

**GYMTRACKER**

**Especificaciones técnicas**

**Equipo de Desarrollo:**

Jonattan Angelino Celis Carvajal

Yesica Viviana García Segura

Diego Alejandro Núñez Núñez

Sthiben Jesus Ruiz Reyes

**FICHA**: 3102803

**Tecnólogo en Análisis y Desarrollo de Software**

SENA CENTRO DE ELECTRICIDAD, ELECTRÓNICA Y TELECOMUNICACIONES

2025



INTRODUCCIÓN

El presente documento expone las especificaciones técnicas fundamentales que guiarán el desarrollo del sistema ***GymTracker.***  A partir de los datos recolectados mediante encuestas a los usuarios y un análisis detallado de las necesidades del proyecto, se han definido las bases técnicas que aseguran el funcionamiento eficiente y la optimización del sistema. A continuación detallan aspectos técnicos clave como la metodología de desarrollos seleccionada, los requerimientos funcionales y no funcionales, así como el diseño del sistema mediante diagramas UML. Además , se incluyen representaciones gráficas y prototipos visuales que ilustran la interfaz y la estructura del sistema, brindando una visión completa de su funcionamiento.

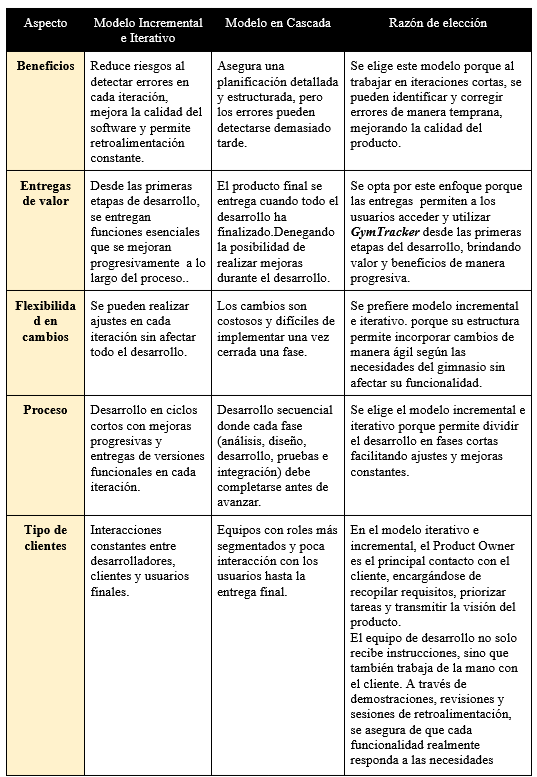
1. DEFINICIÓN TÉCNICA

1.1 Modelo de desarrollo de software

1.1.1 Modelo Incremental e iterativo

Para la ejecución del plan de desarrollo de ***GymTracker***, se ha seleccionado un modelo que prioriza la flexibilidad y la mejora continua: el modelo Incremental e iterativo. Esto permitirá que el sistema evolucione constantemente en lugar de ser un producto estático. Además, en el siguiente documento se establece una comparación con el modelo en cascada, explicando las razones por las cuales se optó por el modelo incremental e iterativo y destacando sus ventajas dentro del proyecto.

*“Para visualizar en detalle el modelo de desarrollo seleccionado ,haga clic en la siguiente imagen .Esto abrirá un archivo con información más detallada”.*

[**](https://docs.google.com/document/d/1XhejZyqemhni4c16ndNsqxYt_99N5NqCLEyIm9iwpG8/edit?tab=t.0)

*(Figura 1.1. Diferencias entre el modelo incremental e iterativo y el modelo cascada.)*

1.2 Historias de usuarios y requerimientos funcionales y no funcionales

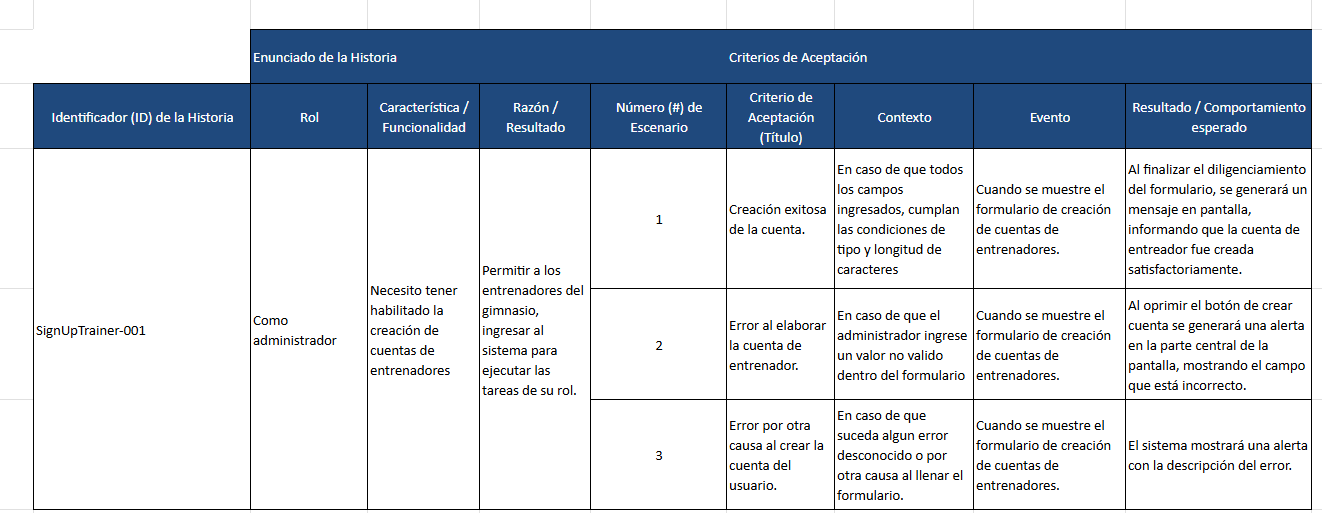
Las historias de usuario describen las necesidades de un usuario o cliente para un producto o software. Estas son esenciales para definir cómo debe funcionar el sistema desde la perspectiva de quien lo utilizará.

Dentro del sistema ***GymTracker,*** estas historias permiten establecer un conjunto de pasos claves para su desarrollo, asegurando que cada funcionalidad cumpla con los criterios adecuados***.***

A continuación la estructura del siguiente documento:

* **Si eres un usuario que quiere reservar clases,** aquí sabrás cómo definir desde el rol hasta el resultado esperado.
* **Si eres un instructor que necesita gestionar sus usuarios,** entenderás por qué cada evento debe responder a un comportamiento del sistema claro y confiable.

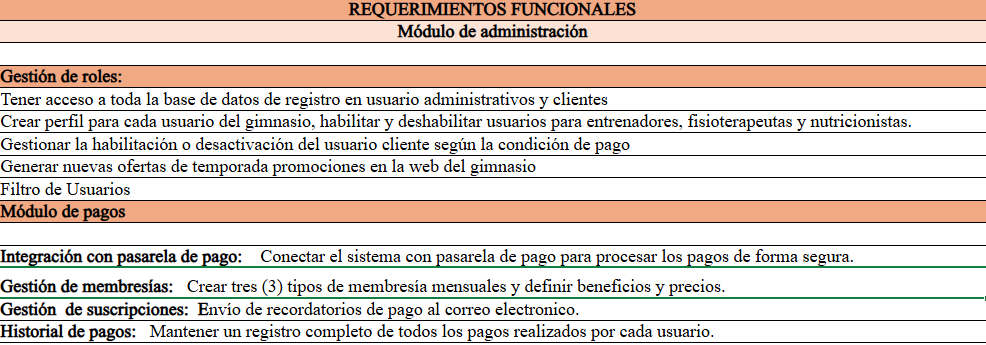
*“Para visualizar el detallado de las historias de usuario . Haga click en la siguiente imagen, esto abrirá un archivo tipo Excel , donde contiene varios libros, cada libro tiene la funcionalidad de cada profesional y administrativos de manera detallada."*

[](https://soysena-my.sharepoint.com/:x:/g/personal/danuez6_soy_sena_edu_co/EeFy5v-3lvZChYV-OJFsWWwBlap0BLLIQdMY91VfJIyAaA?rtime=JHNfLLFl3Ug)

*(Figura 1.2 Historia de usuarios )*

Los requerimientos funcionales y no funcionales establecen las características y condiciones que debe cumplir el software para satisfacer las necesidades del usuario. Los funcionales definen las acciones y capacidades del sistema,mientras que los no funcionales determinan atributos como rendimiento, seguridad y usabilidad ,asegurando la calidad del sistema. En el caso de ***Gymtracker***, estos requisitos forman la base técnica que garantiza su correcto funcionamiento,optimizando la experiencia del usuario y la eficiencia operativa.

*“Para visualizar el detallado de los requerimientos funcionales y no funcionales.Haga click en la siguiente imagen, esto abrirá un archivo tipo Excel, de manera detallada."*

[****](https://soysena-my.sharepoint.com/:x:/g/personal/danuez6_soy_sena_edu_co/EeFy5v-3lvZChYV-OJFsWWwBlap0BLLIQdMY91VfJIyAaA?rtime=JHNfLLFl3Ug)

*(Figura 1.2.1 Requerimientos funcionales y no funcionales del sistema. )*

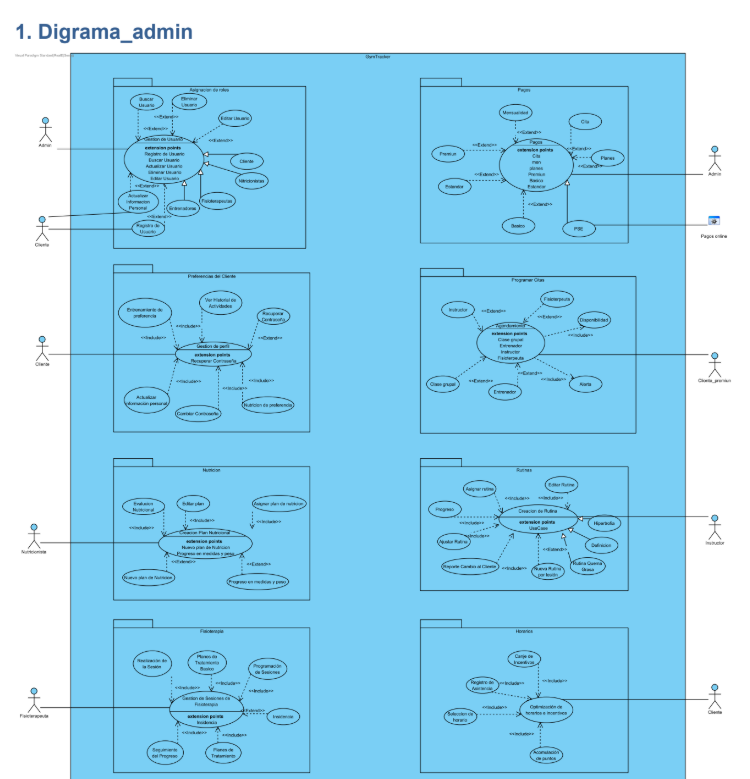
2. DISEÑO DEL SISTEMA UML

2.1 Diagrama de casos de uso

El

diseño del sistema mediante UML(Unified Modeling Language) permite representar visualmente la estructura y el comportamiento de  **GymTracker.** A través de diagramas como los casos de uso, se detallan las interacciones entre los diferentes actores, así como las funcionalidades y relaciones dentro del sistema.

*A continuación, haga clic en la siguiente imagen para acceder al documento con el diagrama detallado.*

[](https://drive.google.com/file/d/1Er6khYiGcQP9Wo66daQPejGIqaP9m-FQ/view?usp=sharing)

*(Figura 2.1. Diagrama de casos de uso. )*

2.2 Diagrama de Clases

El diagrama de clases es una representación esencial dentro de la Programación orientada a Objetos (OOP), ya que define la estructura estática de ***GymTracker.*** En este diagrama, se detallan las clases, sus atributos, métodos y relaciones, incluyendo, asociación, herencia, composición, agregación y otras dependencias.

*A continuación, haga clic en la siguiente imagen para extender la imagen de manera detallada.*

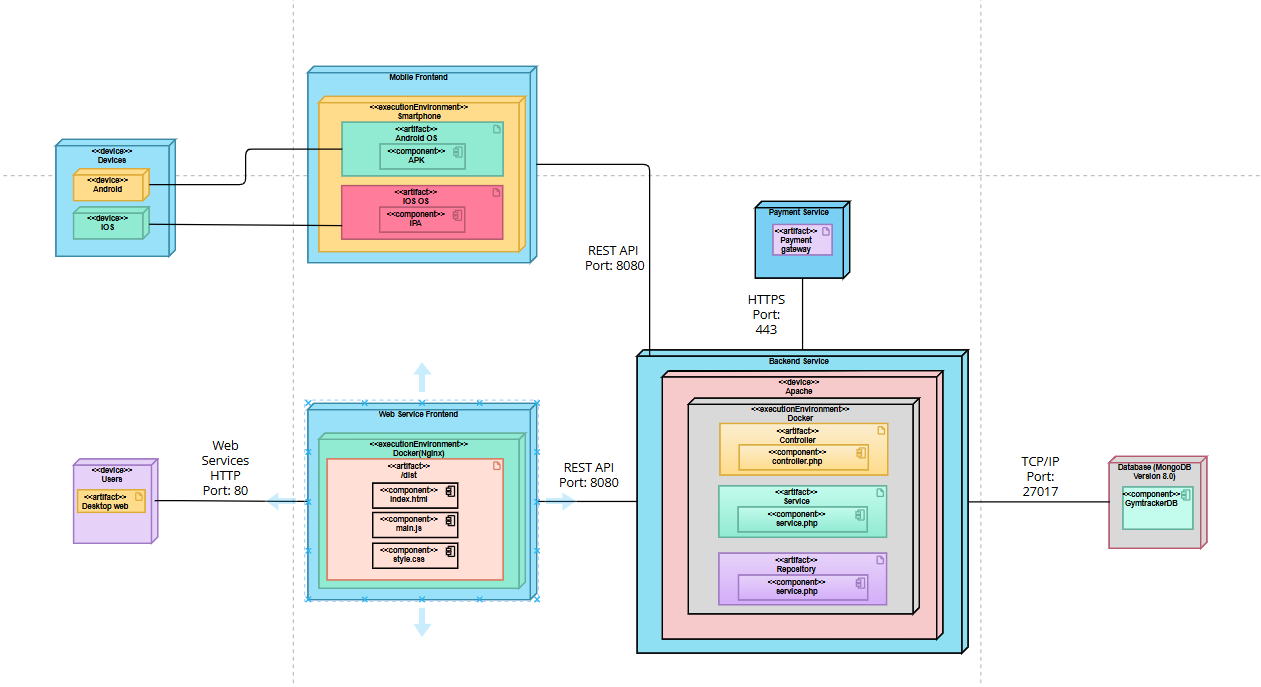
[****](https://soysena-my.sharepoint.com/:i:/g/personal/danuez6_soy_sena_edu_co/EaWDslt-u79Jqpu88r-8kAsBe0mHqz6zK3xlbLjUUE9yoQ?e=rOhU2b)

*(Figura 2.2. Diagrama de clases. )*

2.3 Diagrama de despliegue

El diagrama de despliegue es una representación gráfica dentro de la arquitectura de software que ilustra cómo los componentes del sistema **GymTracker** se distribuyen en el hardware. Este diagrama especifica la relación entre los nodos físicos (servidores, dispositivos móviles, bases de datos) y los artefactos de software, definiendo su comunicación e interacción para garantizar un funcionamiento eficiente del sistema.

*A continuación, haga clic en la siguiente imagen para extender la imagen de manera detallada.*

[](https://soysena-my.sharepoint.com/:i:/g/personal/danuez6_soy_sena_edu_co/EdGkSRViEcRKv28yuX2VTJ4BgE2DVzYr2i_-yUYp9fmA6w?e=GFLJ23)

*(Figura 2.3. Diagrama de despliegue.)*

3. DEFINICIONES GRÁFICAS

3.1 Mockup

Los mockups proporcionan una primera representación visual de las interfaces de ***GymTracker,*** permitiendo anticipar la experiencia del usuario antes del desarrollo. A través de estos prototipos, se pueden evaluar la usabilidad y la disposición de los elementos, asegurando una interacción intuitiva con la plataforma.

En esta sección se ilustran casos clave, como:

* La gestión de clientes mediante el **CRUD** de un administrador.
* La creación de planes alimenticios por parte del rol “Nutricionista”.

Estos modelos sirven como un puente entre la conceptualización y la implementación, facilitando iteraciones y mejoras antes de la fase de programación.

*A continuación, haga clic en la siguiente imagen para ingresar al Mockup de manera extendida y detallada.*

[Selecciona el siguiente diagrama: 
](https://www.figma.com/design/2u45G7e3Ol9YfGHSTCZSLq/GymTracker-Project?node-id=0-1&node-type=canvas)

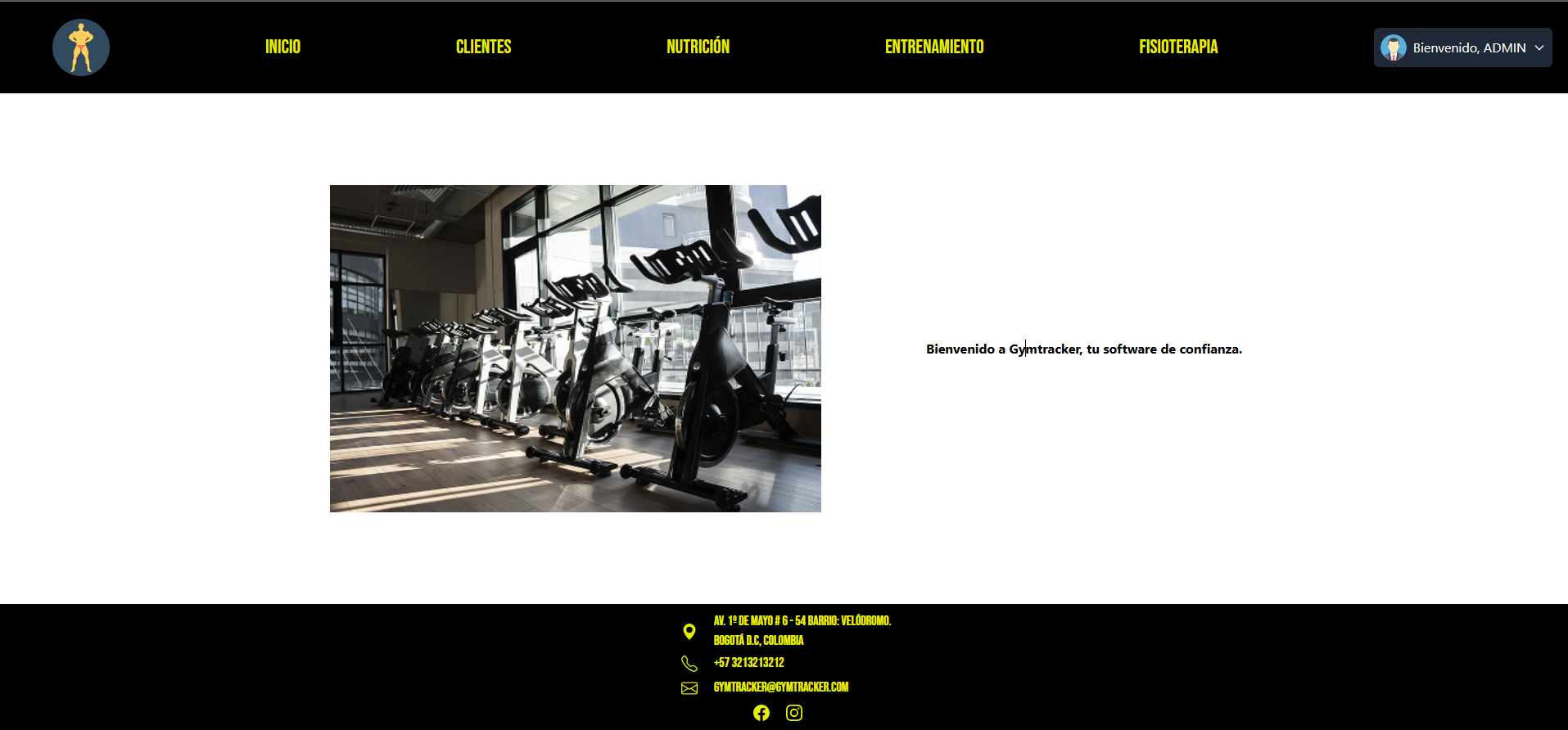
*(Figura 3.1 . Mockup de GymTracker. )*

3.2 Prototipo no funcional

Un prototipo no funcional es una representación visual del sistema que permite evaluar la interfaz y la experiencia del usuario antes del desarrollo de software. En esta etapa, se define la arquitectura de navegación y la disposición de los elementos clave sin implementar su funcionalidad operativa.

Para ***GymTracker***, este prototipo establece las bases de una plataforma optimizada para la gestión de rutinas de ejercicio, monitoreo de progresos y agendamiento de citas.Aunque aún es un concepto visual , representa la estructura sobre la cual se desarrollara un sistema de información web de manera detallada.

*A continuación, haga clic en la imagen para ingresar al prototipo no funcional de manera extendida y detallada.*

[](https://github.com/ddienu/GymTracker_Documentation/tree/Diego_DEV/Mockup)

*(Figura 3.2. Prototipo no funcional de GymTracker. )*

Referencias bibliográficas:

Sommerville, I. (2011). Software engineering (9ª ed.). Pearson.