento de Progreso**: Estadísticas de calorías quemadas, tiempo de ejercicio y objetivos alcanzados.
  5. **Integración con Plataformas de Salud**: Sincronización de datos con Google Fit y Apple Health.
  6. **Comunidad de Usuarios**: Espacio para que los usuarios se motiven y compartan logros.
* **Duración del Proyecto**: 6 meses.
* **Presupuesto**: S/6,000,000.

# CAPÍTULO 2

# ESTUDIO DE FACTIBILIDAD

## Alternativas de Solución

El desarrollo de UniFit considera distintas alternativas tecnológicas para satisfacer las necesidades de los estudiantes universitarios, permitiéndoles acceder a una plataforma de rutinas de ejercicio personalizadas. A continuación, se detallan las alternativas evaluadas:

* Alternativa 1: Stack MERN (MongoDB, Express.js, React.js, Node.js)

La solución principal propuesta emplea el stack MERN, una tecnología moderna y escalable que permite la creación de aplicaciones web interactivas. Este stack es eficiente para el manejo de datos en tiempo real y proporciona una experiencia de usuario dinámica. Esta alternativa también permite personalizar rutinas basadas en el perfil y las metas del usuario, además de realizar un seguimiento de su progreso.

* Alternativa 2: Firebase para la autenticación y base de datos

Otra opción es utilizar Firebase como backend, facilitando la autenticación y el almacenamiento de datos. Aunque Firebase simplifica la infraestructura, presenta limitaciones para personalizar rutinas complejas y generar recomendaciones específicas.

* Alternativa 3: Progressive Web App (PWA)

La última alternativa es una PWA, que permite acceso sin conexión y mayor flexibilidad en dispositivos móviles. Sin embargo, el desarrollo inicial puede ser más demandante y requerir optimización para un acceso rápido y eficiente a los datos de los usuarios.

## Factibilidad Técnica

Para la implementación de UniFit, se requiere una infraestructura técnica que soporte el desarrollo, despliegue y mantenimiento de la aplicación de entrenamiento. La factibilidad técnica se evalúa en términos de los recursos de hardware y software necesarios.

### 

### Hardware: Servidor en la nube

UniFit se desplegará en un entorno de contenedores Docker, lo que requiere servidores en la nube capaces de soportar y escalar esta tecnología. Se recomienda utilizar servicios en la nube como AWS o DigitalOcean, que ofrecen configuraciones adecuadas para aplicaciones web que dependen de bases de datos y servicios en tiempo real. A continuación, se detallan las características mínimas recomendadas:

| **Características** | **Valores** |
| --- | --- |
| Número de núcleos | 12 |
| Memoria caché | 128 MB |
| Frecuencia mínima | 1.2GHz |
| Frecuencia base máxima turbo | 3.3GHz |
| RAM compatible | DDR4 |
| Memoria ECC compatible | No |
| Número de PCI | 1x16 + 1x8 + 1x4 |
| Máximo PCI Express | 8GT/s |
| TDP | 140w |
| Disco Duro | 1TB |
| Tarjeta de red | 1300Mbps – 2.4G |
| Tarjeta de video | 16GB GDDR5 |

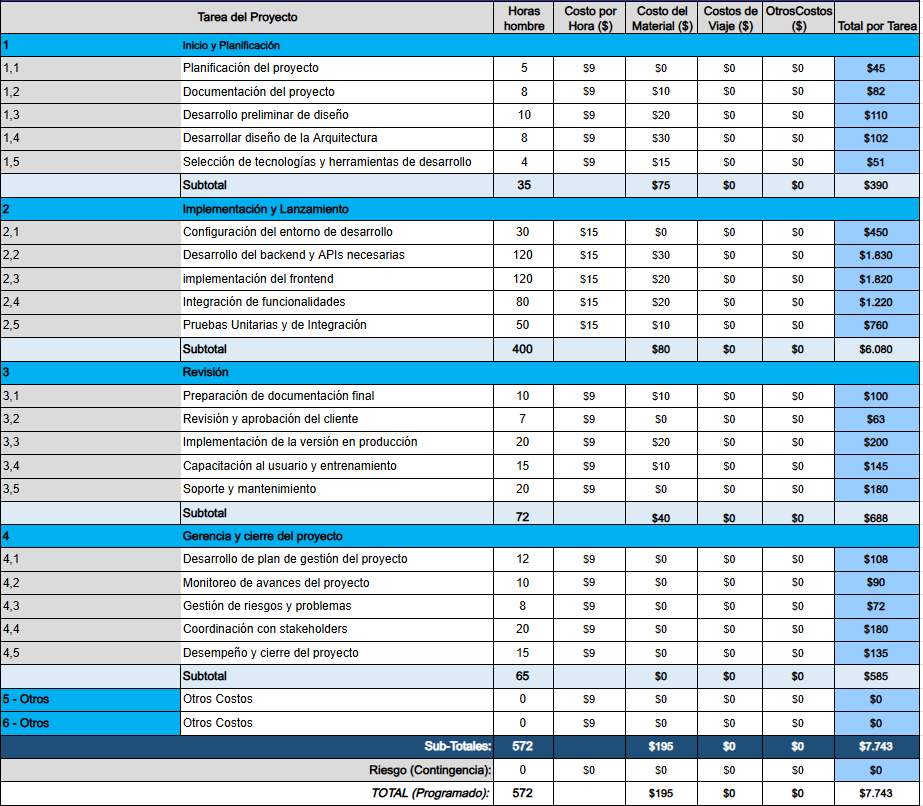
### Software: Stack MERN y Docker

Para el desarrollo de UniFit, se emplearán React y pgAdmin como tecnologías principales, con la siguiente estructura de software:

* React: La biblioteca de JavaScript React se empleará para el desarrollo de la interfaz de usuario, proporcionando una experiencia intuitiva, interactiva y adaptable a dispositivos móviles y de escritorio. React permite construir una interfaz dinámica, optimizada para la consulta y registro de datos de entrenamiento en tiempo real.
* pgAdmin: pgAdmin se utilizará como herramienta de administración para la base de datos, facilitando la gestión de datos almacenados en PostgreSQL. pgAdmin permite una supervisión eficiente de la base de datos, asegurando que los datos de las rutinas y el progreso de los usuarios se manejen de forma segura y organizada.

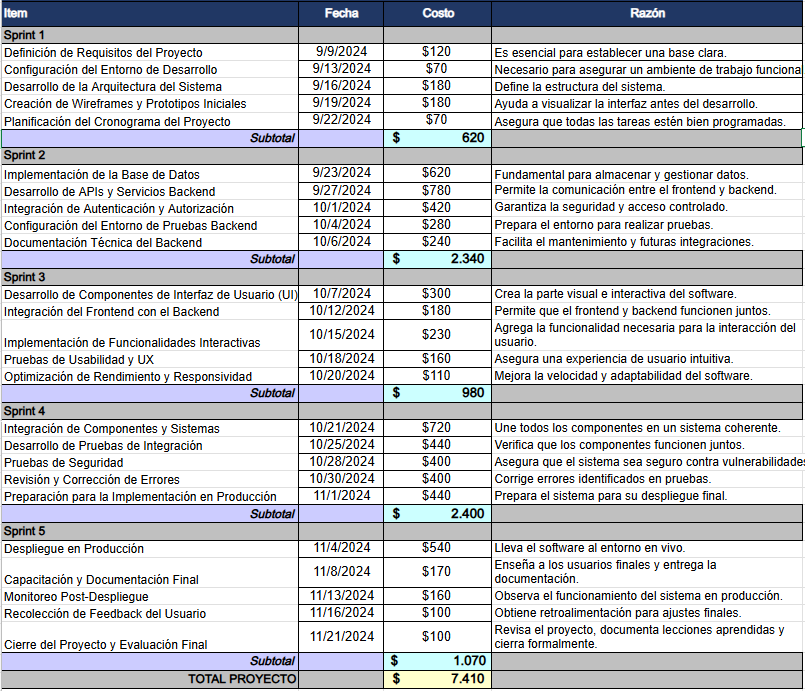
Para la gestión de versiones y el trabajo colaborativo, se utilizará Git junto con la metodología Git Flow, lo que permitirá a los desarrolladores mantener un control de versiones estructurado y efectivo durante el desarrollo del proyecto.

## Factibilidad Económica

El análisis de factibilidad económica detalla los costos estimados para la implementación y el mantenimiento de UniFit, asegurando su viabilidad financiera.

### 

### Gastos generales

Se estiman los siguientes gastos para el desarrollo de UniFit:

La inversión estimada total es de S/ 7,410.00, lo cual es viable dado el alcance y los beneficios del proyecto para los estudiantes universitarios.

## Factibilidad Operacional

La factibilidad operacional asegura que UniFit se pueda implementar con éxito y que sea sostenible para la comunidad universitaria.

### Sistemas de ventas

**2.4.1.1. Fase de análisis**

* Identificación de los requisitos de usuario, interfaz, rendimiento, hardware y software mediante entrevistas y encuestas a estudiantes.
* Definición de la arquitectura del sistema, con énfasis en seguridad de datos y personalización de rutinas.

**2.4.1.2. Fase de diseño y desarrollo**

* Diseño de la interfaz y estructura de la base de datos en MongoDB.
* Creación de módulos para registro de usuario, generación de rutinas y seguimiento de progreso.
* Implementación de recordatorios personalizados y notificaciones.

**2.4.1.3. Fase de transición**

* Realización de pruebas internas y beta con estudiantes seleccionados.
* Ajustes en base a retroalimentación y análisis de rendimiento.

**2.4.1.4. Fase de implementación**

* Lanzamiento de UniFit en la nube.
* Capacitación a los usuarios mediante un manual y asistencia inicial en el uso de la aplicación.

# CAPÍTULO 3

# ANÁLISIS DE REQUERIMIENTOS

## Metas del Sistema de Información

* **Ofrecer planes de entrenamiento personalizados**: Implementar un sistema que permita a los usuarios seleccionar y personalizar sus rutinas de ejercicio, según sus objetivos, disponibilidad de tiempo y equipo disponible en casa.
* **Control de progreso físico**: Desarrollar una funcionalidad que permita a los usuarios registrar y visualizar su progreso en variables como peso, medidas corporales y rendimiento en cada ejercicio, facilitando el seguimiento de resultados.
* **Sistema de recordatorios y motivación**: Incluir un sistema de notificaciones que envíe recordatorios y mensajes de motivación a los usuarios para asegurar que mantengan la constancia en sus rutinas.
* **Acceso a contenido educativo en tiempo real**: Proveer una sección de videos y artículos sobre técnicas de entrenamiento, alimentación saludable y cuidado físico, actualizados periódicamente, para educar y motivar a los usuarios.
* **Generación de reportes de actividad y uso**: Emitir reportes mensuales sobre el uso de la aplicación, el avance en rutinas y la adherencia a los entrenamientos, permitiendo a los administradores conocer patrones de uso y áreas de mejora.
* **Módulo de control de acceso**: Implementar un sistema de gestión de usuarios y control de acceso basado en permisos, donde estudiantes puedan acceder a rutinas personalizadas y ver su historial de actividad.

## Requisitos del Sistema

Para el desarrollo del sistema de aplicación web de entrenamiento personal dirigido a universitarios, se han definido una serie de metas y características esenciales que guiarán la creación del producto. Este sistema busca ofrecer rutinas caseras personalizadas, seguimiento de progreso, y contenido educativo enfocado en la actividad física y la salud, proporcionando una plataforma accesible, intuitiva y motivadora para los estudiantes. Con el propósito de lograr estos objetivos y asegurar que el producto cumpla tanto con las expectativas del usuario como con las necesidades de mantenimiento y escalabilidad a futuro, se han establecido los siguientes requerimientos funcionales y no funcionales.

### Requerimientos funcionales

**Gestión de acceso de usuarios**:

* El sistema deberá contar con acceso restringido mediante nombre de usuario y contraseña.
* El sistema deberá permitir el registro de nuevos usuarios con validación de sus datos.
* El sistema deberá contar con una función de recuperación de contraseña.

**Gestión de rutinas y planes de entrenamiento**:

* El sistema deberá permitir a los usuarios seleccionar rutinas predefinidas y personalizarlas de acuerdo a sus necesidades (nivel, duración, objetivos).
* El sistema deberá mostrar una biblioteca de rutinas con categorías (fuerza, resistencia, flexibilidad, etc.).
* El sistema deberá registrar el progreso del usuario en cada sesión y mostrar un historial de rutinas completadas.

**Seguimiento de progreso físico**:

* El sistema deberá permitir a los usuarios registrar y actualizar medidas corporales (peso, altura, circunferencia muscular, etc.).
* El sistema deberá generar gráficos de progreso para que el usuario visualice sus avances.

**Recordatorios y notificaciones**:

* El sistema deberá enviar recordatorios de próximas sesiones de entrenamiento y alertas personalizadas para fomentar la constancia.
* El sistema deberá notificar al usuario cuando complete una meta de entrenamiento.

**Sistema de reportes y estadísticas**:

* El sistema deberá generar reportes mensuales de actividad para cada usuario, mostrando su nivel de adherencia y progreso.
* El sistema deberá permitir al administrador ver reportes de uso general (número de sesiones, usuarios activos, etc.).

**Acceso a contenido educativo**:

* El sistema deberá ofrecer acceso a videos, guías y artículos sobre entrenamiento y hábitos saludables.
* El sistema deberá actualizar periódicamente el contenido educativo y permitir al administrador cargar nuevo material.

### Requerimientos no funcionales

**Seguridad**:

* El sistema deberá proteger los datos de usuario con encriptación de contraseñas y medidas de seguridad para evitar accesos no autorizados.
* El sistema deberá cumplir con las normativas de protección de datos personales.

**Disponibilidad y rendimiento**:

* El sistema deberá estar disponible al menos el 99% del tiempo, especialmente en horarios pico (mañanas y tardes).
* El sistema deberá cargar en menos de 2 segundos en dispositivos de gama media.

**Compatibilidad**:

* El sistema deberá ser compatible con navegadores populares (Chrome, Firefox, Safari) y funcionar en dispositivos móviles y de escritorio.
* La interfaz de usuario deberá ser responsiva y adaptarse a diferentes resoluciones de pantalla.

**Usabilidad**:

* El sistema deberá ofrecer una interfaz intuitiva y amigable para que los usuarios puedan navegar y utilizar las funciones sin requerir instrucciones adicionales.
* El sistema deberá permitir que los usuarios personalicen su perfil y configuraciones de manera sencilla.

**Escalabilidad**:

* El sistema deberá estar diseñado para soportar un aumento gradual en el número de usuarios sin afectar el rendimiento.
* El sistema deberá permitir la integración de futuras funcionalidades, como opciones de pago en caso de planes premium.

**Mantenimiento y actualización**:

* El sistema deberá permitir realizar actualizaciones sin afectar la disponibilidad para los usuarios.
* La arquitectura del sistema deberá ser modular para facilitar el mantenimiento y la implementación de mejoras.

## Identificación de Actores del Sistema

Un actor es cualquier entidad que tenga comportamientos definidos dentro del sistema. En este caso, los actores no solo representan a los usuarios finales, sino también a roles administrativos y otros sistemas que puedan integrarse con la aplicación. A continuación, se describen los actores identificados para el sistema de entrenamiento personal.

### Administrador

Este perfil incluye al personal de gestión y administradores de la aplicación. El administrador cuenta con acceso total a la información de la plataforma. Sus funciones incluyen la creación, modificación y eliminación de usuarios; gestión de contenido (rutinas, videos, y artículos educativos); generación de reportes de actividad de usuarios; y administración de configuraciones generales del sistema. Además, el administrador puede gestionar los recordatorios y notificaciones enviadas a los usuarios, así como cargar nuevo contenido educativo en la plataforma.

* + 1. **Entrenador**

Este perfil permite a un profesional en entrenamiento físico acceder a la plataforma para crear y personalizar rutinas de ejercicio adecuadas para los usuarios. El entrenador puede generar planes de entrenamiento personalizados, actualizar y modificar las rutinas predefinidas, así como acceder a los reportes de progreso general de los usuarios para evaluar el cumplimiento y la efectividad de los planes de entrenamiento.

* + 1. **Usuario (Estudiante Universitario)**

Este es el actor principal de la aplicación, representando al estudiante universitario que desea entrenar desde casa. El usuario puede registrarse, crear un perfil, acceder a rutinas personalizadas, registrar y consultar su progreso físico, y recibir recordatorios de entrenamiento. Además, el usuario puede visualizar contenido educativo en la aplicación y ajustar las rutinas según su disponibilidad y nivel de habilidad. También tiene la opción de actualizar sus objetivos y recibir recomendaciones automáticas basadas en su progreso.

* + 1. **Nutricionista (opcional)**

En caso de que el sistema incluya asesoría nutricional, este actor representaría a profesionales de nutrición que pueden recomendar planes de alimentación complementarios a las rutinas de ejercicio. Este perfil podría ver el progreso de los usuarios y realizar recomendaciones nutricionales personalizadas, que los estudiantes podrán seguir junto a sus entrenamientos.

# CAPÍTULO 4

# PLANIFICACIÓN DEL PROYECTO

## Definición de Roles de Trabajo

Este proyecto está organizado bajo el marco Scrum, dividiendo el trabajo en sprints con objetivos específicos para una entrega continua de valor. Utilizamos PostgreSQLpara gestionar los datos de usuarios y clases, y Tailwind CSS para desarrollar una interfaz intuitiva y rápida de diseñar.

### Product owner

Define la visión y prioridades del proyecto, alineando las funcionalidades clave con los requerimientos del cliente y gestionando las expectativas.

Encargados:

### Scrum master

Facilita los procesos de Scrum, eliminando obstáculos, liderando reuniones y asegurando que el equipo siga los principios de Scrum.

Encargados:

### Frontend Dev

Implementa el diseño de la interfaz de usuario utilizando Tailwind CSS, con un enfoque en responsividad y usabilidad.

Encargados:

### Backend Dev

Crea y gestiona la API REST para la conexión del frontend con PostgreSQL, asegurando una comunicación eficiente y segura y la estructura de la base de datos en PostgreSQL, optimizando las consultas para mejorar la velocidad de la aplicación, y apoya en pruebas de rendimiento.

Encargados:

### Team member

Implementa el diseño de la interfaz de usuario utilizando Tailwind CSS, con un enfoque en responsividad y usabilidad.

Encargados:

### Tester

Realiza pruebas de interfaz y funcionalidad en cada entrega para detectar errores y asegurar la calidad del producto antes de cada entrega.

Encargados:

## Product Backlog

### El Product Backlog es la lista priorizada de todas las características y mejoras que se requieren para la aplicación. El objetivo es garantizar que el gimnasio cuente con una plataforma intuitiva que permita a los usuarios y administradores gestionar las actividades y membresías de forma eficiente.

### Autenticación de usuario (Registro y Login en modal): Los usuarios deben poder crear cuentas e iniciar sesión fácilmente en un modal. Esto optimiza el flujo sin cambiar de página.

### Panel administrativo de usuarios y membresías: Permite a los administradores gestionar las membresías, historial de usuarios, y datos personales.

### Sistema de reservas de clases: Los usuarios pueden ver y reservar clases disponibles.

### Reportes de estadísticas de asistencia y progreso: El gimnasio puede monitorear la asistencia y el progreso de los clientes.

### Conexión con PostgreSQL: Gestión de datos robusta para usuarios, clases y asistencias.

### Pruebas de UI y rendimiento: Garantizar la eficiencia y usabilidad de la aplicación en dispositivos de diferentes tamaños.

## Sprint Backlog

Cada sprint tiene un objetivo específico basado en el Product Backlog, con entregas funcionales en cada uno para mantener un avance continuo.

### Sprint 1

* Configuración del entorno (instalación de dependencias, creación de estructuras de directorios y repositorios).
* Configuración de PostgreSQL.
* Estructura básica de frontend con Tailwind CSS.

### Sprint 2

* Desarrollar el sistema de registro e inicio de sesión en un modal.
* Implementar validaciones de entrada y mensajes de error amigables para el usuario.
* Pruebas iniciales de autenticación.

### Sprint 3

* Crear vistas y funcionalidades del panel de administración (gestión de membresías, historial de usuarios).
* Conexión con PostgreSQL para la creación, actualización y eliminación de usuarios y membresías.
* Pruebas de interfaz y funcionalidad en el panel de administración.

### Sprint 4

* Diseñar y desarrollar el sistema de reservas, permitiendo a los usuarios ver clases disponibles y hacer reservas.
* Configuración de alertas de notificación para las reservas.
* Pruebas de la funcionalidad de reservas.

### Sprint 5

* Implementar un sistema de reportes para monitorear la asistencia y el progreso de los usuarios.
* Optimización de la interfaz con Tailwind CSS para asegurar una experiencia de usuario fluida y atractiva.
* Pruebas finales de aceptación y revisión de la interfaz en múltiples dispositivos.

## Planificación de Sprints

Cada sprint tiene un conjunto claro de objetivos. Al final de cada sprint, se realiza una revisión y retrospectiva para optimizar el flujo de trabajo. Esto permite que cualquier error se identifique y se corrija antes de la siguiente fase.

### Historias de usuario

Un ejemplo de historia podría ser: “Como usuario, quiero poder reservar clases fácilmente para planificar mis entrenamientos semanales”.

### Priorización de historias de usuario

Primero, se desarrollan funciones esenciales, como la autenticación, luego la gestión de usuarios y finalmente el sistema de reservas, basándonos en su impacto en la experiencia del usuario.

## Cronograma de Actividades

El proyecto se organiza en un calendario de sprints, donde cada uno tiene una duración de dos semanas, con una revisión detallada al final de cada sprint. La planificación se adapta si surgen problemas o nuevas ideas durante la retrospectiva.

| **Sprint** | **Objetivo** | **Duración** |
| --- | --- | --- |
| Sprint 1 | Configuración inicial y setup de Tailwind/PostgreSQL | 2 semanas |
| Sprint 2 | Sistema de autenticación | 2 semanas |
| Sprint 3 | Panel de administración de usuarios | 2 semanas |
| Sprint 4 | Sistema de reservas de clases | 2 semanas |
| Sprint 5 | Reportes y pruebas de aceptación | 2 semanas |

## Gestión de Riesgos

Es fundamental identificar y gestionar los riesgos a lo largo del proyecto para asegurar su éxito. Los riesgos incluyen:

* Retrasos en la integración del frontend y backend: Puede ocurrir debido a problemas de compatibilidad o errores de conexión con PostgreSQL. Se mitiga con pruebas continuas y comunicación frecuente.
* Problemas de rendimiento en la UI: Solucionar el impacto en la experiencia del usuario mediante pruebas de carga en dispositivos móviles y de escritorio.
* Dificultades en la funcionalidad de reservas: Si hay conflictos en la disponibilidad de clases o en la gestión de la capacidad, se resolverán con ajustes en la arquitectura de la base de datos y optimización de consultas SQL.

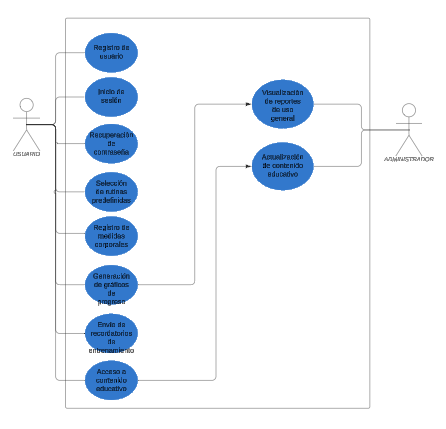
## 

# CAPÍTULO 5

# DISEÑO DEL SISTEMA DE INFORMACIÓN

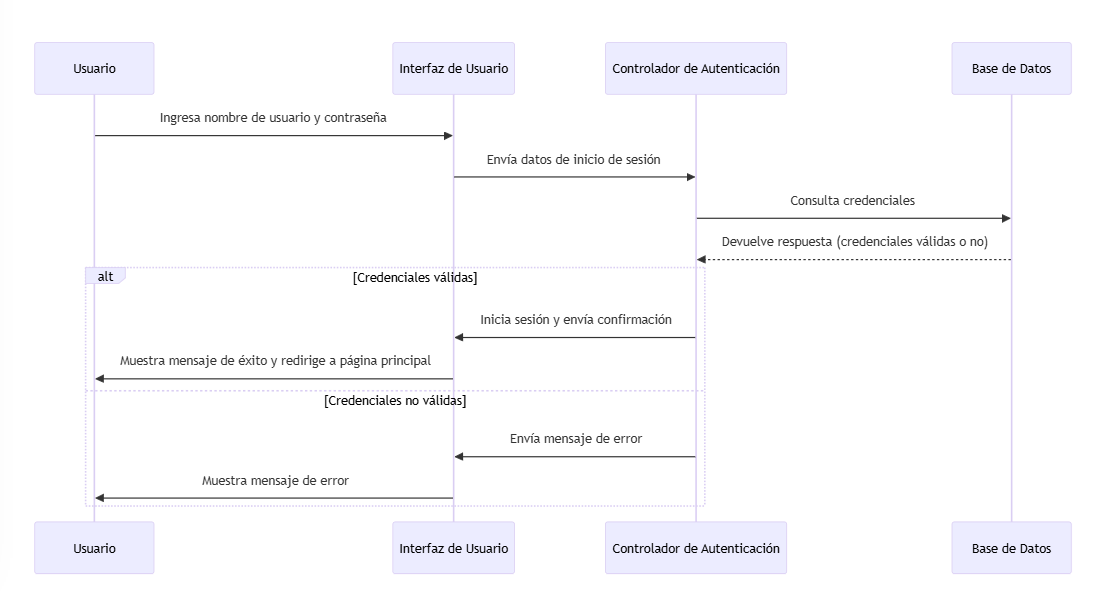
## Diseño de Diagramas UML

### Diagramas de casos de uso

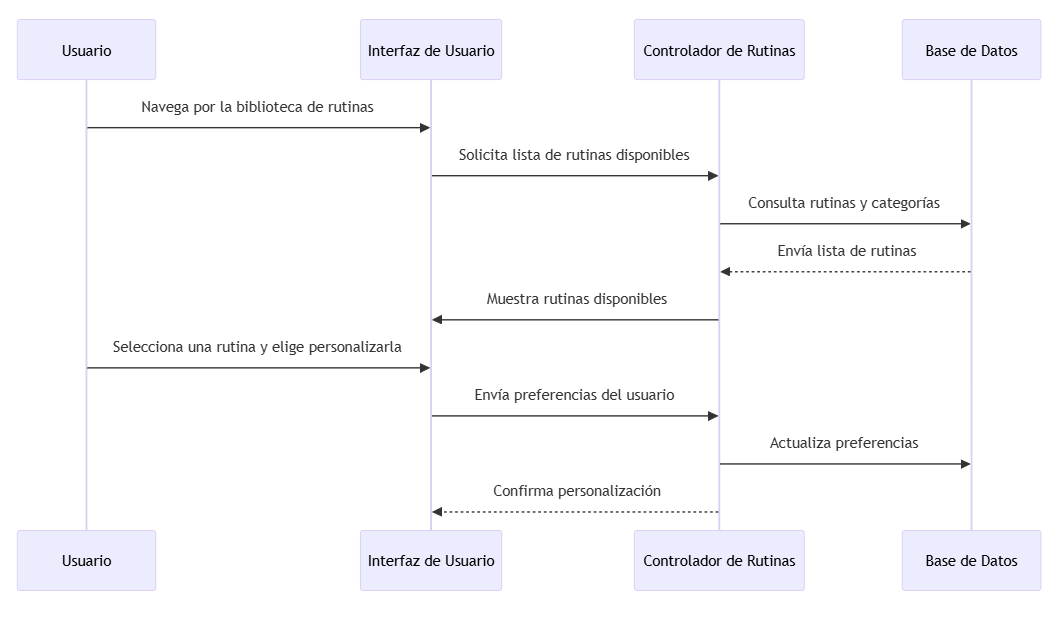


### Diagramas de secuencia:

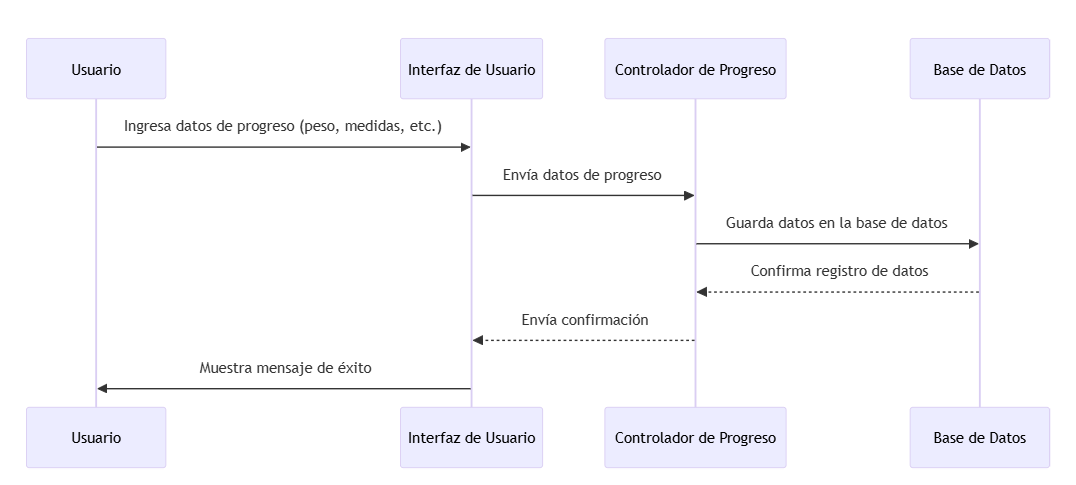
* + - 1. Diagrama de secuencia para el inicio de sesión



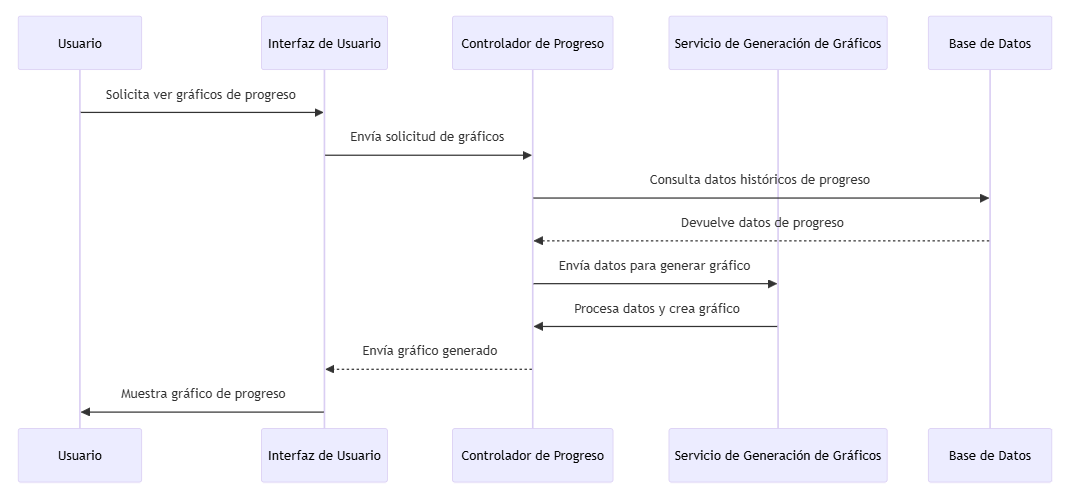
* + - 1. Diagrama de secuencia para la selección y personalización de rutina



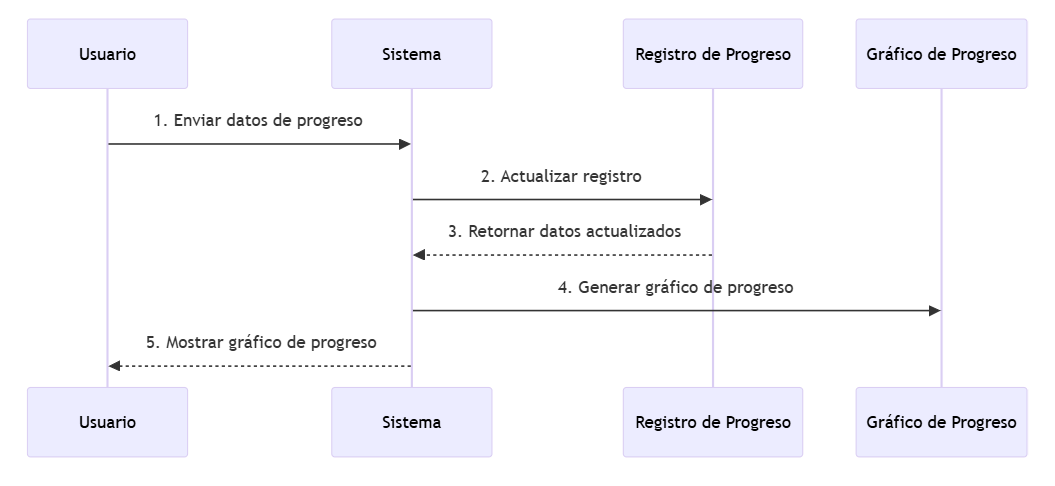
* + - 1. Diagrama de secuencia para el registro de progreso físico



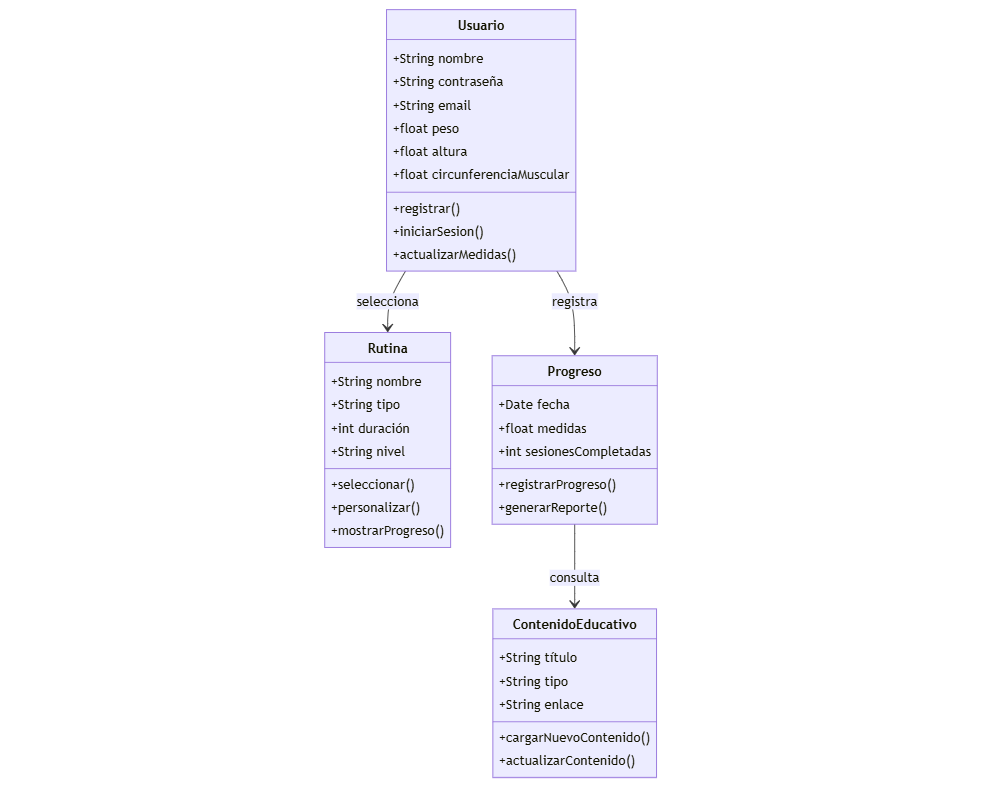
* + - 1. Diagrama de secuencia para la generación y visualización de gráficos de progreso



### Diagramas de colaboración

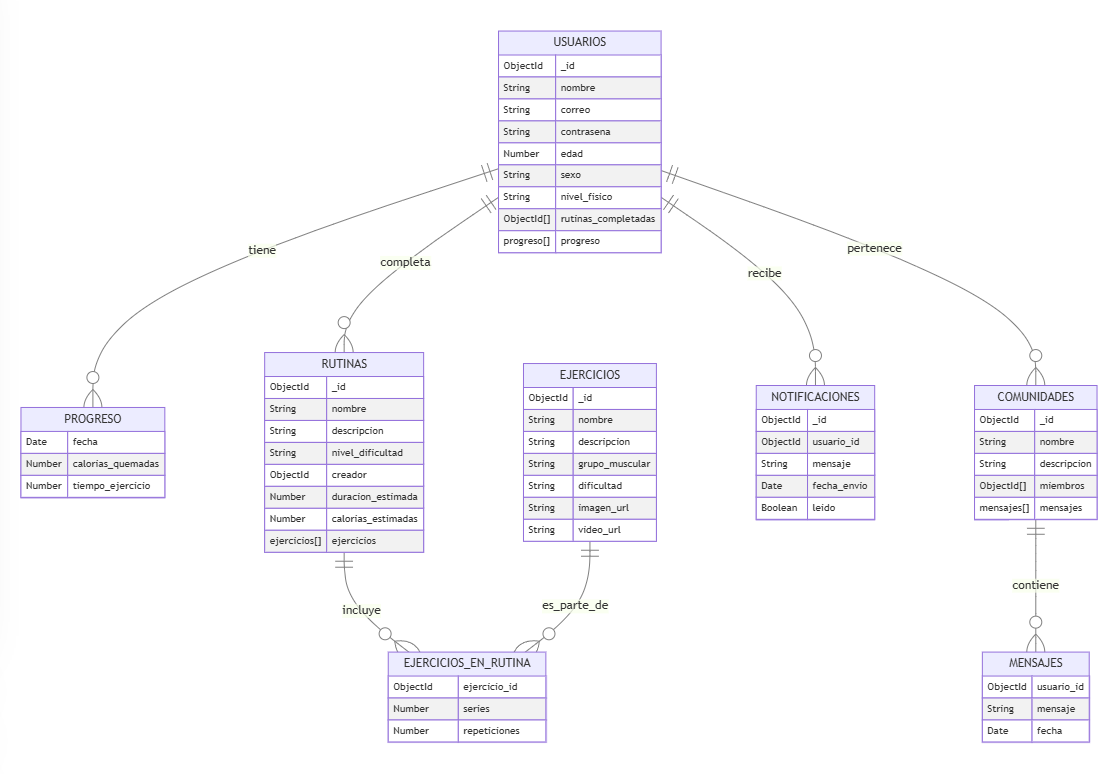


### Diagramas de clases

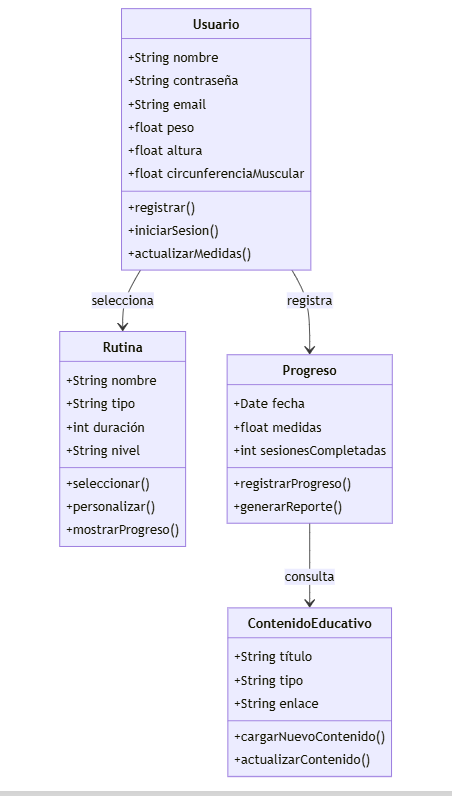


## Diseño de Base de Datos

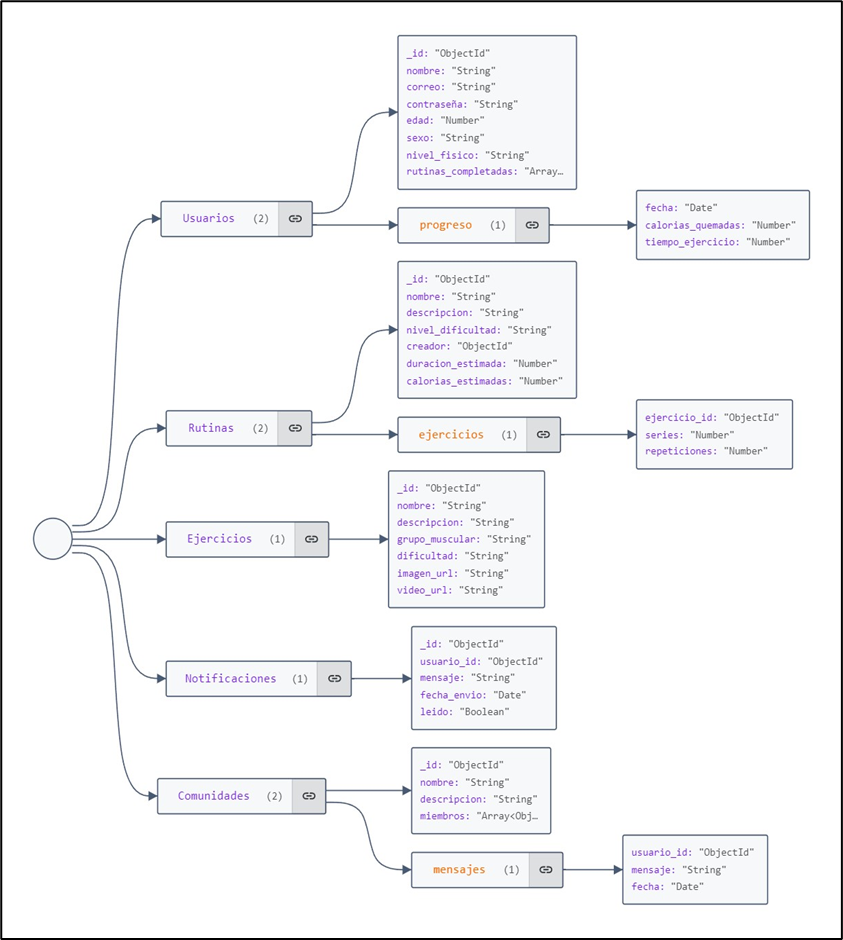
### Diseño conceptual (E/R)



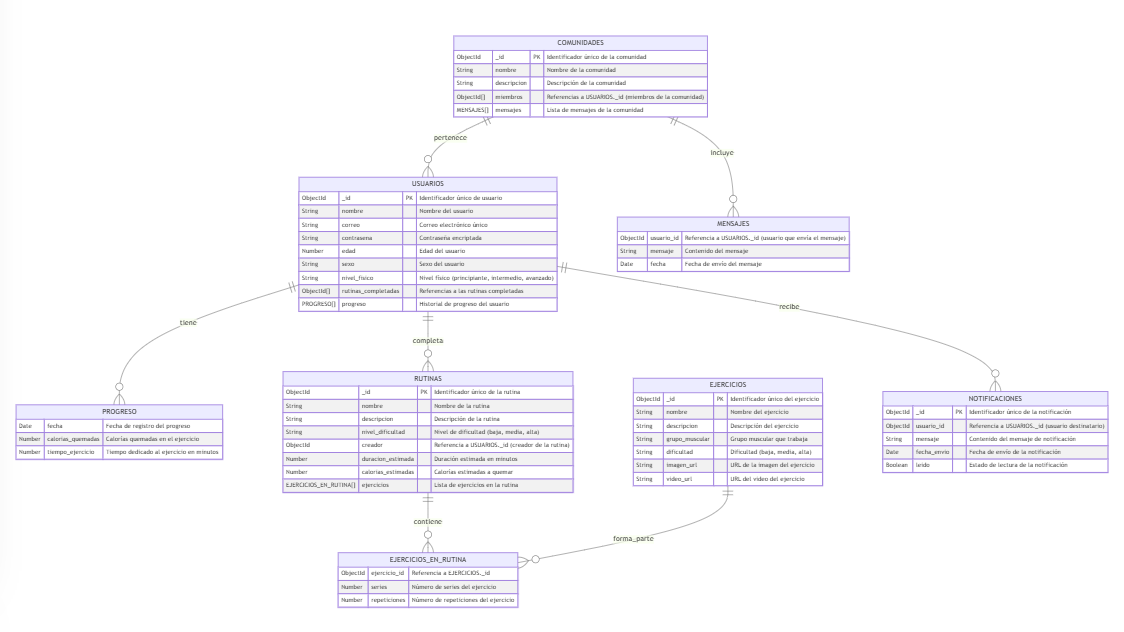
### Diseño lógico



### Diseño físico

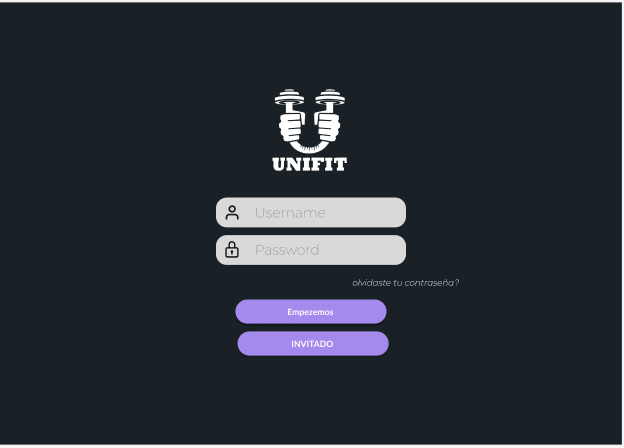


### Modelado de base de datos



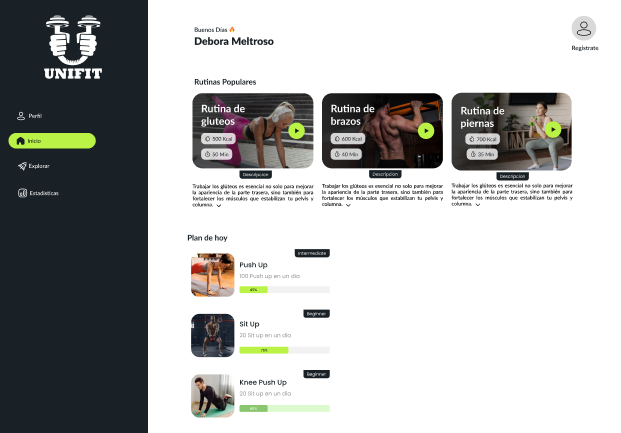
## Diseño de Interfaces Básicas

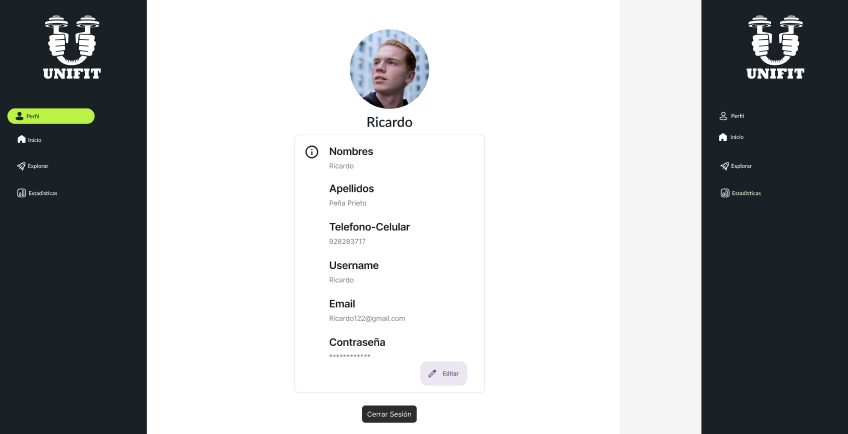
### Acceso login

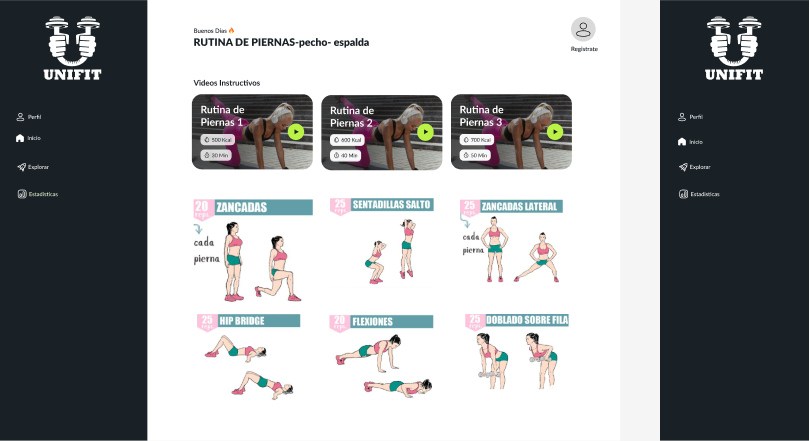


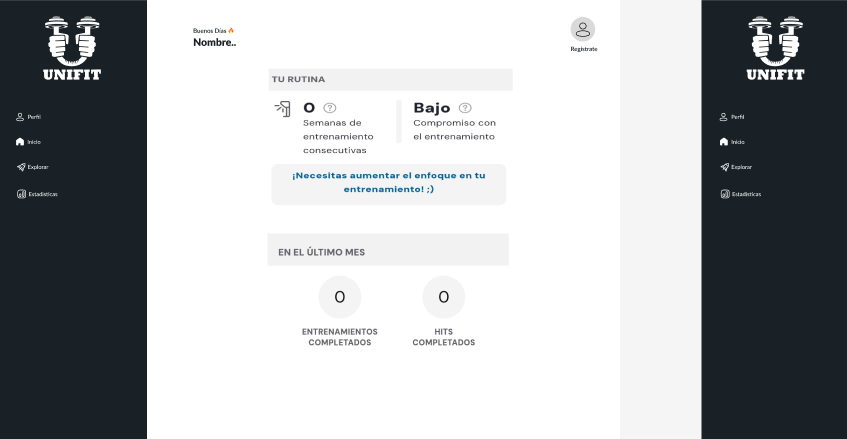
### Interfaz …

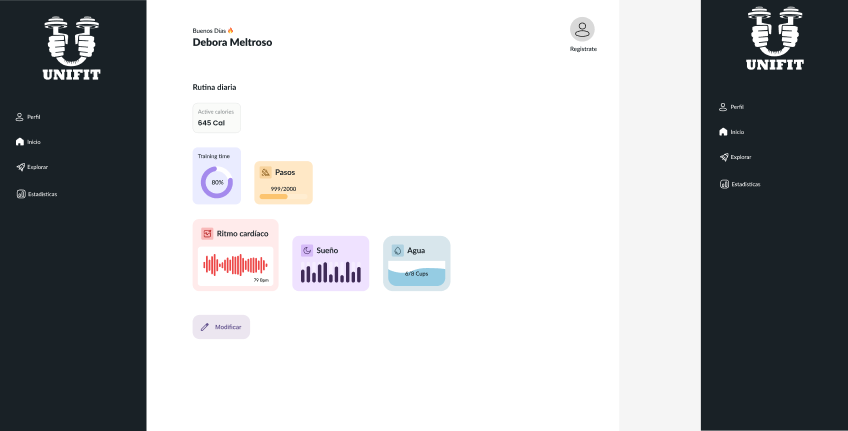












# CAPÍTULO 6

# CODIFICACIÓN DEL SOFTWARE

## Desarrollo del Sprint 1

### Sprint planning

**Actividades principales:**

Diseño e implementación de la interfaz inicial del perfil.

Configuración del sistema de registro y autenticación de usuarios.

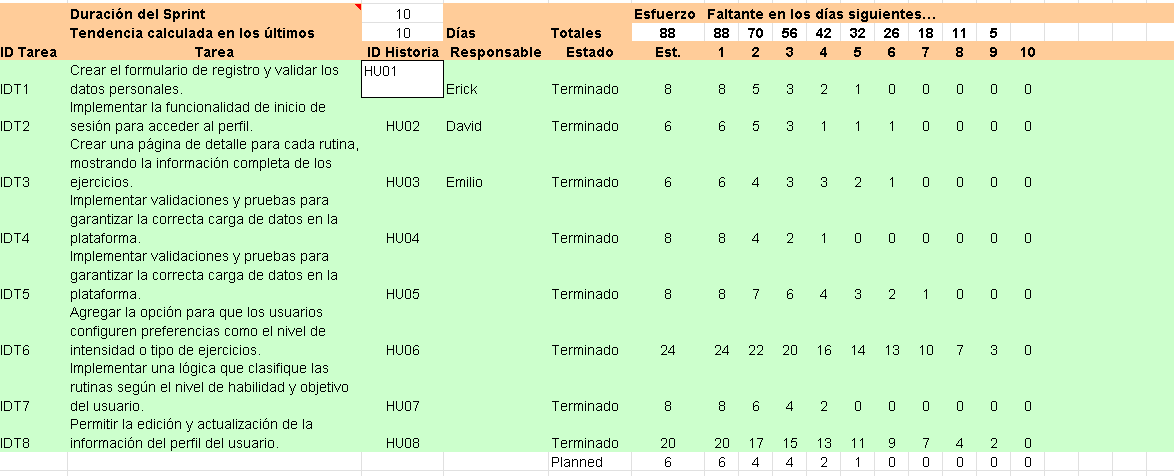
Definición de atributos básicos del perfil como nombre, correo y foto.

Pruebas unitarias para asegurar funcionalidad básica.

Se definieron las siguientes tareas prioritarias:

* Diseño de la interfaz inicial del perfil.
* Configuración del sistema de registro y autenticación.
* Definición de atributos básicos del perfil (nombre, correo, foto).

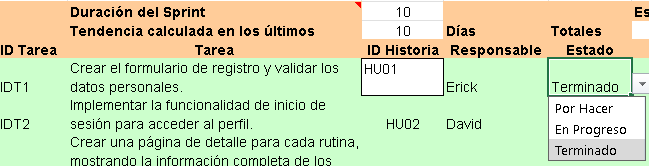
### Sprint backlog



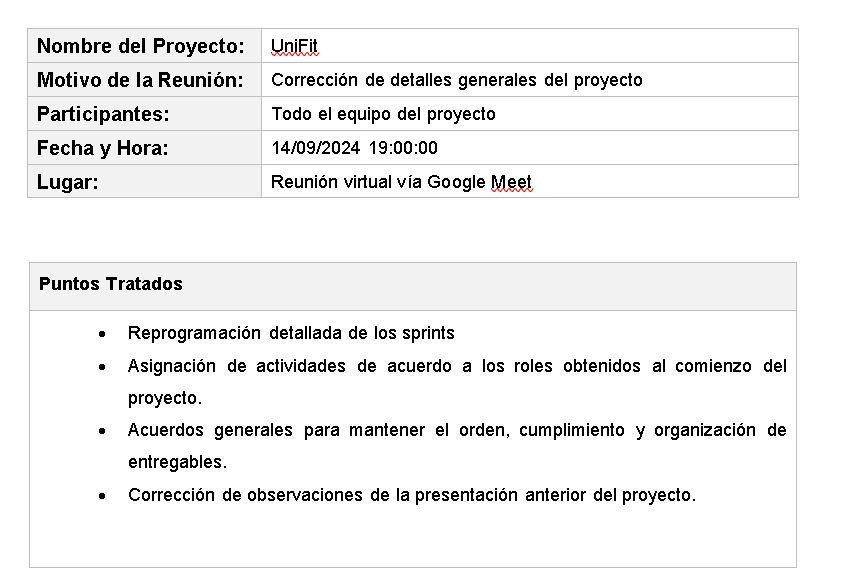
### Historias de usuarios

* Crear el formulario de registro de usuario
* Implementar el sistema de inicio de sesión (login)
* Crear una página principal con las rutinas de entrenamiento
* Crear el espacio de detalles de cada rutina
* Implementar una funcionalidad para que la página de detalles de rutinas personalizadas
* Crear mi perfil
* Implementar la visualización de rutinas personalizadas
* Actualizar mi información personal

### Taskboard

Se organizaron las tareas en columnas de "Por hacer", "En progreso" y "Completado".

### Daily scrum

Reuniones diarias de 15 minutos para revisar el progreso de cada tarea y solucionar bloqueos.

### Sprint review

### Criterios de aceptación

### Resultados del sprint

#### Evidencias.

#### Prueba de desarrollo.

#### ….

### Sprint retrospective

## Desarrollo del Sprint 2

### Sprint planning

Se definieron objetivos para implementar notificaciones y personalización de rutinas:

* Diseñar una interfaz intuitiva para las rutinas.
* Crear un sistema de notificaciones automáticas.

### Sprint backlog

* Programar algoritmos de personalización de rutinas.
* Desarrollar notificaciones push.
* Integrar base de datos para preferencias del usuario.

### Historias de usuarios

* Como usuario, quiero crear y personalizar rutinas de ejercicio según mis objetivos.
* Como usuario, quiero recibir recordatorios automáticos para mantenerme en línea con mis metas.

### Taskboard

Tareas divididas en planificación visual, desarrollo de backend, y pruebas de integración.

### Daily scrum

Se presentó un prototipo funcional de la gestión de rutinas y sistema de notificaciones.

### Sprint review

Se presentó un prototipo funcional de la gestión de rutinas y sistema de notificaciones.

### Criterios de aceptación

* Las notificaciones deben enviarse automáticamente según las preferencias del usuario.
* Las rutinas personalizadas deben ser editables y almacenadas correctamente.

### 

## Desarrollo del Sprint 3

### Sprint planning

Incorporar opciones para compartir progreso en redes sociales.

Mejorar la conectividad entre módulos.

### Sprint backlog

### Historias de usuarios

### Taskboard

### Daily scrum

### Sprint review

### Criterios de aceptación

* Las publicaciones en redes deben ejecutarse sin errores.
* El panel debe reflejar datos en tiempo real.

### 

## Desarrollo del Sprint 4

### Sprint planning

* Incorporar opciones para compartir progreso en redes sociales.
* Mejorar la conectividad entre módulos.

### Sprint backlog

* Desarrollo de API para compartir datos.
* Pruebas de conectividad y rendimiento.

### Historias de usuarios

* Como usuario, quiero compartir mis logros en redes sociales para motivar a otros.
* Como usuario, quiero visualizar mi progreso en un panel de control dinámico

### Taskboard

Organización en diseño de APIs, frontend y backend de panel.

### Daily scrum

Monitoreo del progreso y ajuste en requerimientos.

### Sprint review

### Criterios de aceptación

## Desarrollo del Sprint 5

### Sprint planning

### Sprint backlog

### Historias de usuarios

### Taskboard

### Daily scrum

### Sprint review

### Criterios de aceptación

### 

# CAPÍTULO 7

# PRUEBAS DE SOFTWARE

## Plan de Pruebas

En este capítulo se detalla el proceso de pruebas llevado a cabo durante el desarrollo del software UniFit. Se realizaron pruebas de caja negra para verificar el correcto funcionamiento de los módulos y las interacciones con las API mediante Postman. Adicionalmente, se implementó un plan de pruebas que incluye los pasos seguidos y los resultados obtenidos.

Objetivo del Plan de Pruebas:

* Garantizar que todas las funcionalidades del sistema se ejecuten según lo esperado.
* Identificar errores en los flujos principales de la plataforma.

Estrategia:

* Pruebas de caja negra en módulos clave para evaluar entradas y salidas.
* Verificación de las API usando Postman para asegurar la correcta comunicación entre el frontend y el backend.

**7.1.1 Pruebas de Caja Negra**

Definición:

Se evaluaron los módulos mediante pruebas de caja negra, donde se analizó si las salidas corresponden a las entradas definidas sin revisar el código interno.

Pasos Realizados:

Identificar las funcionalidades principales a probar (registro de usuarios, inicio de sesión, creación de rutinas, etc.).

Diseñar casos de prueba con diferentes escenarios (entradas válidas, entradas no válidas, datos límite).

Ejecutar cada caso de prueba y registrar los resultados en una tabla de Excel.

Comparar los resultados obtenidos con los esperados.

Resultados:

Se logró detectar y corregir errores en las validaciones de datos y en la interfaz de usuario antes de la entrega final.

| **Caso** | **Entrada** | **Salida Esperada** | **Resultado Obtenido** | **Estado** |
| --- | --- | --- | --- | --- |
| Registro usuario válido | Correo y contraseña válidos | Mensaje de confirmación | Mensaje de confirmación | Aprobado |
| Registro usuario inválido | Contraseña vacía | Error: "Campo requerido" | Error: "Campo requerido" | Aprobado |

#### 7.1.2 Pruebas de API con Postman

**Proceso:**

1. Se definieron los endpoints clave a probar, incluyendo:
   * Creación de usuarios.
   * Actualización de datos.
   * Recuperación de rutinas.
2. Para cada endpoint, se diseñaron escenarios con diferentes métodos HTTP (*GET, POST, PUT, DELETE*).
3. Validación de respuestas HTTP (200, 400, 500) y del formato JSON devuelto.

**Resultados:**Las pruebas con Postman confirmaron la correcta integración y comunicación entre las API y los componentes del sistema.

**Ejemplo de Prueba de API:**

* **Endpoint:** /api/register
* **Método:** POST

**Entrada:**json  
Copiar código  
{

"email": "test@unifit.com",

"password": "123456"

}

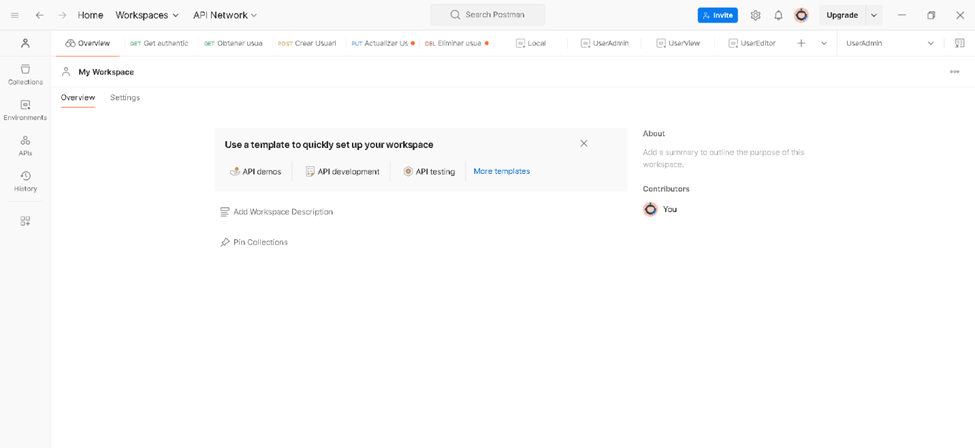
**Salida Esperada:**json  
Copiar código  
{

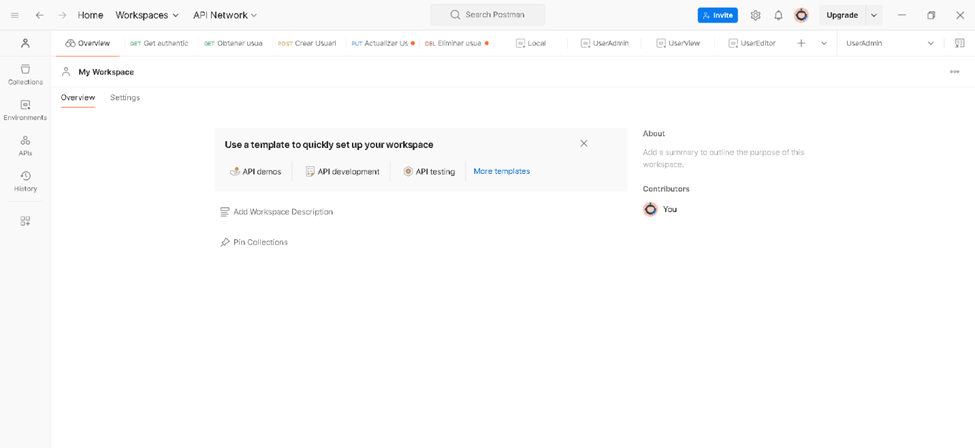
"status": "success",

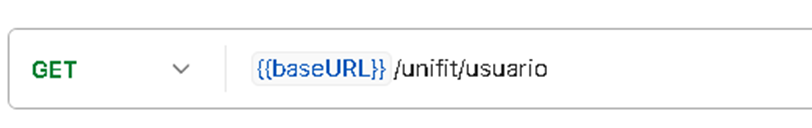
"message": "Usuario registrado exitosamente"

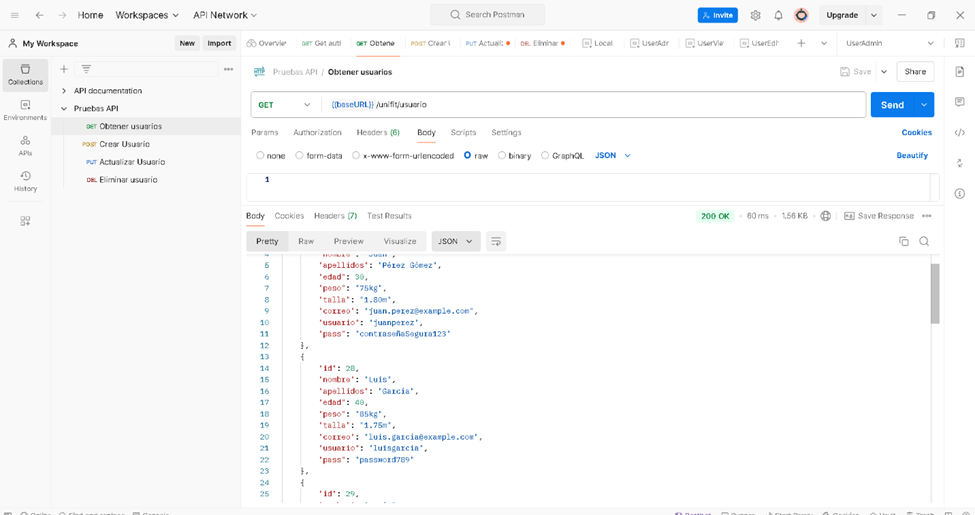
}

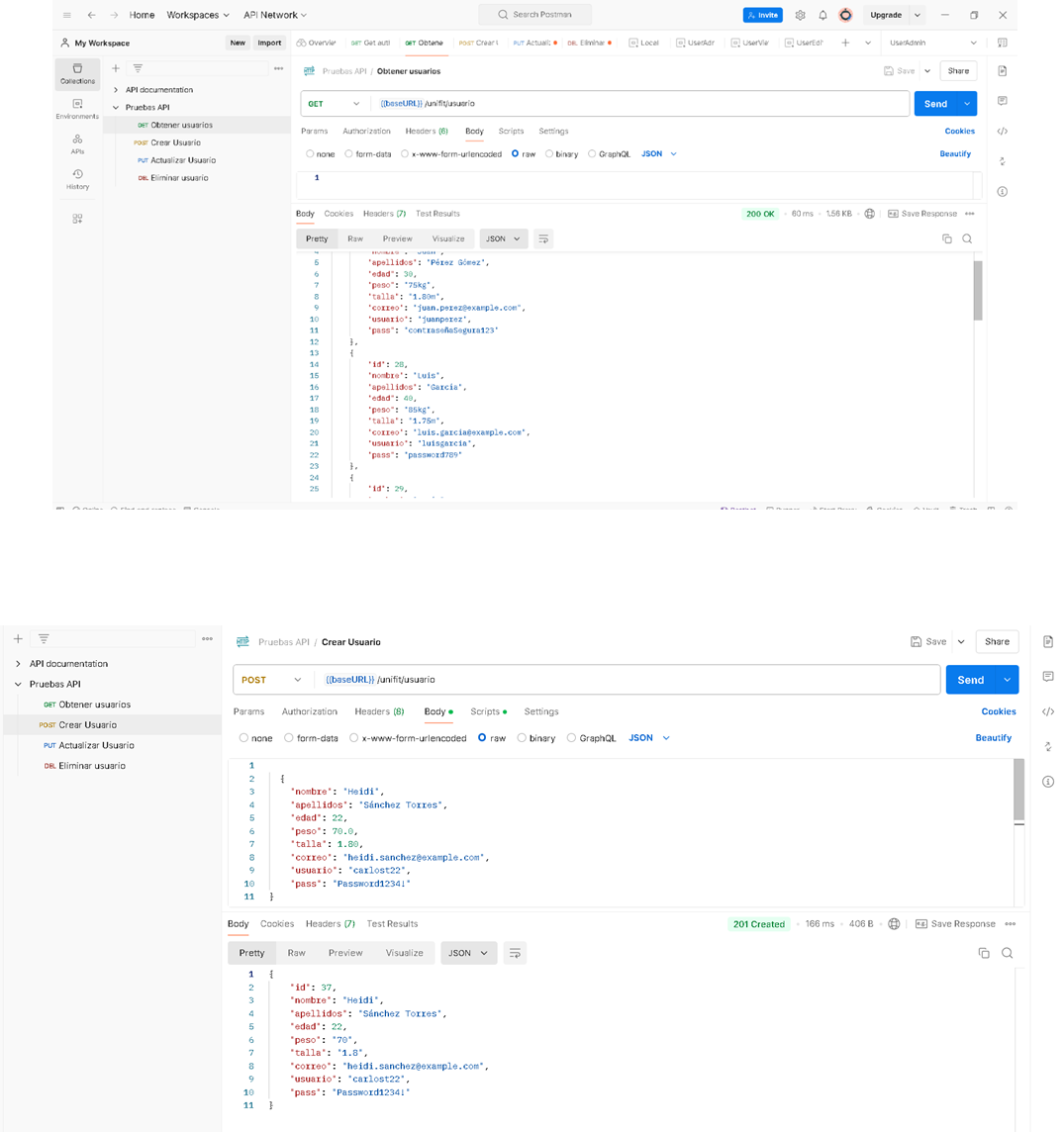
* **Resultado Obtenido:** Igual a la salida esperada.

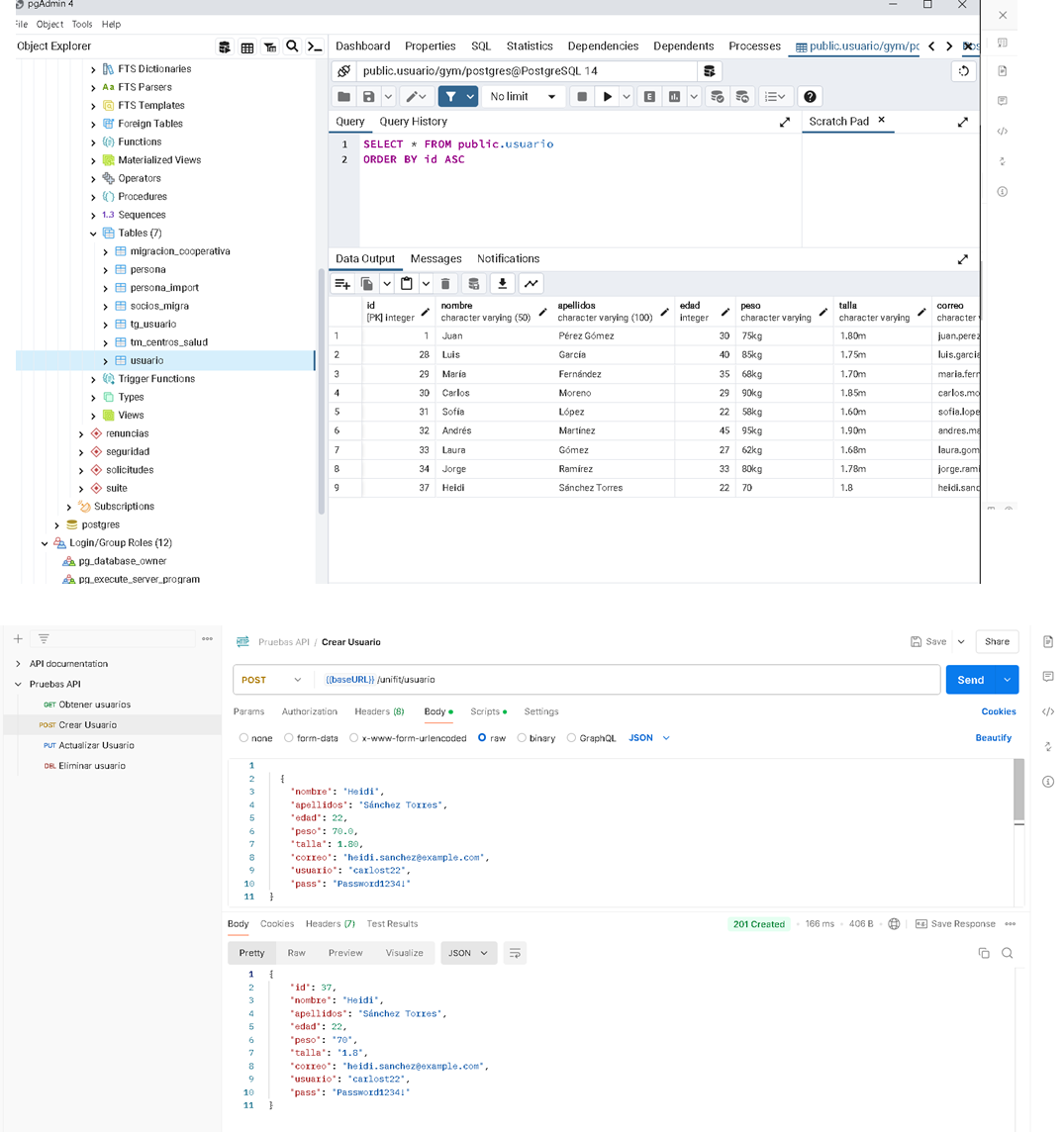


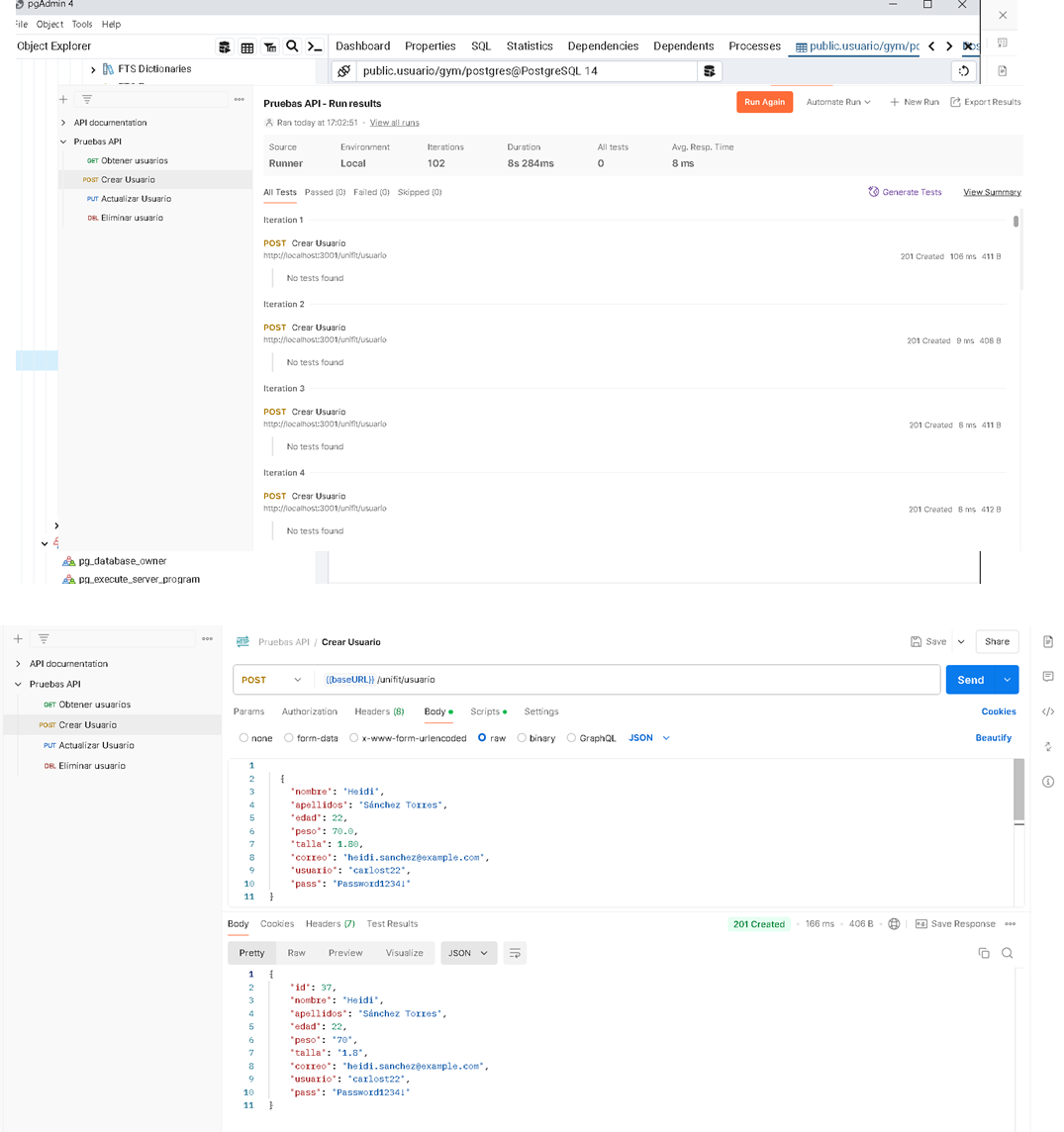












# CONCLUSIONES

1. Cumplimiento de Objetivos

* **UniFit** alcanzó su objetivo de ofrecer una plataforma de entrenamiento adaptada a las necesidades de los estudiantes.
* Las métricas implementadas permitieron un seguimiento continuo y una alineación con el **cronograma** previsto.

1. Métricas Clave

* **Burnup Chart** y **Burndown Chart**: Facilitó la toma de decisiones y la detección de desviaciones.
* **SPI** (Índice de Desempeño del Cronograma): Midió la **efectividad** del equipo respecto al cronograma, permitiendo ajustes oportunos.

# RECOMENDACIONES

1. Mejoras Sugeridas

* Realizar **ajustes en tiempo real** para asegurar que el proyecto siga alineado con los objetivos
* Añadir **Cycle Time** y **Lead Time** para una visión más completa del flujo de trabajo y planificación.

1. Impacto de las Métricas

* Las métricas ayudaron al equipo a **adaptarse a desafíos** y mantener el proyecto en curso.
* La experiencia reafirma la **importancia de la gestión de métricas** en proyectos ágiles, estableciendo una base sólida para **futuras implementaciones**.

# ANEXOS

## Anexo 01. Manual Técnico

## Anexo 02. Manual de Usuario