UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

TFG del Grado en Ingeniería Informática

Herramienta de gestión para el seguimiento de pacientes con video

Presentado por Alberto Lanchares Diez
en Universidad de Burgos — 26 de noviembre de 2025
Tutores: José Luis Garrido Labrador y José Miguel Ramírez Sanz

UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

D. José Luis Garrido Labrador y D. José Miguel Ramírez Sanz, profesores del departamento de Ingeniería Informática, Área de Lenguajes y Sistemas Informáticos,

Exponen:

Que el alumno D. Alberto Lanchares Diez, con DNI 71362969H, ha realizado el Trabajo final de Grado en Ingeniería Informática titulado Herramienta de gestión para el seguimiento de pacientes con video.

Y que dicho trabajo ha sido realizado por el alumno bajo la dirección del que suscribe, en virtud de lo cual se autoriza su presentación y defensa.

En Burgos, 26 de noviembre de 2025

Vº. Bº. del Tutor:	Vº. Bº. del co-tutor:
D. José Luis Garrido Labrador	D. José Miguel Ramírez Sanz

Resumen

En este primer apartado se hace una **breve** presentación del tema que se aborda en el proyecto.

Descriptores

Palabras separadas por comas que identifiquen el contenido del proyecto Ej: servidor web, buscador de vuelos, android ...

Abstract

A **brief** presentation of the topic addressed in the project.

Keywords

keywords separated by commas.

Índice general

Índice de figuras

Índice de tablas

1. Introducción

A medida que la población envejece y los servicios sanitarios se concentran principalmente en ciudades, muchas personas que residen en zonas rurales encuentran dificultades para acceder a terapias de rehabilitación. Esta situación afecta de manera especial a quienes padecen determinados problemas de salud crónicos, como la enfermedad del Parkinson. Ésta es una patología progresiva que provoca rigidez muscular, temblores y dificultad en el movimiento, y para la cual actualmente no existe cura. Sin embargo, mediante terapia, los pacientes pueden ralentizar el avance de la enfermedad, mejorar su movilidad y preservar durante más tiempo su independencia y calidad de vida. Las limitaciones de movilidad y la lejanía de los centros de referencia complican aún más el acceso, repercutiendo en el bienestar de los pacientes y en el esfuerzo de sus familias.

Ante esta problemática, surge **TerapiTrack**, una plataforma web diseñada para facilitar el acompañamiento y el seguimiento de terapias, principalmente para personas diagnosticadas con la enfermedad de Parkinson. Sin embargo, su enfoque flexible y modular permite también su uso en otros casos que requieran terapias a distancia, como personas con otras enfermedades crónicas. TerapiTrack busca eliminar barreras geográficas y de accesibilidad, adaptándose a las necesidades y capacidades de distintos tipos de pacientes.

1.1. Contexto del problema

La Enfermedad de Parkinson es una enfermedad neurodegenerativa que suele requerir una combinación de medicación y diversos ejercicios de fisioterapia o estimulación cognitiva, adaptados a cada fase y persona. En España, esta problemática se acentúa especialmente en la conocida “España

Vaciada", donde la mayoría de la población son personas de edad avanzada, grupo especialmente propenso a padecer este tipo de enfermedades. Para estos pacientes, acudir con regularidad a consultas y sesiones presenciales supone un reto considerable, no solo por la falta de servicios sanitarios especializados en muchas regiones rurales, sino también por la distancia a los centros de salud de referencia y las propias limitaciones físicas que provoca la enfermedad. Esta situación también se da en otros perfiles de pacientes que necesitan un seguimiento constante para evitar recaídas y para mantener su autonomía el mayor tiempo posible.

1.2. Propuesta de solución

El objetivo principal de TerapiTrack es acercar la rehabilitación y el control terapéutico al entorno cotidiano del paciente, apoyándose en la tecnología para romper barreras tradicionales. Las principales funcionalidades que aporta la herramienta son:

- Permitir que los profesionales sanitarios creen ejercicios y rutinas personalizadas.
- Ofrecer la posibilidad de organizar sesiones y ajustar las actividades según la evolución de cada paciente.
- Facilitar el seguimiento remoto mediante la grabación y evaluación de los ejercicios realizados.
- Proporcionar una vía de comunicación sencilla y segura entre pacientes y especialistas, ayudando a resolver dudas y ajustar tratamientos sin desplazamientos.
- Garantizar una interfaz comprensible, accesible y adaptada a usuarios con dificultades motoras o cognitivas.

1.3. Estructura de la memoria

La presente memoria se divide en diferentes capítulos que buscan ofrecer una visión global del trabajo realizado:

- **Capítulo 1. Introducción**: Expone el origen del proyecto y contextualiza el problema a resolver.

- **Capítulo 2. Objetivos del proyecto**: Presenta los objetivos perseguidos, tanto generales como específicos.
- **Capítulo 3. Conceptos teóricos**: Revisa los fundamentos técnicos y sanitarios necesarios para comprender la solución desarrollada.
- **Capítulo 4. Técnicas y herramientas**: Describe las tecnologías, herramientas y metodologías empleadas.
- **Capítulo 5. Aspectos relevantes del desarrollo**: Explica las fases principales del desarrollo y las decisiones adoptadas.
- **Capítulo 6. Trabajos relacionados**: Analiza otras soluciones similares y posiciona TerapiTrack respecto a ellas.
- **Capítulo 7. Conclusiones y líneas de trabajo futuras**: Resume los resultados alcanzados y plantea posibles mejoras o ampliaciones.

Junto a estos capítulos, se incluyen anexos donde se recopila documentación técnica, manuales y otra información complementaria.

1.4. Materiales entregados

Para facilitar la validación, el uso y la posible evolución del sistema, se entrega junto con la memoria un conjunto de materiales adicionales:

- Código fuente completo del proyecto y su historial de versiones.
- Definición de la estructura de la base de datos y un conjunto de datos de prueba.
- Documentación técnica detallada en los anexos.
- Manual de usuario y de instalación.
- Batería de pruebas del sistema, con los resultados obtenidos.

Este conjunto de materiales pretende que cualquier persona interesada pueda comprender, utilizar y mejorar la solución presentada, contribuyendo así a un mejor acceso a la rehabilitación y al acompañamiento terapéutico a través de medios digitales.

2. Objetivos del proyecto

Este capítulo expone de manera clara los propósitos principales del desarrollo de TerapiTrack, distinguiendo entre los objetivos funcionales —enfocados en las necesidades detectadas para usuarios y profesionales— y los objetivos técnicos, que marcan las directrices para la construcción y despliegue de la plataforma. Esta diferenciación permite visualizar tanto la utilidad práctica del sistema como los aspectos tecnológicos que respaldan su funcionamiento.

2.1. Objetivos funcionales

- Gestionar diferentes perfiles de usuario (administrador, paciente, personal sanitario: médico, terapeuta, psicólogo, enfermero) con permisos diferenciados.
- Permitir la vinculación de pacientes con los profesionales responsables de su seguimiento.
- Ofrecer una biblioteca de ejercicios y recursos terapéuticos en formato audiovisual, filtrable según necesidades clínicas.
- Facilitar la creación y programación de sesiones personalizadas de rehabilitación.
- Registrar y evaluar el progreso de cada paciente mediante grabaciones, informes y revisión de ejercicios.
- Impulsar la comunicación bidireccional y las notificaciones entre los usuarios y los profesionales, para un acompañamiento real y adaptado.

- Asegurar la trazabilidad de la evolución terapéutica, con un acceso a la información ajustado a la privacidad y rol de cada usuario.

2.2. Objetivos técnicos

- Desarrollar la aplicación sobre un framework robusto (Flask) y modular, que facilite tanto el mantenimiento futuro como la ampliación de funcionalidades.
- Diseñar e implementar una base de datos relacional segura y eficiente, que garantice la integridad y la protección de la información sensible.
- Crear una interfaz accesible y sencilla, especialmente adaptada a personas con dificultades motoras y/o cognitivas.
- Aplicar mecanismos de cifrado y control de accesos para preservar la privacidad y la seguridad de los datos almacenados y transferidos.
- Implementar el despliegue de la aplicación en la nube (por ejemplo, a través de Heroku), asegurando así disponibilidad remota y alta accesibilidad.
- Desarrollar una batería de pruebas automatizadas que permita validar el correcto funcionamiento y la calidad del sistema.

3. Conceptos teóricos

Este capítulo sintetiza los aspectos teóricos esenciales para entender la naturaleza y el desarrollo del proyecto TerapiTrack. Se abordan tanto conceptos del ámbito sanitario como fundamentos tecnológicos y de ingeniería del software aplicados durante la implementación.

3.1. Telemedicina y rehabilitación remota

La telemedicina hace referencia al uso de las tecnologías de la información y la comunicación para prestar servicios sanitarios a distancia. En el caso concreto de la rehabilitación, estas herramientas permiten realizar el seguimiento y diseño de terapias físicas o cognitivas sin que paciente y profesional coincidan en el mismo lugar, facilitando así la continuidad asistencial en enfermedades crónicas como el Parkinson y superando barreras geográficas.

3.2. Accesibilidad y usabilidad en aplicaciones sanitarias

Las aplicaciones orientadas a la salud deben contemplar criterios estrictos de accesibilidad y usabilidad para garantizar que cualquier paciente, independientemente de sus capacidades motrices o cognitivas, pueda manejar la plataforma con facilidad. Algunos principios fundamentales son:

- **Perceptible**: Que todos los elementos y mensajes sean claros y apreciables por los usuarios.

- **Operable**: Los controles y formularios deben ser sencillos y adaptados para personas con limitaciones motoras.
- **Comprensible**: El lenguaje, la navegación y las acciones deben ser directas y sin ambigüedades.
- **Robusto**: El sistema ha de ser compatible con distintos dispositivos y tecnologías de apoyo.

3.3. Arquitectura Modelo-Vista-Controlador (MVC)

El patrón de diseño MVC es una estrategia habitual para el desarrollo de aplicaciones web, como TerapiTrack, y divide la lógica en tres partes diferenciadas:

- **Modelo**: Encargado de la gestión y estructura de los datos (usuarios, pacientes, profesionales, ejercicios, sesiones, etc.).
- **Vista**: Responsable de mostrar la información utilizando una interfaz clara, jerárquica y accesible.
- **Controlador**: Interpreta las acciones del usuario, comunica modelo y vista, y coordina la lógica de la aplicación (ejemplo: iniciar sesión, asignar sesiones, evaluar ejercicios, etc.).

3.4. Bases de datos relacionales y modelos de información

El almacenamiento de información clínica o terapéutica requiere garantizar la integridad, trazabilidad y seguridad de los datos. Se utilizan bases de datos relacionales (en este caso, SQLite mediante SQLAlchemy) para organizar usuarios, pacientes, profesionales, ejercicios y sesiones. Destacan dos conceptos clave:

Normalización

El proceso de normalización evita la redundancia y asegura la integridad referencial. En TerapiTrack, las tablas principales (Usuario, Paciente, Pro-

fesional, Ejercicio, Sesión) mantienen relaciones claras y funcionales entre sí.

Relaciones de muchos a muchos

Para reflejar la realidad donde un paciente puede trabajar con varios profesionales, y un ejercicio puede asignarse a muchas sesiones, se utilizan tablas intermedias que permiten una asignación flexible y transparente.

3.5. Tecnologías empleadas en el desarrollo del sistema

En la construcción de TerapiTrack se han utilizado las siguientes tecnologías principales:

- **Flask**: Framework web en Python para la gestión de rutas y lógica de servidor (backend).
- **SQLAlchemy**: Herramienta ORM que gestiona la interacción con bases de datos relacionales.
- **SQLite**: Motor de base de datos embebido, ideal para pruebas y despliegues sencillos.
- **Jinja**: Motor de plantillas para renderizado dinámico de vistas e interfaces HTML.
- **Bootswatch**: Colección de temas para Bootstrap que facilita una apariencia visual moderna y accesible.
- **Jira**: Plataforma de gestión de proyectos y seguimiento de tareas.
- **GitBash** y **GitHub**: Control del versionado de código y colaboración entre desarrolladores.

3.6. Consideraciones de seguridad y privacidad

Por tratarse de datos sensibles, el desarrollo tiene en cuenta aspectos de seguridad:

- Cifrado de contraseñas.
- Control de acceso por roles (administrador, profesional, paciente).
- Políticas de retención y eliminación de vídeos terapéuticos.
- Uso de sesiones y validación para autenticación segura.

3.7. Buenas prácticas y validación

Se fomenta la calidad y mantenibilidad del software mediante:

- Pruebas automatizadas para comprobar el funcionamiento de módulos clave.
- Documentación de código, manuales de usuario y de instalación.
- Uso de metodologías ágiles en la organización del trabajo (por ejemplo, con Jira).

3.8. Ejemplos prácticos de uso de la plantilla LaTeX

A continuación se muestran algunos ejemplos de comandos útiles de la plantilla, siguiendo el formato recomendado por la documentación:

Secciones y subsecciones

Las secciones se incluyen con el comando:

```
\section{Nombre de sección}
```

Las subsecciones se incluyen de forma análoga:

```
\subsection{Nombre de subsección}
```

Subsubsecciones

Para mayor detalle se puede emplear el comando:

```
\subsubsection{Nombre de subsubsección}
```

3.9. Referencias

Las referencias a artículos, webs o libros se incluyen con cite, como en el siguiente ejemplo: cite [3]. Para citar varios recursos: [2, 1].

3.10. Imágenes

La plantilla dispone de comandos propios para incorporar imágenes, por ejemplo:

Figura 3.1: Escudo de la Universidad de Burgos

3.11. Listas de items

Existen diversas formas de listar:

- Primer ítem de la lista.
- Segundo ítem relevante.

1. Primer elemento.
2. Segundo elemento.

Herramienta	Ámbito	Descripción
Flask	Backend	Desarrollo de la lógica de servidor y API
SQLAlchemy	Base de datos	ORM para conexión y gestión relacional de datos
SQLite	Base de datos	Motor ligero y embebido para el almacenamiento de infor
Jinja	Plantillas	Renderizado dinámico de las páginas HTML
Bootswatch	Frontend	Apariencia visual basada en Bootstrap
Jira	Gestión	Organización, planificación y seguimiento del proyecto
GitBash	Versionado	Control del código mediante terminal
GitHub	Repositorio	Colaboración y almacenamiento del proyecto

Tabla 3.1: Herramientas utilizadas en el proyecto

Primer ítem Descripción más detallada.

Segundo ítem Segunda descripción relevante.

3.12. Tablas

Se pueden emplear comandos clásicos o los de la plantilla. Ejemplo:

Este capítulo proporciona así el contexto teórico y tecnológico que sustenta toda la memoria y el desarrollo de la aplicación.

4. Técnicas y herramientas

Este capítulo describe las metodologías, tecnologías y utilidades elegidas para el desarrollo de TerapiTrack, justificando cada decisión y recogiendo las principales ventajas de cada alternativa considerada.

4.1. Metodología de desarrollo

Gestión ágil con Jira

Se optó por una organización ágil utilizando **Jira** para la planificación y el control del trabajo. Esta herramienta ha permitido dividir el proyecto en sprints y tareas distribuidas en tableros, facilitando el seguimiento del progreso real y la adaptación ágil ante imprevistos. Jira es compatible con metodologías como Scrum y Kanban, permitiendo organizar el desarrollo por etapas incrementales y gestionar incidencias de forma eficiente.

4.2. Tecnologías de backend

Python y Flask

Flask ha sido el framework seleccionado para el desarrollo web backend debido a su flexibilidad, ligereza y la facilidad con la que se puede integrar con diferentes extensiones. Permite crear aplicaciones a medida y controlar con detalle la estructura de cada módulo, sin imponer capas innecesarias ni patrones rígidos. Otras plataformas como Django fueron consideradas, aunque Flask resultó más idóneo para un proyecto modular y de tamaño medio, donde la sencillez y el control sobre la arquitectura eran prioritarios.

Ventajas de Flask:

- Ligero, flexible y fácil de aprender
- Permite una estructura de proyecto a medida
- Ecosistema de extensiones muy abundante
- Ideal para aplicaciones modulares y APIs RESTful

SQLAlchemy

Para la interacción con la base de datos se ha implementado **SQLAlchemy**, un ORM ampliamente utilizado en el ecosistema Python. SQLAlchemy permite gestionar modelos y relaciones mediante objetos, evitando la escritura directa de SQL y facilitando la migración o ampliación futura del sistema. Su uso minimiza errores y simplifica la validación de datos, además de proporcionar portabilidad entre distintos motores de base de datos.

Alternativas

Se evaluó el uso de otras bibliotecas ORM y la manipulación directa con SQL clásico, pero se descartaron para evitar redundancia y riesgos de inconsistencia.

4.3. Gestión de datos

Base de datos: SQLite

Durante el desarrollo se ha empleado **SQLite** por su sencillez y portabilidad (no requiere servidor y almacena toda la información en un solo archivo), idónea para pruebas y prototipos rápidos.

Herramienta complementaria: DB Browser for SQLite

Durante la fase inicial, DB Browser for SQLite facilitó la visualización y depuración de la base de datos desde un entorno gráfico.

4.4. Tecnologías de frontend

Bootswatch

El diseño y la apariencia de la interfaz de TerapiTrack se basan en **Bootswatch**, una colección de temas CSS construidos sobre Bootstrap. Esta decisión permitió aplicar rápidamente estilos accesibles y coherentes, favoreciendo la navegación intuitiva y el cumplimiento de estándares WCAG de accesibilidad web.

HTML5, JavaScript y Jinja

Se empleó **HTML5** como base estructural de las páginas, **JavaScript** para funcionalidades interactivas y la grabación web, y el motor **Jinja** para el renderizado dinámico del frontend en Flask. La combinación de estas tecnologías aporta flexibilidad y compatibilidad con todos los navegadores modernos.

4.5. Testing y aseguramiento de la calidad

Pytest

El testeo automatizado se llevó a cabo con **Pytest**, por su sintaxis sencilla, el soporte de fixtures reutilizables y la facilidad para parametrizar y agrupar pruebas. Esta herramienta ha permitido obtener cobertura completa en los módulos críticos y facilita la detección precoz de errores antes del despliegue.

4.6. Despliegue e infraestructura

Heroku

Heroku se eligió como plataforma de despliegue por su facilidad de uso, integración con Git y escalado automático de recursos según la carga de trabajo. El sistema de despliegue continuo con *git push* agiliza las pruebas y actualizaciones en entornos cloud, permitiendo centrarse en la funcionalidad sin preocuparse por la gestión del servidor.

4.7. Control de versiones y colaboración

Git y GitHub

Toda la gestión del código se apoyó en **Git** como sistema distribuido de control de versiones y **GitHub** como repositorio central, lo que facilitó el trabajo colaborativo y la trazabilidad de los cambios a lo largo de todas las iteraciones. El uso de GitHub permitió registrar issues, realizar revisiones de código y mantener sincronizado el trabajo entre las distintas ramas del repositorio.

4.8. Otras herramientas de desarrollo

- **Visual Studio Code**: Entorno principal de edición y depuración, con extensiones dedicadas para Python, Flask y control de versiones.
- **Overleaf**: Redacción colaborativa de esta memoria en LaTeX, facilitando la organización de capítulos y anexos.
- **Draw.io**: Creación de diagramas para representar la arquitectura y el flujo de datos del sistema.

4.9. Comparativas y justificación de elecciones

A lo largo del desarrollo, se compararon tecnologías tanto por facilidad de integración como por el soporte ofrecido por la comunidad, documentación y curva de aprendizaje. Herramientas como Flask y SQLAlchemy se adaptaron perfectamente a los requisitos de modularidad y personalización, mientras que plataformas como Heroku y GitHub ofrecieron un entorno sencillo y ágil para desplegar y gestionar el ciclo de vida completo del proyecto.

Este conjunto de herramientas ha permitido desarrollar TerapiTrack de forma organizada, eficiente y con una sólida base para abordar futuras escalas y mejoras del sistema.

5. Aspectos relevantes del desarrollo del proyecto

Este apartado pretende recoger los aspectos más interesantes del desarrollo del proyecto, comentados por los autores del mismo. Debe incluir desde la exposición del ciclo de vida utilizado, hasta los detalles de mayor relevancia de las fases de análisis, diseño e implementación. Se busca que no sea una mera operación de copiar y pegar diagramas y extractos del código fuente, sino que realmente se justifiquen los caminos de solución que se han tomado, especialmente aquellos que no sean triviales. Puede ser el lugar más adecuado para documentar los aspectos más interesantes del diseño y de la implementación, con un mayor hincapié en aspectos tales como el tipo de arquitectura elegido, los índices de las tablas de la base de datos, normalización y desnormalización, distribución en ficheros3, reglas de negocio dentro de las bases de datos (EDVHV GH GDWRV DFWLYDV), aspectos de desarrollo relacionados con el WWW... Este apartado, debe convertirse en el resumen de la experiencia práctica del proyecto, y por sí mismo justifica que la memoria se convierta en un documento útil, fuente de referencia para los autores, los tutores y futuros alumnos.

6. Trabajos relacionados

Este apartado sería parecido a un estado del arte de una tesis o tesina. En un trabajo final grado no parece obligada su presencia, aunque se puede dejar a juicio del tutor el incluir un pequeño resumen comentado de los trabajos y proyectos ya realizados en el campo del proyecto en curso.

7. Conclusiones y Líneas de trabajo futuras

Todo proyecto debe incluir las conclusiones que se derivan de su desarrollo. Éstas pueden ser de diferente índole, dependiendo de la tipología del proyecto, pero normalmente van a estar presentes un conjunto de conclusiones relacionadas con los resultados del proyecto y un conjunto de conclusiones técnicas. Además, resulta muy útil realizar un informe crítico indicando cómo se puede mejorar el proyecto, o cómo se puede continuar trabajando en la línea del proyecto realizado.

Bibliografía

[1] Zachary J Bortolot and Randolph H Wynne. Estimating forest biomass using small footprint lidar data: An individual tree-based approach that incorporates training data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 59(6):342–360, 2005.

[2] John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, 1992.

[3] Wikipedia. Latex — wikipedia, la enciclopedia libre. https://es.wikipedia.org/w/index.php?title=LaTeX&oldid=84209252, 2015. [Internet; descargado 30-septiembre-2015].