Universidad de Valladolid

ESCUELA DE INGENIERÍAS INDUSTRIALES

UNIVERSIDAD DE VALLADOLID

ESCUELA DE INGENIERIAS INDUSTRIALES

Grado en Ingeniería en Electrónica Industrial y Automática

# Aplicación web para la gestión y análisis de datos de un sistema de rehabilitación con realidad aumentada


Autor:

Caballero Cano, Rafael

Tutores:

Miguel Trespaderne, Félix

Fuente López, Eusebio De La


Valladolid, junio 2025.

## Resumen

Este Trabajo de Fin de Grado detalla el desarrollo de una aplicación web con Streamlit, respaldada por una base de datos Supabase (PostgreSQL), para la gestión y análisis de datos en rehabilitación neuromotora con Realidad Aumentada (RA). Como evolución del sistema M3Display (Matilla Plaza, 2023), la plataforma supera el manejo de CSVs locales mediante un almacenamiento centralizado y estructurado de datos de pacientes, sesiones, trayectorias numéricas de movimiento (timestamp, X, Y). La interfaz de Streamlit facilita a los terapeutas el registro de sesiones, la visualización de datos almacenados en Supabase, incluyendo trayectorias y GIFs, y el acceso a análisis cuantitativos. Se implementó un sistema de autenticación seguro y la gestión de archivos en Supabase Storage, demostrando la viabilidad de esta arquitectura para un seguimiento objetivo del paciente.

## Palabras Clave

Streamlit, Supabase, Aplicación Web, Base de Datos, Rehabilitación Neuromotora.

## Abstract

This Final Degree Project details the development of a web application using Streamlit, supported by a Supabase (PostgreSQL) database, for the management and analysis of data in neuromotor rehabilitation with Augmented Reality (AR). As an evolution of the M3Display system (Matilla Plaza, 2023), the platform overcomes local CSV file management through centralized and structured storage of patient, session, and, crucially, numerical movement trajectory data (timestamp, X, Y). The Streamlit interface allows therapists to register sessions, visualize data stored in Supabase, including trajectories and GIFs, and access quantitative analyses. A secure authentication system and file management in Supabase Storage were implemented, demonstrating the viability of this architecture for objective patient monitoring.

## Keywords

Streamlit, Supabase, Web Application, Database, Neuromotor Rehabilitation.

ÍNDICE PRINCIPAL

Resumen ..... 3
Palabras Clave ..... 3
Abstract ..... 3
Keywords ..... 3
CAPÍTULO 1. INTRODUCCIÓN Y OBJETIVOS ..... 11
1.1 DESCRIPCIÓN Y CONTEXTO DEL PROYECTO ..... 11
1.2 MOTIVACIÓN ..... 12
1.3 OBJETIVOS ..... 12
1.4 ALCANCE ..... 13
1.5 ESTRUCTURA DEL TRABAJO ..... 14
CAPÍTULO 2. ESTADO DEL ARTE ..... 17
2.1 CONTEXTO: REHABILITACIÓN NEUROMOTORA Y SUS DESAFIÍOS ..... 17
2.2 EL PAPEL DE LA TECNOLOGÍA EN LA REHABILITACIÓN MODERNA ..... 17
2.3 PANORAMA DE TECNOLOGÍAS EMERGENTES APLICADAS A LA REHABILITACIÓN NEUROLÓGICA ..... 18
2.4 REALIDAD AUMENTADA (RA) EN REHABILIATCIÓN: CAPACIDADES Y APLICACIONES ..... 19
2.5 CASO DE ESTUDIO PREVIO: SISTEMA M3DISPLAY Y JUEGO “COINS” ..... 20
2.6 GESTIÓN DE DATOS CLÍNICOS EN REHABILITACIÓN: PLATAFORMAS Y REQUISITOS ..... 21
2.7 HERRAMIENTAS TECNOLÓGICAS PARA LA IMPLEMENTACIÓN DE PLATAFORMAS DE ANÁLISIS ..... 23
2.8 HERRAMIENTAS PARA LA GESTIÓN DE DATOS CLÍNICOS ..... 24
2.9 REVISIÓN DE SISTEMAS SIMILARES EXISTENTES ..... 24
2.10 ANALISIS COMPARATIVO Y JUSTIFICACIÓN DE LA SOLUCIÓN PROPUESTA ..... 26
CAPÍTULO 3. METODOLOGÍA ..... 29
3.1 ENFOQUE METODOLÓGICO DEL PROYECTO ..... 29
3.2 FASES DEL PROYECTO ..... 30
3.3 TÉCNICAS DE RECOLECCIÓN Y ANÁLISIS DE REQUISITOS ..... 32
3.4 HERRAMIENTAS DE DESARROLLO ..... 33
3.5 CRITERIOS DE EVALUACIÓN Y VALIDACIÓN DEL PROYECTO ..... 36
CAPÍTULO 4. ANÁLISIS DE REQUISITOS ..... 39
4.1 PERFILES DE USUARIOS ..... 39

4.2 REQUISITOS FUNCIONALES (RF) ..... 39
4.2.1 GESTIÓN DE USUARIOS (AUTENTIFICACIÓN Y AUTORIZACIÓN) ..... 39
4.2.2 GESTIÓN DE PACIENTES ..... 40
4.2.3 GESTIÓN DE SESIONES DE REHABILITACIÓN ..... 40
4.2.4 ALMACENAMIENTO Y GESTIÓN DE DATOS DE TRAYECTORIA NUMÉRICOS 40
4.2.5 ANÁLISIS CUANTITATIVO DE DATOS DE MOVIMIENTO ..... 41
4.2.6 VISUALIZACIÓN DE DATOS Y PROGRESO ..... 41
4.2.6 GENERACIÓN DE INFORMES (SIMPLIFICADO) ..... 42
4.3 REQUISITOS NO FUNCIONALES (RNF) ..... 42
4.3.1 SEGURIDAD DE DATOS ..... 42
4.3.2 USABILIDAD ..... 42
4.3.3 RENDIMIENTO ..... 43
4.3.4 COMPATIBILIDAD ..... 43
4.3.5 MANTENIBILIDAD Y EXTENSIBILIDAD (CONSIDERACIONES DE DISEÑO) .. 43
CAPÍTULO 5. DISEÑO DEL SISTEMA ..... 45
5.1 ARQUITECTURA GENERAL DEL SISTEMA ..... 45
5.1.1 VISIÓN GENERAL Y FILOSOFÍA DE DISEÑO ..... 45
5.1.2 DIAGRAMA DE COMPONENTES PRINCIPALES ..... 46
5.1.3 ARQUITECTURA CLIENTE-SERVIDOR Y FLUJO DE DATOS PRINCIPAL ..... 47
5.2 DISEÑO TÉCNICO ..... 49
5.2.1 CONJUNTO DE TECNOLOGÍAS UTILIZADO ..... 49
5.2.2 DISEÑO DE LA BASE DE DATOS (SUPABASE/POSTGRESQL) ..... 57
5.2.3 DISEÑO DEL ALMACENAMIENTO DE ARCHIVOS (SUPABASE STORAGE) ... 62
5.2.4 DISEÑO DE LA LÓGICA DE LA APLICACIÓN Y MÓDULOS DE ANALISIS ..... 63
5.2.5 DISEÑO DE LA SEGURIDAD ..... 64
5.2.6 DISEÑO DEL SCRIPT DE CAPTURA ..... 65
5.3 DISEÑO DE LA INTERFAZ DE USUARIO (UI) Y FLUJO DE NAVEGACIÓN ..... 65
5.3.1 PRINCIPIOS DE DISEÑO DE UI/UX ..... 66
5.3.2 ESTRUCTURA GENERAL DE LA APLICACIÓN Y NAVEGACIÓN Y DISEÑO DE LAS PANTALLAS PRINCIPALES ..... 66
5.3.3 COMPONENTES REUTILIZABES Y ESTILO VISUAL ..... 72
CAPÍTULO 6. IMPLEMENTACIÓN ..... 73
6.1 ENTORNO DE DESARROLLO, ESTRUCTURA DEL PROYECTO Y DEPENDENCIAS ..... 73
6.1.1 CONFIGURACION DEL ENTORNO DE DESARROLLO ..... 73

6.1.2 ORGANIZACIÓN DEL CÓDIGO FUENTE Y MODULARIDAD ..... 74
6.1.3 GESTIÓN DE DEPENDENCIAS ..... 75
6.2 IMPLEMENTACIÓN DETALLADA DE FUNCIONALIDADES CLAVE ..... 76
6.2.1 SISTEMA DE AUTENTIFICACIÓN DE USUARIOS Y GESTIÓN DE SESIONES . 76
6.2.2 IMPLEMENTACIÓN DEL MÓDULO DE GESTIÓN DE ENTIDADES (PACIENTES, TERAPEUTAS, TERAPIAS) ..... 77
6.2.3 IMPLEMENTACIÓN DEL MÓDULO DE SESIONES DE REHABILITACIÓN ..... 78
6.2.4 IMPLEMENTACIÓN DEL MÓDULO DE ANÁLISIS CUANTITATIVO Y VISUALIZACIÓN ..... 82
6.3 RETOS TÉCNICOS DETALLADOS Y SOLUCIONES IMPLEMENTADAS ..... 83
6.4 CONSIDERACIONES SOBRE LA SEGURIDAD DE DATOS MÉDICOS EN LA IMPLEMENTACIÓN ..... 84
6.4.1 AUTENTIFICACIÓN DE USUARIOS Y GESTIÓN DE SESIÓN ..... 84
6.4.2 AUTORIZACIÓN Y CONTROL DE ACCESO A NIVEL DE FILA (ROW LEVEL SECURITY – RLS) ..... 85
6.4.3 SEGURIDAD EN LA TRANSMISIÓN Y ALMACENAMIENTO DE DATOS ..... 85
6.4.4 CONCIENCIACIÓN SOBRE GPDR/HIPAA Y PRÁCTICAS DE DESARROLLO .. 86
CAPÍTULO 7. PRUEBAS Y VALIDACIÓN ..... 87
7.1 ESTRATEGIA Y ENTORNO DE PRUEBAS ..... 87
7.1.1 ESTRATEGIA GENERAL DE PRUEBAS ..... 87
7.1.2 ENTORNO DE PRUEBAS ..... 87
7.2 DATOS DE PRUEBA UTILIZADOS ..... 88
7.3 TIPOS DE PRUEBAS REALIZADAS Y RESULTADOS ..... 89
7.3.1 PRUEBAS FUNCIONALES ..... 89
7.3.2 PRUEBAS DE INTEGRACIÓN ..... 92
7.3.3 PRUEBAS DE USABILIDAD (INFORMALES) ..... 93
7.3.4 PRUEBAS DE REQUISITOS NO FUNCIONALES (SELECCIONADAS) ..... 93
7.4 DISCUSIÓN DE LOS RESULTADOS DE LAS PRUEBAS Y VALIDACIÓN ..... 94
CAPÍTULO 8. CONCLUSIONES Y POSIBLES MEJORAS ..... 97
8.1 LOGROS OBTENIDOS Y EVALUACIÓN DEL CUMPLIMIENTO DE LOS OBJETIVOS ..... 97
8.2 LIMITACIONES DEL TRABAJO REALIZADO ..... 100
8.3 POSIBLES MEJORAS Y LÍNEAS DE TRABAJO FUTURO ..... 101
8.4 IMPACTO POTENCIAL DEL PROYECTO EN EL CAMPO DE LA REHABILITACIÓN ASISTIDA POR TECNOLOGÍA ..... 103
8.5 CONCLUSIÓN FINAL ..... 105

BIBLIOGRAFÍA ..... 107
ANEXO 1: MANUAL DE USUARIO ..... 111
1. Introducción ..... 111
2. Acceso a la Aplicación y Autentificación ..... 111
3. Panel principal y Navegación ..... 112
4. Gestión de Tablas Generales ..... 113
5. Gestión de Sesiones de Rehabilitación ..... 114
6. Análisis de Progreso del Paciente ..... 120
7. Resolución de Problemas Comunes ..... 122

ÍNDICE DE FIGURAS

Figura 1: Diferencias entre realidad virtual, realidad aumentada y realidad mixta. Fuente: (Ordoñez, 2020)..... 19
Figura 2: Configuración hardware del sistema de RA no inmersiva M3Display. (a) Estructura general. (b) Esquema de reflexión en el espejo. (c) Vista resultante para el paciente. Fuente: Matilla Plaza (2023), Figura 8 (basada en Cisnal, 2023)..... 20
Figura 3: Detección y seguimiento de landmarks de la mano mediante MediaPipe Hands, tecnología integrada en M3Display. Fuente: (Fan Zhang, 2020)..... 21
Figura 4: Ejemplo conceptual de un dashboard clínico basado en web para la monitorización de la rehabilitación. Muestra la integración de información del paciente, un gráfico de progreso longitudinal y paneles para visualizar métricas clave del movimiento Fuente: Elaboración propia..... 22
Figura 5: Conjunto de tecnologías usadas en el proyecto, mostrando las principales herramientas y su interrelación conceptual en la arquitectura de la aplicación web. Fuente: Elaboración propia. ..... 35
Figura 6: Diagrama de componentes de alto nivel de la aplicación web. Fuente: Elaboración propia. ..... 46
Figura 7: Diagrama de flujo de datos simplificado para operaciones clave: (a) Autentificación de usuario, (b) Registro de una nueva sesión con captura de datos de codigo_raton.py, y (c) Visualización y análisis de una sesión existente. Fuente: Elaboración propia. ..... 48
Figura 8: Diagrama Entidad-Relación (DER) de la base de datos implementada en Supabase. Muestra las tablas Pacientes, Terapeutas, Terapias, Sesiones y DatosTrayectoria, sus columnas principales y las relaciones entre ellas. Fuente: Captura del esquema ..... 58
Figura 9: Pantalla de Inicio de Sesión/Registro Fuente: Aplicación TFG ..... 66
Figura 10: Vista principal con opciones Fuente: Aplicación TFG..... 67
Figura 11: Vista de Gestión de la Tabla Pacientes Fuente: Aplicación TFG..... 67
Figura 12: Formulario de Adición de Nueva Sesión Manual Fuente: Aplicación TFG . 68
Figura 13: Vista de Historial de Sesiones Fuente: Aplicación TFG ..... 68
Figura 14:Vista de detalles y Análisis de una Sesión Seleccionada Fuente: Aplicación TFG ..... 69
Figura 15: Captura de Sesión con Ratón Fuente: Aplicación TFG..... 70
Figura 16: Vista de Análisis de Progreso del Paciente Fuente: Aplicación TFG ..... 71
Figura 17: Captura de pantalla de la interfaz de Streamlit mostrando las métricas cinemáticas (Duración, Longitud, Velocidad Media) calculadas para una sesión de prueba tras la recuperación y procesamiento de sus datos de trayectoria. Fuente: Aplicación TFG..... 91
Figura 18:Ejemplo de visualización en la aplicación: Gráfico estático de una trayectoria recuperada de la base de datos. Fuente: Aplicación TFG..... 92
Figura 19:Ejemplo del gráfico de evolución de la "Velociad Media" a lo largo un par de sesiones para un paciente de prueba, generado en la vista de "Análisis de Progreso del Paciente". Fuente: Aplicación TFG. ..... 92

Figura A 1: Interfaz de inicio de sesión y registro Fuente: Aplicación TFG ..... 111
Figura A 2: Panel principal de la aplicación. Fuente: Aplicación TFG..... 112
Figura A 3:Vista de gestión de la tabla Pacientes. Fuente: Aplicación TFG..... 113
Figura A 4: Formulario para agregar un nuevo paciente. Fuente: Aplicación TFG ... 114

Figura A 5: Vista del historial de sesiones. Fuente: Aplicación TFG..... 114
Se expandirá automáticamente una sección debajo de la tabla con la información detallada, análisis y acciones para esa sesión. Figura A 6:Detalles y análisis de la sesión seleccionada. Fuente: Aplicación TFG ..... 115
Figura A 7: Formulario de configuración para una nueva sesión con ratón. Fuente: Aplicación TFG ..... 117
Figura A 8: Interfaz del script de captura de movimiento codigo_raton.py Fuente: Aplicación TFG..... 118
Figura A 9: Formulario para agregar una nueva sesión manualmente. Fuente: Aplicación TFG..... 119
Figura A 10: Vista de análisis de progreso longitudinal para un paciente seleccionado. Fuente: Aplicación TFG ..... 121

# CAPÍTULO 1. INTRODUCCIÓN Y OBJETIVOS

## 1.1 DESCRIPCIÓN Y CONTEXTO DEL PROYECTO

El presente Trabajo de Fin de Grado, TFG, se centra en desarrollar una aplicación web para la gestión y análisis de datos de un sistema de rehabilitación con realidad aumenta, dirigida a terapeutas y médicos en entornos hospitalarios. Este TFG surge como la evolución natural del sistema desarrollado por Matilla Plaza (2023), *Desarrollo de un juego serio de realidad aumentada para rehabilitación neuromotora*. Mientras el trabajo previo se centró en el desarrollo del juego terapéutico y la captura inicial de datos de sesión (guardados localmente en archivos CSV), este proyecto aborda el desafío crítico subsecuente: la necesidad de una plataforma centralizada para almacenar, visualizar y analizar cuantitativamente dichos datos, permitiendo un seguimiento preciso y personalizado del progreso del paciente.

En el contexto actual de la salud digital, los sistemas de rehabilitación asistida por tecnología están transformando los tratamientos tradicionales. La realidad aumentada, AR, ha demostrado ser particularmente efectiva en terapias de recuperación motora, ofreciendo entornos interactivos que mejoran la adherencia al tratamiento y facilitan la cuantificación objetiva del progreso. Si bien el trabajo de Matilla Plaza (2023) avanzó en el desarrollo de videojuegos terapéuticos con esta tecnología, el presente TFG se enfoca en el desafío operativo identificado tras su implementación: la ausencia de herramientas integradas que faciliten el análisis cuantitativo y el seguimiento longitudinal del progreso de los pacientes a partir de los datos generados.

Esta solución integra:

- Un módulo de captura de datos de movimiento mediante dispositivos de entrada estándar y sistemas RA.
- Un sistema centralizado de almacenamiento en la nube (Supabase), diseñado para recibir y organizar los datos de las sesiones de rehabilitación.
- Un mecanismo (idealmente automatizado) para la ingesta de datos de movimiento, superando la gestión manual de archivos CSV generados por sistemas como M3Display.
- Herramientas de visualización y análisis de datos clínicos.
- Mecanismos de generación de informes para profesionales sanitarios.

El proyecto se enmarca dentro de las tendencias actuales de eHealth y telemedicina, proporcionando una plataforma accesible desde cualquier dispositivo con navegador web, que sirve como puente entre pacientes, terapeutas y los datos generados durante las sesiones de rehabilitación.

Es importante destacar que el sistema implementará medidas de protección de datos conforme a regulaciones HIPAA/GDPR, incluyendo cifrado de información y controles de acceso

## 1.2 MOTIVACIÓN

La motivación principal para el desarrollo de este proyecto surge de tres factores clave:

1. **Necesidad clínica**: Los procesos tradicionales de rehabilitación física suelen depender de evaluaciones subjetivas y registros manuales, lo que dificulta el seguimiento preciso del progreso del paciente. Existe una demanda creciente de sistemas objetivos que capturen métricas cuantificables.
2. **Oportunidad tecnológica:** La popularización de tecnologías como la realidad aumentada y las plataformas en la nube ofrecen nuevas posibilidades para crear soluciones accesibles que no requieran hardware especializado costoso.
3. **Impacto social:** Las enfermedades neurodegenerativas y las secuelas de accidentes cerebrovasculares afectan a una población cada vez mayor, requiriendo soluciones escalables que puedan atender a más pacientes sin sacrificar la calidad del tratamiento.

Personalmente, este proyecto combina mi interés por el desarrollo de software aplicado al ámbito sanitario con el desafío técnico de integrar diversas tecnologías emergentes en una solución cohesiva y centrada en el usuario final.

## 1.3 OBJETIVOS

### Objetivo principal

El principal objetivo de este TFG es desarrollar una aplicación web integral basada en Streamlit y Supabase para la gestión y análisis de datos clínicos en sistemas de rehabilitación con realidad aumentada, garantizando el cumplimiento de normativas de protección de datos (HIPAA/GDPR). El sistema debe permitir:

- El almacenamiento seguro y estructurado de datos de sesiones terapéuticas (incluyendo información del paciente, parámetros de sesión y datos de trayectoria) en la base de datos Supabase.
- El registro seguro de sesiones terapéuticas con captura automatizada de métricas de movimiento.
- La visualización interactiva de datos para evaluar el progreso de los pacientes.
- La generación de visualizaciones específicas (ej. GIFs animados de trayectorias) para facilitar la interpretación por parte del terapeuta.

Además, se validará su eficacia mediante su implementación real en un entorno controlado, demostrando su aplicabilidad en contextos hospitalarios.

### Objetivos secundarios

Como objetivos secundarios se incluyen:

- Diseñar un modelo de base de datos en Supabase optimizado para almacenar datos heterogéneos como trayectorias o imágenes.
- Definir y, si es factible dentro del alcance, implementar la interfaz o mecanismo (API/función) para permitir que los datos de sesión generados por sistemas externos puedan ser registrados directamente en la base de datos Supabase, automatizando el proceso de carga de datos.
- Implementar algoritmos de procesamiento para análisis cuantitativo de movimiento. Por ejemplo, filtrado de trayectorias.
- Crear interfaces intuitivas adaptadas a los distintos perfiles de usuarios.
- Garantizar el cumplimiento de normativas de protección de datos médicos

### Objetivos académicos

Además de los objetivos previamente descritos, este trabajo también persigue metas formativas:

- Comprender los desafíos clínicos en la rehabilitación motora y el papel de las tecnologías asistidas.
- Profundizar en arquitecturas cloud para la gestión de datos sensibles.
- Adquirir habilidades en tecnologías emergentes, como puede ser dominar el conjunto de tecnologías seleccionado (Streamlit, Supabase, Python).
- Fomentar competencias de investigación científica como puede ser documentar procesos técnicos con rigor académico.

## 1.4 ALCANCE

El presente TFG tiene como alcance:

### Desarrollo teórico-práctico:

- Se implementará un sistema web completo (frontend Streamlit, backend Python/Supabase) para gestión de terapias de rehabilitación motora mediante realidad aumentada.
- La plataforma integrará módulos específicos para captura, almacenamiento y análisis cuantitativo de datos clínicos.

- El desarrollo del frontend con Streamlit permitirá una interacción accesible para profesionales sanitarios.

Modelado y análisis:

- La arquitectura backend con Python y Supabase servirá como base para el procesamiento seguro de información médica.
- Los módulos de terapia permitirán estudiar diferentes patrones de movimiento.
- Las visualizaciones interactivas y animaciones GIF facilitarán el análisis objetivo del progreso de los pacientes.

Validación práctica:

- La funcionalidad de la plataforma web (gestión, análisis, visualización) se validará utilizando conjuntos de datos realistas, que pueden ser:
    - Datos CSV previamente generados por el sistema M3Display (Matilla Plaza, 2023).
    - Datos numéricos de trayectoria generados mediante un script prototipo desarrollado en Python, utilizando su biblioteca gráfica estándar (Tkinter), para simular la captura de movimiento con un dispositivo de entrada convencional (ratón).Datos simulados con características similares.
- El sistema demostrará su aplicabilidad en entornos clínicos controlados.

Limitaciones deliberadas:

- No se incluye el desarrollo ni la modificación del sistema M3Display en sí mismo. El foco está en la plataforma de gestión y análisis *posterior*.
- La implementación práctica de la conexión directa y automática entre M3Display y la base de datos Supabase podría quedar fuera del alcance de la implementación final de este TFG (limitándose a su diseño y definición de interfaz), dependiendo de la complejidad técnica y el tiempo disponible. La validación se realizaría entonces mediante la carga manual o programática de los datos preexistentes (CSV) o simulados.
- La plataforma no realiza diagnóstico médico automatizado, funcionando como herramienta de apoyo a la decisión clínica.
- El alcance se centra en el análisis de movimiento, excluyendo módulos administrativos como facturación o gestión de agendas.

Este enfoque garantiza un proyecto viable técnicamente, con resultados directamente aplicables en contextos reales de rehabilitación neuromotora.

## 1.5 ESTRUCTURA DEL TRABAJO

Este trabajo está estructurado en ocho capítulos, acompañado por un anexo que se recoge al final:

**Capítulo 1. Introducción y objetivos:** Se introduce el contexto del proyecto, centrado en la gestión y análisis de datos de rehabilitación neuromotora asistida por tecnología, específicamente Realidad Aumentada, justificando su necesidad clínica y tecnológica como evolución de trabajos previos. Se detallan los objetivos principales, secundarios y académicos perseguidos, se define el alcance del sistema web a desarrollar, plataforma Streamlit con Supabase, y se presenta la estructura general del documento.

**Capítulo 2. Estado del arte:** Se revisa la literatura relevante sobre la rehabilitación neuromotora y el papel de las tecnologías emergentes. Se analizan trabajos previos sobre el uso de la Realidad Aumentada en terapias, así como soluciones existentes para la gestión y análisis de datos clínicos mediante plataformas web. Finalmente, se describen brevemente las tecnologías clave seleccionadas para el proyecto (Streamlit, Supabase, Python para análisis) y se identifica el nicho específico que este trabajo busca cubrir en la monitorización cuantitativa del progreso del paciente.

**Capítulo 3. Metodología:** Se describe la metodología de desarrollo de software empleada para la gestión y ejecución del proyecto. Se detallan las fases seguidas, desde el análisis inicial hasta la validación final, y se enumeran las herramientas y tecnologías específicas utilizadas en cada etapa del proceso de diseño, implementación y pruebas de la aplicación web.

**Capítulo 4 Análisis de Requisitos:** Se especifican los requisitos que debe cumplir la aplicación web para la gestión y análisis de datos de rehabilitación. Se detallan los requisitos funcionales, describiendo las operaciones que el sistema permitirá realizar a los usuarios (principalmente terapeutas), y los requisitos no funcionales, relacionados con aspectos cruciales como la usabilidad de la interfaz, el rendimiento en la visualización de datos, la seguridad en el manejo de información sensible y el cumplimiento normativo (HIPAA/GDPR).

**Capítulo 5. Diseño del sistema:** Se presenta el diseño arquitectónico de la aplicación web, detallando la interacción entre el frontend desarrollado con Streamlit, la lógica de la aplicación implementada en Python y la base de datos en la nube Supabase. Se incluye el diseño detallado del modelo de datos para almacenar información de pacientes y sesiones, el diseño de las interfaces de usuario clave y la descripción de los módulos principales del sistema, como los de visualización de trayectorias, cálculo de métricas de rendimiento y generación de informes.

**Capítulo 6. Implementación:** Se detalla el proceso de codificación y construcción de la aplicación web utilizando Streamlit y Python, así como la configuración de la base de datos en Supabase. Se explican las decisiones de implementación más relevantes, se muestran fragmentos de código ilustrativos de funcionalidades clave (conexión a Supabase, procesamiento de datos de movimiento, generación de gráficos interactivos) y se comentan los desafíos técnicos encontrados y sus soluciones. Se incluye una descripción del script

auxiliar (Tkinter) utilizado para generar datos de prueba simulando sesiones de terapia.

**Capítulo 7. Pruebas y validación:** Se describen las pruebas realizadas para verificar el correcto funcionamiento, la usabilidad y la fiabilidad de la aplicación desarrollada. Se presentan los resultados obtenidos durante la fase de validación, utilizando datos generados para simular escenarios de uso real en un entorno controlado. Se evalúa el cumplimiento de los requisitos funcionales y no funcionales, y se discute la capacidad del sistema para el análisis efectivo de los datos de rehabilitación.

**Capítulo 8. Conclusiones y posibles mejoras:** Se exponen las conclusiones principales extraídas del desarrollo del proyecto y la validación del sistema web. Se recapitulan los logros alcanzados en relación con los objetivos propuestos para la gestión y análisis de datos. Se discuten las limitaciones del trabajo realizado (como el uso de datos simulados o la no integración directa con un sistema AR) y se proponen posibles líneas de mejora y trabajo futuro, como la integración con hardware AR real, la ampliación de las métricas de análisis o la realización de estudios piloto con usuarios clínicos.

**Anexo 1. Manual de usuario:** Se incluye una guía detallada para el usuario final (personal clínico/terapeuta) sobre cómo instalar (si aplica) y utilizar las distintas funcionalidades de la aplicación web desarrollada con Streamlit, explicando los pasos para la gestión de pacientes, el registro/visualización de sesiones y la interpretación de los análisis presentados, acompañada de capturas de pantalla ilustrativas.

# CAPÍTULO 2. ESTADO DEL ARTE

## 2.1 CONTEXTO: REHABILITACIÓN NEUROMOTORA Y SUS DESAFIÍOS

La rehabilitación neuromotora juega un papel crucial en la recuperación de la funcionalidad tras lesiones neurológicas como el Daño Cerebral Adquirido (DCA) o los accidentes cerebrovasculares (Astudillo Aguiar, 2023). Las secuelas resultantes, incluyendo la paresia (debilidad muscular) y déficits de coordinación, impactan profundamente la autonomía y calidad de vida de los pacientes (BVS, 1963). Los enfoques terapéuticos tradicionales, que abarcan desde movilizaciones pasivas/activas hasta métodos específicos como Rood o Kabat (FisioOnline, 2024; Matilla Plaza, 2023), constituyen la base de la práctica clínica. Sin embargo, estos métodos convencionales enfrentan desafíos significativos que limitan su efectividad óptima.
La monotonía inherente a la repetición de ejercicios a menudo conduce a una disminución de la motivación y, consecuentemente, a una pobre adherencia al tratamiento por parte del paciente, factores críticos para el éxito de la rehabilitación (Matilla Plaza, 2023). Además, la evaluación del progreso frecuentemente se basa en valoraciones subjetivas del terapeuta o en mediciones manuales que pueden carecer de la precisión, resolución y objetividad necesarias para un seguimiento fino de la evolución y para la personalización informada de la terapia (Dobkin, 2023). Esta confluencia de factores subraya la necesidad apremiante de herramientas innovadoras que ofrezcan terapias más atractivas y permitan una cuantificación rigurosa y objetiva del progreso motor.

## 2.2 EL PAPEL DE LA TECNOLOGÍA EN LA REHABILITACIÓN MODERNA

La tecnología se ha consolidado como un agente transformador en la medicina moderna, y su impacto en el campo de la rehabilitación es particularmente notable (APD, 2023). La integración de herramientas digitales ha experimentado un crecimiento exponencial en los últimos años, como reflejan las inversiones en I+D en tecnología médica (EvaluateMedTech, 2018; EvaluateMedTech, 2024). Este auge tecnológico no solo complementa las prácticas existentes, sino que está redefiniendo los paradigmas de tratamiento, abriendo nuevas vías para mejorar la eficacia, la accesibilidad, la personalización, la motivación y la cuantificación objetiva de las intervenciones terapéuticas (Toro, 2023; Fisioform, 2023).

## 2.3 PANORAMA DE TECNOLOGÍAS EMERGENTES APLICADAS A LA REHABILITACIÓN NEUROLÓGICA

Un abanico diverso de tecnologías está contribuyendo a la evolución de la rehabilitación neurológica:

- **Realidad Virtual (RV) y Realidad Aumentada (RA):** La RV sumerge al usuario en entornos completamente virtuales, mientras que la RA superpone elementos digitales sobre el mundo real (Azuma, 1997; Ordoñez, 2020). Ambas permiten crear escenarios terapéuticos controlados, seguros y altamente motivadores (Keshner & Weiss, 2018; ClinicaUner, 2023).
- **Juegos Serios (Serious Games):** Aplican principios de diseño de videojuegos (metas, reglas, feedback, recompensas) a objetivos no lúdicos, como la rehabilitación. Han demostrado ser herramientas eficaces para incrementar el compromiso y la adherencia del paciente (Lynn E. Fiellin, 2014; Matilla Plaza, 2023).
- **Robótica y Exoesqueletos:** Dispositivos mecánicos que asisten o resisten el movimiento, facilitando la realización de terapias intensivas, repetitivas y funcionalmente relevantes, además de proporcionar soporte estructural (Díez, 2020).
- **Telefisioterapia y Tele-rehabilitación:** Utilizan tecnologías de comunicación (videollamadas, plataformas web) para prestar servicios de rehabilitación a distancia, superando barreras geográficas y de movilidad (Pietrusinski et al., 2020).
- **Biofeedback y Sensores Vestibles (Wearables):** El biofeedback proporciona información en tiempo real sobre funciones fisiológicas para aprender a controlarlas. Los wearables (pulseras, relojes inteligentes, etc.) permiten la monitorización continua de parámetros de actividad y movimiento en el entorno del paciente (Dobkin, 2023).
- **Inteligencia Artificial (IA):** Se aplica para analizar grandes conjuntos de datos clínicos y de movimiento, identificar patrones, personalizar tratamientos, asistir en el diagnóstico y mejorar las predicciones de resultados (Matilla Plaza, 2023).

Estas innovaciones, a menudo combinadas entre sí, están impulsando la transición hacia una rehabilitación más precisa, personalizada, cuantificable y centrada en el paciente.

## 2.4 REALIDAD AUMENTADA (RA) EN REHABILIATCIÓN: CAPACIDADES Y APLICACIONES

La RA, al superponer información digital sobre el entorno físico sin aislar al usuario, presenta características idóneas para la rehabilitación motora (Azuma, 1997; Ordoñez, 2020).

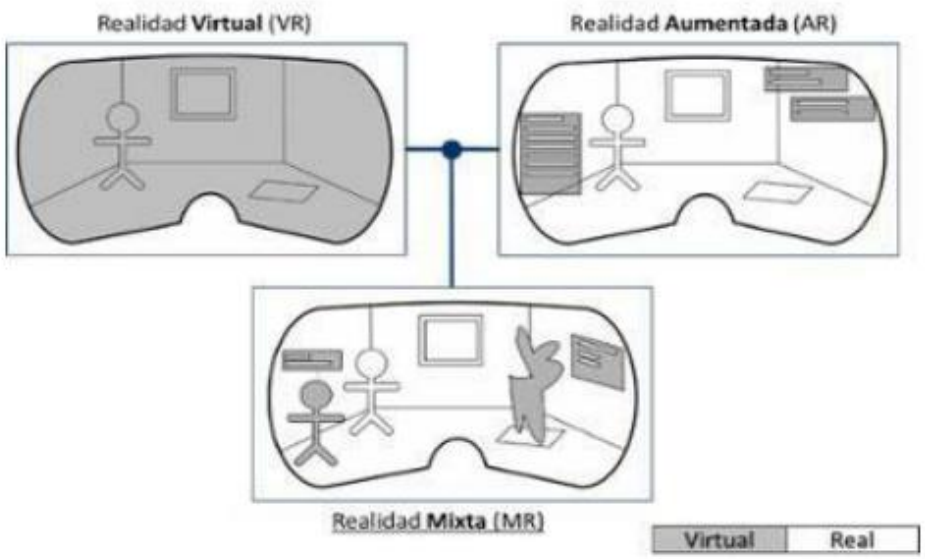


*Figura 1: Diferencias entre realidad virtual, realidad aumentada y realidad mixta. Fuente: (Ordoñez, 2020).*

Sus beneficios específicos incluyen:

- **Motivación y Compromiso:** La gamificación de ejercicios repetitivos aumenta significativamente la adherencia (Chen et al., 2020).
- **Feedback Multimodal Inmediato:** Permite retroalimentación visual y/o auditiva instantánea sobre la calidad del movimiento, crucial para el aprendizaje motor (Khademi et al., 2012).
- **Personalización Dinámica:** Facilita la adaptación de la dificultad y los parámetros de las tareas al estado y progreso del paciente (Chen et al., 2020).
- **Cuantificación Objetiva del Movimiento:** Sistemas de seguimiento integrados (tracking) permiten registrar datos precisos (trayectorias 3D, velocidad, precisión, suavidad) que sirven como biomarcadores digitales para monitorizar la recuperación (Dobkin, 2023; Moreno et al., 2022). Estudios comparativos han mostrado la superioridad de la RA frente a interfaces 2D convencionales en términos de rendimiento motor en tareas de rehabilitación (Hossein Mousavi Hondori et al., 2016).

## 2.5 CASO DE ESTUDIO PREVIO: SISTEMA M3DISPLAY Y JUEGO "COINS"

Un sistema que materializa la aplicación de RA en rehabilitación es M3Display, desarrollado en ITAP de la Universidad de Valladolid (Matilla Plaza, 2023; Cisnal et al., 2023). Se trata de un sistema de RA no inmersiva que utiliza una configuración de monitor, cámara y espejo para proyectar elementos virtuales sobre la imagen reflejada del paciente y sus miembros superiores.

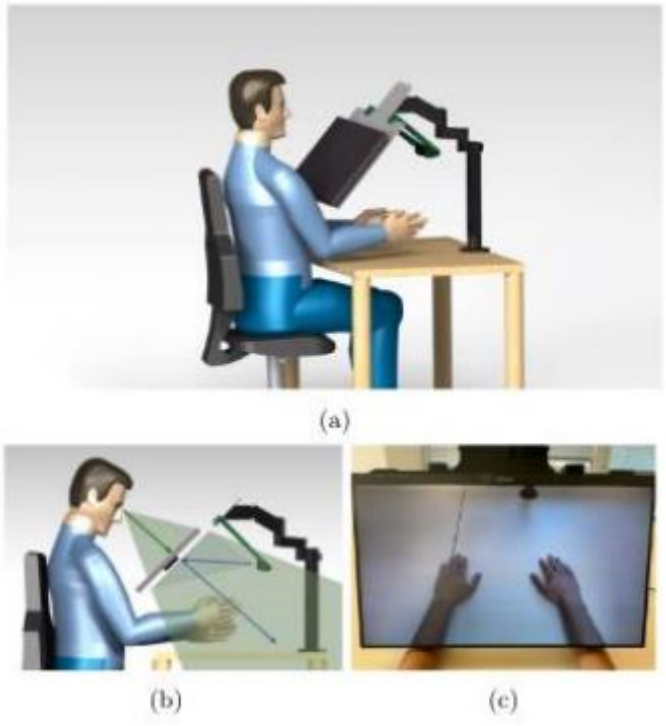


*Figura 2: Configuración hardware del sistema de RA no inmersiva M3Display. (a) Estructura general. (b) Esquema de reflexión en el espejo. (c) Vista resultante para el paciente. Fuente: Matilla Plaza (2023), Figura 8 (basada en Cisnal, 2023).*

Para la interacción, M3Display se basa en el seguimiento de manos mediante MediaPipe Hands (Team M., 2021; Fan Zhang et al., 2020), una tecnología de visión por computador capaz de detectar en tiempo real los puntos clave *(landmarks)* de las manos.

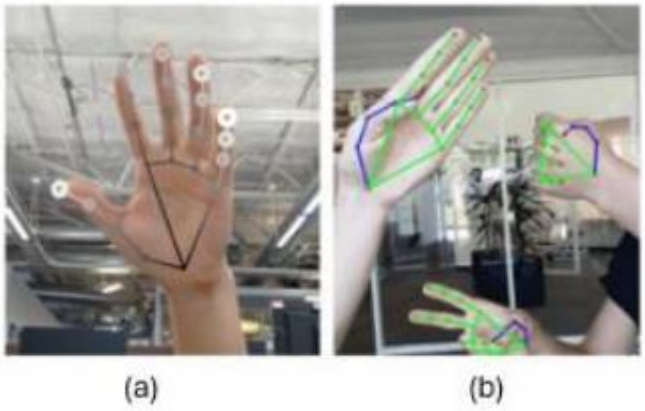


*Figura 3: Detección y seguimiento de landmarks de la mano mediante MediaPipe Hands, tecnología integrada en M3Display. Fuente: (Fan Zhang, 2020).*

El Trabajo de Fin de Grado previo (Matilla Plaza, 2023) utilizó esta plataforma para desarrollar el juego serio "Coins", enfocado en ejercicios de seguimiento de trayectoria para el miembro superior. Una funcionalidad clave implementada fue la generación de archivos de datos en formato CSV al finalizar cada sesión, conteniendo información detallada como la posición de la mano (coordenadas), el tiempo transcurrido y la puntuación obtenida instante a instante (Matilla Plaza, 2023, Sección 6.6 y Figura 52). Si bien la generación de estos datos representa un paso fundamental hacia la cuantificación, su almacenamiento en archivos locales y dispersos constituye una limitación significativa para su uso clínico sistemático y el análisis longitudinal.

## 2.6 GESTIÓN DE DATOS CLÍNICOS EN REHABILITACIÓN: PLATAFORMAS Y REQUISITOS

La explosión de datos cuantitativos generados por las nuevas tecnologías de rehabilitación (RA, wearables, etc.) crea una necesidad imperativa de sistemas eficientes para su gestión, almacenamiento y análisis, transformando datos brutos en información clínicamente relevante (Shortliffe & Cimino, 2013). El desarrollo de plataformas web para este fin debe considerar múltiples requisitos críticos:

- **Almacenamiento Centralizado, Seguro y Escalable:** Se necesitan bases de datos capaces de albergar de forma segura grandes volúmenes de datos heterogéneos (información de pacientes, detalles de sesiones, series temporales de datos de movimiento). Las soluciones Cloud BaaS (Backend-as-a-Service), como Supabase (basada en PostgreSQL), ofrecen infraestructuras gestionadas que simplifican el desarrollo, pero

requieren una configuración experta para garantizar la seguridad (Supabase Documentation, 2023; Fernández, 2022).

- **Seguridad y Cumplimiento Normativo:** El manejo de datos de salud está sujeto a estrictas regulaciones (GDPR en la UE, HIPAA en EE.UU.). Las plataformas deben implementar cifrado de datos (en reposo y en tránsito), mecanismos robustos de autenticación y autorización, y controles de acceso granulares (como Row Level Security - RLS) para proteger la confidencialidad e integridad de la información del paciente (Reglamento GDPR; HIPAA Journal, 2023; AEPD, 2023). El cumplimiento de normativas específicas para dispositivos médicos (como EN ISO 13485:2021) podría ser necesario dependiendo de la clasificación del sistema completo.
- **Interoperabilidad:** Para una integración efectiva en el ecosistema sanitario, es deseable que la plataforma pueda intercambiar datos con otros sistemas (ej., Historia Clínica Electrónica) utilizando estándares como HL7 FHIR o modelos de datos clínicos como openEHR (HL7 FHIR Foundation, 2022; OpenEHR Foundation, 2023).
- **Usabilidad:** Las interfaces de usuario deben ser intuitivas y eficientes para el personal clínico, que puede no tener una formación técnica avanzada, permitiendo un acceso rápido y claro a la información relevante para la toma de decisiones (García et al., 2021).


*Figura 4: Ejemplo conceptual de un dashboard clínico basado en web para la monitorización de la rehabilitación. Muestra la integración de información del paciente, un gráfico de progreso longitudinal y paneles para visualizar métricas clave del movimiento Fuente: Elaboración propia.*

## 2.7 HERRAMIENTAS TECNOLÓGICAS PARA LA IMPLEMENTACIÓN DE PLATAFORMAS DE ANÁLISIS

La construcción de una plataforma web para la gestión y análisis de datos de rehabilitación requiere la selección de un conjunto de tecnologías adecuadas. Las tecnologías elegidas para este TFG buscan un equilibrio entre potencia, flexibilidad y rapidez de desarrollo:

- **Supabase (Backend y Base de Datos):** Seleccionado como plataforma BaaS por ser Open Source, utilizar PostgreSQL (una base de datos relacional robusta y estándar), y ofrecer servicios integrados de autenticación, almacenamiento y APIs en tiempo real, además de características de seguridad como RLS. Simplifica significativamente la gestión de la infraestructura backend (Supabase Documentation, 2023; Fernández, 2022).
- **Streamlit (Frontend / Interfaz Web):** Elegido por ser un framework Python que permite crear aplicaciones web interactivas centradas en datos con un esfuerzo de desarrollo mínimo comparado con frameworks web tradicionales. Es ideal para crear rápidamente dashboards y herramientas de visualización para usuarios no técnicos como los terapeutas (García et al., 2021; Pandey, 2023). Su integración nativa con el ecosistema Python de análisis y visualización es una ventaja clave. La gestión segura de la autenticación es un aspecto a considerar (Wade, 2022).
- **Python y su Ecosistema Científico (Análisis y Visualización):** Python es el lenguaje de facto para el análisis de datos. Se utilizarán librerías fundamentales como:
    - **Pandas:** Para la manipulación eficiente de datos tabulares (lectura de CSV, interacción con la base de datos, limpieza y transformación de datos) (McKinney, 2017).
    - **NumPy:** Para operaciones numéricas y manejo de arrays, base para muchos algoritmos de análisis.
    - **SciPy:** Para implementar algoritmos de procesamiento de señales (ej. filtrado) y análisis estadístico.
    - **Matplotlib:** Para la generación de gráficos estáticos de alta calidad y visualizaciones animadas (como los GIFs de trayectorias) (Hunter, 2007; Unwin, 2023).
    - **Plotly:** Para crear gráficos interactivos (zoom, pan, tooltips) que pueden ser embebidos directamente en la aplicación Streamlit, mejorando la exploración de los datos por parte del usuario (Plotly Medical Team, 2022).

Esta combinación tecnológica permite construir una solución completa, desde la base de datos hasta la interfaz de usuario final, aprovechando herramientas modernas y ampliamente documentadas.

## 2.8 HERRAMIENTAS PARA LA GESTIÓN DE DATOS CLÍNICOS

El verdadero valor de recoger datos detallados de movimiento reside en la capacidad de analizarlos cuantitativamente para extraer información clínicamente significativa. Más allá de la simple visualización, la plataforma propuesta debe integrar técnicas de análisis como:

- **Preprocesamiento y Filtrado:** Los datos de seguimiento (especialmente de visión por computador) suelen contener ruido. Es necesario aplicar técnicas de filtrado (ej. filtros de mediana móvil, filtros de Butterworth, o filtros más avanzados como Kalman) para suavizar las trayectorias y obtener una representación más fiable del movimiento real (Ramírez et al., 2024).
- **Cálculo de Métricas Cinemáticas:** Extracción de parámetros objetivos que describen la calidad del movimiento. Ejemplos relevantes incluyen:
    - **Precisión:** Error respecto a una trayectoria objetivo (ej. desviación media o máxima).
    - **Velocidad:** Perfiles de velocidad instantánea, velocidad media, número de picos de velocidad (relacionado con la fluidez).
    - **Tiempo de Ejecución:** Duración total de la tarea o de segmentos específicos.
- **Optimización del Procesamiento:** Considerar la eficiencia computacional de los algoritmos, especialmente pensando en futuras aplicaciones que puedan requerir análisis más cercano al tiempo real (Nguyen et al., 2023).

La integración de estos análisis en la plataforma web permitirá al terapeuta ir más allá de la observación visual y obtener una comprensión cuantitativa y objetiva del rendimiento y progreso del paciente.

## 2.9 REVISIÓN DE SISTEMAS SIMILARES EXISTENTES

Si bien la combinación específica de análisis de datos de RA en una plataforma Streamlit/Supabase es novedosa, existen diversos esfuerzos en la literatura y desarrollos comerciales que abordan aspectos relacionados con la rehabilitación tecnológica y la gestión de datos:

- **Plataformas de Tele-rehabilitación con Monitorización:** Sistemas como RehabilitaWeb (nombre genérico representativo, ver Pietrusinski et al., 2020 para arquitecturas cloud) a menudo se centran en la gestión

de sesiones remotas, videoconsultas y asignación de ejercicios. Pueden incluir monitorización mediante sensores wearables o cámaras web, pero raramente ofrecen análisis cinemático detallado de datos provenientes de sistemas de RA complejos como M3Display. Su enfoque suele ser más la logística y el seguimiento general del cumplimiento que el análisis biomecánico fino.

- **Sistemas Basados en Sensores Inerciales (IMUs) y Web:** Plataformas como SwiM (System for Wearable Interactive Motion rehabilitation) (Zhou et al., 2018) utilizan sensores IMU en las extremidades y una plataforma web para visualizar el rendimiento y el rango de movimiento. Si bien ofrecen cuantificación y acceso web, la naturaleza de los datos (orientación, aceleración) y el tipo de análisis difieren de los datos posicionales 3D detallados que proporciona un sistema como M3Display basado en visión por computador.
- **Software Propietario de Sistemas de RV/Robótica:** Empresas como Tyromotion con su software TyroS o Motek con Cuevas Medek Exercise (CME) y su software D-Flow, ofrecen soluciones integradas de hardware (robots, plataformas de RV) y software para análisis. Son sistemas potentes y validados clínicamente, pero representan una inversión considerable, son ecosistemas cerrados y no están diseñados para integrar o analizar datos de sistemas externos como M3Display.
- **Proyectos de Investigación con Web Dashboards:** Artículos como el de Lee et al. (2019) describen sistemas web para seguimiento motor, a menudo utilizando tecnologías web más tradicionales (ej., MEAN/MERN stack, Django) y enfocándose en visualizaciones específicas o conjuntos de datos particulares. Plataformas como NeuroRehabLab Online Platform (Faria et al., 2018) buscan integrar datos de diferentes fuentes (Kinect, Leap Motion), pero su enfoque puede ser más amplio o la tecnología subyacente (ej., almacenamiento, framework web) diferente a la propuesta aquí.
- **Herramientas de Código Abierto para Análisis Biomecánico:** Existen librerías y software open-source como OpenSim o Mocap Analyser, pero son herramientas de análisis offline, potentes pero que requieren un alto grado de expertise técnico y no constituyen una plataforma de gestión clínica integrada y basada en web.

**Brecha Identificada:** A pesar de estos avances, persiste una brecha en la disponibilidad de plataformas web accesibles, flexibles y específicamente orientadas a terapeutas para la gestión centralizada y el análisis cinemático detallado de datos de trayectoria 3D provenientes de sistemas de rehabilitación con Realidad Aumentada basados en visión por computador (como M3Display). Las soluciones existentes tienden a ser o bien sistemas cerrados y costosos, o plataformas de tele-rehabilitación con análisis limitados, o herramientas de análisis offline que requieren conocimientos técnicos avanzados, o prototipos de investigación con enfoques tecnológicos diferentes.

Falta una solución que democratice el análisis avanzado de estos datos específicos en un entorno clínico o de investigación mediante un conjunto de tecnologías modernas y de rápido desarrollo como Streamlit/Supabase.

## 2.10 ANALISIS COMPARATIVO Y JUSTIFICACIÓN DE LA SOLUCIÓN PROPUESTA

El análisis del estado del arte y la revisión del trabajo previo de Matilla Plaza (2023) confirman la **necesidad crítica** de una herramienta que supere las limitaciones del almacenamiento de datos de rehabilitación en archivos CSV locales. Estas limitaciones (dificultad en seguimiento longitudinal, análisis manual complejo, falta de visión integrada, problemas de escalabilidad) impiden explotar todo el potencial clínico de los datos generados por sistemas avanzados como M3Display.

Este TFG propone abordar directamente esta brecha mediante el desarrollo de una aplicación web específica (Frontend: Streamlit; Backend/DB: Supabase; Lógica: Python), diseñada explícitamente como una plataforma de gestión, post-procesamiento y análisis cuantitativo para datos de rehabilitación neuromotora con RA.

**Análisis Comparativo y Ventajas Clave:**

La tabla siguiente compara la solución propuesta con el estado anterior (CSVs) y las limitaciones generales de los sistemas revisados:

| Característica | Estado Anterior (CSVs M3Display) | Limitaciones Sistemas Revisados (General) | Solución Propuesta (Streamlit/Supabase) | Ventaja / Novedad |
|---|---|---|---|---|
| **Almacenamiento** | Local, disperso, inseguro | Cerrado/Propietario, Costoso, No específico para RA | Cloud Centralizado, Seguro, Escalable (Supabase) | Gestión Unificada, Acceso Seguro, Flexible |
| **Ingesta Datos** | Manual / Script local | A menudo ligada a hardware específico | Diseño para Ingesta Directa/Automatizada | Eficiencia, Potencial Tiempo Real |
| **Análisis** | Externo, Básico / Manual | Limitado, Genérico, Requiere Expertise Offline | Integrado, Cuantitativo Avanzado (Python/SciPy) | Métricas Objetivas Accesibles, Estandarizado |

| | | | | |
|---|---|---|---|---|
| **Visualización** | Limitada / Externa | Rígida, No interactiva, No específica para trayectoria | Interactiva Web (Streamlit, Plotly, GIFs) | Fácil Interpretación, Seguimiento Visual |
| **Accesibilidad** | Restringida al PC | Requiere software/hardware específico, Costoso | Vía Navegador Web | Ubicuidad, Usabilidad para Terapeutas |
| **Seguimiento** | Complejo, Manual | A menudo limitado o complejo | Longitudinal Automático, Comparativas | Visión Clara de Evolución |
| **Tecnología** | - | Propietaria, Tradicional Web, Compleja | Moderna, Open Source (mayormente), Rápida Dev. | Flexibilidad, Adaptabilidad, Coste Potencial |

**Justificación y Novedad:**

La **justificación** de este proyecto radica en la necesidad clínica y tecnológica de cerrar la brecha entre la *generación* de datos ricos en rehabilitación con RA y su *aplicación* efectiva. La novedad reside en la combinación específica de factores:

1. **Conjunto de tecnologías moderno y accesible:** El uso de Streamlit y Supabase permite un desarrollo rápido y la creación de una herramienta potencialmente de bajo coste y fácil despliegue, democratizando el acceso a análisis avanzados.
2. **Orientación al Terapeuta:** La interfaz y las funcionalidades están pensadas para ser utilizadas por personal clínico, facilitando la interpretación y la toma de decisiones basadas en datos objetivos.
3. **Puente Digital:** Actúa como el eslabón intermedio esencial, transformando datos brutos en conocimiento clínico accionable y maximizando el retorno de la inversión en sistemas de rehabilitación con RA.

En esencia, este TFG no busca crear un nuevo sistema de RA, sino aportar la inteligencia analítica y la usabilidad clínica necesarias para que los datos generados por dichos sistemas tengan un impacto real en la práctica de la rehabilitación neuromotora.

# CAPÍTULO 3. METODOLOGÍA

Este capítulo detalla el enfoque metodológico adoptado para el desarrollo de la aplicación web de gestión y análisis de datos de rehabilitación con Realidad Aumentada. Se describen las fases del proyecto, las técnicas empleadas para la recolección y análisis de requisitos, las herramientas de desarrollo seleccionadas y los criterios que se utilizarán para la evaluación y validación del sistema implementado.

## 3.1 ENFOQUE METODOLÓGICO DEL PROYECTO

Dada la naturaleza evolutiva del proyecto, que parte de un sistema previo (M3Display y el juego "Coins" de Matilla Plaza, 2023) y busca desarrollar una nueva capa de gestión y análisis de datos, se ha optado por un enfoque de desarrollo incremental y prototipado rápido. Esta elección se fundamenta en varias consideraciones:

- **Claridad del Problema Principal, Flexibilidad en Detalles:** Si bien la necesidad de una plataforma de análisis para los datos de M3Display es clara, los detalles específicos de las métricas de análisis, las visualizaciones más útiles y la interfaz de usuario óptima pueden refinarse a medida que se avanza.
- **Tecnologías Nuevas para el Desarrollador:** El uso de Streamlit y Supabase, si bien potentes, puede implicar una curva de aprendizaje. Un enfoque incremental permite desarrollar y probar módulos de forma independiente.
- **Importancia del Feedback (Implícito):** Aunque no se contempla un ciclo formal con usuarios finales (terapeutas) durante el desarrollo de este TFG debido a sus limitaciones, el desarrollo se orienta a crear una herramienta que sea útil para ellos, y el prototipado permite visualizar y ajustar funcionalidades.
- **Viabilidad en el Contexto de un TFG:** Permite entregar funcionalidades concretas de forma progresiva y gestionar el alcance de manera más efectiva dentro de los plazos académicos.

Este enfoque se asemeja a una metodología ágil adaptada, donde se priorizan entregas funcionales de componentes clave (ej., almacenamiento de datos, luego visualización básica, luego análisis, etc.), permitiendo ajustes basados en los descubrimientos realizados durante el desarrollo. Se combina con un desarrollo orientado a prototipos especialmente para la interfaz de usuario con Streamlit, dada la facilidad de este *framework* para crear y modificar interfaces interactivas rápidamente.

## 3.2 FASES DEL PROYECTO

El desarrollo del proyecto se ha estructurado en las siguientes fases principales, que, aunque se presentan secuencialmente, pueden tener solapamientos y retroalimentación entre ellas debido al enfoque incremental:

1. **Fase 1: Investigación y Planificación (Completada)**
    - Revisión bibliográfica exhaustiva (Capítulo 2: Estado del Arte).
    - Análisis del sistema previo de Matilla Plaza (2023) para entender la naturaleza de los datos generados y las limitaciones existentes.
    - Definición de los objetivos principales y secundarios del TFG (Capítulo 1).
    - Establecimiento del alcance y las limitaciones del proyecto (Capítulo 1).
    - Selección del conjunto de tecnologías (Streamlit, Supabase, Python y librerías asociadas).
    - Planificación inicial de las funcionalidades y módulos de la aplicación.
2. **Fase 2: Análisis y Definición de Requisitos (Capítulo 4)**
    - Identificación de los requisitos funcionales (qué debe hacer el sistema) y no funcionales (cómo debe ser el sistema, ej., usabilidad, seguridad).
    - Definición de los perfiles de usuario (principalmente terapeutas/médicos).
    - Técnicas de recolección de requisitos (detalladas en la sección 3.3).
3. **Fase 3: Diseño del Sistema (Capítulo 5)**
    - Diseño de la arquitectura general de la aplicación web (frontend, backend, base de datos).
    - Modelado de la base de datos en Supabase, incluyendo la nueva tabla DatosTrayectoria y sus relaciones.
    - Diseño de la interfaz de usuario (mockups o wireframes conceptuales de las principales vistas en Streamlit).
    - Diseño de los algoritmos para el análisis cuantitativo de datos de movimiento.
    - Definición de la interfaz para la ingesta de datos de sistemas externos (considerando M3Display).

4. **Fase 4: Implementación (Capítulo 6)**
    - Configuración del entorno de desarrollo (Supabase, Python, Streamlit).
    - Desarrollo de la estructura de la base de datos en Supabase.
    - Implementación de la lógica de negocio en Python para la gestión de usuarios, pacientes y sesiones.
    - Modificación del script de captura (*codigo_raton.py*) para guardar datos numéricos de trayectoria.
    - Implementación de las funciones para la carga, almacenamiento, filtrado y recuperación de datos de trayectoria en Supabase.
    - Desarrollo de los algoritmos de análisis cuantitativo (cálculo de métricas).
    - Construcción de la interfaz de usuario en Streamlit, incluyendo formularios, tablas y visualizaciones (gráficos estáticos, GIFs, gráficos de progreso).
5. **Fase 5: Pruebas y Validación (Capítulo 7)**
    - Pruebas unitarias de funciones críticas (ej., cálculo de métricas, conexión a BD).
    - Pruebas de integración entre los componentes (Streamlit-Supabase, Streamlit-Lógica Python).
    - Pruebas funcionales para verificar que el sistema cumple con los requisitos definidos.
    - Pruebas de usabilidad (evaluación de la facilidad de uso de la interfaz).
    - Validación con conjuntos de datos realistas (CSVs de M3Display, datos del script Tkinter modificado, datos simulados).
6. **Fase 6: Documentación y Elaboración de la Memoria (Continua)**
    - Redacción de la memoria del TFG.
    - Comentarios en el código.
    - Elaboración del manual de usuario (Anexo).

## 3.3 TÉCNICAS DE RECOLECCIÓN Y ANÁLISIS DE REQUISITOS

La definición precisa de los requisitos es fundamental para el éxito del proyecto. Las técnicas empleadas incluyen:

1. **Análisis Documental:**
    - Estudio detallado del TFG de Matilla Plaza (2023) para comprender las funcionalidades existentes del sistema M3Display/Coins, el formato de los datos CSV generados y las limitaciones identificadas por el autor.
    - Revisión de la bibliografía científica (artículos, revisiones) sobre sistemas de rehabilitación con RA, gestión de datos clínicos y herramientas de análisis de movimiento para identificar funcionalidades comunes, métricas relevantes y mejores prácticas.
2. **Análisis del Problema:**
    - Identificación de la "brecha" entre la generación de datos por M3Display y su uso clínico efectivo, tal como se describe en la Introducción y el Estado del Arte. Este análisis directo del problema guía la definición de los requisitos de la solución.
3. **Definición de Casos de Uso (Implícitos):**
    - Aunque no se formalicen extensamente, se consideran escenarios típicos de uso por parte de un terapeuta: registrar un nuevo paciente, añadir una sesión de terapia (manual o "capturada"), visualizar los detalles y datos de una sesión, analizar la trayectoria de una sesión, comparar el progreso de un paciente entre sesiones, generar un informe visual. Estos escenarios ayudan a derivar requisitos funcionales.
4. **Prototipado Iterativo (Especialmente para UI):**
    - El uso de Streamlit permite crear prototipos rápidos de la interfaz. Las decisiones sobre qué información mostrar y cómo organizarla se pueden tomar y modificar de forma iterativa durante la fase de implementación, lo que en sí mismo es una forma de refinar los requisitos de la interfaz basados en la viabilidad y la claridad.
5. **Consideración de Requisitos Técnicos y Normativos:**
    - Análisis de las capacidades de Supabase (para diseño de BD y seguridad) y Streamlit (para diseño de UI).

- Investigación de los requisitos derivados de normativas de protección de datos (GDPR/HIPAA) para definir requisitos no funcionales de seguridad y privacidad.

## 3.4 HERRAMIENTAS DE DESARROLLO

Las principales herramientas seleccionadas para el desarrollo de este proyecto son:

- **Lenguajes de Programación:**
  - **Python (versión 3.8+):** Lenguaje principal para la lógica de la aplicación, el análisis de datos y el backend (con Streamlit).
- **Frameworks y Librerías:**
  - **Streamlit:** Framework de Python escogido para el desarrollo rápido del frontend y la interfaz de usuario web, permitiendo transformar scripts de datos en aplicaciones interactivas con mínima codificación de frontend tradicional
  - **Supabase (Python Client Library):** Para la interacción con la base de datos y servicios de Supabase desde la aplicación Python/Streamlit.
  - **Pandas:** Para la manipulación y análisis de datos tabulares (CSV, DataFrames).
  - **NumPy:** Para operaciones numéricas eficientes.
  - **SciPy:** Para algoritmos de procesamiento de señales (ej. filtrado) y análisis estadístico.
  - **Matplotlib / Seaborn:** Para la generación de gráficos estáticos y animaciones (GIFs).
  - **Pillow (PIL):** Para el manejo de imágenes (ej. en el script Tkinter y potencialmente para procesar imágenes subidas).
  - **dotenv:** Para la gestión segura de las credenciales de acceso a Supabase mediante variables de entorno.
- **Base de Datos y Backend:**
  - **Supabase:** Plataforma BaaS que proporciona una base de datos PostgreSQL gestionada, servicios de autenticación de usuarios, almacenamiento de archivos y APIs RESTful automáticas.

- **Entorno de Desarrollo y Control de Versiones:**
    - **Visual Studio Code (VS Code):** Como editor de código principal, por su amplio soporte para Python, integración con Git y extensiones útiles.
    - **Git y GitHub:** Para el control de versiones del código fuente, seguimiento de cambios y gestión del proyecto.
- **Herramienta de Captura de Datos (Prototipo):**
    - **Tkinter (Python):** Para el script *codigo_raton.py* que simula la captura de datos de movimiento y genera datos para la validación.

Adicionalmente a las herramientas listadas, y con el objetivo de optimizar el proceso de desarrollo y explorar soluciones técnicas, se emplearon herramientas de asistencia basadas en inteligencia artificial. Específicamente, GitHub Copilot se utilizó como asistente integrado en el editor Visual Studio Code para la sugerencia y autocompletado de código. Asimismo, se realizaron consultas a modelos de lenguaje accesibles a través de plataformas como AI Studio para la generación de ideas, depuración de fragmentos de código y clarificación de conceptos. Cabe destacar que la utilización de estas herramientas fue siempre supervisada, y cualquier código o sugerencia proveniente de ellas fue rigurosamente revisado, comprendido, adaptado y validado por el autor para asegurar su correcta implementación, eficiencia, seguridad y alineación con los requisitos y objetivos del presente Trabajo de Fin de Grado. La responsabilidad final sobre la totalidad del código desarrollado recae en el autor.

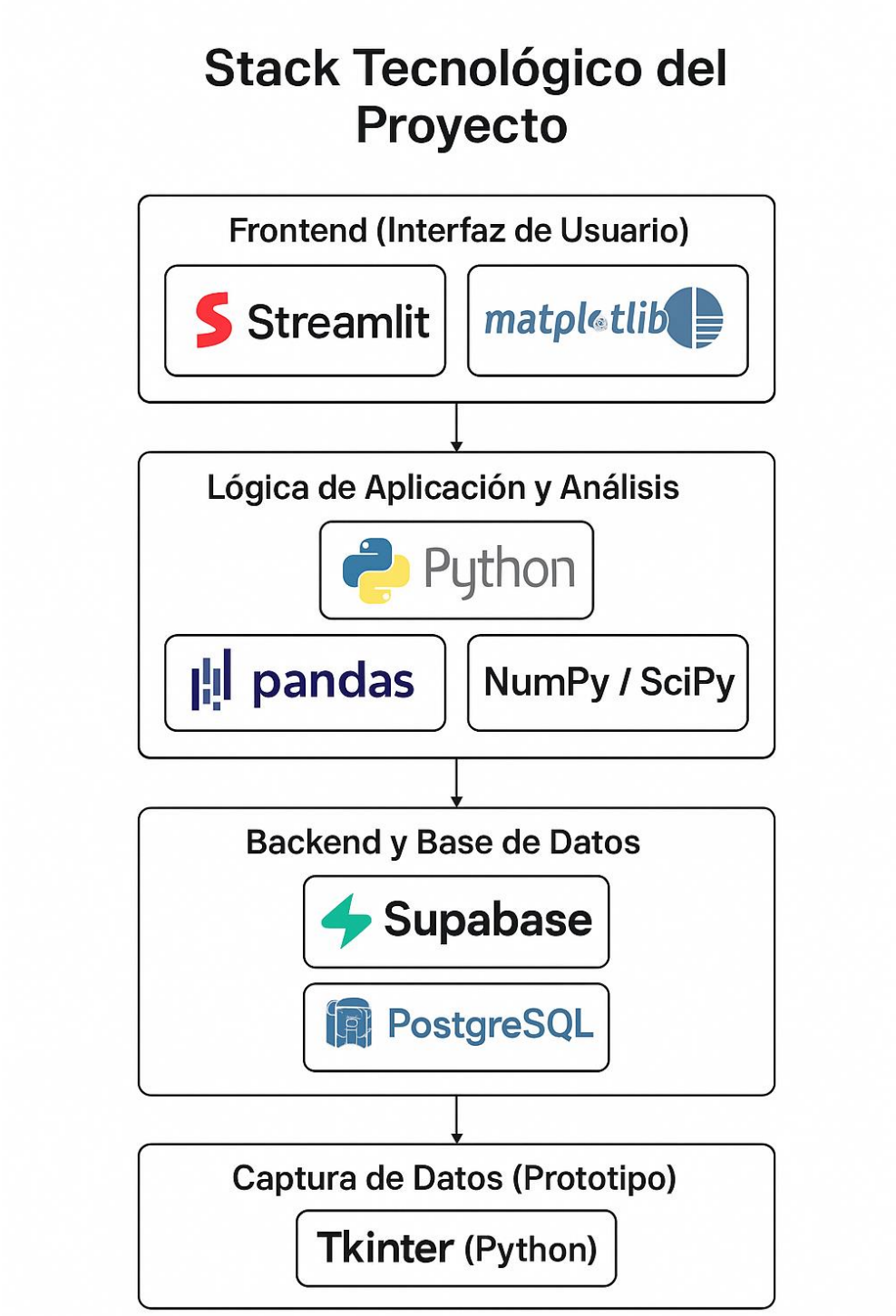


*Figura 5: Conjunto de tecnologías usadas en el proyecto, mostrando las principales herramientas y su interrelación conceptual en la arquitectura de la aplicación web. Fuente: Elaboración propia.*

## 3.5 CRITERIOS DE EVALUACIÓN Y VALIDACIÓN DEL PROYECTO

La evaluación del TFG se centrará en el cumplimiento de los objetivos planteados y la funcionalidad de la aplicación web desarrollada. Los principales criterios de evaluación serán:

1. **Cumplimiento de Requisitos Funcionales:**
    - ¿El sistema permite registrar y gestionar pacientes y sesiones de forma segura en Supabase?
    - ¿Se almacenan correctamente los datos de trayectoria numérica (provenientes del script Tkinter modificado o de CSVs subidos) en la base de datos?
    - ¿La aplicación implementa algoritmos para el análisis cuantitativo del movimiento (cálculo de métricas clave)?
    - ¿Se visualizan interactivamente los datos de sesión, las trayectorias y las métricas calculadas?
    - ¿Se pueden generar informes visuales (ej. GIFs animados de trayectorias) a partir de los datos almacenados?
2. **Cumplimiento de Requisitos No Funcionales (Principales):**
    - **Usabilidad:** ¿La interfaz de usuario desarrollada con Streamlit es intuitiva y fácil de usar para un perfil de terapeuta (evaluado por el propio desarrollador y potencialmente mediante pruebas heurísticas)?
    - **Seguridad (Conceptual):** ¿Se han implementado las medidas básicas de seguridad en Supabase (RLS, autenticación) y se han considerado los principios de protección de datos en el diseño?
    - **Rendimiento (Básico):** ¿La carga de datos y la generación de visualizaciones se realizan en tiempos razonables para los conjuntos de datos de prueba?
3. **Calidad del Software Desarrollado:**
    - ¿El código es modular, legible y está adecuadamente comentado?
    - ¿Se han manejado errores y excepciones de forma apropiada?
4. **Validación con Datos Realistas:**
    - ¿La plataforma procesa y analiza correctamente los datos CSV generados por M3Display (Matilla Plaza, 2023)?
    - ¿La plataforma procesa, analiza y visualiza correctamente los datos numéricos de trayectoria generados por el **script

prototipo desarrollado en Python (utilizando Tkinter, su biblioteca estándar para interfaces gráficas de usuario)**, el cual simula la captura de movimiento con un dispositivo de entrada convencional?

5. Calidad de la Documentación:
    - ¿La memoria del TFG es clara, completa y sigue los estándares académicos?
    - ¿El manual de usuario (si se incluye) es útil y comprensible?

La validación se realizará principalmente mediante pruebas funcionales exhaustivas utilizando los conjuntos de datos mencionados, asegurando que los resultados de los análisis son coherentes y las visualizaciones representan correctamente la información. Dada la naturaleza de un TFG, la validación en un entorno clínico real con terapeutas y pacientes reales excede el alcance, pero el sistema se diseñará para ser *potencialmente aplicable* en dichos contextos.

# CAPÍTULO 4. ANÁLISIS DE REQUISITOS

Este capítulo tiene como objetivo fundamental especificar de manera clara y concisa los requisitos que debe cumplir la aplicación web para la gestión y análisis de datos de rehabilitación con Realidad Aumentada. La correcta definición de estos requisitos es crucial para guiar el diseño, la implementación y la posterior validación del sistema. Se dividen en requisitos funcionales, que describen las operaciones y funcionalidades que el sistema ofrecerá al usuario (principalmente el terapeuta), y requisitos no funcionales, que definen las características de calidad y las restricciones bajo las cuales operará el sistema.

Los requisitos se han derivado principalmente del análisis documental del trabajo previo de Matilla Plaza (2023), la revisión de la literatura científica sobre sistemas similares, la identificación de la brecha existente (descrita en el Capítulo 2), y la consideración de los objetivos del presente TFG (detallados en el Capítulo 1).

## 4.1 PERFILES DE USUARIOS

El sistema está diseñado principalmente para ser utilizado por el siguiente perfil de usuario:

- **Terapeuta/Personal Clínico:** Profesional sanitario (fisioterapeuta, terapeuta ocupacional, médico rehabilitador) encargado de planificar, supervisar y evaluar las sesiones de rehabilitación de los pacientes. Este usuario interactuará directamente con la aplicación web para gestionar pacientes, registrar sesiones, visualizar datos, analizar el progreso y generar informes. Se asume un nivel básico-medio de competencia informática.

*(Aunque no es un usuario directo de esta plataforma de análisis, el sistema maneja datos de los "Pacientes", por lo que sus necesidades de privacidad y la naturaleza de sus datos son consideraciones primordiales).*

## 4.2 REQUISITOS FUNCIONALES (RF)

Los requisitos funcionales describen las acciones específicas que el sistema debe ser capaz de realizar. Se identifican con el prefijo "RF".

### 4.2.1 GESTIÓN DE USUARIOS (AUTENTIFICACIÓN Y AUTORIZACIÓN)

- **RF001:** El sistema deberá permitir el registro de nuevos usuarios terapeutas.

- **RF002:** El sistema deberá permitir a los usuarios terapeutas autenticarse de forma segura para acceder a la aplicación (login/logout).
- **RF003:** El sistema deberá gestionar diferentes niveles de acceso o roles si fuera necesario en futuras ampliaciones (aunque para el alcance actual se considera un único rol de terapeuta con acceso completo a sus pacientes).

### 4.2.2 GESTIÓN DE PACIENTES

- **RF004:** El sistema permitirá al terapeuta registrar nuevos pacientes, almacenando información demográfica básica (ej. identificador, nombre, fecha de nacimiento, notas relevantes).
- **RF005:** El sistema permitirá al terapeuta visualizar una lista de los pacientes registrados.
- **RF006:** El sistema permitirá al terapeuta editar la información de un paciente existente.
- **RF007:** El sistema permitirá al terapeuta eliminar un paciente (considerando implicaciones de integridad de datos de sus sesiones).

### 4.2.3 GESTIÓN DE SESIONES DE REHABILITACIÓN

- **RF008:** El sistema permitirá al terapeuta registrar una nueva sesión de rehabilitación asociada a un paciente específico.
- **RF009:** El registro de una sesión deberá permitir incluir información contextual como la fecha, hora, tipo de ejercicio/juego (ej. "Coins - Sinusoide", "Coins - Reloj"), parámetros de la sesión (ej. tiempo objetivo, dificultad), y notas del terapeuta.
- **RF010:** El sistema permitirá registrar una sesión realizada mediante el script de captura de movimiento con ratón (*codigo_raton.py*), incluyendo la captura de la imagen final del trazado.
- **RF011:** El sistema permitirá registrar una sesión de forma manual, permitiendo la subida de una imagen del resultado (si existe) y un archivo de datos de trayectoria (CSV/JSON).

### 4.2.4 ALMACENAMIENTO Y GESTIÓN DE DATOS DE TRAYECTORIA NUMÉRICOS

- **RF012:** El sistema deberá ser capaz de procesar un archivo de datos de trayectoria (formato CSV con columnas: timestamp, pos_x, pos_y)

generado por el script de captura *codigo_raton.py* (modificado) o subido manualmente.

- **RF013:** El sistema deberá almacenar de forma segura y estructurada la secuencia completa de puntos numéricos (timestamp relativo, coordenada X, coordenada Y) de la trayectoria de cada sesión en la base de datos Supabase, vinculada al id_sesion correspondiente.
- **RF014:** El sistema permitirá recuperar los datos numéricos de trayectoria almacenados para una sesión específica para su posterior análisis y visualización.

### 4.2.5 ANÁLISIS CUANTITATIVO DE DATOS DE MOVIMIENTO

- **RF015:** El sistema deberá implementar algoritmos para aplicar un filtrado (ej. media móvil) a los datos de trayectoria recuperados, con el fin de reducir el ruido.
- **RF016:** El sistema deberá calcular y mostrar métricas cinemáticas básicas a partir de los datos de trayectoria de una sesión, incluyendo al menos:
    - Duración total de la tarea.
    - Longitud de la trayectoria recorrida por el paciente.
    - Velocidad media del movimiento.

### 4.2.6 VISUALIZACIÓN DE DATOS Y PROGRESO

- **RF019:** El sistema permitirá visualizar los detalles de una sesión de rehabilitación específica, incluyendo su información contextual, la imagen asociada (si existe) y las métricas calculadas.
- **RF020:** El sistema deberá visualizar gráficamente la trayectoria de movimiento (X vs Y) recuperada de la base de datos para una sesión seleccionada.
- **RF021:** El sistema permitirá visualizar el progreso de un paciente a lo largo de múltiples sesiones mediante gráficos que muestren la evolución de métricas clave (ej. puntuación (si se integra), duración, error, suavidad) en función del tiempo o número de sesión.
- **RF022:** El sistema permitirá generar y visualizar una animación GIF del movimiento de la mano realizado durante una sesión a partir de los datos de trayectoria almacenados.
- **RF023:** El sistema permitirá la descarga de los GIFs generados.

### 4.2.6 GENERACIÓN DE INFORMES (SIMPLIFICADO)

- **RF024:** Las visualizaciones de progreso y los GIFs generados servirán como una forma de informe visual simplificado para el terapeuta. (No se contempla la generación de informes PDF complejos y estructurados dentro del alcance principal).

## 4.3 REQUISITOS NO FUNCIONALES (RNF)

Los requisitos no funcionales definen las cualidades del sistema y las restricciones bajo las cuales debe operar. Se identifican con el prefijo "RNF".

### 4.3.1 SEGURIDAD DE DATOS

- **RNF001:** El acceso a la aplicación deberá estar protegido mediante un sistema de autenticación de usuarios (login/password).
- **RNF002:** Los datos de los pacientes y sus sesiones deberán almacenarse de forma segura en la base de datos Supabase.
- **RNF003:** El sistema deberá estar diseñado considerando los principios de protección de datos establecidos en normativas como GDPR y HIPAA (ej. minimización de datos, control de acceso).
- **RNF004:** La transmisión de datos entre el cliente (navegador) y el servidor (Supabase/Streamlit) deberá realizarse mediante protocolos seguros (HTTPS).

### 4.3.2 USABILIDAD

- **RNF005:** La interfaz de usuario deberá ser intuitiva, clara y fácil de aprender para los terapeutas, minimizando la necesidad de formación extensa.
- **RNF006:** El sistema deberá proporcionar retroalimentación adecuada al usuario sobre las acciones realizadas (ej. mensajes de éxito, error, carga).
- **RNF007:** La navegación dentro de la aplicación deberá ser lógica y consistente.
- **RNF008:** Los tiempos de respuesta para las interacciones comunes (cargar listas, mostrar detalles de sesión) deberán ser aceptables para no frustrar al usuario.

### 4.3.3 RENDIMIENTO

- **RNF009:** El procesamiento de datos de trayectoria y el cálculo de métricas para una sesión individual deberán realizarse en un tiempo razonable (ej. pocos segundos) tras la selección de la sesión.
- **RNF010:** La generación de gráficos y animaciones GIF deberá ser eficiente para no causar demoras excesivas en la interfaz.
- **RNF011:** El sistema deberá ser capaz de manejar un volumen moderado de datos de pacientes y sesiones (adecuado para un entorno de investigación o una clínica pequeña/mediana, sin pretender escalabilidad masiva en esta fase de TFG).

### 4.3.4 COMPATIBILIDAD

- **RNF012:** La aplicación web deberá ser accesible y funcional en los navegadores web modernos más comunes (ej. Chrome, Firefox, Edge) en sistemas operativos de escritorio (Windows, macOS, Linux).
- **RNF013:** El sistema deberá ser capaz de procesar archivos CSV de trayectoria con un formato predefinido (timestamp, X, Y, separados por punto y coma).

### 4.3.5 MANTENIBILIDAD Y EXTENSIBILIDAD (CONSIDERACIONES DE DISEÑO)

- **RNF014:** El código deberá ser modular y estar bien documentado para facilitar futuras modificaciones y extensiones.
- **RNF015:** El diseño de la base de datos deberá permitir la posible adición de nuevas métricas o tipos de datos en el futuro.

Estos requisitos servirán como base para el diseño y la implementación del sistema, y como referencia para las pruebas y la validación final del TFG.

# CAPÍTULO 5. DISEÑO DEL SISTEMA

Este capítulo presenta el diseño detallado de la aplicación web desarrollada para la gestión y análisis de datos de rehabilitación con Realidad Aumentada. Se aborda la arquitectura general del sistema, el diseño técnico de sus componentes clave, incluyendo la estructura de la base de datos en Supabase, las tecnologías empleadas, y el diseño de la interfaz de usuario con sus flujos de navegación. El objetivo es proporcionar una comprensión clara de cómo se ha estructurado la solución para cumplir con los requisitos funcionales y no funcionales identificados en el capítulo anterior, y cómo se integran las diferentes tecnologías para ofrecer una plataforma robusta y útil para los terapeutas.

## 5.1 ARQUITECTURA GENERAL DEL SISTEMA

### 5.1.1 VISIÓN GENERAL Y FILOSOFÍA DE DISEÑO

La aplicación web desarrollada para la gestión y análisis de datos de rehabilitación se ha concebido siguiendo una arquitectura cliente-servidor basada en web. En este modelo, el cliente es el navegador web utilizado por el terapeuta, mientras que el servidor corresponde a la instancia donde se ejecuta la aplicación desarrollada con Streamlit, un framework de Python de código abierto que facilita la creación rápida de interfaces web interactivas y centradas en datos.

La filosofía de diseño que ha guiado la construcción de la arquitectura se ha centrado en varios principios clave:

- **Modularidad:** Para facilitar el desarrollo, las pruebas y el mantenimiento futuro, buscando la separación de responsabilidades entre los diferentes componentes del sistema.
- **Rapidez de Desarrollo y Prototipado:** Aprovechando las capacidades de Streamlit para construir y iterar sobre la interfaz de usuario de forma eficiente.
- **Escalabilidad Conceptual y Servicios Gestionados:** Utilizando Supabase, una plataforma open-source que ofrece una solución Backend-as-a-Service (BaaS) integral, proporcionando la base de datos PostgreSQL, así como servicios de autenticación y almacenamiento en la nube, lo que permite una gestión simplificada de la infraestructura backend y un potencial de escalabilidad.
- **Seguridad desde el Diseño:** Dada la naturaleza sensible de los datos clínicos manejados, se han considerado desde el inicio los mecanismos de seguridad para el acceso, la transmisión y el almacenamiento de la

información, aprovechando las funcionalidades de autenticación y autorización de Supabase.

El objetivo principal de la arquitectura es, por tanto, proporcionar una plataforma intuitiva y eficiente para el terapeuta, que centralice los datos generados durante las sesiones de rehabilitación y ofrezca herramientas potentes para su análisis cuantitativo y visualización, contribuyendo así a un seguimiento más objetivo y personalizado del progreso del paciente.

### 5.1.2 DIAGRAMA DE COMPONENTES PRINCIPALES

La arquitectura del sistema se compone de varios elementos interconectados que colaboran para ofrecer la funcionalidad completa de la aplicación.

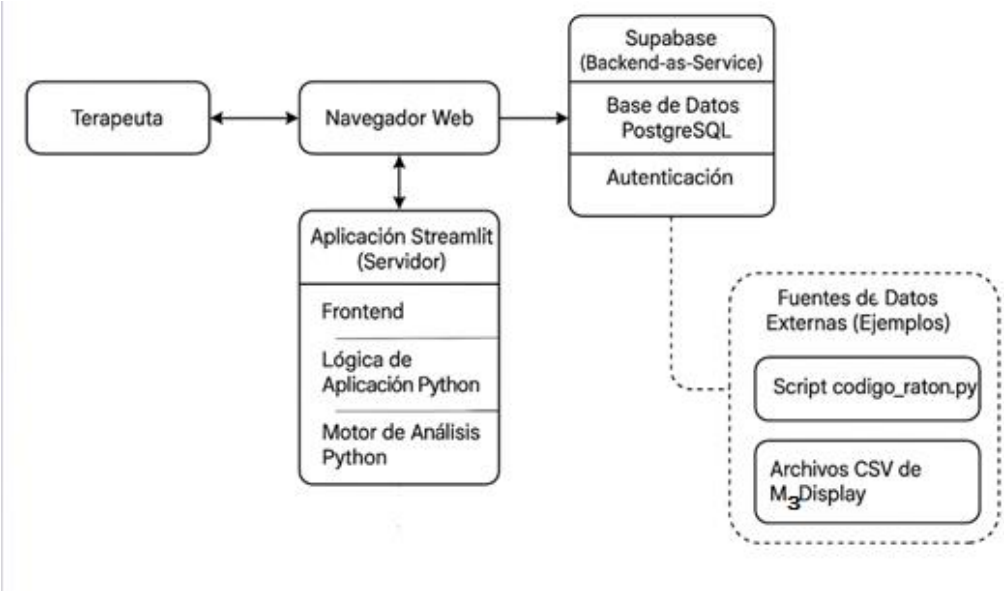


*Figura 6: Diagrama de componentes de alto nivel de la aplicación web. Fuente: Elaboración propia.*

La figura 6 muestra la interacción entre el Terapeuta (usuario), el Navegador Web (cliente), la Aplicación Streamlit (frontend y lógica de servidor), el Motor Python (análisis de datos), y la plataforma Supabase (Base de Datos PostgreSQL, Autenticación, Almacenamiento). También se indica la interacción con fuentes de datos externas como el script *codigo_raton.py* o los CSV del sistema M3Display.

Descripción detallada de la figura 6:

1. **Usuario (Terapeuta):** Interactúa con la aplicación a través de un navegador web.
2. **Navegador Web (Cliente):** Renderiza la interfaz de usuario generada por Streamlit y envía las interacciones del usuario al servidor de Streamlit.
3. **Aplicación Streamlit (Servidor):**

- **Frontend:** Genera dinámicamente la interfaz de usuario web.
- **Lógica de Aplicación Python:** Maneja la lógica de la aplicación, el flujo de la aplicación, la interacción con Supabase para operaciones CRUD (Create, Read, Update, Delete), y la orquestación del análisis de datos.

4. **Motor de Análisis Python:** Conjunto de funciones y librerías (Pandas, NumPy, Matplotlib, SciPy) que se ejecutan en el servidor Streamlit para procesar los datos de trayectoria, calcular métricas y generar visualizaciones.
5. **Supabase (Backend-as-a-Service):**
   - **Base de Datos PostgreSQL:** Almacena de forma persistente los datos de usuarios (implícitamente manejados por Supabase Auth), pacientes, terapeutas, terapias, sesiones y los puntos de trayectoria.
   - **Autenticación:** Gestiona el registro, inicio y cierre de sesión de los usuarios terapeutas.
   - **Almacenamiento (Storage):** Almacena archivos asociados a las sesiones, como las imágenes de los dibujos y los archivos CSV de trayectoria.
6. **Fuentes de Datos Externas (Ejemplos):**
   - **Script *codigo_raton.py*:** Genera un archivo de imagen y un archivo CSV con datos numéricos de trayectoria.
   - **Archivos CSV de M3Display:** (Matilla Plaza, 2023) Datos preexistentes que pueden ser cargados manualmente.

### 5.1.3 ARQUITECTURA CLIENTE-SERVIDOR Y FLUJO DE DATOS PRINCIPAL

El sistema opera bajo un modelo cliente-servidor. El cliente es el navegador web del terapeuta, responsable de presentar la interfaz de usuario generada por Streamlit y de transmitir las acciones del usuario (clics en botones, envíos de formularios) al servidor. El servidor de Streamlit aloja la aplicación Python, interpreta las interacciones del cliente, ejecuta la lógica de negocio (incluyendo el procesamiento y análisis de datos), y se comunica con los servicios de Supabase (base de datos, autenticación, almacenamiento) para la persistencia y recuperación de la información.

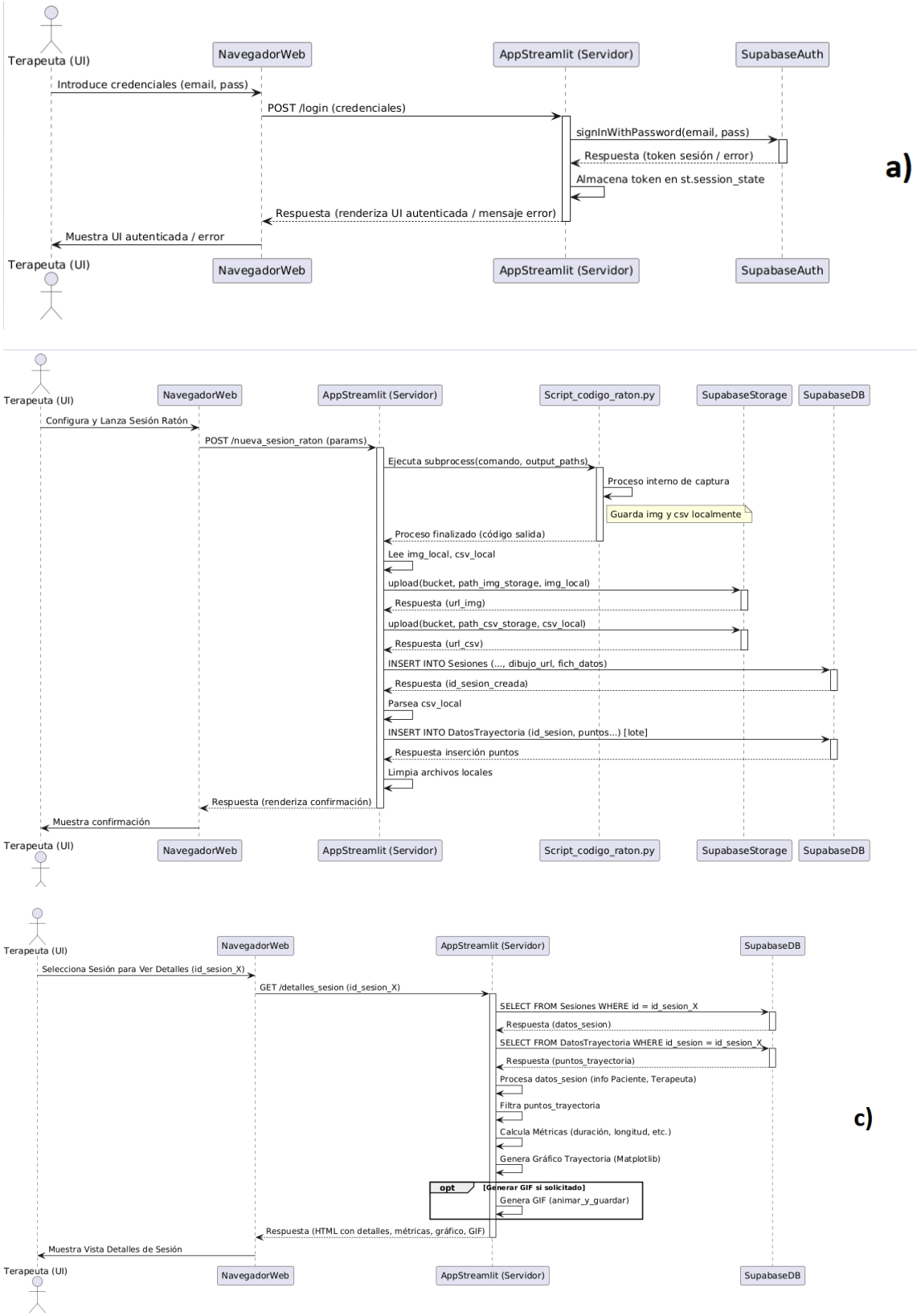


*Figura 7: Diagrama de flujo de datos simplificado para operaciones clave: (a) Autentificación de usuario, (b) Registro de una nueva sesión con captura de*

*datos de codigo_raton.py, y (c) Visualización y análisis de una sesión existente. Fuente: Elaboración propia.*

En la figura 7 se pueden ver 3 diagramas de flujo. Para detallar un poco más los diagramas de la figura:

- **Diagrama a): Autentificación de usuario:**
  - Terapeuta introduce las credenciales en la UI de Streamlit.
  - Streamlit envía las credenciales a Supabase Auth.
  - Supabase Auth valida y devuelve un token de sesión.
  - Streamlit almacena el token y actualiza la UI.
- **Diagrama b): Registro de una nueva sesión con captura de datos de *codigo_raton.py*:**
  - Terapeuta configura y lanza la sesión desde Streamlit.
  - Streamlit ejecuta *codigo_raton.py* (vía subprocess).
  - *codigo_raton.py* guarda imagen y CSV de trayectoria localmente (en TEMP_DIR).
  - Streamlit lee los archivos locales.
  - Streamlit sube la imagen y el CSV a Supabase Storage, obteniendo URLs.
  - Streamlit inserta los metadatos de la sesión (incluyendo URLs) en la tabla Sesiones de Supabase.
  - Streamlit lee el CSV de trayectoria (local o desde Storage) y guarda los puntos numéricos en la tabla DatosTrayectoria de Supabase.
- **Diagrama c): Visualización y análisis de una sesión existente:**
  - Terapeuta selecciona una sesión en la UI de Streamlit.
  - Streamlit (Python) solicita a Supabase los metadatos de la sesión (tabla Sesiones).
  - Streamlit (Python) solicita a Supabase los datos de trayectoria (tabla DatosTrayectoria) para esa sesión.
  - La lógica Python procesa/filtra los datos de trayectoria, calcula métricas.
  - Streamlit renderiza la información de la sesión, las métricas y las visualizaciones (gráficos, GIFs) en la UI.

## 5.2 DISEÑO TÉCNICO

### 5.2.1 CONJUNTO DE TECNOLOGÍAS UTILIZADO

La selección del conjunto de tecnologías usado se basó en la necesidad de un desarrollo rápido y eficiente, la capacidad de manejar y analizar datos de forma

robusta, la creación de interfaces de usuario interactivas y la utilización de herramientas predominantemente de código abierto y bien soportadas por la comunidad. A continuación, se detallan las tecnologías clave y su rol en el proyecto:

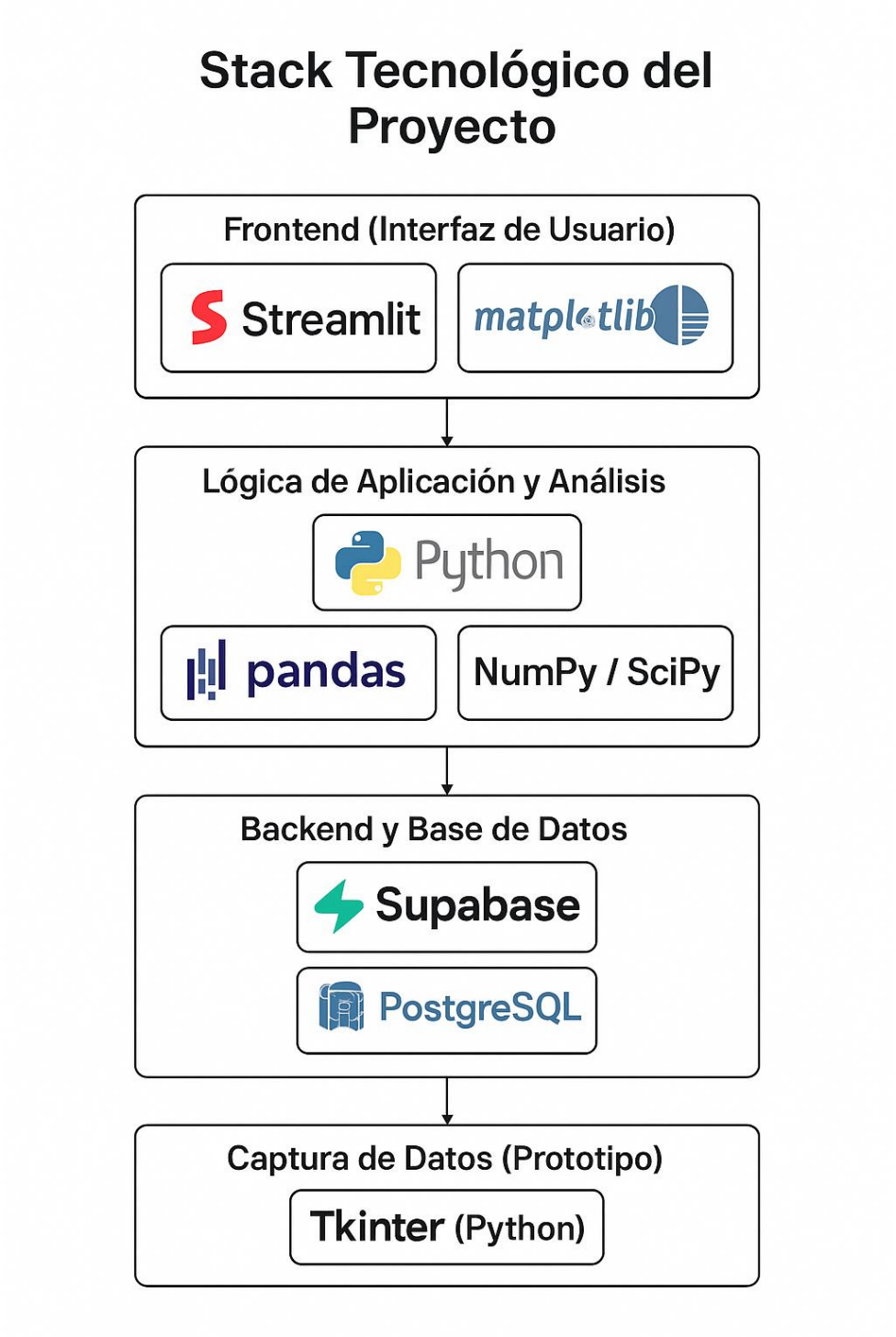


*Figura 5: Conjunto de tecnologías usadas en el proyecto, mostrando las principales herramientas y su interrelación conceptual en la arquitectura de la aplicación web. Fuente: Elaboración propia.*

- **Python**
    - **Descripción:** Python ha sido elegido como el lenguaje de programación principal para la totalidad de la lógica de la aplicación, tanto en el lado del servidor (a través de Streamlit) como para el script de captura de datos (*codigo_raton.py*). Su sintaxis clara y legible, junto con su extenso ecosistema de librerías, lo convierten en una opción ideal para el desarrollo rápido de prototipos y aplicaciones completas, especialmente aquellas que involucran manipulación y análisis de datos.
    - **Ventaja para el TFG:** La facilidad de aprendizaje y la vasta cantidad de recursos disponibles (documentación, tutoriales, foros comunitarios) reducen la curva de aprendizaje y agilizan el desarrollo. Su naturaleza interpretada permite una depuración y experimentación más rápidas. La disponibilidad de librerías especializadas para ciencia de datos, análisis numérico y desarrollo web es crucial para este proyecto. En este TFG, Python se utiliza para:
        - Desarrollar toda la aplicación web con Streamlit.
        - Implementar la lógica de negocio (gestión de pacientes, sesiones, etc.).
        - Interactuar con la base de datos Supabase.
        - Realizar el preprocesamiento y análisis cuantitativo de los datos de trayectoria.
        - Generar las visualizaciones y animaciones.
        - Desarrollar el script auxiliar de captura de datos con Tkinter.

- **Streamlit:**
    - **Descripción:** Streamlit es un framework de Python de código abierto diseñado específicamente para crear aplicaciones web interactivas y centradas en datos con un mínimo de código. Permite a los desarrolladores transformar scripts de datos en aplicaciones web compartibles en cuestión de horas, sin necesidad de experiencia previa en desarrollo frontend (HTML, CSS, JavaScript). Funciona ejecutando el script Python de arriba abajo cada vez que hay una interacción del usuario, y renderizando los cambios en la interfaz de forma eficiente.

- **Ventaja para el TFG:** Su principal atractivo es la **velocidad de desarrollo y la simplicidad**. Permite crear interfaces de usuario funcionales y estéticamente agradables utilizando widgets nativos de Streamlit. La integración directa con librerías de visualización de Python como Matplotlib y Plotly es fundamental para mostrar los resultados del análisis de datos. En este proyecto, Streamlit se utiliza para:
    - Construir todas las vistas de la aplicación: autenticación, gestión de tablas (pacientes, terapeutas, terapias), historial de sesiones, formularios de entrada de datos, vista de detalles de sesión con análisis y visualizaciones, y la vista de análisis de progreso del paciente.
    - Gestionar el estado de la sesión del usuario.
    - Orquestar las llamadas a las funciones de lógica de negocio y análisis implementadas en Python.

- **Supabase (Backend y Base de Datos):**
  La elección de Supabase como la plataforma Backend-as-a-Service (BaaS) para este proyecto se fundamentó en una combinación de su naturaleza Open Source, su robustez tecnológica y la agilidad que aporta al ciclo de desarrollo. Supabase se presenta como una alternativa atractiva a otras soluciones BaaS propietarias, ofreciendo un conjunto de herramientas integradas que simplifican significativamente la construcción y gestión de la infraestructura backend necesaria para una aplicación web moderna como la propuesta.

  Las principales razones para su selección y cómo se aprovechan sus características en este TFG son:
    - **Base de Datos PostgreSQL Gestionada:**
        - **Descripción:** Supabase proporciona una instancia dedicada de PostgreSQL, una de las bases de datos relacionales de código abierto más potentes, maduras y ricas en funcionalidades del mundo. Esto significa que el proyecto se beneficia de la fiabilidad, integridad de datos (gracias a las transacciones ACID), y la capacidad para definir esquemas complejos con relaciones, claves foráneas y restricciones, tal como se detalla en la sección 5.2.2.
        - **Ventaja para el TFG:** Evita la necesidad de configurar, gestionar y mantener un servidor de base de datos PostgreSQL por separado. Supabase se encarga de las copias de seguridad, el escalado básico y la seguridad a

nivel de infraestructura. Permite el uso de SQL estándar para consultas complejas si fuera necesario, aunque muchas interacciones se realizan a través de su cliente Python. El soporte para tipos de datos avanzados como JSONB (aunque no se use extensivamente en el diseño actual para las trayectorias, es una opción para datos semiestructurados) y timestamptz es también valioso.

- Autenticación Integrada (Supabase Auth):
  - **Descripción:** Supabase incluye un servicio de autenticación completo basado en GoTrue (un servidor de API de autenticación Open Source). Soporta múltiples métodos de autenticación, incluyendo el registro y login con email/contraseña (utilizado en este proyecto), así como proveedores OAuth (Google, GitHub, etc.) y magic links. Gestiona la creación de usuarios, la verificación de emails, el reseteo de contraseñas y la emisión de JSON Web Tokens (JWT) para la gestión de sesiones.
  - **Ventaja para el TFG:** Elimina la complejidad de implementar un sistema de autenticación seguro desde cero. La integración con Row Level Security (RLS) de PostgreSQL permite definir políticas de acceso a datos basadas en la identidad del usuario autenticado, lo cual es crucial para la seguridad de los datos clínicos.

- Almacenamiento de Archivos (Supabase Storage):
  - **Descripción:** Ofrece un servicio de almacenamiento de objetos compatible con S3 para guardar y servir archivos de cualquier tipo (imágenes, vídeos, documentos, etc.). Permite organizar los archivos en buckets y carpetas, y gestionar los permisos de acceso de forma granular, incluso integrándolos con las políticas RLS de la base de datos.
  - **Ventaja para el TFG:** Se utiliza para almacenar los archivos generados o subidos durante las sesiones de rehabilitación, como las imágenes de los dibujos (.png) y los archivos CSV originales de las trayectorias. La capacidad de generar URLs públicas (o firmadas) facilita la visualización de estos archivos en la interfaz de Streamlit.

- **APIs Autogeneradas (PostgREST):**
  - **Descripción:** Supabase utiliza PostgREST para generar automáticamente una API RESTful directamente sobre el esquema de la base de datos PostgreSQL. Esto significa que cualquier tabla o vista en la base de datos es accesible a través de puntos finales HTTP bien definidos, con soporte para filtrado, ordenación, paginación y operaciones CRUD.
  - **Ventaja para el TFG:** Aunque la interacción principal se realiza a través de la librería cliente `supabase-py` (que internamente usa estas APIs), esta característica subyacente simplifica la comunicación entre la aplicación Python/Streamlit y la base de datos. También abre la puerta a futuras integraciones con otros sistemas o clientes que puedan consumir esta API directamente (como se menciona en el objetivo secundario de una posible ingesta automatizada desde M3Display).

- **Cliente Python (`supabase-py`):**
  - **Descripción:** Supabase proporciona una librería cliente oficial para Python que facilita la interacción con todos sus servicios (Auth, Database, Storage, Realtime Functions) de una manera idiomática y sencilla desde el código Python.
  - **Ventaja para el TFG:** Permite que toda la lógica de interacción con el backend (autenticación, consultas a la base de datos, subida/descarga de archivos) se implemente directamente en los scripts de Streamlit, manteniendo un desarrollo cohesivo dentro del ecosistema Python.

- **Naturaleza Open Source y Comunidad:**
  - **Descripción:** Al ser una plataforma construida sobre herramientas de código abierto (PostgreSQL, GoTrue, PostgREST, etc.), ofrece transparencia y evita el "vendor lock-in" total de algunas soluciones propietarias. Cuenta con una comunidad activa y una documentación en crecimiento.
  - **Ventaja para el TFG:** Permite una mayor comprensión de cómo funcionan los componentes subyacentes y ofrece la posibilidad de auto-hospedar la plataforma en el futuro si

fuera necesario, aunque para este TFG se utiliza la versión cloud gestionada por Supabase.

- Escalabilidad Conceptual y Plan Gratuito:
  - **Descripción:** Si bien el plan gratuito tiene limitaciones, tiene las funcionalidades necesarias para el desarrollo y prototipado de este TFG. La arquitectura de Supabase está diseñada para escalar a medida que crecen las necesidades de la aplicación, con planes de pago que ofrecen mayores recursos.
  - **Ventaja para el TFG:** Permite desarrollar y probar la aplicación completa sin incurrir en costes iniciales significativos, con una ruta clara hacia una mayor capacidad si el proyecto evolucionara más allá del ámbito académico.

En resumen, Supabase fue seleccionado porque proporciona un conjunto integrado y robusto de servicios backend esenciales, permitiendo al desarrollador centrarse en la lógica de la aplicación específica (gestión de datos de rehabilitación, análisis y visualización) en lugar de en la compleja tarea de construir y mantener la infraestructura subyacente. Su compatibilidad con PostgreSQL y su enfoque en la seguridad (especialmente RLS) son particularmente valiosos para una aplicación que maneja datos clínicos sensibles.

- Librerías Python Claves:
  - Pandas:
    - **Descripción:** Es la librería fundamental en Python para la manipulación y análisis de datos tabulares. Proporciona estructuras de datos de alto rendimiento y fáciles de usar y herramientas para leer y escribir datos en diversos formatos (CSV, Excel, bases de datos SQL, etc.), limpiar datos, transformarlos, fusionarlos, agruparlos y realizar análisis descriptivos.
    - **Ventaja para el TFG:** Indispensable para:
      - Leer los archivos CSV generados por *codigo_raton.py* o los CSV del sistema M3Display.
      - Procesar y limpiar los datos de trayectoria antes de su almacenamiento o análisis.

        - Convertir los datos recuperados de Supabase (que suelen venir como listas de diccionarios) en DataFrames para facilitar su manipulación y análisis.
        - Preparar los datos para las funciones de visualización y cálculo de métricas.
    - **NumPy (Numerical Python):**
        - **Descripción:** Es el paquete fundamental para la computación científica con Python. Proporciona soporte para arrays y matrices multidimensionales de gran tamaño, junto con una amplia colección de funciones matemáticas de alto nivel para operar sobre estos arrays. Muchas otras librerías científicas, incluyendo Pandas y SciPy, se basan en NumPy.
        - **Ventaja para el TFG:** Utilizado para:
            - Operaciones numéricas eficientes en los datos de trayectoria (ej. cálculos vectoriales, diferencias entre coordenadas para calcular distancias).
            - Generación de secuencias numéricas o arrays necesarios para algunos algoritmos de análisis o visualización.
            - Base para el cálculo de métricas cinemáticas que implican operaciones con arrays de coordenadas o tiempos.
    - **Matplotlib**
        - **Descripción:** Matplotlib es una librería de trazado 2D completa para Python que produce figuras de calidad de publicación en una variedad de formatos impresos y entornos interactivos. Ofrece un control muy detallado sobre todos los aspectos de una figura.
        - **Ventaja para el TFG:** Matplotlib es esencial para generar los gráficos estáticos de las trayectorias (X vs Y) que se muestran en los detalles de la sesión. GIF del movimiento de la mano.
    - **Pillow (PIL - Python Imaging Library Fork):**
        - **Descripción:** Es una librería potente para abrir, manipular y guardar muchos formatos de archivo de imagen diferentes. Ofrece capacidades de procesamiento de imágenes.

- **Ventaja para el TFG:**
    - Requerida para guardar las animaciones generadas por FuncAnimation en formato GIF.
    - Utilizada por el script *codigo_raton.py* (a través de PIL.ImageGrab) para capturar y guardar la imagen del canvas al finalizar la terapia.
    - Potencialmente útil en la aplicación Streamlit si se necesitara procesar las imágenes de los dibujos subidas por el usuario (ej. redimensionar, convertir formato), aunque en el alcance actual se enfoca más en su almacenamiento y visualización.

- **dotenv (python-dotenv):**
    - **Descripción:** Esta librería permite cargar variables de entorno desde un archivo .env al entorno de la aplicación Python. Esto es una práctica recomendada para gestionar configuraciones sensibles como claves de API o credenciales de base de datos, evitando codificarlas directamente en el script.
    - **Ventaja para el TFG:** Se utiliza para cargar de forma segura las variables SUPABASE_KEY y SUPABASE_URL desde un archivo .env local, manteniendo estas credenciales fuera del control de versiones y del código fuente principal.

Esta combinación de tecnologías proporciona una base sólida y flexible para el desarrollo de la aplicación, permitiendo desde la creación rápida de la interfaz hasta el análisis complejo de datos, todo dentro del ecosistema Python y con el apoyo de los servicios gestionados de Supabase.

### 5.2.2 DISEÑO DE LA BASE DE DATOS (SUPABASE/POSTGRESQL)

El diseño de la base de datos es un pilar fundamental del sistema, ya que debe almacenar de forma eficiente y estructurada toda la información relativa a los pacientes, terapeutas, tipos de terapia, las sesiones realizadas y los datos detallados de trayectoria de movimiento. Se ha utilizado el editor de tablas de Supabase para definir el esquema.

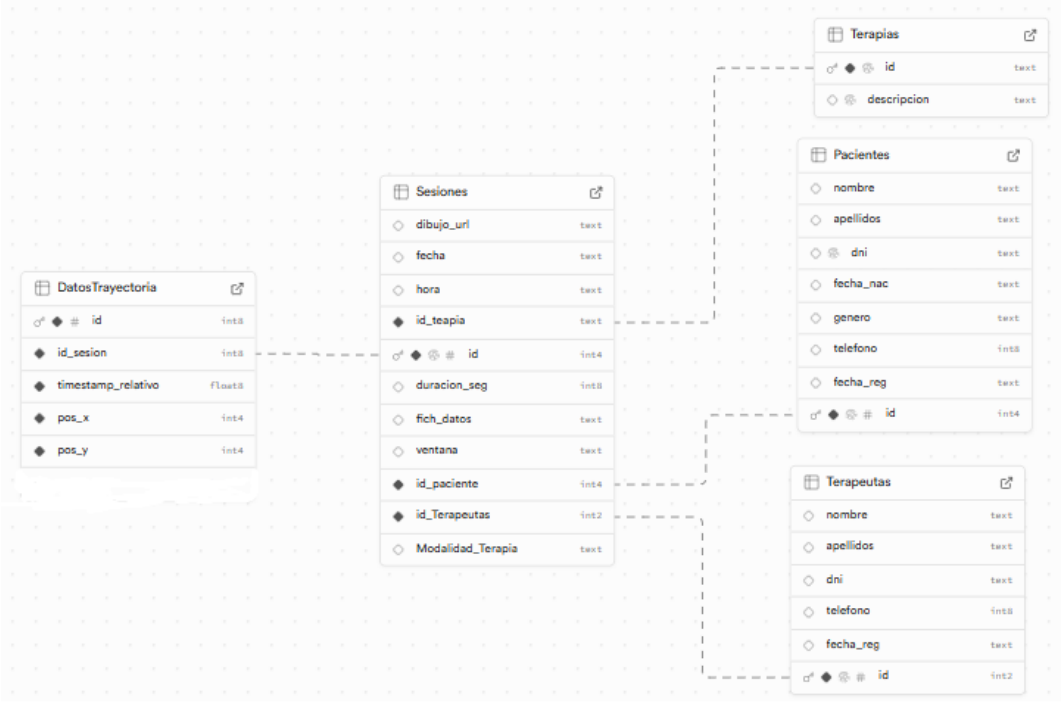


*Figura 8: Diagrama Entidad-Relación (DER) de la base de datos implementada en Supabase. Muestra las tablas Pacientes, Terapeutas, Terapias, Sesiones y DatosTrayectoria, sus columnas principales y las relaciones entre ellas. Fuente: Captura del esquema*

- A continuación, se detalla la estructura de cada tabla:
    - **Tabla: Pacientes**
        - **Propósito:** Almacenar la información demográfica y de registro de los pacientes que participan en las terapias.
        - **Columnas:**
            - id: int4 (Integer), Clave Primaria (PK), Autoincremental, NOT NULL. Identificador único del paciente.
            - nombre: text, NOT NULL. Nombre del paciente.
            - apellidos: text, NOT NULL. Apellidos del paciente.
            - dni: text, UNIQUE (opcional, pero recomendado si se usa como identificador secundario). Documento Nacional de Identidad o similar.
            - fecha_nac: text. Fecha de nacimiento del paciente.
            - genero: text. Género del paciente.

        - telefono: int8 (BigInt). Número de teléfono de contacto.
        - fecha_reg: text NOT NULL. Fecha de registro del paciente en el sistema.

- **Tabla: Terapeutas**
    - **Propósito:** Almacenar la información de los terapeutas que utilizan la aplicación y administran las sesiones.
    - **Columnas:**
        - id: int2 (SmallInt), PK, Autoincremental, NOT NULL. Identificador único del terapeuta.
        - nombre: text, NOT NULL. Nombre del terapeuta.
        - apellidos: text, NOT NULL. Apellidos del terapeuta.
        - dni: text, UNIQUE (opcional).
        - telefono: int8 (BigInt).
        - fecha_reg: text, NOT NULL. Fecha de registro del terapeuta.
- **Tabla: Terapias**
    - **Propósito:** Definir los tipos generales de terapia o ejercicios que se pueden realizar. Esto permite una categorización y, potencialmente, la asociación de parámetros específicos a cada tipo de terapia en el futuro.
    - **Columnas:**
        - id: text, PK, NOT NULL. Identificador textual único para el tipo de terapia (ej. "reloj", "sinusoide").
        - descripcion: text, NOT NULL. Nombre o descripción clara de la terapia general.
- **Tabla: Sesiones**
    - **Propósito:** Registrar cada instancia de una sesión de rehabilitación realizada por un paciente, bajo la supervisión de un terapeuta y asociada a un tipo de terapia. Contiene metadatos de la sesión y enlaces a datos más detallados.
    - **Columnas:**

- id: int4 (Integer), PK, Autoincremental, NOT NULL. Identificador único de la sesión.
- fecha: text, NOT NULL. Fecha en que se realizó la sesión.
- hora: text, NOT NULL. Hora de inicio de la sesión.
- id_paciente: int4 Clave Foránea (FK) a Pacientes(id), ON DELETE CASCADE NOT NULL.
- id_Terapeutas: int2 (SmallInt), FK a Terapeutas(id), ON DELETE CASCADE (o RESTRICT). NOT NULL.
- id_teapia: text, FK a Terapias(id) ON DELETE SET NULL (o RESTRICT). NOT NULL. Referencia al tipo de terapia o ejercicio realizado.
- Modalidad_Terapia: text. Descripción adicional de la modalidad (ej. "Ratón - Terapia del Reloj", "M3Display (Sim) - Seguimiento Curvo").
- duracion_seg: int8 (BigInt). Duración de la sesión en segundos (si se registra).
- dibujo_url: text. URL al archivo de imagen del dibujo (si aplica) almacenado en Supabase Storage.
- fich_datos: text. URL al archivo CSV de trayectoria (si aplica) almacenado en Supabase Storage.
- ventana: text. Entorno o aplicación utilizada para la sesión (ej. "Tkinter Mouse Tracker", "Simulador M3Display v1.0").

- **Tabla: DatosTrayectoria**
  - **Propósito:** Almacenar la secuencia detallada de puntos numéricos (coordenadas y tiempo) que componen la trayectoria de movimiento realizada durante una sesión. Esta tabla permite un análisis fino del rendimiento motor.
  - **Columnas:**
    - id: int8 (BigInt), PK, Autoincremental, NOT NULL. Identificador único del punto de trayectoria.
    - id_sesion: int4 (Integer), FK a Sesiones(id), ON DELETE CASCADE. NOT NULL. Vincula el punto a la sesión correspondiente.

- pos_x: int4 (Integer), NOT NULL. Coordenada X del punto de la trayectoria.
- pos_y: int4 (Integer), NOT NULL. Coordenada Y del punto de la trayectoria.

- **Relaciones Principales:**
  - Un Paciente puede tener muchas Sesiones. Una Sesión pertenece a un único Paciente. (Relación 1:N)
  - Un Terapeuta puede supervisar muchas Sesiones. Una Sesión es supervisada por un único Terapeuta. (Relación 1:N)
  - Una Terapia (tipo general) puede estar asociada a muchas Sesiones. Una Sesión se asocia a un tipo de Terapia. (Relación 1:N)
  - Una Sesión puede tener muchos puntos de DatosTrayectoria. Cada punto de DatosTrayectoria pertenece a una única sesión. (Relación 1:N)
- **Consideraciones de Diseño:**
  - La elección de text para campos de fecha y hora en algunas tablas (Pacientes, Terapeutas, Sesiones) es una decisión que simplifica la entrada desde Python si no se realiza una conversión estricta a tipos date/time de PostgreSQL. Sin embargo, para consultas y ordenamientos basados en fecha/hora, los tipos nativos date, time son preferibles y más eficientes. En la aplicación Streamlit se realizan conversiones para la visualización y el ordenamiento.
  - La tabla DatosTrayectoria es crucial para el análisis cuantitativo. Almacenar cada punto como una fila permite consultas flexibles y cálculos de métricas. Se ha optado por int4 para pos_x y pos_y asumiendo coordenadas de pantalla o canvas que suelen ser enteras; si fueran flotantes, float8 sería más apropiado.
  - Se ha definido ON DELETE CASCADE para las FK de id_paciente, id_Terapeutas en Sesiones, y para id_sesion en DatosTrayectoria. Esto significa que, si se elimina un paciente o un terapeuta, todas sus sesiones asociadas (y los datos de trayectoria de esas sesiones) también se eliminarán. Si se elimina una sesión, todos sus puntos de trayectoria se eliminarán. Esta es una decisión de diseño para mantener la integridad

referencial; una alternativa podría ser ON DELETE RESTRICT para evitar borrados accidentales u ON DELETE SET NULL si se quisiera mantener el registro de la sesión anonimizando el paciente/terapeuta (lo cual complicaría la lógica de la aplicación).

- Para id_teapia en Sesiones referenciando Terapias(id), se sugiere ON DELETE SET NULL para que, si se elimina un tipo de terapia general, las sesiones que usaban ese tipo no se borren, sino que simplemente pierdan esa referencia específica, manteniendo el id_teapia textual que también se guarda.

### 5.2.3 DISEÑO DEL ALMACENAMIENTO DE ARCHIVOS (SUPABASE STORAGE)

Supabase Storage se utiliza para almacenar archivos binarios asociados a las sesiones.

- **Depósito:** Se utiliza un único déposito principal denominado sesiones.
- **Políticas de Acceso al Depósito:** Se configura para permitir el acceso público de lectura (SELECT) a los archivos dentro de las carpetas public/, ya que las URLs generadas son públicas. Las operaciones de subida (INSERT), actualización (UPDATE) y borrado (DELETE) de archivos se gestionan a través del backend de la aplicación Python utilizando las claves de servicio de Supabase, que tienen permisos elevados, o mediante políticas de Storage si se configuran para usuarios autenticados específicos.
- **Estructura de Carpetas:** Para organizar los archivos, se utiliza la siguiente estructura dentro del depósito sesiones:
  - public/sesiones_dibujos/: Para almacenar las imágenes (.png, .jpg, etc.) de los dibujos o capturas de pantalla de las terapias. Los nombres de archivo incluyen identificadores de paciente, terapeuta, tipo de terapia y timestamp para asegurar unicidad.
  - public/sesiones_trayectorias_csv/: Para almacenar los archivos CSV originales que contienen los datos numéricos de trayectoria, como una copia de seguridad o referencia. Los nombres de archivo siguen una convención similar.
- **Generación de URLs:** La aplicación Python, tras subir un archivo, obtiene la URL pública de acceso a dicho archivo desde Supabase Storage. Estas URLs son las que se almacenan en las columnas dibujo_url y fich_datos de la tabla Sesiones.

### 5.2.4 DISEÑO DE LA LÓGICA DE LA APLICACIÓN Y MÓDULOS DE ANALISIS

La lógica principal de la aplicación reside en el código Python ejecutado por Streamlit.

- **Funciones de Interacción con Supabase:** Se han desarrollado funciones reutilizables para encapsular las operaciones CRUD y otras interacciones con Supabase:
    - check_auth(): Verifica si existe un token de sesión activo.
    - handle_auth_error(): Gestiona errores comunes de autenticación.
    - obtener_datos(tabla): Recupera todos los registros de una tabla.
    - eliminar_datos(tabla, id_registro): Elimina un registro específico.
    - iniciar_sesion(email, password): Autentica al usuario.
    - registrar_usuario(email, password): Registra un nuevo usuario.
    - mostrar_detalles_sesion(sesion): Recupera información relacionada de Pacientes, Terapeutas y Terapias para una sesión dada.
    - agregar_registro(tabla, data): Inserta un nuevo registro en una tabla.
    - upload_file_to_storage(...): Sube un archivo local a Supabase Storage.
    - obtener_opciones(...): Recupera datos para rellenar selectores (dropdowns).
    - guardar_trayectoria_csv_a_supabase(id_sesion, csv_filepath): Procesa un CSV y guarda los puntos en DatosTrayectoria.
    - obtener_trayectoria_supabase(id_sesion): Recupera los puntos de trayectoria de una sesión.
- **Módulo de Análisis de Trayectoria:**
    - **Entrada:** Los datos de trayectoria se recuperan de la tabla DatosTrayectoria como un DataFrame de Pandas.
    - **Filtrado:** La función filtra_trayectoria(datos_x, datos_y, filter='median', size_f=5) aplica un filtro de mediana móvil (configurable a media) a las coordenadas X e Y para suavizar el ruido inherente a la captura. El tamaño de la ventana es adaptable.

- **Cálculo de Métricas:** Se han implementado funciones para calcular:
    - calcular_duracion(df_trayectoria): Duración total a partir de los timestamps relativos.
    - calcular_longitud_recorrida(df_trayectoria): Suma de las distancias euclidianas entre puntos consecutivos.
    - calcular_velocidad_media(df_trayectoria): Longitud total dividida por la duración total.
- **Generación de Visualizaciones:**
    - **Gráfico Estático de Trayectoria:** Se utiliza Matplotlib para generar un gráfico 2D (X vs Y) de la trayectoria (original o filtrada), mostrando puntos de inicio y fin.
    - **Animación GIF:** La función animar_y_guardar(data_x, data_y, ...) utiliza matplotlib y PIllowWriter para crear un GIF animado que reproduce la secuencia del movimiento.
    - **Gráficos de Progreso Longitudinal:** Para un paciente seleccionado, se recuperan las métricas calculadas de todas sus sesiones, se ordenan cronológicamente y se utilizan gráficos de líneas de Streamlit (st.line_chart) para visualizar la evolución de cada métrica a lo largo del tiempo.

### 5.2.5 DISEÑO DE LA SEGURIDAD

La seguridad de los datos clínicos es una prioridad.

- **Autenticación:** Se utiliza el sistema de autenticación integrado de Supabase (basado en GoTrue), que maneja el registro con confirmación por email, inicio de sesión con email/contraseña, y la gestión de tokens de sesión (JWT). La aplicación Streamlit gestiona el estado de la sesión del usuario.
- **Autorización (Row Level Security - RLS):** Se han implementado políticas RLS para todas las tablas. Las políticas actuales permieten a cualquier usuario autentificado (auth.role() = 'authenticated') realizar operaciones de SELECT, INSERT, UPDATE y DELETE sobre todos los datos.
    - **Justificación:** Para el alcance de este TFG, y asumiendo un entorno donde todos los usuarios autenticados son terapeutas de confianza con acceso al sistema completo, esta política simplifica la implementación.

- **Consideración Futura:** Para un sistema en producción real o con múltiples roles, estas políticas RLS deberían ser mucho más granulares. Por ejemplo:
    - Un terapeuta solo podría ver/modificar los pacientes y sesiones que le han sido asignados o que ha creado.
    - La tabla Terapeutas podría tener permisos más restrictivos para la modificación.
    - auth.uid() (el ID del usuario autenticado) se usaría en las políticas para filtrar los datos a los que cada usuario tiene acceso.

- **Seguridad de Conexión:** La comunicación con Supabase (tanto para la base de datos como para el storage) se realiza a través de HTTPS, asegurando el cifrado de los datos en tránsito.
- **Gestión de Credenciales:** Las claves de Supabase (URL y anon key o service_role key si se usara directamente en el backend, aunque aquí se usa el cliente Python con autenticación de usuario) se gestionan mediante variables de entorno (.env file), no se codifican directamente en el script.

### 5.2.6 DISEÑO DEL SCRIPT DE CAPTURA

El script Tkinter proporcionado se ha modificado conceptualmente para:

- **Registrar Timestamps Relativos:** Además de las coordenadas X e Y, se registra el tiempo transcurrido desde el inicio del trazado para cada punto.
- **Guardar Datos Numéricos en CSV:** Al finalizar la captura, además de la imagen PNG, el script guarda la secuencia de puntos (timestamp, X, Y) en un archivo CSV con un formato definido (separado por punto y coma, con cabecera Timestamp;PosX;PosY). El nombre del archivo CSV se deriva del nombre del archivo PNG para facilitar la asociación.
- **Mantenimiento de Funcionalidad Original:** Se mantiene la capacidad de dibujar sobre figuras objetivo (sinusoide, reloj) y guardar la imagen visual del trazado.

## 5.3 DISEÑO DE LA INTERFAZ DE USUARIO (UI) Y FLUJO DE NAVEGACIÓN

La interfaz de usuario se ha desarrollado utilizando Streamlit, buscando un diseño limpio, funcional y fácil de usar para los terapeutas.

### 5.3.1 PRINCIPIOS DE DISEÑO DE UI/UX

- **Simplicidad:** Minimizar la complejidad visual y de interacción.
- **Consistencia:** Utilizar patrones de diseño y elementos de UI consistentes a lo largo de la aplicación.
- **Claridad:** Presentar la información de forma clara y comprensible.
- **Feedback al Usuario:** Proporcionar mensajes sobre el estado de las operaciones (éxito, error, carga).
- **Eficiencia:** Permitir a los terapeutas realizar sus tareas con el menor número de pasos posible.

### 5.3.2 ESTRUCTURA GENERAL DE LA APLICACIÓN Y NAVEGACIÓN Y DISEÑO DE LAS PANTALLAS PRINCIPALES

La aplicación se estructura en torno a una barra lateral para la autenticación y la navegación principal, y un área de contenido principal donde se muestran las diferentes vistas.

Se describen las vistas clave implementadas con Streamlit, ilustradas con capturas de pantalla de la aplicación funcional.

- **Vista de Autentificación (Login/Registro):**

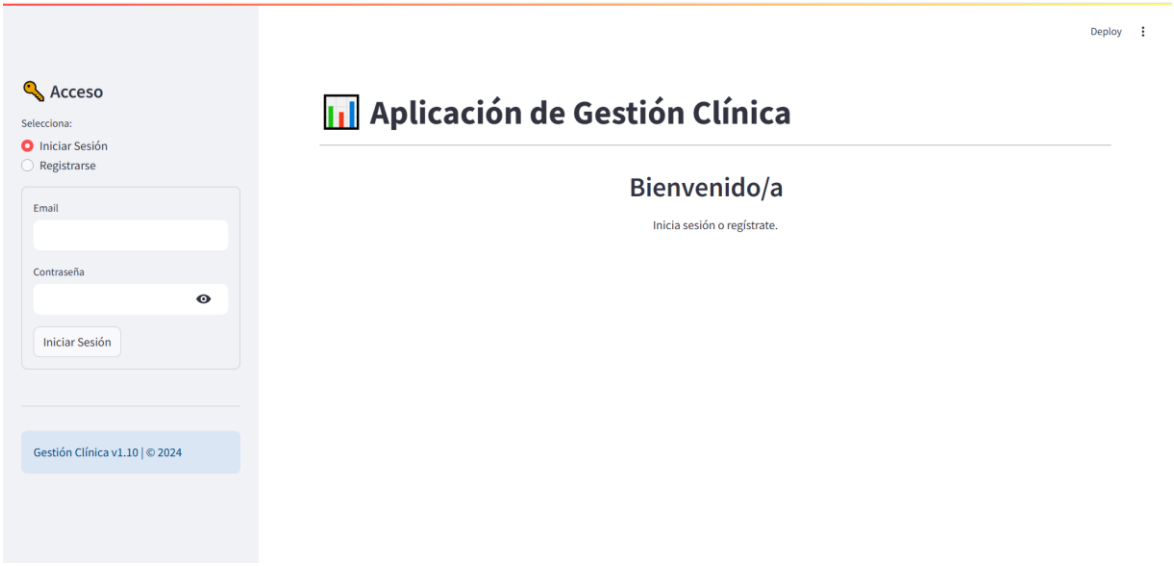


*Figura 9: Pantalla de Inicio de Sesión/Registro Fuente: Aplicación TFG*

Descripción de la Figura 9: La barra lateral presenta opciones para “Iniciar Sesión” o “Registrarse”. Cada opción despliega un formulario con campos para email y contraseña, y un botón de envío. Se proporcionan mensajes de estado (éxito, error)

- **Vista Principal (Autentificado):**


*Figura 10: Vista principal con opciones Fuente: Aplicación TFG*

Descripción de la Figura 10: Una vez autentificado, el usuario ve panel central con dos secciones principales: “Gestión de Tablas Generales” (Botones para Pacientes, Terapeutas y Terapias) y “Gestión de Sesiones” (Botones para Historial, Nueva sesión con ratón, Nueva sesión con M3Display (Simulada), y Análisis de Progreso del Paciente).

- **Vista de Gestión de Tablas (Pacientes, Terapeutas, Terapias):**

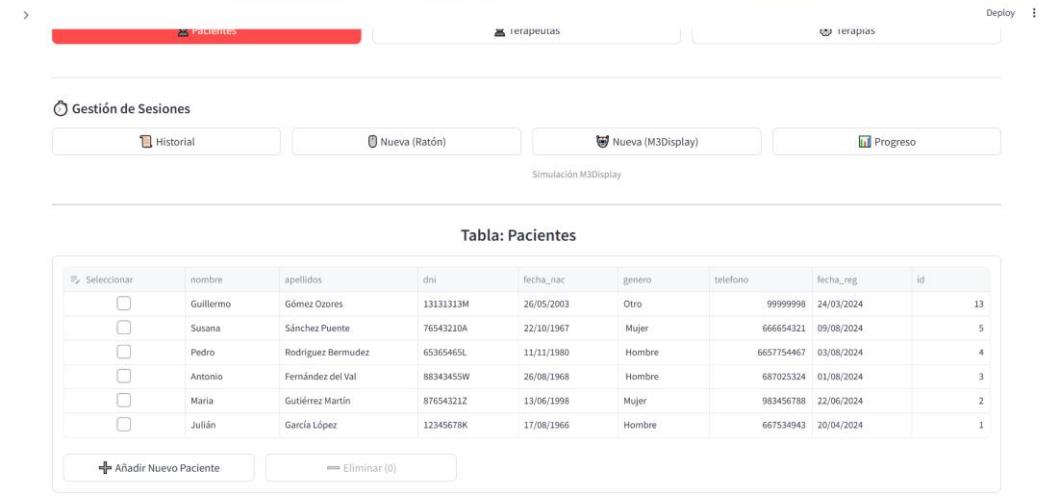


*Figura 11: Vista de Gestión de la Tabla Pacientes Fuente: Aplicación TFG*

Descripción de la Figura 11: Al seleccionar una tabla general, se muestra una vista que presenta los datos en un formato tabular. Incluye una columna “Seleccionar” (*checkbox*) para operaciones de eliminación. Las columnas de datos están configuradas para visualización, pero deshabilitadas para edición directa. Se proporcionan botones para “Añadir Nuevo [Elemento]” y “Eliminar Seleccionados”.

- **Vista de Formularios de Adición (Pacientes, Terapeutas, Terapias, Sesión Manual):**

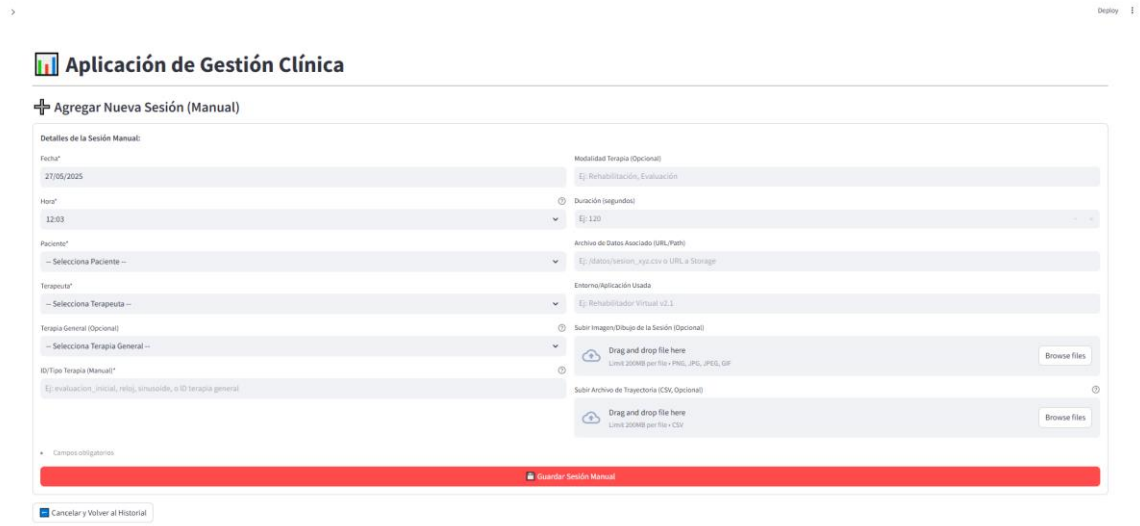


*Figura 12: Formulario de Adición de Nueva Sesión Manual Fuente: Aplicación TFG*

Descripción de la Figura 12: Formularios estructurados. Utiliza widgets de Streamlit para la entrada de datos. Se realizan validaciones básicas, campos obligatorios, y se proporcionan mensajes de feedback.

- **Vista de Historial de Sesiones:**

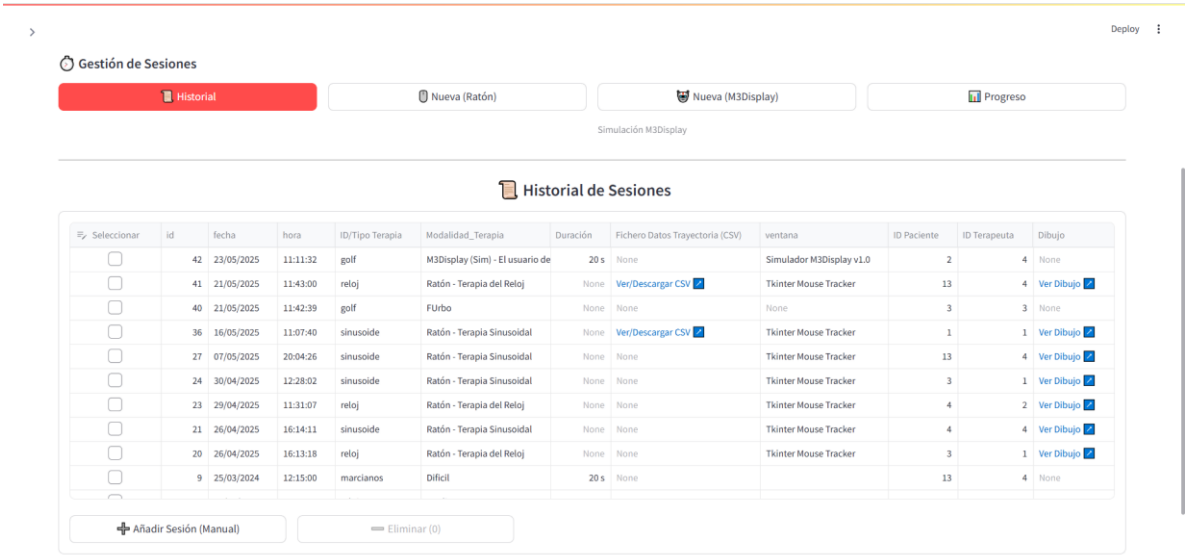


*Figura 13: Vista de Historial de Sesiones Fuente: Aplicación TFG*

Descripción de la Figura 13: Similar a la gestión de tablas generales, muestra las sesiones registradas con columnas configuradas para visualización (IDs, fechas, horas, enlaces a dibujos/CSV). Permite la selección de una única sesión para ver detalles o múltiples para eliminar. Incluye un botón para “Añadir Sesión Manual”.

- Vista de Detalles de sesión y Análisis de Sesión Seleccionada:


*Figura 14:Vista de detalles y Análisis de una Sesión Seleccionada Fuente: Aplicación TFG*

Descripción de la figura 14: Al seleccionar una sesión del historial la vista se expande mostrando:

- Información contextual de la sesión (Paciente, Terapeuta, Tipo de Terapia asociados, recuperados mediante consultas a Supabase).

- La imagen del dibujo de la sesión (si existe, cargada desde la URL de Supabase Storage).
- Una sección de "Análisis de la Trayectoria (Datos de BD)":
  - Calcula y muestra las métricas cinemáticas (Duración, Longitud, Velocidad Media).
  - Genera y muestra un gráfico estático (Matplotlib) de la trayectoria (X vs Y) con puntos de inicio/fin.
- Una sección para "Generar/Descargar GIF Animado":
  - Permite generar un GIF a partir de los datos de trayectoria de la BD (originales o filtrados).
  - Muestra el GIF generado y ofrece un botón de descarga.
  - También ofrece una opción para subir un archivo CSV y generar el GIF a partir de él.
- Una sección para "Archivos Adjuntos" que enlaza al CSV de trayectoria si fue subido a Storage.

- **Vista de Captura de Sesión con Ratón:**

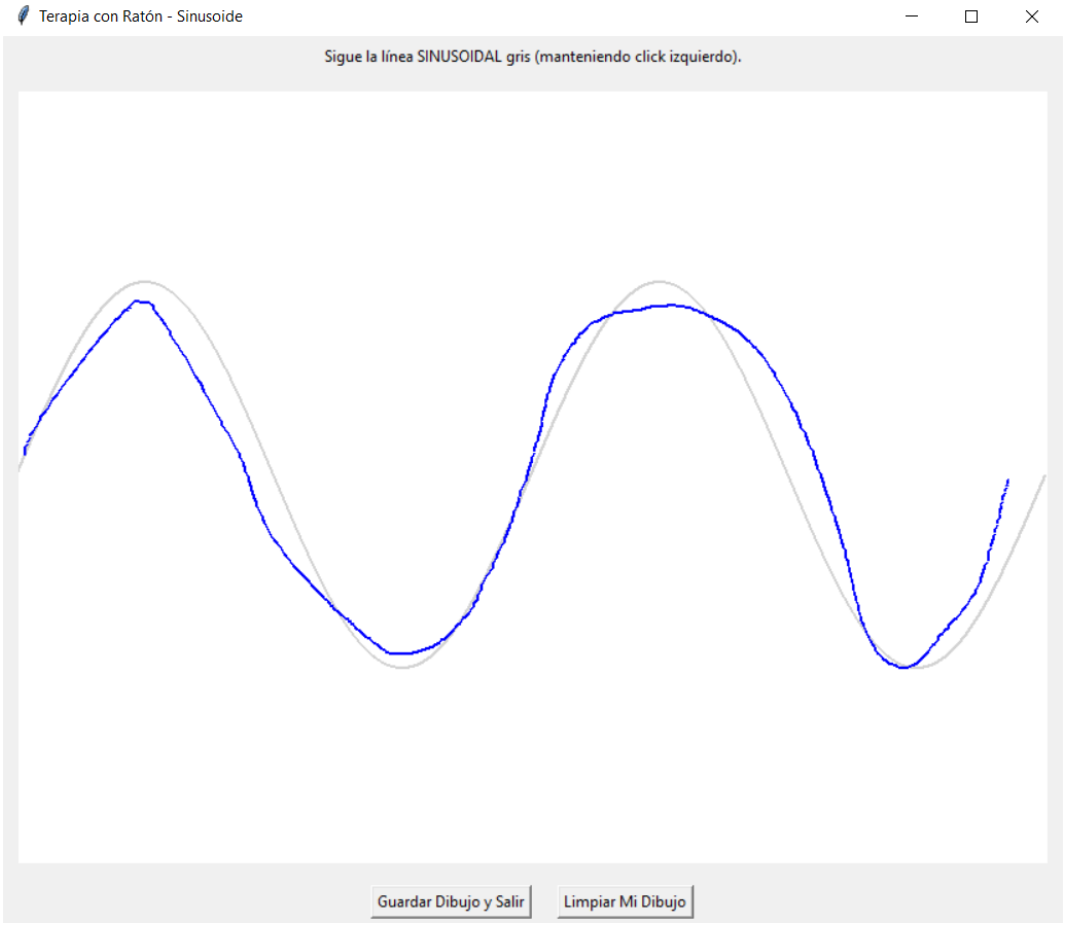


*Figura 15: Captura de Sesión con Ratón Fuente: Aplicación TFG*

Descripción de la Figura 15: Se describe brevemente la interacción: el terapeuta selecciona parámetros en Streamlit, se lanza el script Python/Tkinter en una ventana separada, el paciente/usuario realiza la tarea, y al cerrar y guardar, se generan los archivos .png y .csv que la aplicación Streamlit luego procesa.

- Vista de Análisis de Progreso del Paciente:


*Figura 16: Vista de Análisis de Progreso del Paciente Fuente: Aplicación TFG*

Descripción de la Figura 16: Permite al terapeuta seleccionar un paciente de una lista. Una vez seleccionado, la aplicación recupera todas sus sesiones, calcula las métricas clave para cada una (Duración, Longitud, Velocidad Media), y presenta gráficos de líneas mostrando la evolución de estas métricas a lo largo del tiempo (o número de sesión). También se muestra una tabla resumen con los datos de progreso.

### 5.3.3 COMPONENTES REUTILIZABES Y ESTILO VISUAL

- Se han creado funciones Python reutilizables para la interacción con Supabase (ej. `obtener_datos`, `agregar_registro`), formateo de datos para selectores (ej. `format_persona`, `format_terapia`), y para el análisis y visualización (ej. `filtra_trayectoria`, `animar_y_guardar`, funciones de cálculo de métricas).
- El estilo visual se mantiene coherente con las capacidades por defecto de Streamlit, buscando una apariencia limpia y profesional, utilizando contenedores (`st.container(border=True)`), columnas (`st.columns`) para la disposición y elementos de feedback como st.success, st.error, st.info o st.spinner.

# CAPÍTULO 6. IMPLEMENTACIÓN

Este capítulo describe en detalle el proceso de desarrollo y construcción de la aplicación web para la gestión y análisis de datos de rehabilitación con Realidad Aumentada. Se profundiza en la configuración del entorno de desarrollo, la estructura del código fuente, la implementación pormenorizada de las funcionalidades clave del sistema, la configuración específica de la base de datos y el almacenamiento en Supabase, y un análisis más extenso de los retos técnicos encontrados y las soluciones adoptadas. El objetivo es proporcionar una visión exhaustiva de cómo los conceptos de diseño (Capítulo 5) se materializaron en una aplicación funcional, destacando las decisiones de implementación cruciales tomadas con el conjunto de tecnologías seleccionado (Capítulo 3).

## 6.1 ENTORNO DE DESARROLLO, ESTRUCTURA DEL PROYECTO Y DEPENDENCIAS

### 6.1.1 CONFIGURACION DEL ENTORNO DE DESARROLLO

El desarrollo de la aplicación se llevó a cabo en un entorno basado en Python, utilizando la versión 3.11.5. Para la gestión de paquetes y dependencias, se optó por el uso de entornos virtuales, una práctica estándar en el desarrollo con Python que asegura el aislamiento de las librerías del proyecto y la reproducibilidad del entorno en diferentes máquinas. Todas las dependencias se registraron en un archivo requirements.txt, facilitando su instalación mediante pip install -r requirements.txt.

Visual Studio Code (VS Code) fue el Entorno de Desarrollo Integrado (IDE) principal, seleccionado por su robusto soporte para Python, sus capacidades de depuración, la integración nativa con Git y su amplio ecosistema de extensiones que agilizan el desarrollo.

La configuración de las credenciales sensibles para la conexión con Supabase, nuestra solución de backend y base de datos PostgreSQL en la nube, (URL del proyecto y clave API anónima) se manejó mediante variables de entorno. Se utilizó la librería python-dotenv para cargar estas variables desde un archivo .env local, el cual fue excluido del sistema de control de versiones (Git) para proteger la información confidencial. Esta práctica es fundamental para la seguridad y permite diferentes configuraciones para entornos de desarrollo y producción.

### 6.1.2 ORGANIZACIÓN DEL CÓDIGO FUENTE Y MODULARIDAD

El núcleo de la aplicación web reside en un script principal de Python, convencionalmente denominado app.py, que es ejecutado por el servidor de Streamlit, el framework que renderiza la interfaz web interactiva. A pesar de que Streamlit fomenta un estilo de scripting lineal para la creación rápida de interfaces, se ha puesto un énfasis considerable en la modularidad para mejorar la legibilidad, mantenibilidad y la capacidad de prueba del código. Esto se ha logrado mediante la definición de numerosas funciones Python, cada una con una responsabilidad bien definida.

La estructura funcional se puede agrupar de la siguiente manera:

- **Funciones de Configuración e Inicialización:** Incluyen la configuración de la página de Streamlit (st.set_page_config), la carga de variables de entorno, la inicialización del cliente Supabase y la creación de directorios temporales.
- **Funciones de Autenticación y Gestión de Sesión de Usuario:** Agrupan toda la lógica relacionada con el registro (registrar_usuario), inicio de sesión (iniciar_sesion), cierre de sesión, verificación del estado de autenticación (check_auth), y manejo de errores de sesión (handle_auth_error). Estas funciones interactúan directamente con el servicio Supabase Auth.
- **Funciones de Interacción con la Base de Datos (CRUD):** Para cada entidad principal del sistema (Pacientes, Terapeutas, Sesiones, Terapias, DatosTrayectoria) se han creado funciones específicas para las operaciones de Crear, Leer, Actualizar y Eliminar (ej., obtener_datos, agregar_registro, eliminar_datos). Estas funciones encapsulan las llamadas al cliente Supabase-py, manejan las respuestas y los posibles errores de la API
- **Funciones de Interacción con Supabase Storage:** Funciones como upload_file_to_storage se encargan de la subida de archivos (imágenes de dibujos, CSVs de trayectoria) al bucket designado en Supabase, gestionando el tipo MIME y la obtención de la URL pública.
- **Funciones de Lógica de Negocio y Orquestación de Vistas:** Controlan el flujo de la aplicación, qué vista se muestra al usuario (st.session_state['current_view']), y cómo se transita entre ellas. Estas funciones llaman a las funciones CRUD y de análisis según sea necesario.
- **Funciones de Análisis de Datos de Trayectoria:** Este es un módulo crítico que incluye:

- guardar_trayectoria_csv_a_supabase: Procesa el archivo CSV de trayectoria (generado por código_raton.py o subido manualmente) y guarda cada punto en la tabla DatosTrayectoria.
- obtener_trayectoria_supabase: Recupera la secuencia de puntos de una sesión específica desde Supabase.
- filtra_trayectoria: Aplica un filtro de mediana móvil a los datos de coordenadas para suavizar el ruido.
- calcular_duracion, calcular_longitud_recorrida, calcular_velocidad_media: Calculan las métricas cinemáticas básicas.

- **Funciones de Visualización:**
  - animar_y_guardar: Genera la animación GIF de la trayectoria utilizando Matplotlib.
  - Funciones auxiliares para preparar y mostrar gráficos estáticos de trayectoria y gráficos de progreso de métricas.
- **Funciones de Utilidad:** Incluyen funciones para formatear datos para su visualización en selectores (ej., format_persona, format_terapia), manejo de fechas y horas, y otras tareas auxiliares.

El script *codigo_raton.py* (basado en Tkinter) se mantiene como un archivo Python separado e independiente, invocado como un subproceso. Se creó un directorio temp_files en la raíz del proyecto, el cual también fue añadido al archivo .gitignore para evitar que los archivos temporales generados durante la ejecución sean versionados.

### 6.1.3 GESTIÓN DE DEPENDENCIAS

Todas las librerías externas necesarias para el proyecto se especificaron en un archivo requirements.txt. Esto permite recrear el entorno de ejecución de forma consistente en diferentes máquinas o al desplegar la aplicación. Las dependencias clave incluyen:

- **streamlit:** Versión 1.42.0 Framework para la UI.
- **supabase**: Versión 2.13.0. Cliente Python para Supabase.
- **pandas**: Versión 2.0.3. Para manipulación de datos.
- **numpy**: Versión 1.24.3. Para cálculo numérico.
- **matplotlib**: Versión 3.10.1. Para gráficos y animaciones.
- **Pillow**: Versión 9.4.0. Para manejo de imágenes y guardado de GIFs.
- **python-dotenv**: Versión 0.21.0. Para variables de entorno.

- **pyarrow**: Versión 11.0.0. Para optimización de DataFrames en Streamlit.

## 6.2 IMPLEMENTACIÓN DETALLADA DE FUNCIONALIDADES CLAVE

A continuación, se describe la implementación de las funcionalidades más relevantes de la aplicación, destacando las decisiones técnicas y cómo se utilizaron las herramientas seleccionadas.

### 6.2.1 SISTEMA DE AUTENTIFICACIÓN DE USUARIOS Y GESTIÓN DE SESIONES

La seguridad del acceso a la aplicación es primordial. Se implementó utilizando Supabase Auth.

- **Registro y Confirmación:** El flujo de registro (registrar_usuario) utiliza supabase.auth.sign_up(). Supabase automáticamente envía un correo electrónico de confirmación al nuevo usuario. Hasta que el correo no es confirmado, el usuario no puede iniciar sesión (Supabase devuelve un error "Email not confirmed").
- **Inicio de Sesión:** El inicio de sesión (iniciar_sesion) se realiza con supabase.auth.sign_in_with_password(). Si las credenciales son válidas y el email ha sido confirmado, Supabase devuelve un objeto Session que contiene el access_token y el refresh_token, además de un objeto User. Estos tokens se almacenan en st.session_state de Streamlit. El access_token (JWT) tiene una vida corta (ej. 1 hora por defecto en Supabase) y se utiliza para autenticar las peticiones a la base de datos y storage. El refresh_token tiene una vida más larga y se usa para obtener un nuevo access_token cuando este expira, sin necesidad de que el usuario reintroduzca sus credenciales.
- **Mantenimiento de la Sesión en Streamlit:** Antes de cada operación que requiere autenticación (prácticamente todas las llamadas a Supabase), la función check_auth() verifica la presencia de los tokens en st.session_state. Las funciones que llaman a Supabase utilizan supabase.auth.set_session(st.session_state['access_token'], st.session_state['refresh_token']) para asegurar que el cliente Supabase está utilizando la sesión activa del usuario. Esto es crucial porque la instancia del cliente Supabase es global, pero la sesión es por usuario.

- **Manejo de Expiración de Sesión:** La función handle_auth_error(e) se diseñó para interceptar excepciones de Supabase que indican una sesión inválida o expirada (ej., errores JWT). Cuando se detecta, limpia el st.session_state de las credenciales, muestra un mensaje al usuario y fuerza un st.rerun() que redirige a la vista de login.
- **Cierre de Sesión:** El botón de "Cerrar Sesión" invoca supabase.auth.sign_out(), que invalida los tokens en el servidor de Supabase, y luego limpia el st.session_state localmente, redirigiendo al login.

### 6.2.2 IMPLEMENTACIÓN DEL MÓDULO DE GESTIÓN DE ENTIDADES (PACIENTES, TERAPEUTAS, TERAPIAS)

La gestión de estas entidades fundamentales sigue un patrón CRUD (Crear, Leer, Actualizar, Eliminar), implementado con formularios de Streamlit y funciones Python que interactúan con Supabase.

- **Visualización y Listado (obtener_datos):** Se utiliza st.data_editor para mostrar los datos. Esta elección permite una visualización tabular interactiva y la funcionalidad de selección mediante checkboxes. Se prestó atención a la configuración de columnas (column_config) para formatear fechas, horas, números y para deshabilitar la edición directa en la tabla, ya que las modificaciones se gestionan mediante formularios o acciones específicas. Se implementó un manejo de errores para ArrowTypeError que puede surgir si los datos de una columna tienen tipos mixtos o inesperados, ofreciendo un retroceso a st.dataframe (menos interactivo, pero más robusto ante tipos de datos inconsistentes) para asegurar que el usuario al menos pueda ver los datos. Los datos se ordenan (generalmente por ID descendente) antes de mostrarlos.
- **Creación de Nuevos Registros (agregar_registro y Formularios):** Para cada entidad, se creó una vista de formulario separada (ej. st.session_state['current_view'] = 'add_patient'). Estos formularios utilizan widgets de Streamlit (st.text_input, st.date_input, etc.) para la entrada de datos. Se realizan validaciones básicas en el frontend (ej. campos obligatorios). Al enviar el formulario, los datos se recogen, se empaquetan en un diccionario y se pasan a la función genérica agregar_registro(tabla, data). Esta función realiza la inserción en la tabla de Supabase correspondiente. Se proporciona feedback al usuario (éxito con st.balloons(), o mensajes de error detallados devueltos por la API de Supabase, como violaciones de claves únicas o foráneas).

- **Eliminación de Registros (eliminar_datos):** La interfaz st.data_editor permite al usuario seleccionar múltiples filas. Los IDs de los registros seleccionados se pasan a la función eliminar_datos(tabla, id_registro) que se invoca iterativamente. Se implementó una barra de progreso y mensajes de estado para operaciones de eliminación masiva. Se consideraron los errores comunes, como intentar eliminar un registro que tiene otros registros dependientes (violación de clave foránea con código '23503'), informando al usuario. La eliminación de la tabla Terapias (cuyo id es text) requirió asegurar que la función eliminar_datos (que internamente espera un int para el id en su lógica original) no falle catastróficamente, aunque la eliminación directa de Terapias con ID textual no funcionaría con la lógica original de conversión a int de eliminar_datos sin una adaptación específica para esa tabla.

### 6.2.3 IMPLEMENTACIÓN DEL MÓDULO DE SESIONES DE REHABILITACIÓN

Este módulo es central para la aplicación, ya que gestiona cómo se registran las sesiones de terapia, cómo se asocian a los pacientes y terapeutas, y cómo se manejan los datos de rendimiento y los archivos multimedia asociados. Se han implementado tres flujos principales para el registro de sesiones:

- **Sesión con Captura de Movimiento mediante el Script *codigo_raton.py*:** Este flujo permite al terapeuta iniciar una sesión interactiva donde los datos de movimiento se capturan en tiempo real utilizando el script auxiliar *codigo_raton.py*.

    1. **Configuración e Invocación:** Desde la interfaz de Streamlit, el terapeuta selecciona el paciente, el terapeuta supervisor y el tipo de terapia de ratón específica (Reloj o Sinusoide) a través de un formulario dedicado. Al confirmar, la aplicación Streamlit construye un comando para ejecutar el script *codigo_raton.py*. Este script se invoca como un subproceso utilizando subprocess.run(). Se le pasan argumentos de línea de comandos para indicar el tipo de figura objetivo (--target) y la ruta de salida (--output) donde el script guardará los archivos generados.
    2. **Ejecución de la Terapia Externa:** Se abre una ventana de Tkinter donde el paciente (o el usuario de prueba) realiza la tarea de seguimiento del cursor. El script *codigo_raton.py* (previamente modificado) registra la trayectoria como una secuencia de puntos (timestamp,

x, y) y, al finalizar, guarda dos archivos en el directorio temporal TEMP_DIR de la aplicación Streamlit: una imagen (.png) del trazado final sobre el objetivo y un archivo de datos de trayectoria (.csv) con los puntos numéricos.

3. **Procesamiento Post-Captura en Streamlit:** Una vez que el script *codigo_raton.py* finaliza y devuelve el control a Streamlit, la aplicación verifica la correcta generación y el tamaño de los archivos .png y .csv.

4. **Almacenamiento de Archivos en Supabase Storage:**

    - La imagen del dibujo (.png) se sube al bucket sesiones de Supabase Storage, dentro de la carpeta public/sesiones_dibujos/, utilizando la función upload_file_to_storage. Se genera un nombre de archivo único que incluye identificadores y un timestamp.

    - De manera similar, el archivo CSV de trayectoria también se sube al mismo bucket, pero a la carpeta public/sesiones_trayectorias_csv/.
      La función upload_file_to_storage devuelve las URLs públicas de estos archivos.

5. **Persistencia de Metadatos de Sesión:** Se crea un nuevo registro en la tabla Sesiones de Supabase. Este registro incluye los IDs del paciente y terapeuta, la fecha y hora, el id_teapia (ej. “reloj”, “sinusoide”), la Modalidad_Terapia, la URL del dibujo (dibujo_url) y la URL del archivo CSV de la trayectoria (fich_datos).

6. **Almacenamiento de Datos de Trayectoria Numéricos:** El archivo CSV de trayectoria se procesa mediante la función guardar_trayectoria_csv_a_supabase. Esta función lee el CSV con Pandas, normaliza los datos si es necesario y luego inserta cada punto de la trayectoria (timestamp_relativo, pos_x, pos_y) como una fila individual en la tabla de DatosTrayectoria de Supabase, vinculada al id_sesion de la sesión recién creada. Esta inserción se realiza en lote para mayor eficacia.

7. **Limpieza:** Finalmente, los archivos temporales .png y .csv generados localmente en TEMP_DIR son eliminados.

- **Registro de Sesión Manual (con Posibilidad de Datos Externos)**
  Este flujo permite al terapeuta registrar una sesión que se ha realizado externamente o cuyos datos se quieren introducir de

forma retrospectiva. Es el mecanismo principal por el cual los datos de un sistema como M3Display (que genera CSVs según Matilla Plaza, 2023) podrían ser incorporados para su análisis en esta plataforma.

1. **Entrada de Datos vía Formulario:** El terapeuta completa un formulario detallado en Streamlit que incluye: paciente, terapeuta, fecha, hora, un id_teapia manual (que puede ser un identificador textual o coincidir con un ID de la tabla Terapias), modalidad, duración, y opcionalmente, un campo de texto para la ruta/URL de un archivo de datos y un widget st.file_uploader para subir una imagen del resultado (si existe) y otro st.file_uploader para un archivo CSV de trayectoria.
2. **Procesamiento de Archivos Subidos (si los hay):**
    - Si se sube una imagen, se guarda temporalmente, se sube a public/sesiones_dibujos/ en Supabase Storage, y se obtiene su URL.
    - Si se sube un archivo CSV de trayectoria, se guarda temporalmente.
3. **Persistencia de Metadatos de Sesión:** Se crea un registro en la tabla Sesiones con toda la información proporcionada, incluyendo la URL de la imagen si se subió. El campo fich_datos podría inicialmente contener la ruta/URL textual proporcionada por el terapeuta o, si se implementa una subida directa del CSV de M3Display a Storage, la URL de ese archivo.
4. **Almacenamiento de Datos de Trayectoria Numéricos (desde CSV subido):** Si se subió un archivo CSV de trayectoria a través del st.file_uploader y la sesión principal se guardó correctamente, la función guardar_trayectoria_csv_a_supabase se invoca para procesar este CSV y almacenar sus puntos en la tabla DatosTrayectoria, de forma idéntica al flujo de *codigo_raton.py*. Esto permite que cualquier archivo CSV con el formato correcto (timestamp;PosX;PosY) pueda ser analizado, independientemente de su origen (M3Display, otro sistema, o generado manualmente).
5. **Limpieza:** Los archivos temporales subidos se eliminan.

- **Simulación de Nueva Sesión con M3Display:** Dada la limitación de no modificar el sistema M3Display existente ni tener una conexión directa con él, se ha

implementado una **simulación** para ilustrar cómo se registraría una sesión proveniente de un sistema de este tipo.

1. **Configuración de la Simulación:** El terapeuta selecciona un paciente, un terapeuta y un tipo de terapia general (previamente definido en la tabla Terapias) en un formulario de Streamlit. También puede definir una duración simulada para la terapia.
2. **Ejecución Simulada:** La aplicación muestra una barra de progreso y mensajes que simulan la ejecución de la terapia en el M3Display durante el tiempo especificado. **No hay captura real de movimiento ni generación de archivos de trayectoria o dibujo en este flujo simulado.**
3. **Persistencia de Metadatos de Sesión:** Al finalizar la simulación, se crea un registro en la tabla Sesiones. Los campos dibujo_url y fich_datos se dejan como None (o vacíos), ya que no se generan archivos reales. La Modalidad_Terapia indicaría que es una "M3Display (Sim) - [Descripción de la Terapia]". La duración sí se registra.

- **Propósito de la Simulación:** Esta funcionalidad sirve principalmente para demostrar el flujo de registro de sesiones con un tipo de modalidad diferente al *codigo_raton.py* y para poblar el historial con diversos tipos de sesiones, aunque carezcan de datos de trayectoria analizables por la plataforma actual.

- **Visualización de Historial y Detalles de Sesión:** Independientemente de cómo se haya registrado la sesión, el historial de sesiones (st.data_editor) muestra todos los registros de la tabla Sesiones. Al seleccionar una única sesión, la vista de detalles:
  - Recupera y muestra toda la información contextual de la sesión, así como los datos del paciente, terapeuta y tipo de terapia asociados.
  - Muestra la imagen del dibujo si la dibujo_url existe y es válida.
  - Intenta recuperar y analizar los datos de la tabla DatosTrayectoria asociados a esa id_sesion. Si existen, se realizan el filtrado, cálculo de métricas, visualización de la trayectoria estática y generación de GIF, como se describió en la sección 6.2.4.

- Proporciona un enlace para descargar el archivo CSV de trayectoria original si el campo fich_datos contiene una URL válida.

Este enfoque multifacético para el registro de sesiones permite tanto la captura de nuevos datos (vía *codigo_raton.py*), la importación de datos existentes (vía subida manual de CSV, útil para datos de M3Display), como la simulación de otros tipos de interacciones.

### 6.2.4 IMPLEMENTACIÓN DEL MÓDULO DE ANÁLISIS CUANTITATIVO Y VISUALIZACIÓN

Esta funcionalidad se ha integrado en la vista de "Detalles de Sesión" y en la nueva vista de "Análisis de Progreso del Paciente".

1. **Recuperación y Filtrado de Trayectorias:** Al visualizar los detalles de una sesión, la función obtener_trayectoria_supabase recupera los puntos de la trayectoria almacenados en DatosTrayectoria como un DataFrame de Pandas. Estos datos se pasan a la función filtra_trayectoria, que aplica un filtro de mediana móvil para suavizar el ruido.
2. **Cálculo y Presentación de Métricas Cinemáticas:** Sobre los datos de trayectoria, preferiblemente filtrados, las funciones calcular_duracion, calcular_longitud_recorrida y calcular_velocidad_media extraen parámetros cuantitativos del movimiento. Estos se muestran de forma clara en la interfaz usando st.metric.
3. **Visualización de Trayectoria Estática:** Se utiliza Matplotlib para generar un gráfico 2D (X vs Y) de la trayectoria recuperada de la base de datos, destacando los puntos de inicio y fin. Este gráfico se muestra directamente en la aplicación Streamlit.
4. **Generación de Animaciones GIF:** La función animar_y_guardar toma los datos de trayectoria y crea una animación GIF que reproduce el movimiento, utilizando matplotlib.animation.FuncAnimation. El GIF se muestra en la interfaz y se ofrece para su descarga.
5. **Análisis de Progreso Longitudinal:** Se ha implementado una vista específica donde el terapeuta selecciona un paciente. La aplicación recupera todas las sesiones de dicho paciente, calcula las métricas clave para cada una (procesando sus datos de trayectoria) y presenta gráficos de líneas (st.line_chart) que ilustran la evolución de estas métricas a lo largo del tiempo, permitiendo una evaluación visual del progreso.

## 6.3 RETOS TÉCNICOS DETALLADOS Y SOLUCIONES IMPLEMENTADAS

El desarrollo de una aplicación web con las características descritas, integrando diferentes tecnologías y manejando datos sensibles, presentó varios desafíos técnicos. A continuación, se detallan los más significativos y las soluciones adoptadas.

- Manejo de Datos y Tipos en Supabase/Pandas/Streamlit:
    - **Reto:** Asegurar la consistencia de los tipos de datos entre PostgreSQL (Supabase), Pandas DataFrames y los widgets de Streamlit (especialmente con st.data_editor). Por ejemplo, las fechas y horas requieren un manejo cuidadoso para su correcta visualización y ordenamiento. Los identificadores numéricos (IDs) de Supabase (tipos como bigint o integer) requerían un manejo especial al ser procesados con Pandas, especialmente si contenían valores nulos. Para asegurar que estos IDs se trataran como números enteros dentro de la aplicación y al mismo tiempo se pudieran representar correctamente los valores ausentes, se utilizó el tipo de dato entero, Int64, de Pandas, que está diseñado para manejar esta situación sin convertir incorrectamente los IDs a números de punto flotante.
    - **Solución:** Se realizaron conversiones explícitas de tipos al leer datos de Supabase y antes de pasarlos a st.data_editor (ej. pd.to_datetime, pd.to_numeric, .astype('Int 64')). Se configuraron los formatos de las columnas en st.data_editor para fechas y horas. Se manejaron los valores None o NaN adecuadamente antes de las inserciones en Supabase para columnas que no los permiten.
- Integración del Script Externo *codigo_raton.py*:
    - **Reto:** Ejecutar un script Tkinter (que crea su propia ventana GUI) desde una aplicación Streamlit basada en web, capturar su salida (archivos) y gestionar su ciclo de vida.
    - **Solución:** Se utilizó el módulo subprocess.run() de Python para ejecutar *codigo_raton.py* como un proceso separado. Se configuraron argumentos de línea de comandos para pasarle al script la ruta de salida de los archivos. Se implementó la espera de finalización del proceso y la verificación de los archivos generados. El manejo de errores del subprocess y la limpieza de archivos temporales fueron cruciales.
- Generación y Visualización Eficiente de GIFs:

- **Reto:** La generación de GIFs puede ser intensiva en recursos y tiempo, especialmente para trayectorias largas. Mostrar y descargar el GIF en Streamlit requiere manejar archivos binarios.
- **Solución:** La función animar_y_guardar se optimizó para usar blit=True en FuncAnimation cuando es posible y se guarda el GIF localmente en TEMP_DIR antes de leerlo y mostrarlo con st.image y ofrecerlo para descarga con st.download_button. Se implementó la limpieza del archivo GIF temporal tras su uso. Se añadió un st.spinner para dar feedback al usuario durante la generación.

- **Manejo de Errores y Feedback al Usuario:**
  - **Reto:** Proveer información útil al usuario cuando ocurren errores (API de Supabase, lectura de archivos, ejecución de scripts) sin abrumarlo con detalles técnicos.
  - **Solución:** Se implementaron bloques try-except extensivos alrededor de las operaciones críticas (llamadas a Supabase, procesamiento de archivos, ejecución de subprocess). Se utilizan mensajes de st.error, st.warning, st.info y st.success para comunicar el estado de las operaciones. La función handle_auth_error se diseñó para detectar específicamente errores de sesión y guiar al usuario. Para errores más complejos, se usa st.exception(e) para mostrar el traceback en el log durante el desarrollo/depuración.

La superación de estos retos ha permitido construir una aplicación funcional que cumple con los objetivos principales del TFG, sentando una base sólida para futuras mejoras y extensiones.

## 6.4 CONSIDERACIONES SOBRE LA SEGURIDAD DE DATOS MÉDICOS EN LA IMPLEMENTACIÓN

La protección de los datos de los pacientes ha sido un principio rector en la implementación, aplicando las capacidades de seguridad ofrecidas por el conjunto de tecnologías elegido.

### 6.4.1 AUTENTIFICACIÓN DE USUARIOS Y GESTIÓN DE SESIÓN

La base de la seguridad de acceso se confía al robusto sistema Supabase Auth.

- **Implementación:** Se configuró el registro de usuarios (terapeutas) con confirmación obligatoria por correo electrónico. El inicio de sesión se realiza mediante email y contraseña, y Supabase gestiona el hashing seguro de las contraseñas y la emisión de JSON Web Tokens (JWT).

- **Tokens de Sesión:** La aplicación Streamlit almacena los access_token (corta duración) y refresh_token (larga duración) en st.session_state. El access_token se utiliza para autenticar cada solicitud a los servicios de Supabase (base de datos y storage). La lógica de handle_auth_error ayuda a gestionar la expiración y renovación de estos tokens, solicitando un nuevo inicio de sesión si es necesario.
- **Cierre de Sesión:** La funcionalidad de cierre de sesión no solo limpia los tokens del estado local de Streamlit, sino que también invoca supabase.auth.sign_out() para invalidar la sesión en el servidor de Supabase.

### 6.4.2 AUTORIZACIÓN Y CONTROL DE ACCESO A NIVEL DE FILA (ROW LEVEL SECURITY – RLS)

- **Implementación:** Se ha habilitado Row Level Security (RLS) para todas las tablas de la base de datos PostgreSQL en Supabase que contienen datos de pacientes o sesiones (Pacientes, Terapeutas, Sesiones, Terapias, DatosTrayectoria).
- **Políticas Implementadas (Alcance del TFG):** Para este proyecto, se han implementado políticas RLS que, una vez que un usuario está autenticado (es decir, auth.role() = 'authenticated'), le permiten realizar todas las operaciones CRUD (SELECT, INSERT, UPDATE, DELETE) sobre todos los datos de estas tablas.
- **Justificación y Limitaciones:** Esta configuración inicial simplifica el desarrollo y las pruebas en el contexto de un TFG donde el acceso multi-usuario con diferentes niveles de permiso no es el foco principal. Sin embargo, se reconoce que, para un entorno de producción clínica real, estas políticas RLS necesitarían ser significativamente más granulares y restrictivas (ver Sección 5.2.5 para un diseño más detallado de políticas futuras). La infraestructura RLS está activa y lista para la implementación de reglas más específicas.

### 6.4.3 SEGURIDAD EN LA TRANSMISIÓN Y ALMACENAMIENTO DE DATOS

- **HTTPS:** Toda la comunicación entre el navegador del cliente y el servidor de Streamlit (especialmente si se despliega en plataformas como Streamlit Community Cloud) y entre la aplicación Streamlit y los puntos finales de Supabase (API de base de datos, Auth, Storage) se realiza obligatoriamente a través de HTTPS. Esto asegura el cifrado de los datos en tránsito, protegiéndolos contra interceptaciones.
- **Almacenamiento de Archivos:** Las imágenes y archivos CSV se suben a Supabase Storage. Aunque las URLs generadas para acceder a los

archivos en las carpetas public/ son inherentemente públicas, la capacidad de subir nuevos archivos está restringida a usuarios autenticados a través de la lógica de la aplicación Python. Para un mayor control, se podrían utilizar URLs firmadas con tiempo de expiración.

- **Gestión de Claves API:** Las claves de Supabase (URL y anon key) se gestionan mediante variables de entorno y el archivo .env, que está excluido del control de versiones, evitando su exposición en el código fuente.

### 6.4.4 CONCIENCIACIÓN SOBRE GPDR/HIPAA Y PRÁCTICAS DE DESARROLLO

Aunque la certificación completa del cumplimiento con GDPR/HIPAA excede el alcance de un TFG, se han adoptado prácticas y consideraciones de diseño orientadas a facilitar un futuro cumplimiento:

- **Minimización de Datos:** Los formularios de entrada de datos se han diseñado para recoger solo la información esencial para la funcionalidad de la aplicación.
- **Consentimiento (Conceptual):** Se asume que, en un entorno real, el terapeuta obtendría el consentimiento informado de los pacientes para el uso de sus datos en esta plataforma.
- **Derechos del Sujeto (Preparación para):** La estructura de la base de datos con identificadores únicos permitiría, en futuras iteraciones, implementar funcionalidades para que los pacientes (a través de sus terapeutas) ejerzan sus derechos de acceso, rectificación o supresión.
- **Documentación:** La documentación clara de cómo se manejan los datos (como en esta memoria) es un paso hacia la transparencia requerida por estas normativas.

La implementación actual establece un nivel de seguridad fundamental mediante la autenticación robusta, el cifrado en tránsito y la habilitación de RLS, proporcionando una base sólida sobre la cual se podrían construir controles de acceso y medidas de seguridad más avanzados para un sistema en producción.

# CAPÍTULO 7. PRUEBAS Y VALIDACIÓN

Este capítulo detalla el conjunto de pruebas realizadas para verificar el correcto funcionamiento, la fiabilidad y la usabilidad de la aplicación web desarrollada para la gestión y análisis de datos de rehabilitación. El objetivo principal de esta fase es asegurar que el sistema cumple con los requisitos funcionales y no funcionales especificados en el Capítulo 4, y que las funcionalidades implementadas (Capítulo 6) operan de manera coherente y eficiente. Se presentarán los tipos de pruebas ejecutadas, el entorno en el que se realizaron, los datos de prueba utilizados y los resultados obtenidos, discutiendo la capacidad del sistema para el análisis efectivo de los datos de rehabilitación en un contexto simulado.

## 7.1 ESTRATEGIA Y ENTORNO DE PRUEBAS

### 7.1.1 ESTRATEGIA GENERAL DE PRUEBAS

La estrategia de pruebas adoptada se centró en un enfoque multifacético, combinando pruebas funcionales para cada módulo y característica, pruebas de integración para verificar la correcta interacción entre los componentes (especialmente entre la interfaz Streamlit, la lógica Python y la base de datos Supabase), y pruebas de usabilidad informales para evaluar la experiencia del usuario (terapeuta). Dada la naturaleza de un TFG y la metodología incremental seguida, muchas pruebas se realizaron de forma continua durante el ciclo de desarrollo.

### 7.1.2 ENTORNO DE PRUEBAS

Todas las pruebas se llevaron a cabo en un entorno de desarrollo local que replicaba la configuración de las herramientas utilizadas:

- **Sistema Operativo:** Windows 10
- **Python:** Versión 3.11.5
- **Streamlit:** Ejecutando la aplicación localmente (streamlit run app.py).
- **Supabase:** Se utilizó el plan gratuito de Supabase, accediendo a la instancia del proyecto en la nube para la base de datos PostgreSQL, autenticación y almacenamiento.
- **Navegador Web:** Pruebas realizadas principalmente en Google Chrome. Se realizaron comprobaciones esporádicas en otros navegadores para verificar compatibilidad básica.
- **Script *codigo_raton.py*:** Ejecutado localmente, invocado desde la aplicación Streamlit.

## 7.2 DATOS DE PRUEBA UTILIZADOS

Para una validación representativa, se utilizaron diversos conjuntos de datos:

1. **Datos Generados por *codigo_raton.py*:**
    - Se realizaron múltiples ejecuciones del script *codigo_raton.py* (con las modificaciones para guardar trayectoria numérica y timestamps) simulando diferentes tareas de seguimiento (objetivos "reloj" y "sinusoide").
    - Esto generó un conjunto de archivos .png y sus correspondientes archivos .csv de trayectoria, con variaciones en la duración y la precisión del trazado. Estos datos fueron fundamentales para probar el flujo completo de registro de "Nueva Sesión (Ratón)", incluyendo la subida a Storage y el guardado de puntos en DatosTrayectoria.
2. **Datos CSV Provenientes del Sistema M3Display (Matilla Plaza, 2023):**
    - Se utilizaron archivos CSV de ejemplo generados por el sistema M3Display en el TFG previo (Matilla Plaza, 2023), o archivos con una estructura idéntica (Timestamp;PosX;PosY, con separador ';' y decimal ',').
    - Estos datos se emplearon para probar la funcionalidad de "Añadir Sesión (Manual)" con subida de CSV de trayectoria, verificando que la plataforma puede ingestar y analizar datos de una fuente externa relevante.
3. **Datos Simulados/Manuales:**
    - Se crearon manualmente registros de pacientes, terapeutas y tipos de terapia directamente en la interfaz de Streamlit para poblar la base de datos.
    - Se registraron sesiones manuales sin archivos de trayectoria o con datos de trayectoria CSV creados artificialmente (ej. con patrones geométricos simples) para probar la robustez del sistema ante diferentes entradas.
    - Se crearon escenarios con múltiples sesiones para un mismo paciente para probar la funcionalidad de "Análisis de Progreso del Paciente".

## 7.3 TIPOS DE PRUEBAS REALIZADAS Y RESULTADOS

### 7.3.1 PRUEBAS FUNCIONALES

La verificación de que cada requisito funcional (detallado en el Capítulo 4.2) se cumpliera correctamente fue un componente esencial de la fase de validación. Dado el alcance y la naturaleza de desarrollo rápido de un Trabajo de Fin de Grado utilizando un framework como Streamlit, el enfoque principal para las pruebas funcionales se centró en la ejecución de pruebas manuales exhaustivas y exploratorias. Estas pruebas consistieron en la interacción directa con cada una de las interfaces y funcionalidades de la aplicación, simulando los escenarios de uso típicos de un terapeuta.

Se consideraron las capacidades de testeo de aplicaciones Streamlit que la propia librería ofrece (Streamlit, 2024), las cuales permiten escribir pruebas automatizadas para verificar el estado de la aplicación tras interacciones específicas del usuario (ej. clics en botones, cambios en widgets). Estas pruebas, típicamente implementadas con frameworks como pytest, son fundamentales en entornos de desarrollo de software más amplios para asegurar la regresión y la estabilidad del código ante cambios continuos. Sin embargo, la implementación de una batería completa de pruebas automatizadas de interfaz de usuario para cada interacción y estado posible de la aplicación Streamlit, aunque deseable, se consideró que excedía el tiempo y los recursos disponibles para este TFG, cuyo foco principal residía en la implementación de la lógica de negocio, el análisis de datos y la integración con Supabase.

De manera similar, para la lógica implementada en Python (funciones de interacción con la base de datos, cálculo de métricas, etc.), se reconoce la importancia de frameworks de pruebas unitarias como unittest o pytest en Python. Estos permiten crear pruebas automatizadas para cada función o clase individualmente, asegurando su correcto comportamiento de forma aislada y facilitando la detección temprana de errores durante refactorizaciones o adiciones de código. No obstante, en este proyecto, la validación de estas funciones se realizó principalmente a través de su integración en los flujos de prueba funcionales manuales de la aplicación completa, observando sus entradas y salidas en escenarios de uso real.

A continuación, se detallan las pruebas funcionales manuales realizadas y sus resultados para cada módulo principal:

- **Gestión de Usuarios (Autentificación y Autorización):**
    - **Prueba:** Registro de nuevos usuarios, envío de email de confirmación, inicio de sesión con credenciales válidas/inválidas, manejo de sesión expirada, cierre de sesión.

    - **Resultado:** El sistema de autenticación con Supabase Auth funcionó correctamente. Los usuarios pueden registrarse, deben confirmar su email, y el inicio/cierre de sesión es funcional. La gestión de errores de sesión redirige correctamente al login. Las políticas RLS básicas (acceso para autenticados) se verificaron indirectamente al poder acceder a los datos tras el login.

- **Gestión de Pacientes, Terapeutas y Terapias (CRUD):**
    - **Prueba:** Creación, visualización en tabla, selección y eliminación de registros en las tablas `Pacientes`, `Terapeutas` y `Terapias` a través de la interfaz de Streamlit.
    - **Resultado:** Las operaciones CRUD básicas para estas entidades se implementaron y funcionaron correctamente. La visualización con `st.data_editor` permitió la interacción esperada. La eliminación simple y múltiple fue exitosa. Se observó que la eliminación de `Terapias` con `id` textual fallaría si la función `eliminar_datos` no se adaptara específicamente, pero esto se considera una limitación menor dado el flujo principal del TFG.
- **Gestión de Sesiones de Rehabilitación:**
    - **Prueba (Sesión con *codigo_raton.py*):** Configuración de la sesión, ejecución del script Tkinter, guardado de imagen y CSV local, subida de ambos archivos a Supabase Storage, creación del registro en `Sesiones` (con URLs correctas), y guardado de los puntos de trayectoria en `DatosTrayectoria`.
    - **Resultado:** Este flujo completo se probó satisfactoriamente en múltiples ocasiones. Se verificó que las URLs en Sesiones eran correctas y que los datos números se almacenaban adecuadamente en DatosTrayectoria asociados al id_sesion correcto. La limpieza de archivos temporales funcionó como se esperaba
    - **Prueba (Sesión Manual):** Creación de una sesión manual, subida opcional de imagen y/o archivo CSV de trayectoria. Verificación del almacenamiento en `Sesiones` y `DatosTrayectoria` (si se subió CSV).
    - **Resultado:** La creación de sesiones manuales fue exitosa. La subida de imágenes y su almacenamiento/visualización funcionaron. La subida de CSVs de trayectoria y su posterior procesamiento para llenar `DatosTrayectoria` también operó correctamente, demostrando la capacidad de ingestar datos de fuentes externas con el formato adecuado.
- **Almacenamiento y Gestión de Datos de Trayectoria Numéricos:**
    - **Prueba:** Verificación directa en la base de datos Supabase (tabla `DatosTrayectoria`) de que los puntos (timestamp, x, y,

id_sesion) se almacenaban correctamente tras los flujos de registro de sesión.

- **Resultado:** Los datos se almacenaron de forma estructurada y precisa, con los tipos de datos correctos y las relaciones de clave foránea intactas.

- **Análisis Cuantitativo de Datos de Movimiento:**
  - **Prueba:** Para sesiones con datos de trayectoria, verificar la aplicación del filtro de mediana móvil y el cálculo de las métricas (duración, longitud, velocidad media).
  - **Resultado:** La función filtra_trayectoria aplicó el suavizado esperado. Las métricas calcular_duracion, calcular_longitud_recorrida y calcular_velocidad_media devolvieron valores consistentes y correctos basados en los datos de entrada.

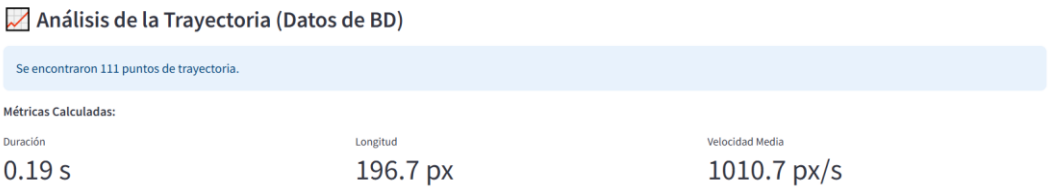


*Figura 17: Captura de pantalla de la interfaz de Streamlit mostrando las métricas cinemáticas (Duración, Longitud, Velocidad Media) calculadas para una sesión de prueba tras la recuperación y procesamiento de sus datos de trayectoria. Fuente: Aplicación TFG.*

- **Visualización de Datos y Progreso:**
  - **Prueba (Detalles de Sesión):** Visualización de información contextual, imagen del dibujo (desde URL de Storage), gráfico estático de trayectoria (desde DatosTrayectoria), y generación/visualización/descarga de GIF (desde DatosTrayectoria).
  - **Resultado:** Todas estas visualizaciones se generaron correctamente. El gráfico de trayectoria representó fielmente los datos almacenados. El GIF animado se creó y mostró adecuadamente, y la funcionalidad de descarga funcionó.

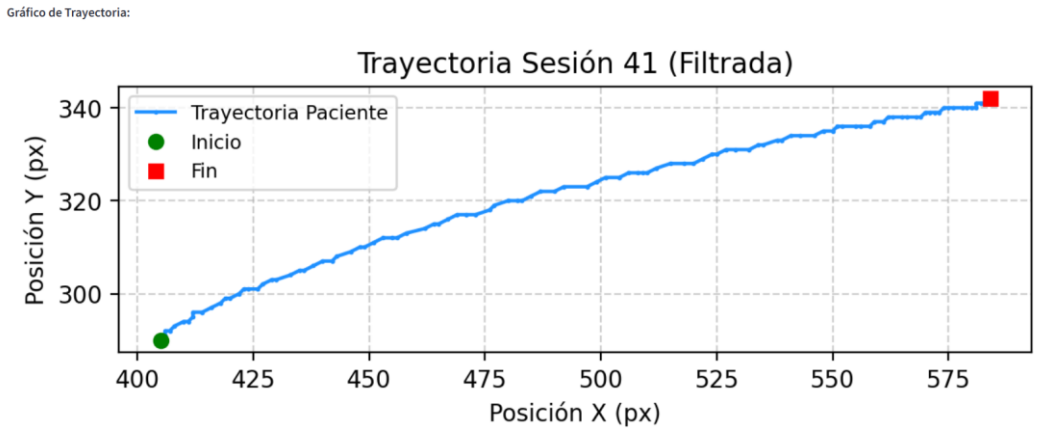


*Figura 18:Ejemplo de visualización en la aplicación: Gráfico estático de una trayectoria recuperada de la base de datos. Fuente: Aplicación TFG.*

- **Prueba (Análisis de Progreso del Paciente):** Selección de un paciente con múltiples sesiones (con datos de trayectoria) y verificación de los gráficos de evolución de métricas.
- **Resultado:** La vista de análisis de progreso cargó correctamente las sesiones del paciente seleccionado, calculó las métricas para cada una y generó los gráficos de líneas mostrando la evolución de la duración, longitud y velocidad media a lo largo de las sesiones.

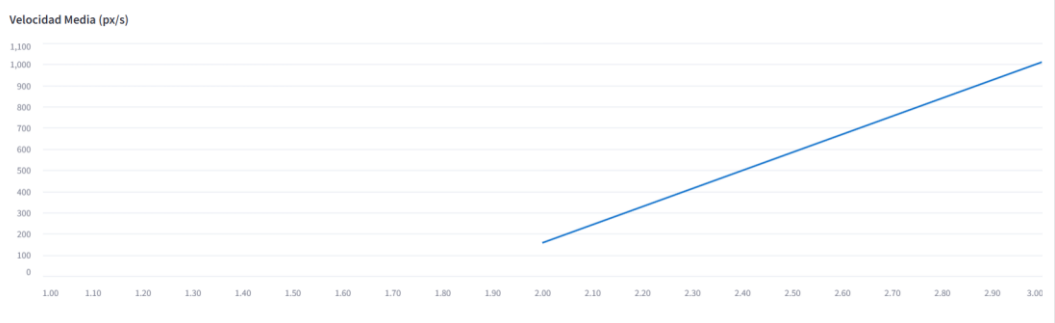


*Figura 19:Ejemplo del gráfico de evolución de la "Velociad Media" a lo largo un par de sesiones para un paciente de prueba, generado en la vista de "Análisis de Progreso del Paciente". Fuente: Aplicación TFG.*

### 7.3.2 PRUEBAS DE INTEGRACIÓN

- **Streamlit - Supabase:** Se verificó continuamente la correcta comunicación entre la aplicación Streamlit y los servicios de Supabase (Auth, Database, Storage). Las funciones de la librería supabase-py respondieron como se esperaba, y el manejo de errores de API fue adecuado.

- **Streamlit - Lógica Python:** Se aseguró que los datos fluyeran correctamente desde los widgets de la interfaz de Streamlit hacia las funciones de lógica de Python y viceversa.
- **Script *codigo_raton.py* - Streamlit:** Se probó la invocación del script externo, el paso de argumentos, y la correcta recepción y procesamiento de los archivos de salida.

### 7.3.3 PRUEBAS DE USABILIDAD (INFORMALES)

- **Prueba:** Se realizaron pruebas informales por parte del desarrollador, navegando por todas las vistas de la aplicación, realizando las acciones típicas de un terapeuta (registrar pacientes, sesiones, analizar datos) y evaluando la claridad de la interfaz, la facilidad de navegación y la utilidad de la información presentada.
- **Resultado:** La interfaz de Streamlit demostró ser intuitiva y fácil de usar para las tareas principales. Los flujos de trabajo son lógicos. La retroalimentación al usuario (mensajes de éxito, error) mejora la experiencia. Se identificaron pequeñas áreas de mejora en la presentación de algunos mensajes de error, que se ajustaron. La organización de las vistas y la navegación se consideraron adecuadas para el perfil de usuario objetivo.

### 7.3.4 PRUEBAS DE REQUISITOS NO FUNCIONALES (SELECCIONADAS)

- **Seguridad (RNF001-RNF004):**
    - **Prueba:** Intentos de acceso sin autenticación, verificación de uso de HTTPS (implícito en despliegue y Supabase), verificación de que las claves API no están en el código. Revisión conceptual de las políticas RLS implementadas.
    - **Resultado:** El acceso sin autenticación fue denegado. La comunicación es vía HTTPS. Las claves están en .env. Las políticas RLS básicas (acceso para autentificados) están activas.
- **Rendimiento (RNF009-RNF011):**
    - **Prueba:** Se midieron tiempos de respuesta informales para la carga de tablas con ~50-100 registros, el procesamiento de trayectorias con ~500-1000 puntos, y la generación de GIFs.

    - **Resultado:** Para el volumen de datos de prueba utilizado, los tiempos de respuesta fueron generalmente aceptables (pocos segundos para la mayoría de las operaciones). La generación de GIFs para trayectorias muy largas (ej. >2000 puntos) podía tardar algo más, pero dentro de límites razonables para una operación no instantánea. El sistema manejó sin problemas el volumen de datos correspondiente a un escenario de prueba con varios pacientes y múltiples sesiones cada uno.

- **Compatibilidad (RNF012):**
    - **Prueba:** Uso de la aplicación en las últimas versiones de Google Chrome y Mozilla Firefox en Windows.
    - **Resultado:** La aplicación funcionó correctamente en ambos navegadores sin problemas de renderizado o funcionalidad significativos.

## 7.4 DISCUSIÓN DE LOS RESULTADOS DE LAS PRUEBAS Y VALIDACIÓN

Las pruebas realizadas demuestran que la aplicación web desarrollada cumple con la gran mayoría de los requisitos funcionales y no funcionales establecidos. El sistema es capaz de:

- Gestionar de forma segura la información de usuarios, pacientes, terapeutas, terapias y sesiones.
- Almacenar y procesar datos numéricos de trayectoria provenientes tanto del script de captura *codigo_raton.py* como de archivos CSV externos (simulando datos de M3Display).
- Aplicar filtrado a los datos de trayectoria.
- Calcular métricas cinemáticas básicas (duración, longitud, velocidad media).
- Visualizar las trayectorias y las métricas de forma integrada.
- Generar animaciones GIF del movimiento.
- Mostrar la evolución del progreso de un paciente a través de múltiples sesiones.

La interfaz de usuario es funcional e intuitiva gracias a Streamlit. Las medidas de seguridad implementadas (autenticación, HTTPS, RLS básica) proporcionan un nivel de protección adecuado para el contexto de un TFG.

La principal limitación validada es la dependencia de la carga manual de CSV para datos de sistemas como M3Display, ya que la conexión automática quedó fuera del alcance de implementación, tal como se previó. Sin embargo, la

capacidad de procesar estos CSV una vez subidos permite igualmente su análisis dentro de la plataforma.

En general, la aplicación demuestra ser una herramienta valiosa y funcional para la gestión y, fundamentalmente, para el análisis cuantitativo y la visualización detallada de datos de rehabilitación, superando las limitaciones del manejo de archivos CSV aislados y proporcionando una plataforma centralizada para el seguimiento del progreso del paciente.

# CAPÍTULO 8. CONCLUSIONES Y POSIBLES MEJORAS

El presente Trabajo de Fin de Grado se ha centrado en el diseño y desarrollo de una aplicación web robusta y funcional, con el objetivo primordial de ofrecer una solución para la gestión centralizada y el análisis cuantitativo de datos generados por sistemas de rehabilitación neuromotora asistidos por Realidad Aumentada. Este esfuerzo surge como una respuesta directa y una evolución necesaria frente a las limitaciones identificadas en trabajos previos, como es el caso del sistema M3Display desarrollado por Matilla Plaza (2023). En dicho sistema, aunque se lograba una valiosa captura de datos de sesión, estos quedaban almacenados como archivos CSV locales, lo que restringía significativamente su potencial analítico y dificultaba un seguimiento longitudinal efectivo.

La solución desarrollada en este TFG busca cerrar esta brecha fundamental, proporcionando a terapeutas y personal clínico herramientas que permitan un seguimiento del progreso del paciente más preciso, objetivo y personalizado. A lo largo de esta memoria, se ha procedido a detallar el contexto que justificó la necesidad del proyecto, los objetivos específicos que se buscaron alcanzar, la metodología de desarrollo incremental y de prototipado que guio su ejecución, el diseño arquitectónico y técnico del sistema implementado, su materialización práctica mediante el código desarrollado, y finalmente, las pruebas realizadas para su validación.

Este capítulo final tiene como propósito principal consolidar las conclusiones más relevantes extraídas de todo el proceso de desarrollo y evaluación. Se realizará una valoración crítica del grado de cumplimiento de los objetivos inicialmente planteados, se reconocerán de forma transparente las limitaciones inherentes al trabajo realizado dentro del marco de un TFG, y se propondrá un conjunto de líneas de mejora y trabajo futuro. Estas últimas no solo buscan subsanar las limitaciones actuales, sino también expandir las capacidades de la plataforma, con la visión de potenciar aún más su impacto y utilidad en el ámbito de la rehabilitación tecnológica.

## 8.1 LOGROS OBTENIDOS Y EVALUACIÓN DEL CUMPLIMIENTO DE LOS OBJETIVOS

El desarrollo de este TFG ha resultado en la creación de una aplicación web funcional y cohesiva, construida sobre el conjunto de tecnologías formado por Streamlit, Supabase y Python, que aborda con éxito el objetivo principal y la gran mayoría de los objetivos secundarios definidos al inicio del proyecto. Los logros más significativos son:

- **Implementación de una Plataforma Web Integral:** Se ha materializado una aplicación web completa que no solo permite la gestión básica de entidades (usuarios autenticados, pacientes, terapeutas, tipos de terapia), sino que se especializa en la gestión detallada de sesiones de rehabilitación. Esta plataforma actúa como un repositorio centralizado en la nube (Supabase), superando la dispersión de datos.
- **Gestión Avanzada de Datos de Trayectoria Numéricos:** Un logro fundamental ha sido el diseño e implementación de un sistema para el almacenamiento estructurado y el manejo eficiente de los datos numéricos de trayectoria (secuencias de puntos timestamp, x, y). Esto incluye la capacidad de procesar archivos CSV (generados por el script *codigo_raton.py* modificado o subidos manualmente, simulando datos de sistemas como M3Display) y persistir cada punto de movimiento en una tabla dedicada (DatosTrayectoria) en Supabase, relacionada intrínsecamente con su sesión correspondiente. Este es un avance crucial respecto al trabajo de Matilla Plaza (2023).
- **Capacidades de Análisis Cuantitativo del Movimiento:** La aplicación va más allá del simple almacenamiento, incorporando algoritmos de procesamiento y análisis:
    - **Ingesta y Procesamiento de Datos Externos:** La funcionalidad para subir y procesar archivos CSV de trayectoria permite la incorporación de datos de diversas fuentes, crucial para la versatilidad de la plataforma.
    - **Filtrado de Ruido:** Se ha implementado un filtro de mediana móvil (filtra_trayectoria) que se aplica a los datos de trayectoria recuperados, mejorando la calidad de la señal para análisis posteriores y visualizaciones más claras.
    - **Cálculo de Métricas Cinemáticas Fundamentales:** El sistema calcula automáticamente y presenta métricas objetivas clave del rendimiento motor para cada sesión con datos de trayectoria, incluyendo la **duración total de la tarea, la longitud de la trayectoria recorrida y la velocidad media del movimiento**. Estas métricas proporcionan una base cuantitativa para la evaluación.
- **Visualización Interactiva y Multifacética de Datos:** Se ha puesto un gran énfasis en la capacidad de la plataforma para presentar la información de manera clara y útil para el terapeuta, utilizando las capacidades de Streamlit y Matplotlib:
    - **Detalles de Sesión Enriquecidos:** La vista de detalles de sesión no solo muestra información contextual, sino que integra la visualización de la imagen del dibujo (si existe, cargada desde

Supabase Storage) y las métricas cinemáticas calculadas para esa sesión específica.

- **Gráficos de Trayectoria desde Base de Datos:** Se generan y muestran gráficos estáticos de la trayectoria de movimiento (X vs Y) directamente a partir de los datos numéricos almacenados en DatosTrayectoria, eliminando la dependencia de archivos externos para esta visualización.
- **Generación de Animaciones GIF:** La plataforma puede generar dinámicamente animaciones GIF del movimiento realizado durante una sesión, también a partir de los datos almacenados en la base de datos, ofreciendo una representación visual dinámica del rendimiento. Estos GIFs son descargables.
- **Análisis de Progreso Longitudinal:** Se ha implementado una vista dedicada que permite al terapeuta seleccionar un paciente y visualizar la evolución de sus métricas cinemáticas clave (duración, longitud, velocidad media) a través de múltiples sesiones, mediante gráficos de líneas interactivos, facilitando la identificación de tendencias y la evaluación del progreso a lo largo del tiempo.

- **Interfaz de Usuario Centrada en el Terapeuta:** La interfaz, desarrollada con Streamlit, ha demostrado ser intuitiva y fácil de navegar durante las pruebas informales, con flujos de trabajo lógicos para las tareas de gestión y análisis. Se ha priorizado la claridad en la presentación de la información.
- **Implementación de Medidas de Seguridad Fundamentales:** Se ha asegurado el acceso mediante autenticación de usuarios (Supabase Auth con confirmación por email), toda la comunicación se realiza por HTTPS, las credenciales sensibles se gestionan mediante variables de entorno, y se ha habilitado Row Level Security (RLS) en Supabase, sentando las bases para un control de acceso granular.
- **Consecución de Objetivos Académicos:** El proyecto ha permitido al autor adquirir un profundo conocimiento sobre los desafíos en la rehabilitación asistida por tecnología, así como dominar el conjunto de tecnologías seleccionado (Streamlit, Supabase, Python y sus librerías), aplicando estos conocimientos para resolver un problema real y documentando el proceso con rigor.

En resumen, se ha logrado el objetivo principal de crear una aplicación web funcional que no solo gestiona datos clínicos de rehabilitación, sino que los transforma en información analítica y visualmente interpretable, proporcionando una herramienta de apoyo significativa para el seguimiento del paciente.

## 8.2 LIMITACIONES DEL TRABAJO REALIZADO

Es fundamental reconocer las limitaciones inherentes a un proyecto de esta naturaleza desarrollado en el marco de un Trabajo de Fin de Grado. Estas limitaciones, algunas de las cuales fueron decisiones conscientes para asegurar la viabilidad del proyecto, incluyen:

- **Integración Automatizada con Sistemas Externos (M3Display):** Como se explicitó en el alcance, la implementación de una conexión directa y completamente automática para la ingesta de datos desde sistemas de captura como el M3Display (Matilla Plaza, 2023) quedó fuera del alcance práctico de la implementación, limitándose a su diseño conceptual y a la capacidad de la plataforma de procesar los CSV generados por dicho sistema mediante subida manual. Esto implica que, para datos de M3Display, persiste un paso manual.

- **Validación Clínica Formal:** Las pruebas y la validación del sistema se realizaron principalmente en un entorno de desarrollo controlado, utilizando datos generados por el script *codigo_raton.py* (adaptado para simular captura de movimiento) y archivos CSV de ejemplo. Si bien se verificó la funcionalidad técnica, no se llevó a cabo una validación formal con terapeutas y pacientes reales en un entorno clínico. Dicha validación sería indispensable para evaluar en profundidad la usabilidad real de la interfaz en el flujo de trabajo clínico, la relevancia y utilidad percibida de las métricas y visualizaciones implementadas, y el impacto general de la herramienta en la práctica diaria de rehabilitación. No obstante, la arquitectura modular del sistema y la flexibilidad del conjunto de tecnologías utilizado (Python, Pandas, Matplotlib) permitirían, con relativa facilidad, modificar las métricas existentes o implementar nuevas pruebas y visualizaciones específicas que un profesional clínico considerase pertinentes tras una evaluación o piloto inicial.

- **Alcance de las Métricas de Análisis Implementadas:** Si bien se implementaron métricas cinemáticas básicas y fundamentales (duración, longitud recorrida y velocidad media), que proporcionan una primera capa de análisis cuantitativo, se reconoce que el catálogo de análisis podría ser considerablemente más extenso y profundo. Métricas más sofisticadas, como el error de la trayectoria del paciente respecto a una trayectoria objetivo (lo que implicaría definir y almacenar dichas trayectorias de referencia), o indicadores de la calidad del movimiento como la suavidad (mediante análisis de jerk o el índice SPARC), la eficiencia de la ruta, o el número de submovimientos, no fueron implementadas en la versión actual. Sin embargo, es importante destacar que la estructura de datos actual (especialmente la

tabla DatosTrayectoria que almacena la secuencia temporal de coordenadas) ha sido diseñada con la previsión de permitir la futura incorporación de estos análisis más complejos. La implementación de tales métricas se beneficiaría enormemente de la colaboración directa con terapeutas y profesionales clínicos, quienes podrían guiar la selección de los indicadores más relevantes para diferentes patologías o fases de la rehabilitación, así como ayudar a definir los algoritmos y umbrales adecuados para su interpretación.

- **Granularidad de las Políticas de Seguridad RLS:** Las políticas de Row Level Security implementadas en Supabase, aunque activas, son generales (permiten acceso completo a los datos a cualquier usuario autenticado). En un entorno de producción real con múltiples terapeutas y potencialmente diferentes roles (ej. administrador, investigador), se requerirían políticas RLS mucho más detalladas y restrictivas para garantizar una segregación de datos y un control de acceso más fino, asegurando que cada terapeuta solo acceda a la información de sus pacientes.
- **Ausencia de Funcionalidades de Diagnóstico o Recomendación:** Conforme al alcance definido, la plataforma actúa como una herramienta de apoyo para la visualización y análisis de datos, y no implementa ninguna funcionalidad de diagnóstico médico automatizado ni ofrece recomendaciones de tratamiento. Su rol es descriptivo y analítico, no prescriptivo.
- **Exclusión de Módulos Administrativos Avanzados:** La aplicación se centra exclusivamente en la gestión de pacientes, sesiones y el análisis de movimiento, excluyendo módulos administrativos más complejos que podrían encontrarse en un sistema de gestión hospitalaria completo (ej. gestión de citas, facturación, inventario de recursos).

Reconocer estas limitaciones es importante no como una deficiencia del trabajo realizado dentro de su contexto, sino como una hoja de ruta clara para el desarrollo y la investigación futuros.

## 8.3 POSIBLES MEJORAS Y LÍNEAS DE TRABAJO FUTURO

La plataforma desarrollada constituye una base robusta y prometedora que puede ser significativamente expandida y mejorada en futuras iteraciones. Las líneas de trabajo futuro más relevantes incluyen:

- **Desarrollo e Implementación de la Interfaz de Ingesta Automática:** Priorizar la implementación del mecanismo (API o sistema de escucha de archivos) diseñado para la ingesta directa y automática

de datos desde sistemas de captura como M3Display. Esto eliminaría el paso de carga manual de CSVs y permitiría un flujo de datos más eficiente y cercano al tiempo real.

- **Ampliación Sustancial del Catálogo de Métricas de Análisis Cinemático:**
    - **Métricas de Error:** Implementar el cálculo de errores respecto a trayectorias objetivo (ej. Desviación Media Absoluta, RMSE, área entre curvas). Esto requeriría un módulo para definir y almacenar/generar estas trayectorias objetivo para cada tipo de ejercicio.
    - **Métricas de Suavidad:** Incorporar algoritmos para calcular métricas de suavidad del movimiento, como el jerk normalizado (derivada de la aceleración) o el Spectral Arc Length (SPARC), que son indicadores importantes de la calidad y coordinación del movimiento.
    - **Otras Métricas Relevantes:** Investigar e implementar otras métricas validadas en la literatura, como el número de unidades de movimiento, la eficiencia de la ruta, el tiempo para alcanzar picos de velocidad, etc.
- **Mejoras Avanzadas en Visualización y Generación de Informes:**
    - **Superposición de Trayectoria Objetivo:** Añadir la capacidad de visualizar la trayectoria objetivo superpuesta a la trayectoria del paciente en los gráficos estáticos y en las animaciones GIF.
    - **Informes Personalizables y Exportables:** Desarrollar un módulo de generación de informes más sofisticado que permita al terapeuta seleccionar las métricas y sesiones de interés y exportar un resumen (ej. a PDF) para la historia clínica del paciente o para compartir con otros profesionales.
    - **Visualizaciones 3D (si aplica):** Si se integran datos de sistemas de captura 3D más completos, explorar librerías de visualización 3D (ej. Plotly) para representar las trayectorias en su espacio tridimensional.
- **Refinamiento de la Seguridad y Gestión de Permisos:**
    - Implementar políticas RLS granulares y basadas en roles en Supabase para un control de acceso más estricto, asegurando que cada terapeuta solo acceda a los datos de sus pacientes asignados.
    - Explorar la implementación de logs de auditoría para rastrear el acceso y las modificaciones a los datos sensibles.

- **Validación Clínica Exhaustiva y Recogida de Feedback de Usuarios:**
    - Diseñar y ejecutar estudios piloto en entornos clínicos reales con la participación activa de terapeutas y pacientes. Esto permitiría evaluar rigurosamente la usabilidad, la utilidad clínica de las métricas y visualizaciones, y el impacto general de la plataforma en la práctica rehabilitadora.
    - Establecer un mecanismo para recoger feedback continuo de los usuarios finales para guiar las futuras iteraciones de desarrollo.
- **Integración de Módulos de Inteligencia Artificial y Aprendizaje Automático:** Utilizar los datos históricos almacenados para entrenar modelos de Machine Learning que puedan, por ejemplo, clasificar patrones de movimiento, predecir la probabilidad de alcanzar ciertos hitos de recuperación, o incluso sugerir adaptaciones en los parámetros de la terapia de forma personalizada.
- **Desarrollo de una Interfaz para Pacientes:** Considerar la creación de una vista simplificada y segura (con acceso controlado por el terapeuta) para que los pacientes puedan visualizar su propio progreso de forma comprensible, lo que podría tener un efecto positivo en su motivación y compromiso con la terapia.
- **Mejoras en la Parametrización y Gestión de "Tipos de Terapia":** Permitir que en la tabla Terapias se puedan asociar parámetros específicos a cada tipo de terapia (ej. umbrales de éxito, duración recomendada, trayectorias objetivo estándar, métricas clave a monitorizar). Esto facilitaría la configuración de nuevas sesiones y el análisis comparativo estandarizado.
- **Optimización del Rendimiento y Escalabilidad:** Para grandes volúmenes de datos de trayectoria (muchos puntos por sesión, muchas sesiones), investigar y aplicar técnicas de optimización en las consultas a la base de datos, en el procesamiento de datos con Pandas/NumPy, y en la generación de visualizaciones para asegurar que la aplicación siga siendo responsiva.

## 8.4 IMPACTO POTENCIAL DEL PROYECTO EN EL CAMPO DE LA REHABILITACIÓN ASISTIDA POR TECNOLOGÍA

La aplicación web desarrollada en este TFG, a pesar de sus limitaciones actuales, representa una contribución significativa y tiene el potencial de generar un impacto positivo en el campo de la rehabilitación neuromotora asistida por tecnología:

- **Para los Profesionales Clínicos (Terapeutas y Médicos):** La plataforma ofrece una transición desde la evaluación subjetiva o el análisis manual y fragmentado de datos hacia un seguimiento objetivo, cuantitativo y centralizado del progreso del paciente. Al proporcionar métricas claras y visualizaciones interactivas (incluyendo la evolución longitudinal), se empodera al terapeuta con herramientas para:
    - Tomar decisiones clínicas más informadas sobre la efectividad de las intervenciones.
    - Personalizar los planes de tratamiento de manera más precisa.
    - Identificar patrones sutiles en la recuperación que podrían pasar desapercibidos.
    - Comunicar el progreso al paciente y a otros profesionales de forma más efectiva.
    - Reducir la carga de trabajo asociada al análisis manual de datos, permitiendo dedicar más tiempo a la interacción directa con el paciente.
- **Para la Investigación Clínica y el Desarrollo de Nuevas Terapias:**
    - La capacidad de almacenar de forma estructurada datos detallados de movimiento y métricas asociadas crea un repositorio valioso para la investigación. Estos datos pueden ser utilizados para estudiar la efectividad de diferentes protocolos de rehabilitación con RA, identificar nuevos biomarcadores digitales del movimiento, o entender mejor los mecanismos de recuperación neuromotora.
    - Facilita la estandarización en la recogida y análisis de datos en estudios que utilicen sistemas de RA similares.
- **Para la Evolución de Sistemas de Captura como M3Display:** Este proyecto actúa como un complemento crucial para sistemas de captura como el M3Display (Matilla Plaza, 2023). Mientras M3Display se enfoca en la interacción y la captura de datos brutos durante la terapia, esta plataforma proporciona la capa de inteligencia y usabilidad posterior que transforma esos datos en conocimiento práctico. Cierra el ciclo desde la generación de datos hasta su interpretación clínica, aumentando significativamente el valor y la aplicabilidad de dichos sistemas de captura.
- **Impulso a la Rehabilitación Basada en Datos y la eHealth:** La solución se alinea con las tendencias globales hacia una medicina y rehabilitación más personalizadas y basadas en la evidencia. Al facilitar la cuantificación objetiva y el seguimiento detallado, contribuye a mover el campo de la rehabilitación hacia prácticas donde las decisiones

terapéuticas están cada vez más respaldadas por datos concretos. Además, al ser una aplicación web, promueve la accesibilidad y se enmarca en el paradigma de la eHealth.

## 8.5 CONCLUSIÓN FINAL

El desarrollo de la "Aplicación web para la gestión y análisis de datos de un sistema de rehabilitación con realidad aumentada" ha representado un esfuerzo significativo por abordar una necesidad tangible en el campo de la eHealth y la rehabilitación tecnológica. Partiendo de las bases sentadas por el sistema M3Display y el juego "Coins" de Matilla Plaza (2023), que demostraron la viabilidad de la captura de datos de movimiento mediante RA, este TFG se propuso y logró con éxito crear la infraestructura de software necesaria para llevar esos datos un paso más allá: desde archivos locales aislados a una plataforma web centralizada, segura y, fundamentalmente, analítica.

La elección de un conjunto de tecnologías moderno y accesible, compuesto por Streamlit para la interfaz, Supabase para el backend y la base de datos, y Python con su robusto ecosistema de librerías para el análisis, ha demostrado ser acertada, permitiendo un desarrollo ágil y la implementación de funcionalidades complejas como el almacenamiento de trayectorias numéricas, el cálculo de métricas cinemáticas y la generación de visualizaciones dinámicas. Se ha conseguido crear una herramienta que no solo gestiona la información de pacientes y sesiones, sino que ofrece al terapeuta una visión cuantitativa y longitudinal del progreso motor, algo que era notablemente difícil de obtener con los métodos de análisis manual previos.

Si bien se reconocen las limitaciones inherentes al alcance de un TFG, como la ausencia de una validación clínica formal o la implementación completa de una ingesta de datos totalmente automatizada desde sistemas como M3Display, la plataforma actual es funcional, cumple con sus objetivos principales y establece una base sólida y prometedora. Las funcionalidades de filtrado de datos, cálculo de métricas básicas (duración, longitud, velocidad) y las diversas formas de visualización (gráficos estáticos de trayectoria, GIFs animados, gráficos de evolución de métricas) ya proporcionan un valor añadido considerable.

El verdadero impacto de este trabajo reside en su potencial para empoderar a los profesionales de la rehabilitación con herramientas que faciliten una práctica más informada y basada en la evidencia. Al transformar datos crudos de movimiento en conocimiento accionable, esta aplicación contribuye a la optimización de las terapias, a una mejor comprensión de la recuperación del paciente y, en última instancia, a mejorar los resultados de la rehabilitación. Las numerosas líneas de trabajo futuro identificadas demuestran la escalabilidad y el potencial de crecimiento de la solución, abriendo la puerta a

futuras investigaciones y desarrollos que podrían refinar aún más sus capacidades y su aplicabilidad en entornos clínicos reales.

En definitiva, este TFG ha logrado construir un "puente digital" significativo, conectando la captura de datos innovadora con el análisis clínico práctico, y representa un paso adelante en la utilización efectiva de la tecnología para avanzar en el campo de la rehabilitación neuromotora.

# BIBLIOGRAFÍA


- Agencia Española de Protección de Datos (AEPD). (2023). *Guía para aplicaciones de salud*. Recuperado el 2 de mayo de 2025, de https://www.aepd.es/es/documento/guia-apps-salud.pdf
- APD. (2023). *Tecnología en la medicina: beneficios y avances más importantes*. Redacción APD. Recuperado el 18 de abril de 2025, de https://www.apd.es/tecnologia-medicina-beneficios-avances/
- Astudillo Aguiar, A. (2023). *Dispositivo de ayuda a la rehabilitación neuromotora de personas con daño cerebral adquirido*. Proyecto Fin de Grado, Arquitectura y Tecnología de Sistemas Informáticos (UPM).
- Azuma, R. T. (1997). A Survey of Augmented Reality. *Presence: Teleoperators and Virtual Environments*, 6(4), 355-385.
- BVS. (1963). *Paresia*. Recuperado el 18 de abril de 2025, de https://decs.bvsalud.org/es/ths/resource/?id=10480
- Chen, Y., Fanchiang, H. D., & Howard, A. (2020). Augmented Reality-Based Rehabilitation Systems for Stroke Patients: A Systematic Review. *IEEE Access*, 8, 106011-106023.
- Cisnal, A., et al. (2023). M3Display: Sistema de realidad aumentada para la rehabilitación de la función motora del miembro superior. En *XLI Congreso Anual de la Sociedad Española de Ingeniería Biomédica* (pp. 161-164).
- ClinicaUner. (2023). *Realidad virtual y aumentada: mejorando la rehabilitación con nuevas tecnologías*. Recuperado el 21 de abril de 2025, de https://clinicauner.es/realidad-virtual-y-aumentada-mejorando-la-rehabilitacion-neurologica-con-nuevas-tecnologias/
- Díez, Ú. C. (2020). *Robótica para la rehabilitación*. (Citar fuente original si se encuentra).
- Dobkin, B. H. (2023). Digital Biomarkers for Motor Rehabilitation: Wearables and AR. *Nature Digital Medicine*, 6(1), 1-14.
- EN ISO 13485:2021. *Medical devices — Quality management systems*.
- EvaluateMedTech. (2018). *Evolución anual de la inversión en investigación y desarrollo en sector de tecnología médica a nivel global desde 2011 a 2024*.
- EvaluateMedTech. (2024). *Variación Porcentual del mercado global de tecnología médica de 2010 a 2022*.

- Fan Zhang, V. B.-L. (2020). *MediaPipe Hands: On-device Real-time Hand Traking*. CVPR Workshop on Computer Vision for Augmented and Virtual Reality, Seattle, WA, USA.
- Faria, A. L., Vourvopoulos, A., Cameirão, M. S., Fernandes, J., & Bermúdez i Badia, S. (2018). NeuroRehabLab Online Platform: A Scalable Cloud-Based Framework for Sensorimotor Rehabilitation. *Journal of NeuroEngineering and Rehabilitation*, 15(1), 111.
- Fernández, C. (2022). Supabase vs Firebase in HIPAA-Compliant Apps: A Security Benchmark. *Health Tech Journal*, 15(3), 112-125.
- Fisioform. (2023). *Avances en fisioterapia y tendencias emergentes*. Recuperado el 18 de abril de 2025, de https://fisioformcursos.com/avances-en-fisioterapia-y-tendencias-emergentes/
- FisioOnline. (2024). *Paresia*. Recuperado el 20 de abril de 2025, de https://www.fisioterapia-online.com/glosario/paresia
- García, B., Pérez-Rey, D., & Alonso-Calvo, R. (2021). Streamlit for Healthcare: Building Interactive Web Applications for Medical Data Analysis. *Journal of Medical Systems*, 45(8), 78.
- Gobierno de España. (2020). *Plan España Digital 2025*. Programas para el Avance Digital.
- HL7 FHIR Foundation. (2022). *FHIR Release 4.0*. Recuperado el 17 de abril de 2025, de https://www.hl7.org/fhir/
- HIPAA Journal. (2023). *Healthcare Data Security Best Practices*. Recuperado el 17 de abril de 2025, de https://www.hipaajournal.com
- Hossein Mousavi Hondori, M. K., Khademi, M., Lopes, C. V., Dodakian, L., & Cramer, S. C. (2016). Choice of Human–Computer Interaction Mode in Stroke Rehabilitation. *Neurorehabilitation and Neural Repair*, 30(7), 628-635. doi:10.1177/1545968315593805
- Hunter, J. D. (2007). Matplotlib: A 2D Graphics Environment. *Computing in Science & Engineering*, 9(3), 90-95.
- Keshner, E. A., & Weiss, P. T. (2018). Virtual Reality and Augmented Reality: What Do We Know About the Effects on Gait Rehabilitation? *Current Physical Medicine and Rehabilitation Reports*, 6(3), 156-166.
- Khademi, M., Mousavi Hondori, H., Lopes, C. V., Dodakian, L., & Cramer, S. C. (2012). Haptic Augmented Reality to Monitor Human Arm's. En *2012 IEEE-EMBS Conference on Biomedical Engineering and Sciences* (pp. 892-895). Langkawi, Malaysia. doi:10.1109/IECBES.2012.6498168

- Lee, S., Shin, H., & Lee, J. (2019). Web-Based Motion Tracking System for Upper Limb Rehabilitation. *Sensors*, 19(10), 2276.
- Lynn E. Fiellin, K. D. (2014). Videogames, Here for Good. *Pediatrics*, 134(5), 849-851. doi:10.1542/peds.2014-0941
- Martínez-Méndez, R., et al. (2023). Dynamic Time Warping for Movement Pattern Analysis in Stroke Patients. *IEEE Journal of Biomedical and Health Informatics*, 27(2), 789-800.
- Matilla Plaza, A. (2023). *Desarrollo de un juego serio de realidad aumentada para rehabilitación neuromotora*. Trabajo de Fin de Grado, Grado en Ingeniería Electrónica Industrial y Automática, Universidad de Valladolid.
- McKinney, W. (2017). *Python for Data Analysis*. O'Reilly.
- Moreno, A., et al. (2022). Real-Time Hand Tracking for Rehabilitation Using Low-Cost AR. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 30, 1549-1558.
- Motek. (s.f.). *Human Movement Analysis Software - D-Flow*. Recuperado el 17 de abril de 2025, de https://www.motekmedical.com/software/d-flow/
- Nguyen, T. H., et al. (2023). Edge Computing for AR Rehabilitation: Reducing Latency in Motion Data Processing. *Sensors*, 23(5), 2567.
- OpenEHR Foundation. (2023). *Clinical Data Models for Rehabilitation*. Recuperado el 18 de abril de 2025, de https://specifications.openehr.org/
- Ordoñez, J. L. (2020). *Realidad Virtual y Realidad Aumentada*. CEDRO.
- Pandey, S. (2023). *Building Clinical Dashboards with Streamlit*. O'Reilly.
- Pietrusinski, M., Cajamarca, G., & Solanki, D. (2020). Cloud-Based Platform for Tele-Rehabilitation. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 28(1), 34-42.
- Plotly Medical Team. (2022). *Interactive 3D Motion Visualization in Web Apps*. Recuperado el 18 de abril de 2025, de https://plotly.com/medical-dashboards/
- Ramírez, J. L., et al. (2024). Comparative Study of Median vs. Kalman Filters for Noise Reduction in Rehabilitation Motion Data. *Medical Engineering & Physics*, 113, 103982.
- Reglamento (UE) 2016/679 del Parlamento Europeo y del Consejo, de 27 de abril de 2016 (GDPR).
- Shortliffe, E. H., & Cimino, J. J. (2013). *Biomedical Informatics: Computer Applications in Health Care and Biomedicine*. Springer.

- Supabase. (2023). *Supabase Documentation*. Recuperado el 22 de abril de 2025, de https://supabase.com/docs
- Team M. (2021). *MediaPipe Hands*. Google AI Blog. Recuperado el 17 de abril de 2025, de https://ai.googleblog.com/2019/08/on-device-real-time-hand-tracking-with.html
- Toro, P. d. (2023). *Avances Tecnológicos en fisioterapia y su impacto en la rehabilitación*. (Citar fuente original si se encuentra).
- Tyromotion GmbH. (s.f.). *TyroS Software*. Recuperado el 18 de abril de 2025, de https://tyromotion.com/en/products/tyros-software/
- Unwin, A. (2023). *Graphics for Medical Time Series Data*. CRC Press.
- Wade, R. (2022). Secure Authentication in Streamlit for Healthcare Apps. *Journal of Open Source Medical Software*, 7(12), e45.
- Zhou, H., Hu, H., & Tao, Y. (2018). SwiM: A System for Wearable Interactive Motion Rehabilitation Based on Inertial Sensors. *Sensors*, 18(1), 180.

# ANEXO 1: MANUAL DE USUARIO

## Aplicación Web para la Gestión y Análisis de Datos de Rehabilitación

## 1. Introducción

Bienvenido/a al manual de usuario de la aplicación web para la gestión y análisis de datos de rehabilitación. Esta herramienta ha sido diseñada para facilitar a los terapeutas el registro, seguimiento y análisis del progreso de los pacientes sometidos a terapias de rehabilitación, especialmente aquellas que involucran la captura de datos de movimiento.

Este manual le guiará a través de las principales funcionalidades de la aplicación, desde el inicio de sesión hasta la visualización y análisis de los datos de las sesiones.

## 2. Acceso a la Aplicación y Autentificación

**2.1 Acceso:** Para acceder a la aplicación, abra su navegador web preferido (se recomienda Google Chrome, Mozilla Firefox o Microsoft Edge actualizados) y diríjase a la URL proporcionada para la aplicación.

**2.2. Inicio de Sesión y Registro:** Al acceder, verá una barra lateral a la izquierda con las opciones de "Iniciar Sesión" o "Registrarse".

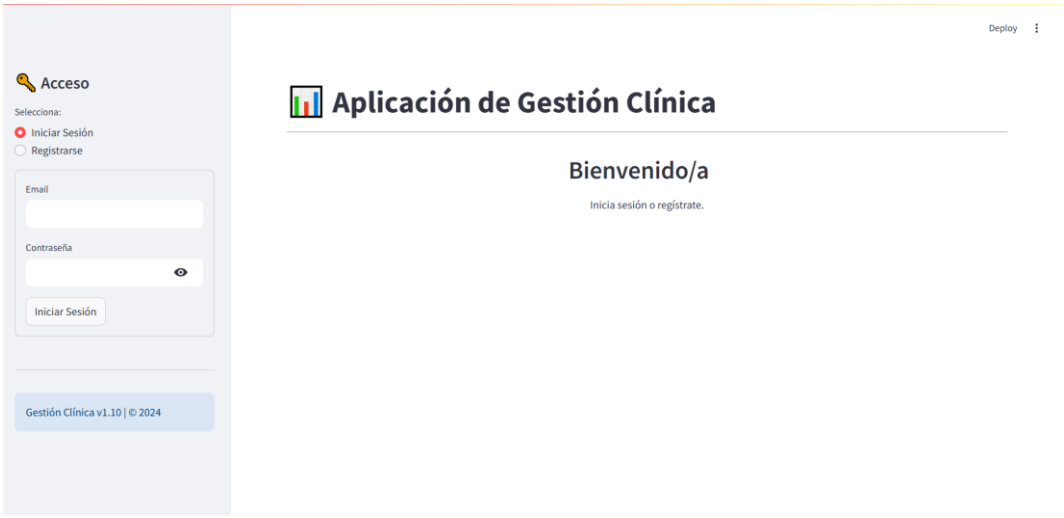


*Figura A 1: Interfaz de inicio de sesión y registro Fuente: Aplicación TFG*

- **Para Iniciar Sesión:**
  - Seleccione la opción "Iniciar Sesión".
  - Ingrese su dirección de correo electrónico y contraseña en los campos correspondientes.
  - Haga clic en el botón "Iniciar Sesión".

- Si las credenciales son correctas y su cuenta está verificada, accederá al panel principal de la aplicación.
- **Para Registrarse (Nuevos Usuarios):**
- Seleccione la opción "Registrarse".
- Ingrese una dirección de correo electrónico válida y una contraseña (mínimo 6 caracteres).
- Haga clic en el botón "Registrarse".
- Recibirá un correo electrónico de confirmación en la dirección proporcionada. Deberá hacer clic en el enlace de dicho correo para verificar su cuenta antes de poder iniciar sesión. Revise su carpeta de spam si no lo recibe.

- **Cierre de Sesión:** Una vez dentro de la aplicación, en la barra lateral, bajo su información de usuario, encontrará el botón "🔒 Cerrar Sesión". Púlselo para finalizar su sesión de forma segura.

## 3. Panel principal y Navegación

Tras iniciar sesión, accederá al panel principal. Desde aquí podrá navegar a las diferentes secciones de gestión y análisis.


*Figura A 2: Panel principal de la aplicación. Fuente: Aplicación TFG*

La navegación se divide principalmente en:

- **Gestión de Tablas Generales:** Permite administrar la información de Pacientes, Terapeutas y Tipos de Terapia.
- **Gestión de Sesiones:** Permite ver el historial de sesiones, registrar nuevas sesiones (con el script de ratón, de forma manual, o simular sesiones M3Display) y analizar el progreso de los pacientes.

# 4. Gestión de Tablas Generales

**4.1. Acceso a la Gestión de Tablas:** En el panel principal, haga clic en los botones "👤 Pacientes", "👤 Terapeutas" o "⚙ Terapias" para acceder a la gestión de cada entidad.

**4.2. Visualización y Eliminación de Registros:** Al seleccionar una tabla, se mostrarán los registros existentes en un formato tabular interactivo.

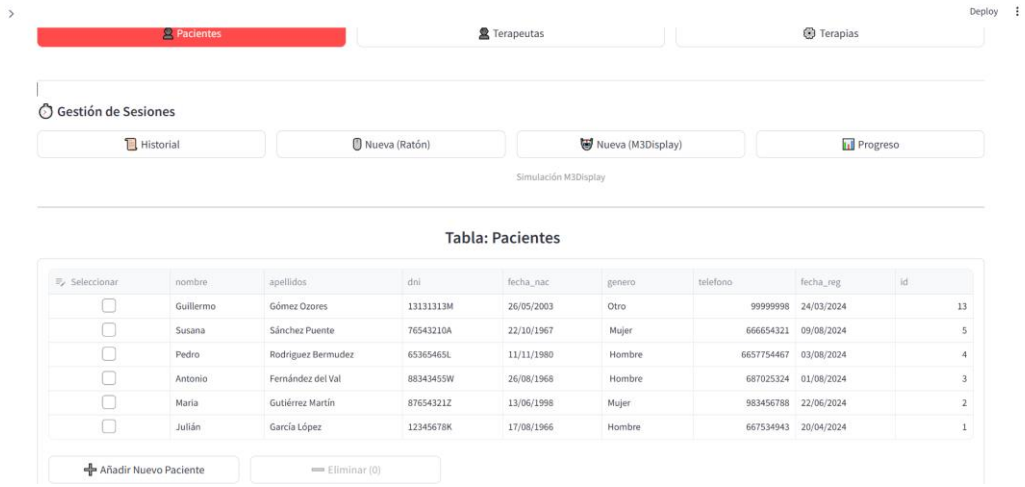


*Figura A 3:Vista de gestión de la tabla Pacientes. Fuente: Aplicación TFG*

- **Visualización:** La tabla muestra la información de cada registro. Puede que necesite desplazarse horizontalmente para ver todas las columnas.
- **Eliminación:**
    - Marque la casilla "Seleccionar" de las filas que desea eliminar.
    - Haga clic en el botón "— Eliminar ([n])", donde [n] es el número de filas seleccionadas.
    - Confirme la acción si se le solicita.

**4.3. Añadir Nuevos Registros:**

- En la vista de gestión de la tabla correspondiente (ej. Pacientes), haga clic en el botón "+ Añadir Nuevo Paciente" (o el equivalente para Terapeutas/Terapias).

- Se le redirigirá a un formulario de adición.

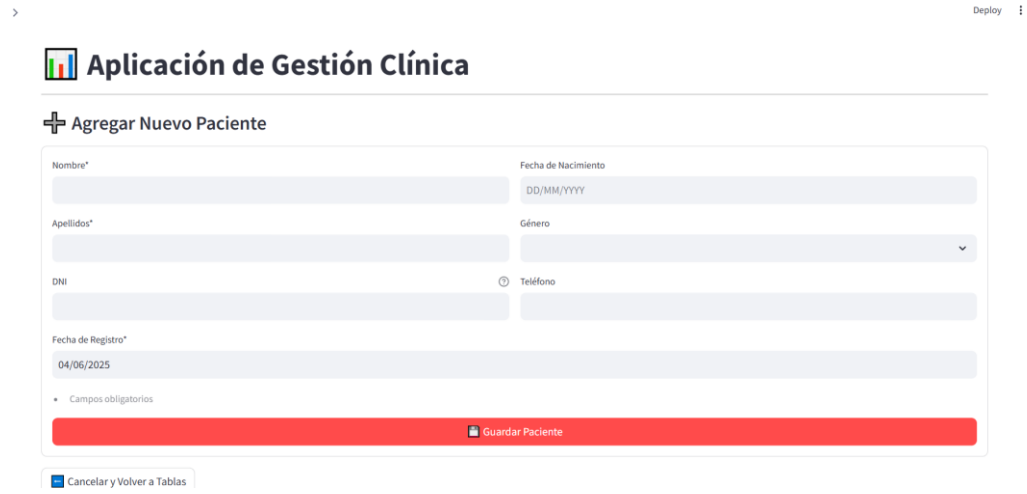


*Figura A 4: Formulario para agregar un nuevo paciente. Fuente: Aplicación TFG*

- Complete los campos del formulario. Los campos marcados con un asterisco (*) son obligatorios.
- Haga clic en "💾 Guardar [Entidad]".
- Si los datos son válidos, el registro se guardará y será redirigido de nuevo a la lista de la tabla.

## 5. Gestión de Sesiones de Rehabilitación

### 5.1. Ver Historial de Sesiones:

1. En el panel principal, en la sección "Gestión de Sesiones", haga clic en "📜 Historial".
2. Se mostrará una tabla con todas las sesiones de rehabilitación registradas.

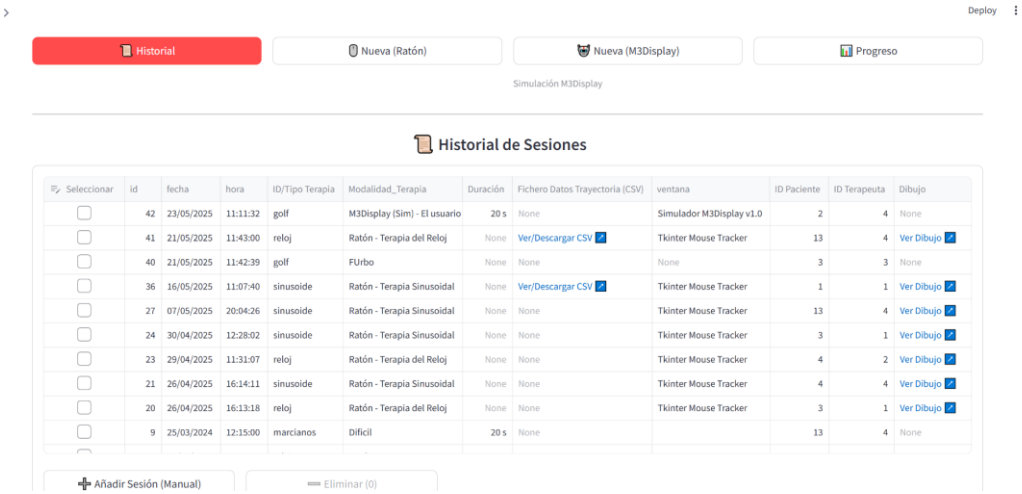


*Figura A 5: Vista del historial de sesiones. Fuente: Aplicación TFG*

3. **Eliminar Sesiones:** Similar a la gestión de tablas generales, puede seleccionar sesiones y eliminarlas.
4. **Ver Detalles y Análisis de una Sesión:**
    - Marque la casilla "Seleccionar" de **una única sesión** de la que desee ver los detalles.

*Se expandirá automáticamente una sección debajo de la tabla con la información detallada, análisis y acciones para esa sesión.*


*Figura A 6:Detalles y análisis de la sesión seleccionada. Fuente: Aplicación TFG*

- **Información Contextual:** Verá datos del paciente, terapeuta y tipo de terapia asociados.
- **Dibujo Asociado:** Si la sesión tiene una imagen de dibujo, se mostrará. Puede hacer clic en "Abrir ↗" para verla en una nueva pestaña.
- **Análisis de Trayectoria:** Si la sesión tiene datos numéricos de trayectoria almacenados, se mostrarán:
    - **Métricas Calculadas:** Duración, Longitud Recorrida, Velocidad Media.
    - **Gráfico de Trayectoria:** Una visualización estática del movimiento X vs Y.
- **Generar/Descargar GIF:**
    - Haga clic en "Generar GIF (Datos BD)" para crear una animación del movimiento a partir de los datos almacenados.
    - Una vez generado, el GIF se mostrará y aparecerá un botón "↓ Descargar GIF (BD)".
    - También puede subir un archivo CSV de trayectoria manualmente en esta sección para generar un GIF a partir de él si los datos de BD no estuvieran disponibles o quisiera probar con otros.
- **Archivos Adjuntos:** Si la sesión tiene un archivo CSV de trayectoria almacenado en Supabase Storage, aparecerá un enlace para "Descargar [nombre_archivo_csv]".

### 5.2. Registrar Nueva Sesión con Captura de Movimiento (Ratón):

1. En el panel principal, en "Gestión de Sesiones", haga clic en "🖱 Nueva (Ratón)".

2. Complete el formulario de configuración de Nueva Sesión con Ratón

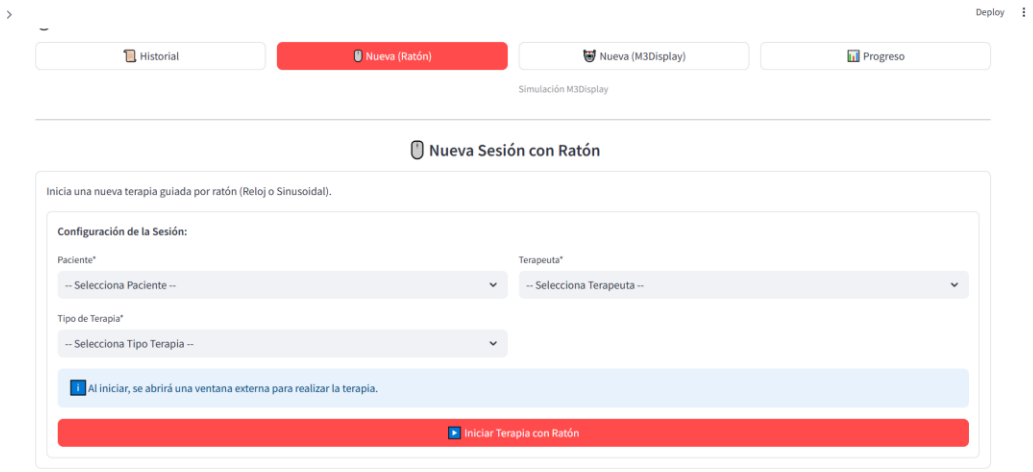


*Figura A 7: Formulario de configuración para una nueva sesión con ratón. Fuente: Aplicación TFG*

3. Haga clic en "▶ Iniciar Terapia con Ratón".

4. Se abrirá una **ventana externa** (la aplicación Tkinter *codigo_raton.py*). Siga las instrucciones en esa ventana para realizar la tarea de seguimiento.

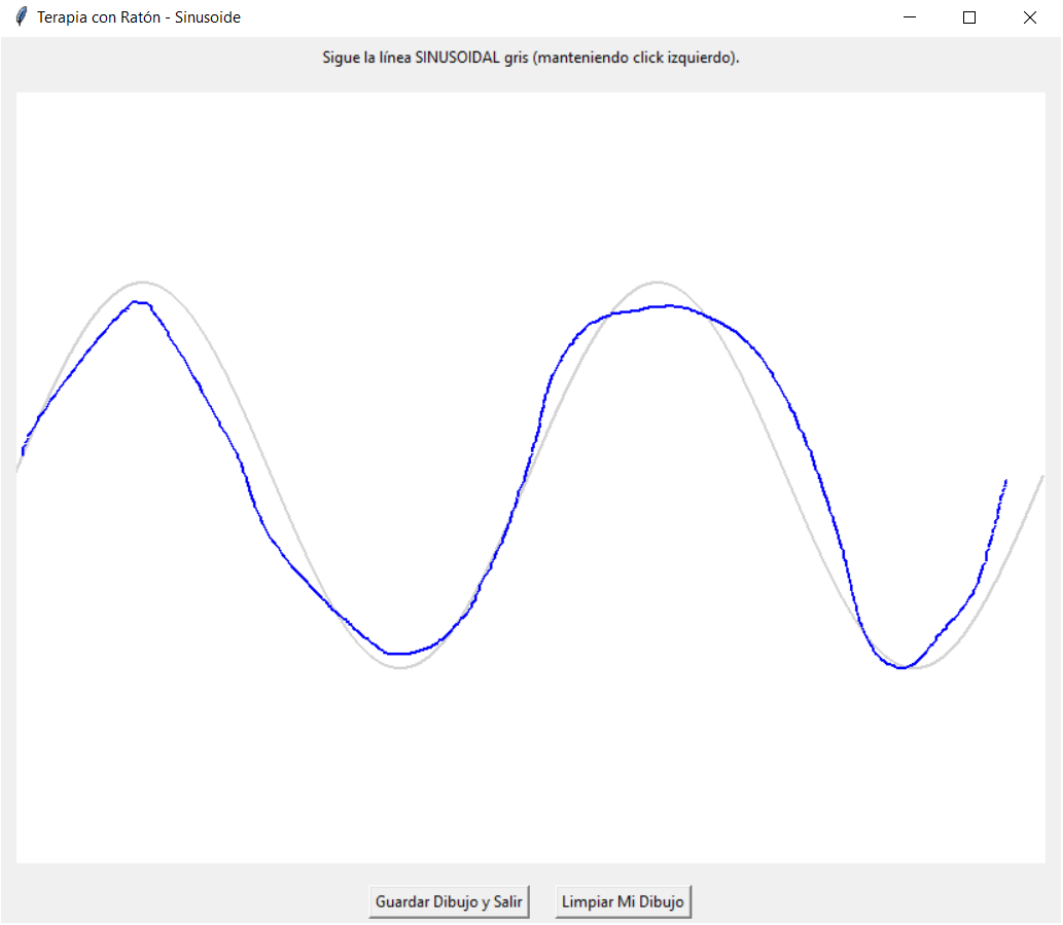


*Figura A 8: Interfaz del script de captura de movimiento codigo_raton.py Fuente: Aplicación TFG.*

Al finalizar la tarea en la ventana externa, haga clic en el botón **"Guardar Dibujo y Salir"** dentro de esa ventana.

5. La ventana externa se cerrará. Vuelva a la aplicación Streamlit. La aplicación procesará los archivos generados (imagen y CSV de trayectoria), los subirá a Supabase, guardará la sesión y los datos de trayectoria, y mostrará mensajes de estado. Si todo es correcto, será redirigido al historial de sesiones.

### 5.3. Registrar Nueva Sesión (Manual):

1. Desde el "Historial de Sesiones" o el panel principal (si se añade un botón directo), acceda al formulario "➕ Agregar Nueva Sesión (Manual)".

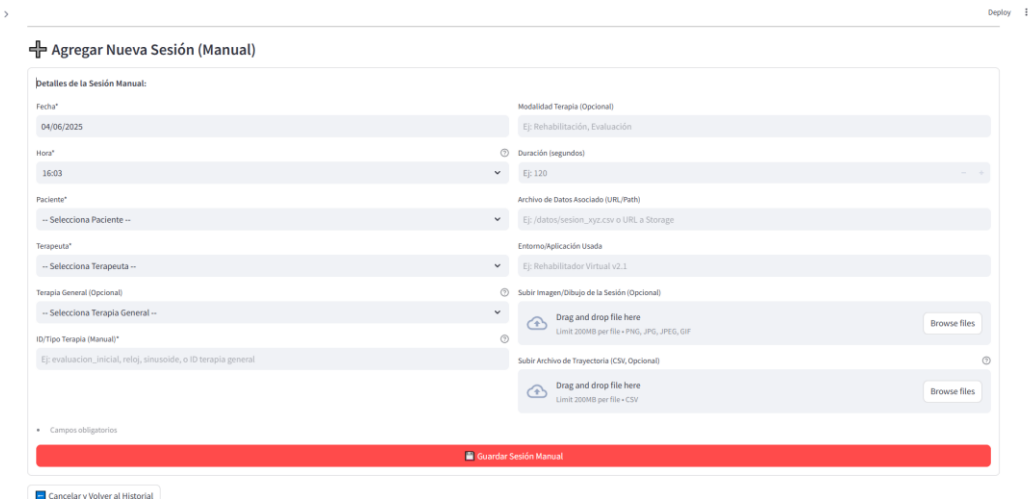


*Figura A 9: Formulario para agregar una nueva sesión manualmente. Fuente: Aplicación TFG.*

2. Complete todos los campos obligatorios (*): Fecha, Hora, Paciente, Terapeuta, ID/Tipo Terapia (Manual).

3. Opcionalmente:

- Complete los demás campos (Modalidad, Duración, URL de Fichero de Datos, Entorno).
- Suba una imagen del resultado de la sesión usando el widget "Subir Imagen/Dibujo...".
- Suba un archivo CSV con los datos numéricos de trayectoria usando el widget "Subir Archivo de Trayectoria (CSV...)" (formato: Timestamp;PosX;PosY, separador ';', decimal ','). **Importante:** La subida de este CSV aquí tiene como objetivo principal que sus puntos se almacenen en la tabla DatosTrayectoria para su análisis, no necesariamente que el archivo CSV en sí mismo se guarde en Supabase Storage (aunque el código actual también lo hace).

4. Haga clic en "💾 Guardar Sesión Manual".

5. La sesión se registrará. Si se subió un CSV de trayectoria, sus puntos se procesarán y almacenarán.

### 5.4. Registrar Nueva Sesión (Simulación M3Display):

1. En el panel principal, en "Gestión de Sesiones", haga clic en "👹 Nueva (M3Display)".

2. Complete el formulario de configuración, seleccionando Paciente, Terapeuta, un Tipo de Terapia general (debe haber terapias definidas en la tabla "Terapias") y la duración simulada.

3. Haga clic en "▶ Iniciar Sesión M3Display (Simulada)".

4. La aplicación simulará el progreso de la terapia. Al finalizar, se creará un registro de sesión con la modalidad "M3Display (Sim)". **Nota:** Este flujo es una simulación y no captura ni almacena datos reales de trayectoria.

## 6. Análisis de Progreso del Paciente

Esta sección permite visualizar la evolución de las métricas clave de un paciente a lo largo de todas sus sesiones registradas que contengan datos de trayectoria.

1. En el panel principal, en "Gestión de Sesiones", haga clic en "📊 Progreso".

   Seleccione un paciente de la lista desplegable.

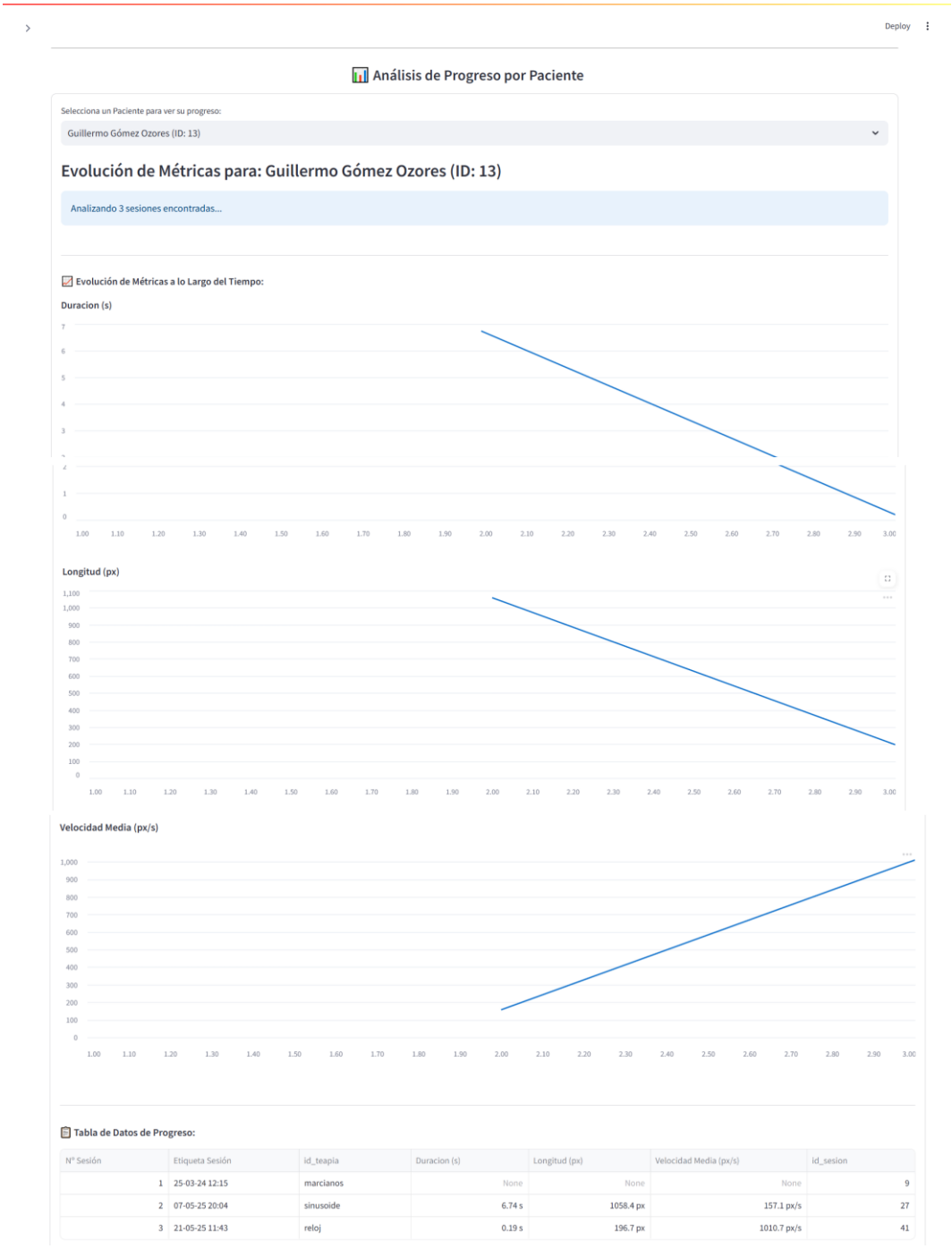


*Figura A 10: Vista de análisis de progreso longitudinal para un paciente seleccionado. Fuente: Aplicación TFG*

2. La aplicación cargará todas las sesiones del paciente, procesará los datos de trayectoria de cada una, calculará las métricas (Duración, Longitud, Velocidad Media) y mostrará:

- **Gráficos de Líneas:** Para cada métrica, un gráfico mostrará su evolución a lo largo de las sesiones del paciente (ordenadas cronológicamente o por número de sesión).
- **Tabla de Datos de Progreso:** Una tabla resumen con las métricas calculadas para cada sesión.

## 7. Resolución de Problemas Comunes

- **Error al Iniciar Sesión:** Verifique sus credenciales. Si es un usuario nuevo, asegúrese de haber confirmado su correo electrónico.
- **Sesión Expirada:** Si la aplicación le indica que su sesión ha expirado, simplemente vuelva a iniciar sesión.
- **Error al Cargar Tablas (ArrowTypeError):** Puede indicar un problema con los tipos de datos en la base de datos. La aplicación intentará mostrar los datos con un visor alternativo. Contacte al administrador si el problema persiste.
- **GIF no se Genera:** Asegúrese de que la sesión seleccionada tenga datos de trayectoria numérica almacenados. Verifique que no haya errores durante la lectura del CSV (si usa la opción de subida manual).
- **Script de Ratón no se Inicia o Falla:** Asegúrese de que Python esté correctamente instalado y en el PATH del sistema donde se ejecuta la aplicación Streamlit (si es local). Verifique que el archivo *codigo_raton.py* esté en la ubicación correcta.