



ugr | Universidad
de **Granada**

TRABAJO FIN DE GRADO
INGENIERÍA EN INFORMÁTICA

Sugerencias de juegos con LangChain

Desarrollo de un sistema web de recomendaciones de videojuegos personalizadas a través de LangChain

Autor: Juan Manuel Garzón Ferrer

Director: Ignacio Javier Pérez Gálvez



ESCUELA TÉCNICA SUPERIOR DE INGENIERÍAS INFORMÁTICA Y DE
TELECOMUNICACIÓN
Granada, junio de 2025

Prefacio

Sugerencias de juegos con LangChain: Desarrollo de un sistema web de recomendaciones de videojuegos personalizadas a través de LangChain

Juan Manuel Garzón Ferrer

Palabras clave: TFG, LangChain, LangGraph, LangSmith, modelos generativos de lenguajes, recomendaciones, sugerencias, videojuegos, recomendación de videojuegos, sugerencias de videojuegos.

Resumen

Este proyecto se centra en la creación de una plataforma web que ofrezca recomendaciones personalizadas de videojuegos, utilizando la novedosa tecnología Langchain como aspecto clave para orquestar no solo las preferencias del usuario sino también las opiniones de diferentes LLMs como ChatGPT o GEMINI. La plataforma permitirá tanto a usuarios registrados como no registrados interactuar con el sistema, proporcionando recomendaciones basadas en los juegos que han jugado o en nuevas preferencias introducidas en tiempo real. Los usuarios registrados podrán beneficiarse de un historial que mejora las recomendaciones con el tiempo, mientras que los usuarios no registrados recibirán sugerencias inmediatas basadas únicamente en la información proporcionada en la sesión actual y novedades del momento.

La plataforma se integrará con APIs de los diferentes LLMs a través de LangChain para obtener datos extra sobre los videojuegos, como nuevos lanzamientos o clásicos de antaño, popularidad en comunidades específicas, u otros aspectos relevantes como accesibilidad y disponibilidad. El sistema analizará esta información para ofrecer recomendaciones adaptadas a los gustos personales, garantizando que los usuarios descubran nuevos juegos adecuados a sus intereses y necesidades.

Además, el uso de LangChain permite utilizar múltiples LLMs, lo que optimiza el rendimiento del sistema al facilitar la consulta de diferentes fuentes de datos en paralelo. Esto no solo mejora la rapidez de las recomendaciones, sino que también enriquece la calidad de las respuestas al incorporar

una variedad de perspectivas y enfoques. LangChain proporciona una arquitectura flexible que se adapta a las necesidades del proyecto, permitiendo escalar y mejorar continuamente las capacidades de la plataforma conforme se integran nuevos datos y se reciben más interacciones de los usuarios.

Suggestions for games with LangChain: Development of a web system for personalized video game recommendations using LangChain.

Juan Manuel Garzón Ferrer

Keywords: Bachelor's Thesis, LangChain, LangGraph, LangSmith, large language model, LLMs recommendations, suggestions, video games, video game recommendation, video game suggestions.

Abstract

This project focuses on the creation of a web platform that offers personalized video game recommendations, using the innovative LangChain technology as a key aspect to orchestrate not only user preferences but also the opinions of different LLMs such as ChatGPT or GEMINI. The platform will allow both registered and non-registered users to interact with the system, providing recommendations based on the games they have played or new preferences introduced in real-time. Registered users will benefit from a history that improves recommendations over time, while non-registered users will receive immediate suggestions based solely on the information provided during the current session and current trends.

The platform will integrate with APIs from various LLMs via LangChain to collect additional data about video games, such as new releases or classic titles, popularity in specific communities, or other relevant aspects like accessibility and availability. The system will analyze this information to offer recommendations tailored to personal tastes, ensuring that users discover new games suited to their interests and needs.

Furthermore, the use of LangChain allows the utilization of multiple LLMs, optimizing the system's performance by enabling queries from different data sources in parallel. This not only speeds up the recommendations but also enhances the quality of the responses by incorporating a variety of perspectives and approaches. LangChain provides a flexible architecture that adapts to the project's needs, allowing for continuous scaling and improvement of the platform's capabilities as new data is integrated and more user interactions are received.

Yo, **Juan Manuel Garzón Ferrer**, alumno de la titulación **TITULACIÓN** de la **Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación de la Universidad de Granada**, autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Grado en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.

Fdo: Juan Manuel Garzón Ferrer

Granada a 15 de junio de 2025 .

D. **Ignacio Javier Pérez Gálvez**, Profesor del Departamento de Ciencias de la Computación e Inteligencia Artificial de la Universidad de Granada.

Informa:

Que el presente trabajo, titulado *Título del proyecto, Subtítulo del proyecto*, ha sido realizado bajo su supervisión por **Juan Manuel Garzón Ferrer**, y autorizamos la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada a 15 de junio de 2025.

Los directores:

Ignacio Javier Perez Galvez

Agradecimientos

Antonio Villena Díaz, estudiante de la Universitat Oberta de Catalunya,
por probar la aplicación y proporcionar su feedback.

Licencia

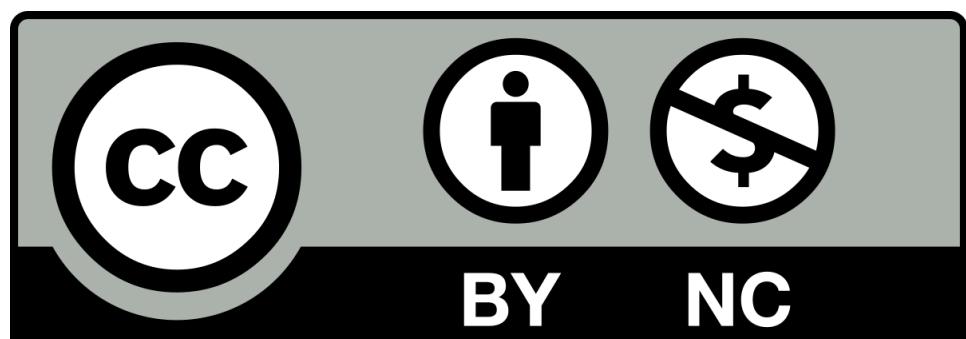
Este Trabajo Fin de Grado está publicado bajo la licencia **Creative Commons Atribución-NoComercial 4.0 Internacional (CC BY-NC 4.0)**. Esto significa que cualquier persona es libre de:

- Compartir: copiar y redistribuir el material en cualquier medio o formato.
- Adaptar: remezclar, transformar y construir a partir del material.

Bajo las siguientes condiciones:

- **Atribución:** Se debe dar crédito adecuado, proporcionar un enlace a la licencia e indicar si se han realizado cambios.
- **No comercial:** No se puede utilizar el material con fines comerciales sin autorización explícita del autor.

Más información sobre esta licencia en: <https://creativecommons.org/licenses/by-nc/4.0/deed.es>



*Licencia Creative Commons Atribución-NoComercial 4.0 Internacional
(CC BY-NC 4.0)
<https://creativecommons.org/licenses/by-nc/4.0/deed.es>*

Índice general

1. Introducción	17
2. Contexto y estado del arte	21
2.1. Contexto	22
2.1.1. Modelos generativos de lenguajes	22
2.1.2. LangChain	24
LangChain básico	25
Métodos	25
LangGraph	27
LangSmith	27
2.2. Estado del arte	28
3. Objetivos y metodología de desarrollo	31
3.1. Objetivos	32
3.2. Metodología de desarrollo	33
3.2.1. Presupuesto	34
4. Análisis	38
4.1. Obtención de Requisitos	39
4.2. Requisitos funcionales	41
4.3. Requisitos no funcionales	42
4.4. Requisitos de información	43
4.5. Actores	44
4.6. Casos de usos	45
4.6.1. Casos de uso del usuario no logueado	45
4.6.2. Casos de uso del usuario logueado	49
4.6.3. Casos de usos de administrador	54
5. Diseño	57
5.1. Diseño de la arquitectura	58
5.2. Elementos de la arquitectura	60
5.2.1. Base de datos	60
5.2.2. Backend	60
5.2.3. Frontend	61

ÍNDICE GENERAL

5.3. Diseño inicial de la interfaz	62
6. Implementación	65
6.1. Sistema operativo	66
6.2. GitHub	67
6.3. Entorno de desarrollo	68
6.4. Base de datos: MongoDB	69
6.5. BackEnd: Python, FastAPI y LangChain	70
6.5.1. DB	72
6.5.2. Utils	72
6.5.3. Modelos	72
6.5.4. Gestores	73
6.5.5. Servicios	73
6.5.6. Rutas	76
6.5.7. Main	76
6.6. FrontEnd: Angular	77
6.6.1. Componentes	78
Home	79
Registro	83
Login	83
Principal	84
Ajustes de cuenta	85
6.7. Puesta en marcha	87
7. Pruebas	89
7.1. Pruebas en la <i>base de datos</i>	90
7.2. Pruebas en el <i>backend</i>	91
7.2.1. Pruebas unitarias y de integración en el <i>backend</i>	92
7.3. Pruebas en el <i>frontend</i>	94
7.3.1. Pruebas unitarias en el <i>frontend</i>	94
7.4. Pruebas extras	96
7.5. Conclusión	97
8. Conclusiones	99
8.1. Conclusiones generales	100
8.2. Conclusiones alineadas con los objetivos	102
8.3. Alineamiento con los Objetivos de Desarrollo Sostenible (ODS)	104
8.4. Trabajos futuros	105
8.5. Conclusiones personales	107
Bibliografía	109

Índice de figuras

1.1. Captura de The Last of Us Parte II.	17
2.1. Ejemplos de modelos generativos de lenguajes.	23
2.2. Arquitectura productos LangChain.	24
3.1. Planificación en diagrama de Gantt.	33
4.1. Diagrama de casos de uso de usuario no logueado . . .	48
4.2. Diagrama de casos de uso de usuario logueado y ad- ministrador	55
5.1. Representación visual de la arquitectura	59
5.2. Diseño preliminar de la página de inicio.	62
5.3. Diseño preliminar de la página principal tras iniciar sesión.	62
5.4. Diseño preliminar de la página de ajustes.	63
6.1. Logo de Ubuntu.	66
6.2. Logo de GitHub.	67
6.3. Interfaz de Visual Studio Code.	68
6.4. Logo de MongoDB.	69
6.5. Logo de Python.	70
6.6. Logo de FastAPI.	71
6.7. Esquema recomendación básica.	74
6.8. Esquema recomendación personalizada.	75
6.9. Logo de Angular.	77
6.10. Imagen de home 1.	79
6.11. Imagen de home 2.	80
6.12. Modo claro.	80
6.13. Vista móvil 1.	81
6.14. Vista móvil 2.	82
6.15. Registro.	83
6.16. Inicio de sesión.	84
6.17. Página principal 1.	85

ÍNDICE DE FIGURAS

6.18. Página principal 2.	85
6.19. Página de ajustes de la cuenta.	86
7.1. Vista general de Swagger.	91
7.2. Vista del uso de un método en Swagger.	92
7.3. Vista de las herramientas de desarrollo del navegador.	94

Capítulo 1

Introducción

En la actualidad, los videojuegos se han convertido en el medio de entretenimiento más relevante, no solo a nivel económico, siendo la industria que más ingresos genera [1], sino también a nivel profesional, donde se utilizan como simuladores en el ámbito militar, en la recreación de lugares históricos, entre otros [2]. Además, desempeñan un papel clave en el ámbito social, formando grandes comunidades a nivel global. Historias como la de Mats Steen [3] demuestran que los videojuegos son mucho más que simples programas, ayudando a quienes no están familiarizados con este medio a comprender su impacto y valor.



Figura 1.1: **Captura de The Last of Us Parte II**, elegido Juego del Año 2020 por los principales medios de la industria. Este videojuego alcanza un nivel gráfico y narrativo sobresaliente. <https://static1.srcdn.com/wordpress/wp-content/uploads/2024/01/tlou2r-joel-horse.JPG>

Capítulo 1: Introducción

También, gracias a la emulación de consolas clásicas, las ofertas diarias, los videojuegos gratuitos, los títulos free-to-play y los servicios de suscripción como Game Pass o PS Plus, es posible acceder a una gran variedad de videojuegos de forma gratuita o a un precio reducido.

No obstante, el aumento en los costos y recursos necesarios para desarrollar un videojuego *triple A* (término que hace referencia a títulos de gran producción, con una alta inversión en desarrollo y publicidad) ha provocado que los lanzamientos de las sagas más conocidas se espacien cada vez más. Ejemplos de ello son el esperado *Grand Theft Auto VI*, que ha tardado 12 años en llegar desde el lanzamiento de *Grand Theft Auto V*, y *The Elder Scrolls VI*, cuya espera alcanza los más de 14 años desde *The Elder Scrolls V*, según la fecha de realización de este trabajo.

Debido a esta situación, si queremos disfrutar de experiencias similares a nuestros videojuegos favoritos, a menudo debemos recurrir a entregas anteriores de la misma saga que no jugamos en su momento o bien buscar alternativas en títulos de características similares.

Algunas plataformas actuales, como Xbox o Steam, recomiendan videojuegos en función de nuestras compras, de los títulos que se han jugado recientemente o de manera aleatoria. Sin embargo, estas recomendaciones no tienen en cuenta factores importantes, como los recursos de los que disponemos en ese momento, la accesibilidad, el deseo de cambiar de género de videojuego o la disponibilidad de otra plataforma, entre otros.

Ante esta necesidad, se propone una solución basada en una plataforma capaz de tener en cuenta todos estos factores de manera automatizada, con el objetivo de ofrecer recomendaciones personalizadas según las características y preferencias del usuario.

Parecería que contemplar todos los géneros de videojuegos, sus características y las distintas plataformas es una tarea ardua. No obstante, gracias al avance de la tecnología, contamos con modelos generativos de lenguajes que han sido entrenados con millones de datos, incluidos videojuegos.

Si aprovechamos esta tecnología a nuestro favor, podemos obtener fácilmente recomendaciones en tiempo real según los requisitos especificados, utilizando información de consultas previas e incluso teniendo en cuenta los videojuegos que poseemos en nuestra biblioteca, gracias a la vinculación de plataformas de videojuegos con nuestra propia plataforma.

Los avances recientes en inteligencia artificial han permitido el desarrollo de modelos generativos de lenguaje entrenados con grandes cantidades de datos, incluidos videojuegos, lo que abre nuevas posibilidades para crear recomendaciones más inteligentes y contextuales.

A lo largo de este trabajo, se explorarán distintas tecnologías relacionadas con el procesamiento del lenguaje natural y su aplicación en sistemas de recomendación, con el fin de desarrollar una plataforma eficiente, escalable e intuitiva.

El resto del documento se estructura de la siguiente manera:

Capítulo 1: Introducción

- **Capítulo 2:** Se explica qué son los modelos generativos de lenguaje y en qué consiste LangChain, además de analizar algunos trabajos previamente realizados.
- **Capítulo 3:** Se presentan los objetivos del proyecto, así como la planificación temporal y el presupuesto estimado.
- **Capítulo 4:** Se analizan los distintos requisitos de la aplicación, explicando también el proceso seguido para su obtención.
- **Capítulo 5:** Se describe la arquitectura general del sistema, detallando los diferentes componentes que lo conforman y su interacción.
- **Capítulo 6:** Se explica la implementación de la plataforma, incluyendo las tecnologías utilizadas, el desarrollo de cada módulo y el procedimiento para su ejecución.
- **Capítulo 7:** Se recogen las pruebas realizadas para verificar y validar el correcto funcionamiento de la aplicación.
- **Capítulo 8:** Se exponen las conclusiones obtenidas a lo largo del desarrollo del trabajo, así como posibles líneas futuras de mejora o ampliación.

Capítulo 2

Contexto y estado del arte

En este capítulo se especificarán las tecnologías que se utilizarán en el proyecto. Describiremos en qué consisten, qué nos ofrecen y cómo se aplican específicamente al desarrollo de esta plataforma.

Además, revisaremos otros trabajos realizados que empleen estas tecnologías, con un enfoque particular en *LangChain*, que constituye el pilar fundamental de este proyecto.

Capítulo 2: Contexto y estado del arte

2.1. Contexto

2.1.1. Modelos generativos de lenguajes

Un **modelo generativo de lenguajes** es un software diseñado para entender, analizar, interpretar y generar lenguaje humano mediante un ordenador. Estos sistemas permiten a las máquinas interactuar con los seres humanos de una manera más natural, utilizando los lenguajes que empleamos en la vida cotidiana. Para lograr este objetivo, los modelos generativos de lenguajes se basan en modelos de aprendizaje automático que, en términos generales, crean estructuras computacionales capaces de aprender patrones y estructuras lingüísticas a partir de grandes volúmenes de datos provenientes de diversos ámbitos, incluidos los videojuegos.

El desarrollo de un LLM (siglas en inglés de large language model) sigue un proceso estructurado que incluye varias etapas. En primer lugar, se diseñan modelos matemáticos y algoritmos que permiten a las máquinas "comprender" la estructura y la semántica del lenguaje. A continuación, estos modelos se entrenaan utilizando millones de datos lingüísticos, como textos, conversaciones y otras fuentes, lo que les permite adquirir la capacidad de realizar tareas complejas, como la traducción o la generación de texto en respuesta a preguntas o peticiones específicas.

Algunos de los modelos generativos de lenguajes más avanzados en la actualidad son **ChatGPT**, desarrollado por OpenAI, y **Gemini**, creado por Google. Ambos sistemas emplean tecnologías basadas en redes neuronales profundas y han alcanzado un nivel de sofisticación que les permite realizar tareas complejas, como la redacción de textos, la resolución de problemas y la generación de contenido con una precisión similar a la de un experto en la materia.

Gracias a su capacidad de comprensión, conocimiento y razonamiento, un uso adecuado de estos sistemas puede generar recomendaciones precisas en prácticamente cualquier ámbito. En nuestro caso, el sector de los videojuegos se beneficia enormemente, ya que existe una gran cantidad de información en la red, que es utilizada en el entrenamiento de muchos modelos. Existen miles de análisis de videojuegos que los clasifican según su género, plataforma, calidad y otros criterios. Además, hay información relevante sobre requisitos técnicos, títulos similares o la mejor plataforma para jugar cada juego. Buscar esta información de manera manual puede ser complicado, ya que no siempre conocemos las mejores fuentes de análisis o los idiomas en los que están disponibles. Sin embargo, los modelos generativos de lenguajes han sido entrenados con esta información y pueden proporcionarnos respuestas de manera sencilla, interactiva e incluso interpretando nuestras intenciones implícitas.

En resumen, los modelos generativos de lenguajes no solo están diseñados para facilitar la comunicación entre humanos y máquinas, sino que también

Capítulo 2: Contexto y estado del arte

están transformando la manera en que interactuamos con la tecnología. En este proyecto, desempeñarán un papel fundamental en la generación de recomendaciones personalizadas de videojuegos.

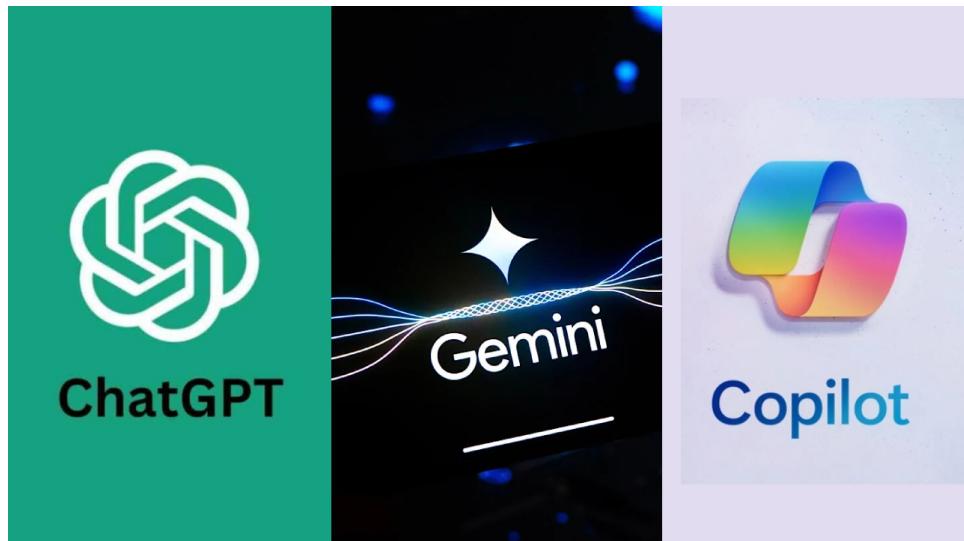


Figura 2.1: Ejemplos de modelos generativos de lenguajes. ChatGPT, Gemini y Copilot, tres de los LLMs más avanzados y reconocidos en la actualidad. <https://static1.srcdn.com/wordpress/wp-content/uploads/2024/01/tlou2r-joe-horse.JPG>

Capítulo 2: Contexto y estado del arte

2.1.2. LangChain

Para comprender mejor esta tecnología, obtendremos la información de su página web oficial.

LangChain es una tecnología que permite a las aplicaciones razonar y procesar información de manera más inteligente. Es utilizada por múltiples empresas de gran renombre, como Google y Rakuten TV. LangChain facilita la construcción de aplicaciones que integran LLMs, permitiendo que estas interactúen de forma dinámica y contextual con los usuarios.

Para cadenas y flujos de recuperación más sencillos, se recomienda usar LangChain junto con LangChain Expression Language para ensamblar componentes. Si estamos creando agentes o necesitamos una orquestación más compleja, utilizamos LangGraph.

Por otro lado, **LangGraph Platform** se encarga de la ejecución, permitiendo desplegar y ejecutar las aplicaciones de manera eficiente. Además, **LangSmith** ofrece herramientas para gestionar y probar las aplicaciones, asegurando su correcto funcionamiento y mejorando el rendimiento. [4] [5]

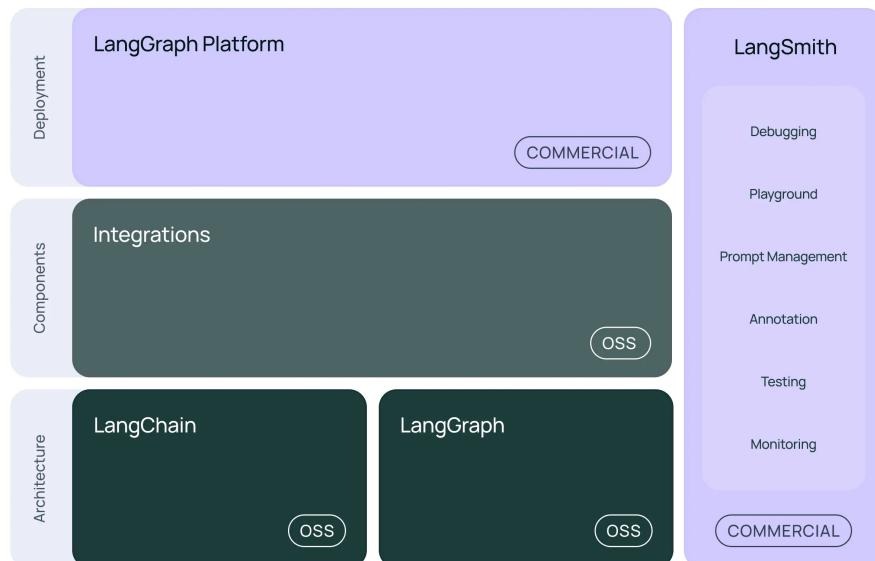


Figura 2.2: **Arquitectura productos LangChain.** La arquitectura de referencia que las empresas adoptan para el éxito. Los productos de LangChain pueden usarse de forma independiente o combinada para lograr un impacto multiplicador, guiando en la construcción, ejecución y gestión de aplicaciones LLM. [4]

Capítulo 2: Contexto y estado del arte

Procedamos a analizar cada producto por separado.

LangChain básico

Es la tecnología base de todos los productos. Es compatible con JavaScript y Python. Cuenta con una amplia biblioteca de componentes para usar de principio a fin en las aplicaciones. Conecta los modelos generativos de lenguajes con cualquier fuente de datos o conocimiento, y APIs para crear aplicaciones conscientes del contexto y con capacidad de razonamiento, utilizando métodos populares como RAG (Retrieval-Augmented Generation), que recupera la información del usuario de, por ejemplo, bases de datos antes de generar una respuesta, para mejorarla; y cadenas simples, en las que se solicita la pregunta al usuario, se pasa al LLM y se envía la respuesta al usuario. LangChain es fácil de comenzar a usar y ofrece elección, flexibilidad y potencia a medida que escalas. Cuenta con más de 3000 colaboradores, la comunidad más grande de cualquier framework centrado en LLM, más de 600 integraciones y es fácil de usar, pero robusto para producción. Además, es de código abierto. [6] [5]

Métodos

En LangChain, diversos métodos avanzados permiten optimizar la interacción con los modelos generativos de lenguajes, facilitando la integración de datos, la mejora en el procesamiento y la generación de respuestas. Estos métodos permiten adaptar el comportamiento del sistema a necesidades específicas, desde la recuperación de información relevante hasta la ejecución de tareas automatizadas mediante agentes. Vamos a explorar algunos de los métodos clave utilizados en LangChain, como la Generación Aumentada por Recuperación (RAG), los agentes inteligentes y las herramientas de evaluación de rendimiento, que son fundamentales para el desarrollo de aplicaciones inteligentes y contextualmente conscientes.

RAG

La **Generación Aumentada por Recuperación (RAG)** con LangChain y sus métodos integrados de gestión y recuperación conecta los datos con el poder de los LLMs. Dispone de herramientas de importación de documentos para cualquier tipo de datos.

Cuenta con potentes algoritmos de recuperación como:

- **Time-Weighted Vector Store:** combina la similitud semántica con un decaimiento temporal para tener en cuenta la actualidad en las recuperaciones.
- **Parent Document Retriever:** incorpora fragmentos pequeños, que son mejores para la búsqueda por similitud, pero recupera fragmentos más grandes que ayudan en la generación de contenido.

Capítulo 2: Contexto y estado del arte

- **Self Query Retriever:** analiza la consulta en lenguaje natural y escribe una consulta estructurada para ejecutarla en el VectorStore subyacente.
- **Contextual Compression:** comprime el documento recuperado utilizando el contexto de la consulta, de modo que solo se devuelve la información relevante de la fuente.
- **Multi Vector Retriever:** realiza consultas en múltiples vectores almacenados por documento, incluidos fragmentos más pequeños, resúmenes y preguntas hipotéticas.

La API de indexación de LangChain sincroniza los datos desde cualquier fuente hacia un almacén vectorial, ayudándote a ahorrar tiempo y recursos.

[7] [5]

Agentes

Los agentes pueden crear borradores iniciales para revisión a posteriori, actuar en tu nombre o esperar tu aprobación antes de la ejecución. LangGraph ofrece control para flujos de trabajo personalizados de agentes y multi-agentes, interacciones fluidas con intervención humana, y soporte nativo para transmisión de datos, lo que mejora la fiabilidad y ejecución de los agentes. Permite llamar de manera forzada a una herramienta, esperar la aprobación de intervención humana, la colaboración de múltiples agentes en un objetivo común y transmitir los pasos intermedios a medida que ocurren. Además, LangSmith ofrece la capacidad de explicar por qué los agentes se desvían y cómo hacer que vuelvan a funcionar correctamente. [8] [5]

Evaluación

LangSmith permite medir el rendimiento de la aplicación a lo largo de todo su ciclo de desarrollo con multitud de métricas. Se puede usar evaluación offline, con integración continua y online.

Un marco de pruebas sólido comienza con la creación de un conjunto de datos de referencia, una tarea que a menudo resulta tediosa. LangSmith simplifica esto permitiéndote guardar las trazas de depuración y producción en conjuntos de datos. Los conjuntos de datos son colecciones de entradas y salidas ejemplares o problemáticas que deben ser replicadas o corregidas, respectivamente. También te permite hacer un seguimiento de cómo se comparan las diferentes versiones de tu aplicación según los criterios de evaluación que hayas definido. Además, facilita la evaluación humana y en línea, no solo latencia, errores y costos, sino también medidas cualitativas para asegurar que tu aplicación responda de manera efectiva y cumpla con las expectativas del usuario. [9] [5]

Capítulo 2: Contexto y estado del arte

LangGraph

LangGraph controla, modera y guía las acciones de los agentes, evitando que se desvíen del curso previsto. Permite diseñar flujos de control diversos, tales como simples, multi-agentes, jerárquicos o secuenciales, todo dentro de un único marco de trabajo. Además, tiene la capacidad de guardar los historiales de conversación y los datos de la sesión, lo que permite mantener el contexto a lo largo del tiempo y asegurar transferencias suaves de información, incluso hacia atrás.

LangGraph ofrece múltiples opciones de despliegue, incluye un estudio visual y es de código abierto, lo que facilita su integración y personalización.

Asimismo, cuenta con una plataforma diseñada para desplegar y escalar aplicaciones LangGraph, con una API orientada a la creación de interfaces de usuario (UX) para agentes, además de un entorno de desarrollo integrado. Sin embargo, a diferencia de su núcleo, esta plataforma no es de código abierto.[\[10\]](#) [\[5\]](#)

LangSmith

LangSmith es una plataforma todo en uno diseñada para desarrolladores, que abarca cada etapa del ciclo de vida de las aplicaciones impulsadas por modelos generativos de lenguajes, tanto si utilizan LangChain como si no.

Esta herramienta permite depurar, colaborar, realizar pruebas y monitorear, ofreciendo una visibilidad completa de toda la secuencia de llamadas. Esto facilita la identificación precisa de errores y cuellos de botella en el rendimiento en tiempo real. Además, permite revisar, etiquetar y evaluar el comportamiento de los agentes de forma sencilla, mejorando así la eficiencia y la fiabilidad del sistema.

LangSmith también puede ejecutarse de forma local, proporcionando flexibilidad y control total sobre el entorno de desarrollo. [\[11\]](#) [\[5\]](#)

Capítulo 2: Contexto y estado del arte

2.2. Estado del arte

La revisión de la literatura existente nos permite comprobar si ya se ha desarrollado un proyecto similar al nuestro, lo que puede servir como fuente de inspiración, guía o referencia para evaluar la novedad de nuestro trabajo.

Tras una exhaustiva búsqueda, no se ha encontrado ningún trabajo o artículo que aborde específicamente el uso de LangChain para la recomendación de videojuegos. Esto sugiere que nuestro proyecto podría ser innovador en este ámbito.

Sin embargo, en una búsqueda más general, se han identificado artículos relacionados que merecen ser destacados:

Un caso relevante es el de las **recomendaciones** en otros ámbitos **utilizando LangChain**. En un trabajo publicado por el Journal of Industrial Engineering and Applied Science, se observa cómo un grupo de estudiantes diseñó un sistema para recomendar anime de manera personalizada, utilizando LangChain. Los resultados obtenidos sometidos a numerosas métricas fueron muy satisfactorios gracias a las capacidades de esta tecnología. [12]

Otro trabajo notable es el de Johans Quintero, quien desarrolló un sistema que recomienda recursos mediante mecanismos de Procesamiento de Lenguaje Natural, utilizando agentes de LangChain. Este proyecto se centra principalmente en la recomendación de libros, pero su enfoque podría extrapolarse a otros ámbitos. [13]

En cuanto a la **recomendación de videojuegos**, se han encontrado trabajos relevantes que utilizan otras tecnologías. Por ejemplo, Daniel Yéalamos Pérez creó un sistema de recomendación de videojuegos basado en las opiniones de otros usuarios, empleando técnicas de extracción e interpretación semántica de datos en un entorno informal. [14]

Por último, cabe mencionar el trabajo de Alejandro Roldán Soblechero, quien desarrolló una aplicación móvil que permite a los usuarios opinar sobre videojuegos. Esta aplicación integra datos de IGDB (Internet Game Database) y otras bases de conocimiento, garantizando la solidez y actualidad de las sugerencias generadas por el sistema de recomendación. [15]

La siguiente tabla resume las características principales de cada trabajo:

Capítulo 2: Contexto y estado del arte

Cuadro 2.1: Comparativa entre trabajos analizados y el presente proyecto

Trabajo	C1	C2	C3	C4	C5
<i>Unlocking Personalized Anime Recommendations: Langchain and LLM at the Forefront</i>	✓	✗	✓	✗	✓
<i>Sistema de recomendación con agente LangChain</i>	✓	✗	✓	✗	✓
<i>Recomendación de videojuegos basado en análisis semántico y minería de opinión</i>	✗	✓	✗	✗	✗
<i>Aplicación de recomendación de videojuegos (IGDB)</i>	✗	✓	✓	✗	✗
Este trabajo	✓	✓	✓	✓	✓

Leyenda de columnas:

C1 Uso de LangChain

C2 Dominio: Recomendación de videojuegos

C3 Uso de múltiples fuentes de datos

C4 Personalización contextual (recursos, accesibilidad, etc.)

C5 Implementación práctica con tecnologías actuales

Estos estudios evidencian que tanto el ámbito de las recomendaciones de videojuegos como el uso de LangChain en sistemas de recomendación son temas explorados en la literatura, ofreciendo un valioso punto de referencia y posibles fuentes de inspiración. Sin embargo, la ausencia de trabajos que combinen específicamente LangChain con la recomendación de videojuegos destaca la originalidad y el carácter innovador de nuestro proyecto.

Capítulo 3

Objetivos y metodología de desarrollo

En este capítulo se define el objetivo principal del proyecto, junto con los objetivos específicos que permitirán alcanzarlo de manera efectiva.

Asimismo, se detallará la metodología de trabajo utilizada en el desarrollo de la plataforma, describiendo los enfoques y herramientas empleados en el desarrollo para este trabajo.

Capítulo 3: Objetivos y metodología de desarrollo

3.1. Objetivos

Para alcanzar una meta, es fundamental definir con precisión qué se desea lograr y establecer los pasos y condiciones necesarios para lograrlo de manera efectiva.

El **objetivo principal** de este trabajo es **desarrollar un sistema de recomendación de videojuegos** que ofrezca sugerencias personalizadas y alineadas con las preferencias y necesidades del usuario.

Para alcanzar este objetivo de manera óptima, es esencial cumplir con los siguientes objetivos secundarios:

- **Diseñar una interfaz clara, intuitiva y accesible** para garantizar una experiencia de usuario fluida y sin barreras.
- **Mejorar la personalización del sistema** mediante el aprendizaje y almacenamiento de las interacciones del usuario, así como la integración de datos provenientes de otras plataformas.
- **Implementar y coordinar múltiples modelos de lenguaje** para optimizar la precisión y diversidad de las recomendaciones.
- **Incorporar factores adicionales en las recomendaciones**, como disponibilidad del juego, requisitos técnicos, accesibilidad y presupuesto del usuario.
- **Facilitar la actualización del sistema** con nuevas tendencias y lanzamientos en el sector de los videojuegos, considerando las limitaciones en el entrenamiento de los modelos de lenguaje.
- **Optimizar el rendimiento del sistema** para ofrecer respuestas rápidas, precisas y justificadas, mejorando la experiencia de usuario.
- **Garantizar la privacidad y seguridad de los datos** mediante buenas prácticas de almacenamiento y procesamiento de información.

El cumplimiento de estos objetivos asegurará el desarrollo de un sistema de recomendación eficaz y adaptado a las necesidades del usuario.

Capítulo 3: Objetivos y metodología de desarrollo

3.2. Metodología de desarrollo

Para el **desarrollo** de la plataforma, se ha adoptado un enfoque basado en **metodologías ágiles**, priorizando la **flexibilidad** y la **adaptación** continua a los desafíos y necesidades emergentes a lo largo del proceso. Las metodologías ágiles se centran en la entrega **incremental** de valor, permitiendo **ajustes rápidos y frecuentes** en función de los comentarios, necesidades e inconvenientes surgidos a lo largo del desarrollo. Este enfoque se caracteriza por ciclos de trabajo cortos, denominados "**sprints**", en los cuales se planifica, ejecuta, prueba y evalúa el progreso de manera continua, asegurando la flexibilidad y la rápida implementación de cambios.

La **planificación** inicial fue un proceso caótico. Aunque comencé con una idea general del proyecto, a medida que avanzaba, fui desarrollando cada sección de la memoria de forma paralela, sin perder de vista cómo se conectaban entre sí los capítulos y cómo se iban integrando los distintos elementos del trabajo. La planificación fue en constante evolución, adaptándose a los tiempos limitados y a los factores externos imprevistos, lo que permitió cumplir con los objetivos, aunque de manera más flexible y con ajustes frecuentes en el camino.

A medida que el desarrollo fue avanzando, la planificación se fue optimizando mediante el establecimiento de objetivos más concretos y plazos más definidos para cada sprint. De esta manera, se lograron entregas parciales de la plataforma, que permitieron validar la funcionalidad y realizar mejoras progresivas en cada ciclo. Además, la iteración continua y la retroalimentación temprana de las soluciones implementadas ayudaron a reducir riesgos y garantizar la calidad del producto final.

La planificación se podría resumir en este diagrama de Gantt:



Figura 3.1: **Planificación en diagrama de Gantt**. Apreciamos como el proyecto se ha realizado a través de los meses. La memoria y la investigación son apartados que evolucionan continuamente. Los distintos elementos de la aplicación se desarrollan por partes, pero en ciertos puntos se integran de forma paralela. Las prueba y presentación se realiza al final, pero durante todo el proceso se tiene en mente.

Capítulo 3: Objetivos y metodología de desarrollo

Sin embargo, es importante volver a señalar que, debido a mis circunstancias personales, no pude dedicarle mucho tiempo a la organización temporal del proyecto. En mi caso, la gestión del tiempo no ha sido un aspecto en el que pudiera destacar, ni en este trabajo ni en otros aspectos de toda mi vida. Por lo tanto, preferí concentrarme en lo verdaderamente importante: la realización efectiva del trabajo, sin gastar tiempo innecesario en crear planificaciones detalladas solo para cumplir con una formalidad. La prioridad fue avanzar en el desarrollo y en los resultados concretos, adaptándome a lo que surgiera en el camino. *Carpe Diem*.

La **investigación** ha sido fundamental en la metodología de trabajo, recurriendo a documentación oficial, artículos técnicos y foros especializados para resolver dudas y optimizar la implementación. Además, se adoptó un enfoque práctico basado en la experimentación, probando diversas soluciones y herramientas para garantizar el correcto funcionamiento de la plataforma y su integración.

El uso de herramientas como *ChatGPT* facilitó la corrección ortográfica, la mejora de la redacción y la traducción de textos en otros idiomas, acelerando así el proceso de adquisición de conocimiento y optimizando el tiempo dedicado a la investigación.

3.2.1. Presupuesto

Dado que el proyecto se ha desarrollado de manera autónoma y sin recursos financieros dedicados, los **costes** asociados se limitan al tiempo invertido, el uso de mis recursos propios, como ordenador y wifi, y al uso de herramientas gratuitas. A continuación, se detalla el presupuesto estimado basado en los costes de un desarrollador informático y los recursos utilizados que podría adecuarse a una empresa:

- **Coste del desarrollador informático:** Según los estándares actuales, el coste medio por hora de un informático en España para un trabajo de este tipo es de aproximadamente 15,27€ por hora[16]. Dado que un TFG dura 300 horas, el cálculo sería el siguiente:

$$300 \text{ horas} \times 15,27 \text{ €} = 4.581 \text{ €}$$

Este coste es aproximado y puede variar dependiendo de la experiencia y la localización del profesional.

- **Equipo informático:** El coste estimado de un ordenador adecuado para el desarrollo de software es de aproximadamente 1.420€, más un ratón de unos 26 €, considerando un equipo de gama media-alta, como es mi caso. Aunque como este desarrollo no requiere de una GPU de alta prestaciones, ya que no se genera casi gráficos ni se hacen muchos cálculos, es perfectamente viable usar un equipo más barato que no

Capítulo 3: Objetivos y metodología de desarrollo

tenga una GPU dedicada, o que esta sea una básica destinada a la ofimática.

- **Consumo de electricidad:** El consumo de un ordenador durante un período de 300 horas puede suponer un coste de aproximadamente 42€ (suponiendo un consumo medio de 0.2 kWh(portátil enchufado más una lámpara) por hora[17] y un coste de unos 0.14€ por kWh). El coste de la electricidad es un factor muy importante a tener en cuenta. Aunque se ha estimado un gasto general, es necesario señalar que este costo puede ser en realidad mayor debido a las tarifas diferenciadas por franjas horarias, las cuales, lamentablemente, están bastante elevadas en las franjas activas. Esta situación resulta aún más agravante al considerar el aumento constante de los precios de la energía, lo que incrementa considerablemente los costes operativos de los proyectos que requieren un uso constante de equipos eléctricos. Es incluso más vergonzoso ver cómo la precariedad de nuestros dirigentes —más interesados en montar un circo entre ellos que en hacer las cosas bien y con responsabilidad— nos deja sin luz casi un día entero, costándonos a muchos una valiosa jornada de trabajo, e incluso, en algunos casos, algo aún peor: la vida. Es lamentable comprobar cómo no hay consecuencias para los culpables, sino aplausos de sus fieles, sumisos como ganado.
- **Conexión a internet:** El coste de un servicio básico de internet(sin contar las líneas móviles) es de alrededor de 30€ con MasMovil. Suponiendo que el proyecto dure 6 meses, el coste estimado de la conexión a internet sería:
$$30 \text{ €} \times 6 \text{ meses} = 180 \text{ €}$$
- **Herramientas y servicios en línea:** Las herramientas utilizadas, como *GitHub*, *LangChain*, *LangGraph* y *ChatGPT*, fueron utilizadas en sus versiones gratuitas, lo que ha permitido minimizar los costes. No obstante, en un entorno profesional, estos servicios pueden tener un coste mensual que suele variar entre 10€ y 100€ dependiendo de las funcionalidades requeridas.
- **Juego en Steam:** Steam requiere un gasto directo de al menos 5 dólares para habilitar ciertas funciones, como la lista de amigos, actividades o el acceso a sus APIs. En mi caso, no había alcanzado ese umbral, ya que suelo adquirir juegos mediante plataformas de keys, que generalmente ofrecen mejores precios y no dependen de los períodos de rebajas oficiales de Steam. Compré Cult of the Lamb por 11,49€, lo que activó esas funcionalidades.

Resumen del presupuesto estimado:

Capítulo 3: Objetivos y metodología de desarrollo

- **Coste del desarrollador informático:** 4.581€
- **Equipo informático:** 1.446€ (incluyendo ratón)
- **Consumo de electricidad:** 42€
- **Conexión a internet:** 180€
- **Herramientas en línea (versión gratuita):** 0€
- **Juego de Steam:** 11.49€

Total estimado de costes (sin incluir costes de servicios pagos):
6.260.49€

Este presupuesto refleja el coste real de desarrollar el proyecto de manera autónoma, sin considerar gastos adicionales relacionados con licencias de software, servicios profesionales o materiales externos.

Conclusión

La metodología ágil permitió una planificación flexible y adaptable a los retos del proyecto, lo que garantizó el éxito del desarrollo de la plataforma. Además, el uso de herramientas gratuitas y la optimización de recursos han permitido mantener los costes del proyecto en niveles razonables, asegurando que el objetivo final se alcanzara dentro de los plazos establecidos, sin comprometer la calidad del trabajo realizado.

Capítulo 4

Análisis

En este capítulo se analizarán en detalle los requisitos de la plataforma, abarcando tanto los requisitos funcionales como los no funcionales y los de información. También se describirán los casos de uso. Se examinarán las necesidades del usuario y las características esenciales del sistema.

Además, se abordará el proceso de obtención de requisitos.

Este análisis servirá como base para la fase de diseño y desarrollo, garantizando que la plataforma cumpla con los objetivos definidos y proporcione una experiencia óptima para los usuarios.

Capítulo 4: Análisis

4.1. Obtención de Requisitos

La definición de los requisitos del sistema se ha basado en mi extensa experiencia en el sector de los videojuegos, combinada con la retroalimentación de amigos y jugadores con los que he interactuado a lo largo de los años. Además, he tenido la oportunidad de observar y analizar constantemente el panorama del mundo de los videojuegos a través de internet, lo que ha proporcionado una visión amplia sobre las necesidades y preferencias de los jugadores.

Para la recopilación de los requisitos, se utilizaron varias fuentes y enfoques, los cuales incluyeron:

- **Experiencia personal y observación directa:** Como jugador habitual y entusiasta del mundo de los videojuegos, he tenido acceso a una gran variedad de plataformas, consolas y configuraciones de PC, lo que me ha permitido comprender las distintas necesidades de los usuarios según el hardware disponible y los gustos personales de cada uno. Esta experiencia directa ha sido fundamental para definir las expectativas y características que debe tener la plataforma de recomendación de videojuegos.
- **Conversaciones con jugadores:** He mantenido numerosas conversaciones con amigos y conocidos, tanto jugadores casuales como experimentados, sobre sus preferencias, frustraciones con las plataformas actuales... Esto me ha permitido obtener información valiosa sobre lo que los jugadores realmente buscan en una herramienta de recomendación, así como las funcionalidades que consideran imprescindibles.
- **Análisis de tendencias en línea:** A través de la observación de foros, redes sociales y sitios web especializados en videojuegos, he podido identificar tendencias en cuanto a lo que los jugadores desean y lo que consideran que falta en las plataformas actuales. Este análisis ha sido crucial para determinar qué funcionalidades deberían ser priorizadas y cómo diferenciarse de la competencia.
- **Revisión de sistemas existentes:** He estudiado y analizado plataformas similares, como Steam, Epic Games, Xbox entre otras, para comprender qué aspectos funcionan bien y cuáles podrían mejorarse. Este análisis comparativo me ha permitido identificar buenas prácticas y detectar carencias en la experiencia del usuario que podrían ser abordadas en la nueva plataforma.

A partir de toda esta información, he podido elaborar un conjunto de requisitos iniciales que reflejan las necesidades reales de los usuarios, lo que me ha permitido validar las ideas y ajustar las funcionalidades mediante

Capítulo 4: Análisis

la retroalimentación constante de los jugadores. De este modo, el sistema propuesto no solo responde a mis propios conocimientos, sino también a las expectativas y deseos de los usuarios del sector.

Capítulo 4: Análisis

4.2. Requisitos funcionales

En esta sección se describen las funciones y características directamente relacionadas con las acciones que pueden realizar los usuarios dentro de la aplicación.

RF-1: Gestión de usuarios

La aplicación debe permitir al usuario realizar las tareas básicas relacionadas con la gestión de su cuenta personal.

- **RF-1.1:** El usuario debe poder registrarse o iniciar sesión en la aplicación utilizando un nombre de usuario, una contraseña y otra información relevante.
- **RF-1.2:** El usuario puede modificar su nombre de usuario, contraseña e información personal asociada a su cuenta.
- **RF-1.3:** El usuario puede eliminar su cuenta si así lo desea.

RF-2: Gestión de información sobre videojuegos

La aplicación debe permitir al usuario gestionar toda la información relacionada con videojuegos.

- **RF-2.1:** El usuario debe poder comunicarse con los agentes para introducir datos sobre su hardware, los videojuegos disponibles, los últimos juegos jugados o adquiridos, y cualquier otro aspecto relevante.
- **RF-2.2:** El usuario puede eliminar completamente la información relacionada con videojuegos si así lo desea.
- **RF-2.3:** El usuario puede vincular y desvincular su cuenta con plataformas externas de videojuegos, permitiendo la importación rápida y actualizada de su información personal sobre videojuegos.

RF-3: Recomendaciones

La aplicación debe ofrecer recomendaciones de videojuegos tanto a usuarios logueados como no logueados.

- **RF-3.1:** Un usuario no logueado puede acceder a recomendaciones básicas, limitadas por la ausencia de historial e información personalizada, pero beneficiándose del uso de modelos generativos de lenguajes.
- **RF-3.2:** Un usuario logueado puede acceder a recomendaciones personalizadas, generadas a partir de sus conversaciones previas, la información introducida manualmente y los datos recuperados desde otras plataformas, con el objetivo de ofrecer sugerencias de alta calidad.

Capítulo 4: Análisis

4.3. Requisitos no funcionales

A continuación, se detallan las necesidades que no están directamente relacionadas con la funcionalidad del sistema, pero que son clave para su buen rendimiento y la satisfacción del usuario.

RNF-1: Rendimiento

El sistema debe responder con la mayor brevedad posible.

RNF-2: Seguridad y privacidad

- **RNF-2.1:** No se almacenará información privada del usuario que no esté relacionada con videojuegos.
- **RNF-2.2:** La conexión con plataformas externas se realizará de forma segura y privada.

RNF-3: Disponibilidad

Si un LLM falla o tarda demasiado en responder, el sistema debe ser capaz de continuar funcionando sin depender de él.

RNF-4: Accesibilidad

El sistema tendrá un diseño y funcionalidades sencillas e intuitivas, de modo que cualquier persona pueda utilizarlo de forma efectiva, independientemente de sus capacidades.

RNF-5: Legales

El sistema hará un uso legal del material no propio, basado en el principio de *fair use* [18].

Capítulo 4: Análisis

4.4. Requisitos de información

Veamos qué datos deberá gestionar nuestro sistema.

RI-1: Datos del usuario

Nombre de usuario y contraseña.

RI-2: Datos sobre videojuegos del usuario

Plataformas disponibles, componentes del ordenador, videojuegos en posesión o accesibles mediante suscripciones, así como juegos marcados como gustados o no.

RI-3: Consultas previas del usuario

Los agentes tendrán la capacidad de recordar conversaciones anteriores con el usuario.

Capítulo 4: Análisis

4.5. Actores

Actor	ACT-1 (Usuario no logueado)
Descripción	Usuario que no tiene cuenta o no ha iniciado sesión.
Características	Solo puede crear una cuenta, iniciar sesión con una existente o acceder a recomendaciones básicas.
Relaciones	RF-1.1, RF-2.1, RF-3.1, RNF-1, RNF-3, RNF-4, RNF-5

Cuadro 4.1: Detalles del Actor: Usuario no logueado

Actor	ACT-2 (Usuario logueado)
Descripción	Usuario que ha iniciado sesión en la aplicación.
Características	Puede gestionar su información, importar datos de otras plataformas, modificar o eliminar su cuenta y recibir recomendaciones personalizadas.
Relaciones	RF-1, RF-2, RF-3.2, RNF-1, RNF-2, RNF-3, RNF-4, RNF-5

Cuadro 4.2: Detalles del Actor: Usuario logueado

Actor	ACT-3 (Administrador)
Descripción	Administrador del sistema, con permisos extendidos respecto al usuario logueado (ACT-2).
Características	Tiene acceso avanzado para depurar y administrar el sistema.
Relaciones	RNF-1, RNF-3, RNF-4

Cuadro 4.3: Detalles del Actor: Administrador

Capítulo 4: Análisis

4.6. Casos de usos

4.6.1. Casos de uso del usuario no logueado

Caso de uso	CU-01: Obtener recomendación de videojuegos sin loguearse
Actores	ACT-1 (Usuario no logueado)
Tipo	Primario
Precondición	-
Poscondición	Se muestra la recomendación básica.
Propósito	Generar una recomendación en función de lo escrito por el usuario.

Cuadro 4.4: Caso de uso de obtención de recomendación básica

Paso	Actor	Sistema
1	El usuario ingresa su búsqueda.	
2		El sistema procesa la información.
3		Se genera una recomendación personalizada.
4		El sistema muestra la recomendación al usuario.

Cuadro 4.5: Flujo de eventos del caso de uso CU-01

Curso alterno de eventos	Descripción
1a. El usuario no ingresa suficientes datos o son ambiguos.	El sistema muestra un mensaje de error indicando que se requieren más datos o más precisos.

Cuadro 4.6: Curso alternativo del caso de uso CU-01

Capítulo 4: Análisis

Caso de uso	CU-02: Crear cuenta
Actores	ACT-1 (Usuario no logueado)
Tipo	Primario
Precondición	El nombre de usuario no debe corresponder con el de otro usuario.
Poscondición	La cuenta es creada.
Propósito	Crear una cuenta en la plataforma para el usuario.

Cuadro 4.7: Caso de uso de crear cuenta

Paso	Actor	Sistema
1	El usuario solicita crear una cuenta.	
2		El sistema solicita al usuario el nombre de usuario y la contraseña.
3	El usuario introduce el nombre y la contraseña.	
4		Se genera la nueva cuenta.
5		El sistema muestra que la cuenta ha sido creada correctamente y el usuario pasa a estar logueado.

Cuadro 4.8: Flujo de eventos del caso de uso CU-02

Curso alterno de eventos	Descripción
4a. El nombre de usuario ya está en uso.	El sistema muestra un mensaje de error indicando que el nombre ya está en uso y solicita uno nuevo.

Cuadro 4.9: Curso alternativo del caso de uso CU-02

Caso de uso	CU-03: Iniciar sesión
Actores	ACT-1 (Usuario no logueado)
Tipo	Primario
Precondición	El usuario debe tener una cuenta creada.
Poscondición	El usuario queda logueado.
Propósito	Permitir al usuario acceder a su cuenta previamente creada.

Cuadro 4.10: Caso de uso de iniciar sesión

Capítulo 4: Análisis

Paso	Actor	Sistema
1	El usuario solicita acceder a su cuenta.	
2		El sistema solicita el nombre de usuario y la contraseña.
3	El usuario introduce el nombre y la contraseña.	
4		El sistema comprueba los datos.
5		El sistema confirma el acceso y el usuario pasa a estar logueado.

Cuadro 4.11: Flujo de eventos del caso de uso CU-03

Curso alterno de eventos	Descripción
4a. El nombre de usuario o la contraseña son incorrectos.	El sistema muestra un mensaje de error indicando que el nombre de usuario o la contraseña son incorrectos y solicita que se introduzcan de nuevo.

Cuadro 4.12: Curso alternativo del caso de uso CU-03

Capítulo 4: Análisis

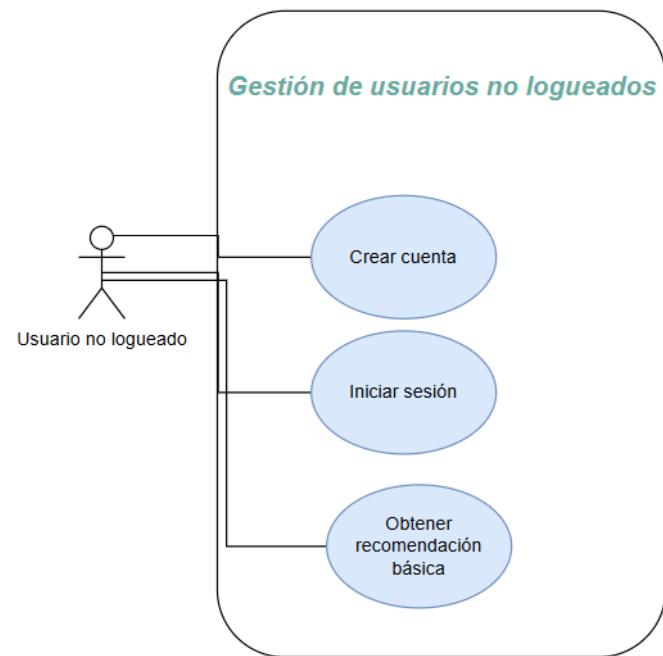


Figura 4.1: **Diagrama de casos de uso de usuario no logueado**

Capítulo 4: Análisis

4.6.2. Casos de uso del usuario logueado

Caso de uso	CU-04: Modificar información sobre videojuegos.
Actores	ACT-2 (Usuario logueado)
Tipo	Primario
Precondición	-
Poscondición	La información sobre el usuario queda actualizada.
Propósito	Permitir que el usuario modifique su información sobre videojuegos.

Cuadro 4.13: Caso de uso de modificar información sobre videojuegos

Paso	Actor	Sistema
1	El usuario escribe su modificación en texto y lo envía a los agentes.	
2		El sistema recibe la petición y la envía a los agentes, quienes la procesan y toman las medidas oportunas sobre la información de videojuegos del usuario.
3		El sistema muestra un mensaje indicando que los cambios se han realizado correctamente.

Cuadro 4.14: Flujo de eventos del caso de uso CU-04

Curso alterno de eventos	Descripción
2a. El texto del usuario no es correcto o está vacío.	El sistema muestra un mensaje de error indicando que el mensaje no es válido.

Cuadro 4.15: Curso alternativo del caso de uso CU-04

Capítulo 4: Análisis

Caso de uso	CU-05: Modificar información sobre la cuenta.
Actores	ACT-2 (Usuario logueado)
Tipo	Secundario
Precondición	-
Poscondición	La información sobre la cuenta del usuario queda actualizada.
Propósito	Permitir al usuario modificar su contraseña o nombre de usuario.

Cuadro 4.16: Caso de uso de modificar información sobre la cuenta del usuario

Paso	Actor	Sistema
1		El sistema pide al usuario la contraseña actual.
2	El usuario escribe la contraseña actual.	
3	El usuario escribe la nueva contraseña y/o el nuevo nombre de usuario.	
4		El sistema recibe los datos y efectúa el cambio.

Cuadro 4.17: Flujo de eventos del caso de uso CU-05

Curso alterno de eventos	Descripción
2a. La contraseña no es correcta.	El sistema muestra un mensaje de error indicando que la contraseña es incorrecta y pide que la introduzca nuevamente.
4a. La nueva contraseña no cumple con los requisitos y/o el nuevo nombre de usuario ya está en uso.	El sistema muestra un mensaje de error indicando que la contraseña no cumple con los requisitos y/o el nuevo nombre de usuario ya está en uso, y pide que se modifiquen los datos.

Cuadro 4.18: Curso alternativo del caso de uso CU-05

Capítulo 4: Análisis

Caso de uso	CU-06: Eliminar cuenta.
Actores	ACT-2 (Usuario logueado)
Tipo	Primario
Precondición	-
Poscondición	La cuenta y los datos del usuario han sido eliminados.
Propósito	Eliminar la cuenta del usuario.

Cuadro 4.19: Caso de uso de eliminar la cuenta del usuario

Paso	Actor	Sistema
1		El sistema pide al usuario la contraseña actual.
2	El usuario escribe la contraseña actual.	
3		El sistema pregunta si el usuario está seguro de eliminar su cuenta.
4	El usuario confirma.	
5		El sistema cierra sesión al usuario y borra su cuenta y su información.

Cuadro 4.20: Flujo de eventos del caso de uso CU-06

Curso alterno de eventos	Descripción
2a. La contraseña no es correcta.	El sistema muestra un mensaje de error indicando que la contraseña es incorrecta y la pide de nuevo al usuario.
3a. El usuario no está seguro de eliminar la cuenta.	El sistema regresa al usuario a la sección anterior.

Cuadro 4.21: Curso alternativo del caso de uso CU-06

Capítulo 4: Análisis

Caso de uso	CU-07: Obtener datos con API externa.
Actores	ACT-2 (Usuario logueado)
Tipo	Primario
Precondición	Tener una cuenta en la plataforma externa.
Poscondición	La plataforma obtiene y guarda los datos de videojuegos de la plataforma externa.
Propósito	Obtener información de plataformas externas.

Cuadro 4.22: Caso de uso de obtener datos con API externa

Paso	Actor	Sistema
1	El usuario pide conectarse a la API.	
2		El sistema llama a la API.
3		#Include api.
4		El sistema recibe el resultado de la API y almacena la información obtenida.

Cuadro 4.23: Flujo de eventos del caso de uso CU-07

Caso de uso	CU-08: Obtener recomendaciones personalizadas.
Actores	ACT-2 (Usuario logueado)
Tipo	Primario
Precondición	-
Poscondición	El usuario obtiene recomendaciones según su información y lo preguntado.
Propósito	Obtener recomendaciones personalizadas.

Cuadro 4.24: Caso de uso de obtener recomendaciones personalizadas

Paso	Actor	Sistema
1	El usuario escribe su petición.	
2		El sistema pasa la petición a los LLMs junto a la información del usuario, y estos la procesan, almacenan y crean una respuesta.
3		El sistema devuelve la respuesta al usuario.

Cuadro 4.25: Flujo de eventos del caso de uso CU-08

Capítulo 4: Análisis

Curso alterno de eventos	Descripción
2a.	La petición es invalida. El sistema pide al usuario que la introduzca de nuevo.

Cuadro 4.26: Curso alternativo del caso de uso CU-08

Capítulo 4: Análisis

4.6.3. Casos de usos de administrador

Similar a usuario logueado, pero con acceso a información de debugueo, opciones especiales...

Caso de uso	CU-09: Opciones avanzadas y obtener métricas.
Actores	ACT-3 (Administrador)
Tipo	Primario
Precondición	-
Poscondición	El administrador obtiene métricas.
Propósito	Obtener métricas y ejecutar opciones avanzadas.
Extiende	CU-04 → CU-09.

Cuadro 4.27: Caso de uso de Opciones avanzadas y obtener métricas

Paso	Actor	Sistema
1	El administrador pide ejecutar un método, con posibles opciones avanzadas.	
2		El sistema ejecuta ese método con las opciones avanzadas especificadas, además de ofrecer métricas.

Cuadro 4.28: Flujo de eventos del caso de uso CU-09

Capítulo 4: Análisis

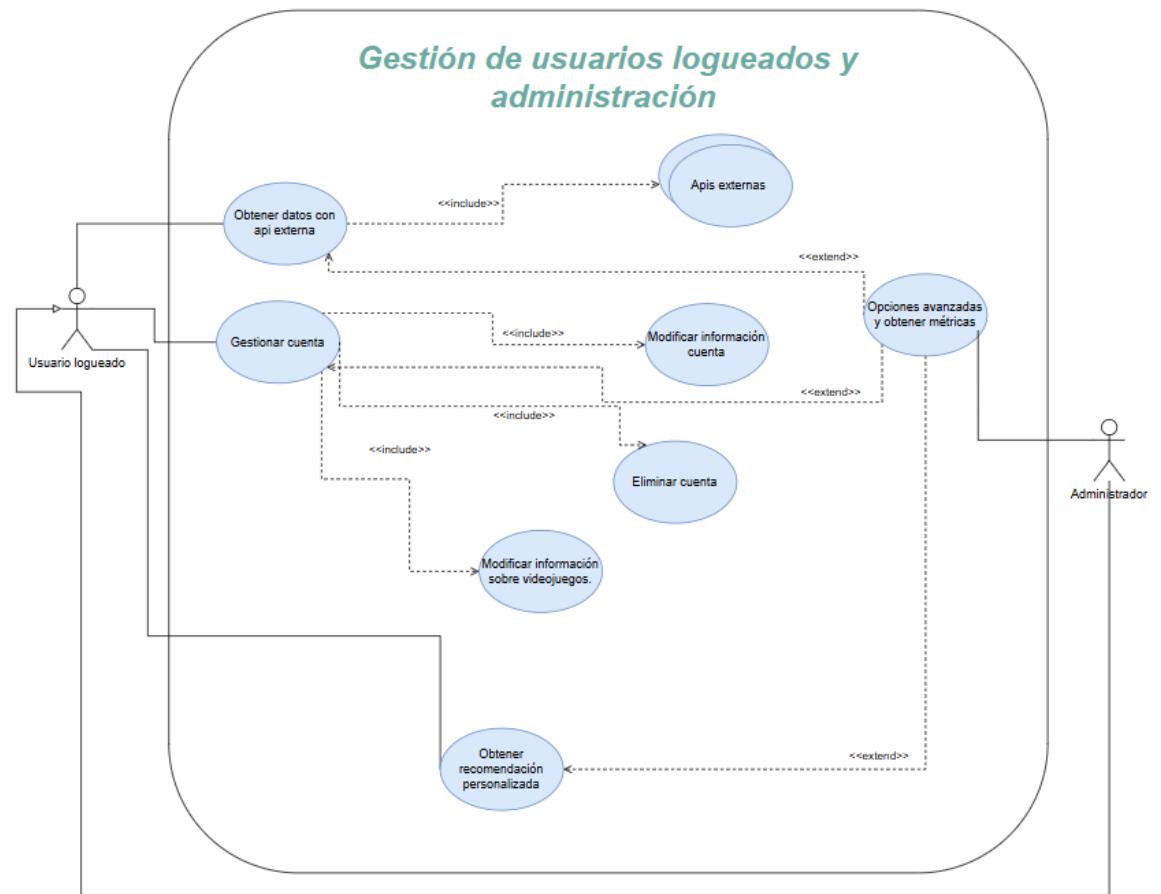


Figura 4.2: Diagrama de casos de uso de usuario logueado y administrador

Capítulo 5

Diseño

En este capítulo, nos enfocaremos en el diseño de nuestra plataforma, especificando sus elementos clave, cómo se comunican entre sí y cómo optimizar su funcionamiento para sacar el máximo provecho de sus capacidades.

También abordaremos la arquitectura general del sistema, el flujo de datos entre los distintos módulos y las decisiones de diseño tomadas para garantizar la escalabilidad, la flexibilidad y la facilidad de uso.

En adición se considerarán aspectos relacionados con la experiencia del usuario para asegurar que la plataforma sea lo más intuitiva y eficiente.

Capítulo 5: Diseño

5.1. Diseño de la arquitectura

A continuación, se define la arquitectura general de la plataforma, describiendo sus componentes principales y cómo se relacionan entre sí.

La arquitectura se sustenta sobre tres pilares fundamentales:

1. **Base de datos:** Es la encargada de almacenar la información del usuario, así como otros datos esenciales para el funcionamiento de la aplicación. Se comunica directamente con el backend, enviando y recibiendo información según sea necesario.

2. **Backend:** Responsable de la lógica y el procesamiento de los datos de la plataforma. En esta capa se encuentra integrado LangChain, el eje principal de este proyecto. El backend se comunica con la base de datos, enviando peticiones y controlando el flujo de datos. En la capa superior, llamada controlador, se gestiona la comunicación con el frontend, recibiendo solicitudes y transmitiendo los datos necesarios.

3. **Frontend:** Se encarga de la presentación visual de la aplicación, siendo el medio con el que el usuario interactúa y visualiza la información. Se comunica con el backend para mostrar los datos al usuario y enviar nuevas solicitudes.

Capítulo 5: Diseño

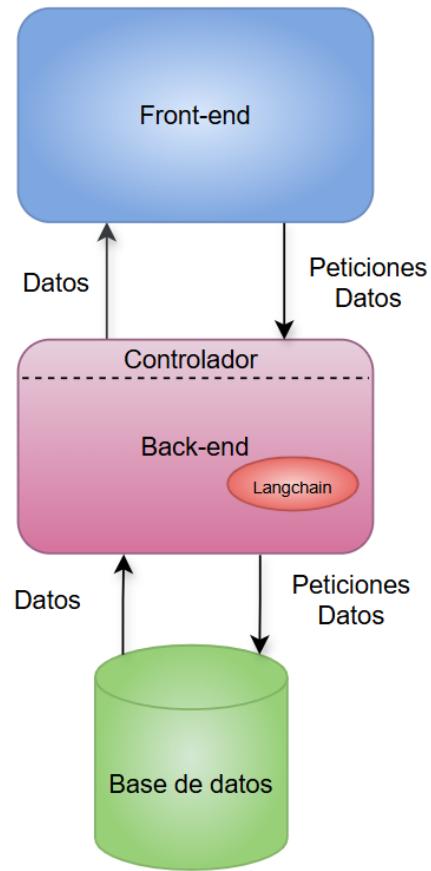


Figura 5.1: **Representación visual de la arquitectura.** En esta imagen se ilustran los elementos clave de la arquitectura, sus interacciones y cómo cada uno de los componentes descritos previamente se conecta y depende entre sí.

Capítulo 5: Diseño

5.2. Elementos de la arquitectura

A continuación, se describen detalladamente los distintos elementos que componen la arquitectura de la plataforma.

5.2.1. Base de datos

La base de datos tiene como objetivo almacenar toda la información relevante de los usuarios. Esto incluye tanto datos personales, como nombre de usuario, contraseña o foto de perfil, como información relacionada con su actividad en videojuegos: consolas que posee, componentes de su PC, juegos jugados, títulos que le han gustado o no, suscripciones activas, así como datos proporcionados por APIs externas (por ejemplo, número de horas jugadas) o necesidades especiales que el usuario haya indicado.

Esta información debe almacenarse de forma segura, garantizando tanto la integridad ante posibles pérdidas de datos como la protección de información sensible, especialmente las contraseñas.

Asimismo, la base de datos debe ser capaz de gestionar peticiones de forma rápida y eficiente, permitiendo operaciones de lectura, inserción y modificación de datos sin comprometer el rendimiento del sistema.

En definitiva, la base de datos debe garantizar la disponibilidad, consistencia, flexibilidad y seguridad de la información en todo momento.

5.2.2. Backend

El backend es el núcleo funcional de la plataforma, encargado de procesar la lógica de la aplicación, gestionar los datos de los usuarios y coordinar la comunicación entre el frontend, la base de datos y los modelos de lenguaje.

Una de sus principales responsabilidades es la gestión de usuarios. Para ello, debe mantener un modelo interno que represente al usuario actual, proporcionando métodos que permitan modificar su información, sincronizar los cambios con la base de datos, y manejar correctamente el estado de sesión. Además, debe facilitar el envío y recepción de datos al frontend de manera eficiente y segura.

Otro aspecto central del backend es el sistema de recomendación basado en [LangChain](#). Cuando el usuario realiza una solicitud (por ejemplo, pedir una recomendación de juegos), el backend debe encargarse de preparar el contexto necesario, incluyendo los datos del usuario (si está autenticado), y enviarlo a LangChain. El modelo generará una respuesta personalizada, que luego será procesada y enviada al frontend para su visualización.

En esta capa también se define la configuración de los modelos generativos de lenguajes implicados, cómo interactúan entre sí y cómo se les indica qué tarea deben realizar. Esto incluye desde la selección de cadenas de herramientas hasta el uso de agentes, dependiendo del flujo de la conversación.

Capítulo 5: Diseño

o la consulta del usuario.

El controlador, ubicado en la parte superior del backend, actúa como intermediario entre el frontend y las capas lógicas inferiores. Expone una serie de métodos que permiten al frontend obtener y enviar datos, mientras que las capas inferiores se encargan de realizar operaciones sobre la base de datos y gestionar la lógica de recomendación.

En conjunto, el backend asegura que la plataforma funcione correctamente, manteniendo la coherencia de los datos, ejecutando las operaciones necesarias y respondiendo de forma eficaz a las interacciones del usuario.

5.2.3. Frontend

El frontend es la capa visual de la plataforma y el punto de interacción directa con el usuario. Su diseño debe priorizar la usabilidad, la claridad en la presentación de la información y la adaptación a distintos dispositivos. Para ello, es fundamental distribuir correctamente los elementos en pantalla, emplear una paleta de colores agradable y coherente, y aplicar un diseño responsive que se adapte a diferentes resoluciones y tamaños de pantalla.

La interfaz contará con una página principal accesible para todos los usuarios, en la que podrán iniciar sesión, registrarse o acceder a una recomendación sencilla sin necesidad de autenticarse. Una vez que el usuario haya iniciado sesión, podrá acceder a funcionalidades más avanzadas, como la edición de su información personal, la solicitud de recomendaciones personalizadas o la consulta de contenido actualizado. Este contenido incluirá noticias destacadas, últimos lanzamientos adaptados a sus consolas y juegos favoritos, o recomendaciones populares entre usuarios con intereses similares.

Desde el punto de vista técnico, el frontend debe comunicarse con el backend mediante peticiones al controlador. A través de estas peticiones podrá recuperar la información necesaria (como recomendaciones o datos del perfil del usuario) y enviar nuevos datos introducidos por el usuario, como actualizaciones de su perfil o nuevas preferencias.

El diseño del frontend no solo debe ser estéticamente atractivo, sino también funcional, guiando al usuario en su experiencia y facilitando la interacción con la plataforma de forma intuitiva y eficiente.

Capítulo 5: Diseño

5.3. Diseño inicial de la interfaz

Para facilitar el desarrollo de la plataforma, es útil realizar un esbozo preliminar de la interfaz con la que interactuará el usuario. En las siguientes figuras se presentan algunos bocetos que muestran cómo se espera que el usuario visualice y navegue por la plataforma.

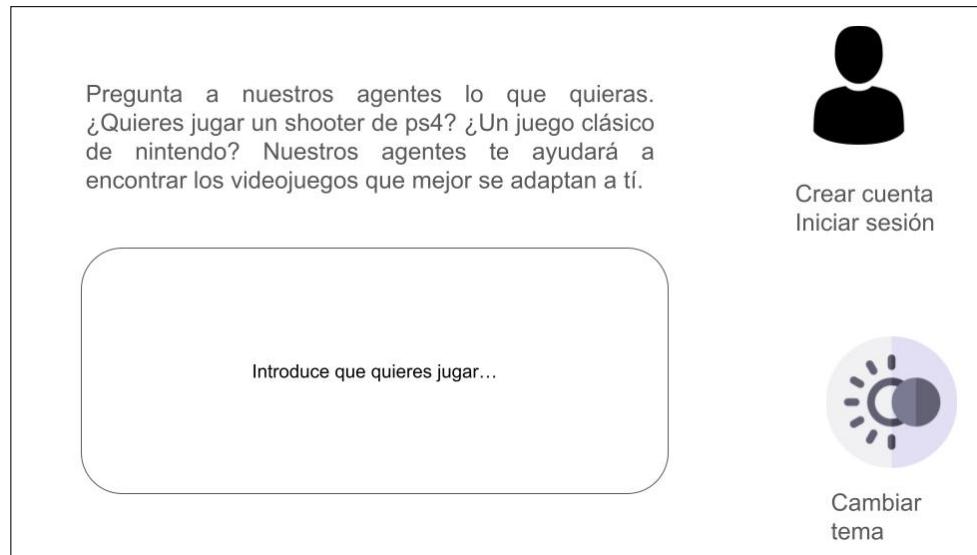


Figura 5.2: Diseño preliminar de la página de inicio.

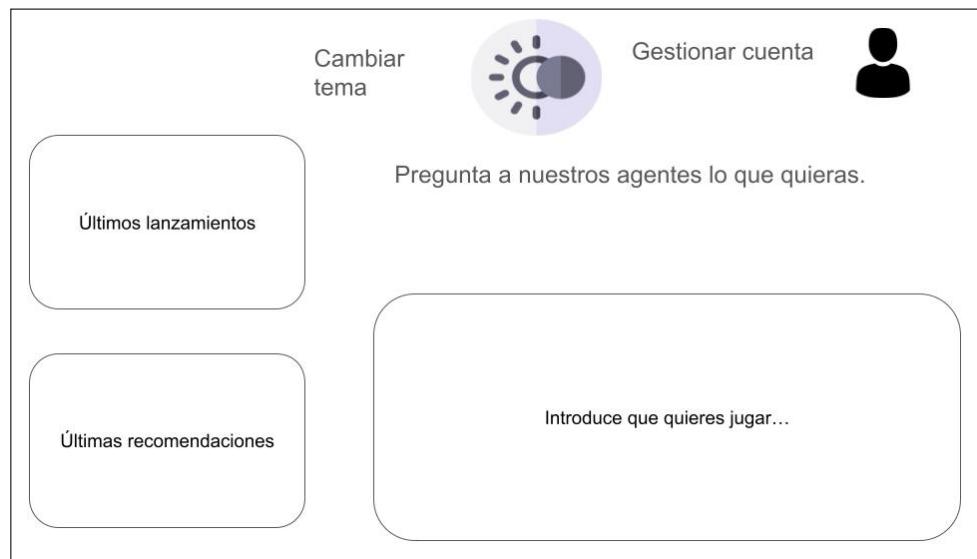


Figura 5.3: Diseño preliminar de la página principal tras iniciar sesión.

Capítulo 5: Diseño



Figura 5.4: Diseño preliminar de la página de ajustes.

Capítulo 6

Implementación

En este capítulo, nos centraremos en la implementación del diseño detallado en el capítulo anterior.

Describiremos los pasos y técnicas utilizadas para llevar a cabo la construcción de la plataforma, desde la configuración del entorno de desarrollo hasta la integración de los diferentes componentes del sistema.

También se detallará el proceso de programación, las herramientas empleadas y las decisiones técnicas tomadas para asegurar que el diseño se traduzca eficazmente en una solución funcional.

Además, abordaremos los retos encontrados durante la implementación y las soluciones adoptadas.

Capítulo 6: Implementación

6.1. Sistema operativo

Para el desarrollo de la plataforma se ha utilizado el sistema operativo Ubuntu, en concreto su versión 24.04 [19].

Ubuntu es una distribución de Linux orientada al software libre y a la facilidad de uso. Es gratuita, fácil de instalar y ampliamente utilizada en el ámbito académico, especialmente en titulaciones relacionadas con la informática como la nuestra.

Además, gracias al uso de un script ejecutable, es posible instalar fácilmente todas las dependencias necesarias para el proyecto, lo que facilita la configuración del entorno de desarrollo en otros equipos.



Figura 6.1: **Logo de Ubuntu**. Una de las mejores distribuciones de Linux, gratuita e ideal para programadores. <https://www.drouiz.com/wp-content/uploads/2015/12/ubuntu-logo2.jpg>.

Capítulo 6: Implementación

6.2. GitHub

Para mantener nuestro trabajo seguro, con posibilidad de recuperar versiones anteriores y documentar adecuadamente los cambios realizados, utilizamos GitHub como sistema de control de versiones.

GitHub permite crear un repositorio que contiene todos los archivos del proyecto. Este repositorio puede clonarse en cualquier ordenador, facilitando así el trabajo colaborativo o desde distintos entornos. Una de sus principales funcionalidades es la posibilidad de crear distintas ramas, lo que permite, por ejemplo, desarrollar nuevas funcionalidades o corregir bugs de forma aislada. Posteriormente, estos cambios pueden fusionarse con la rama principal, conservando un historial detallado de versiones.

Gracias a GitHub, no solo almacenamos nuestro trabajo de forma remota en la nube, sino que también facilitamos el control de cambios, la colaboración con otras personas y la organización del desarrollo del proyecto [20].

El enlace al repositorio del proyecto es el siguiente: <https://github.com/juuaann03/TFG>



Figura 6.2: **Logo de GitHub.** GitHub es una plataforma gratuita de desarrollo colaborativo que permite a los programadores gestionar proyectos y compartir códigos de manera eficiente y segura. https://cdn.prod.website-files.com/5f5a53e153805db840dae2db/64e79ca5aff2fb7295bfddf9_github-que-es.jpg.

Capítulo 6: Implementación

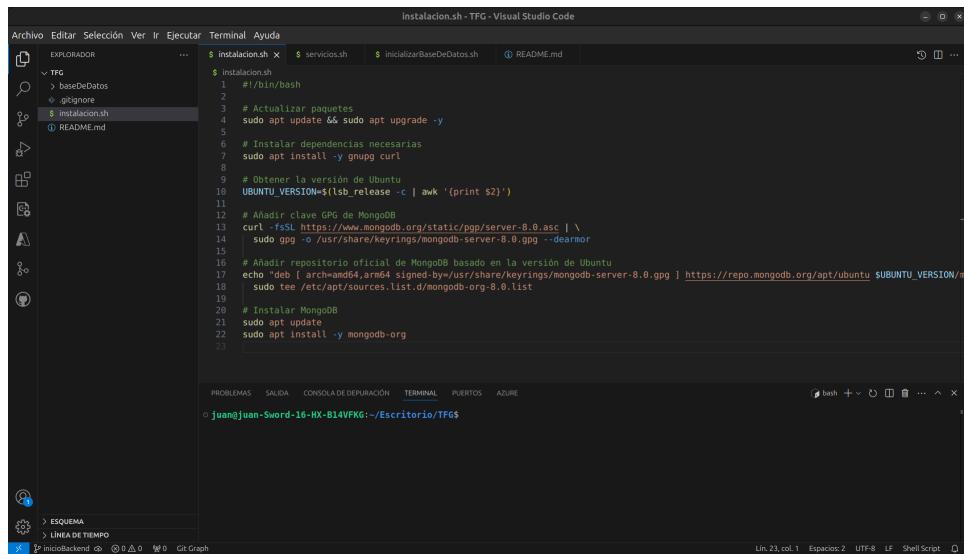
6.3. Entorno de desarrollo

Para el desarrollo de la plataforma se ha utilizado un entorno de desarrollo integrado (IDE), concretamente Visual Studio Code.

Un entorno de desarrollo proporciona un conjunto de herramientas que facilitan la programación, tales como la visualización estructurada de archivos, resaltado de sintaxis, autocompletado, depuración, y la integración directa con sistemas de control de versiones como Git y GitHub.

Visual Studio Code es un IDE moderno, ligero y altamente configurable, compatible con una amplia variedad de lenguajes de programación. Ofrece una gran cantidad de extensiones, incluyendo soporte avanzado para Python, HTML, JavaScript... Además, su integración con Git y GitHub permite realizar operaciones de control de versiones de forma sencilla y eficiente desde el propio editor.

Su interfaz intuitiva y su flexibilidad lo convierten en una herramienta ideal para el desarrollo de nuestra plataforma.



The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows a folder structure with 'TFG' containing 'baseDeDatos', '.gitignore', 'instalacion.sh', and 'README.md'. A file named 'servicios.sh' is also listed in the root.
- Terminal:** The 'instalacion.sh' file is open in the terminal tab, displaying a shell script for MongoDB installation. The script includes commands for updating packages, installing dependencies, adding a GPG key, and adding the official MongoDB repository to the Ubuntu package list.
- Status Bar:** Shows the path 'juan@juan-Sword-16-HX-B14VFKQ:~/Escritorio/TFG\$', the line number 'Lin. 23, col. 1', and the encoding 'UTF-8 LF Shell Script'.

Figura 6.3: **Interfaz de Visual Studio Code.** Su interfaz es llamativa, limpia, intuitiva y personalizable. Permite ver todos los archivos de manera estructurada, gestionar las ramas de Git y resolver los conflictos de manera fácil y eficiente, entre otras funciones muy útiles.

Capítulo 6: Implementación

6.4. Base de datos: MongoDB

Para el almacenamiento de datos en nuestra plataforma, se ha optado por utilizar **MongoDB**, una base de datos NoSQL ampliamente utilizada debido a su potencia, escalabilidad y flexibilidad. MongoDB almacena los datos en documentos BSON (una representación binaria de JSON), lo cual permite una estructura de datos más dinámica en comparación con las bases de datos relacionales tradicionales.

Una de las principales ventajas de MongoDB es su capacidad para manejar documentos con esquemas variables. Por ejemplo, un usuario puede tener 20 videojuegos marcados como favoritos, otro usuario solo 5, y otro ninguno, sin que ello suponga un problema a nivel de estructura de base de datos. Además, permite añadir o eliminar campos fácilmente durante el desarrollo sin necesidad de redefinir esquemas rígidos, algo que sería considerablemente más complejo en bases de datos SQL o Oracle [21] [22].

Para nuestra plataforma se ha creado una única base de datos denominada **LangGames** —una combinación de *LangChain* y *video games*— y una colección principal llamada **Usuarios**, donde se almacenan los perfiles y preferencias de los usuarios.

Con el fin de facilitar la configuración del entorno de desarrollo, se han creado tres scripts bash:

- Un script para la instalación de MongoDB, que configura los repositorios oficiales y realiza la instalación de los paquetes necesarios.
- Un script para lanzar y habilitar el servicio de MongoDB al inicio del sistema.
- Un script que inicializa la base de datos *LangGames* y crea la colección *Usuarios*, asegurando que el sistema esté listo para su uso desde el primer momento.



Figura 6.4: **Logo de MongoDB.** MongoDB es una base de datos orientada a documentos, altamente flexible y escalable, ideal para el desarrollo ágil de aplicaciones. https://www.ovhcloud.com/sites/default/files/styles/large_screens_1x/public/2022-03/black.png.

Capítulo 6: Implementación

6.5. BackEnd: Python, FastAPI y LangChain

El backend de la plataforma ha sido desarrollado en **Python**, un lenguaje de programación interpretado, multiparadigma y de alto nivel. Su sintaxis sencilla y su extensa comunidad lo han posicionado como una de las principales opciones en el desarrollo de software moderno, especialmente en áreas como la inteligencia artificial, ciencia de datos y desarrollo web [23].

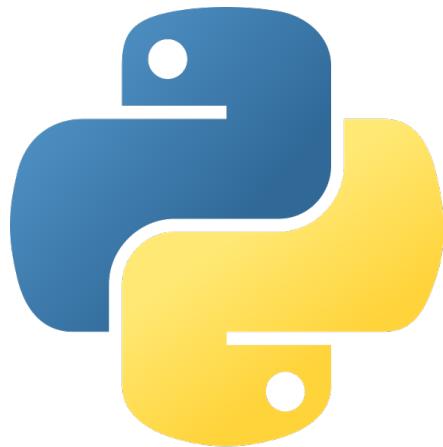


Figura 6.5: **Logo de Python.** Python es uno de los lenguajes de programación más populares y versátiles del panorama actual. <https://upload.wikimedia.org/wikipedia/commons/thumb/c/c3/Python-logo-notext.svg/640px-Python-logo-notext.svg.png>.

Para construir la Api que comunica el backend con el frontend, se ha utilizado la biblioteca **FastAPI**. Esta herramienta permite definir endpoints de manera declarativa y estructurada, facilitando la validación de datos mediante el uso de modelos definidos con *Pydantic*. Además, gracias a su integración con herramientas como OpenAPI, proporciona una documentación automática e interactiva.

El servidor de desarrollo que ejecuta la aplicación FastAPI se gestiona mediante **Uvicorn**, un servidor ASGI (Asynchronous Server Gateway Interface) ligero y de alto rendimiento, especialmente diseñado para aplicaciones web modernas basadas en Python asíncrono. [24] [25]

Capítulo 6: Implementación

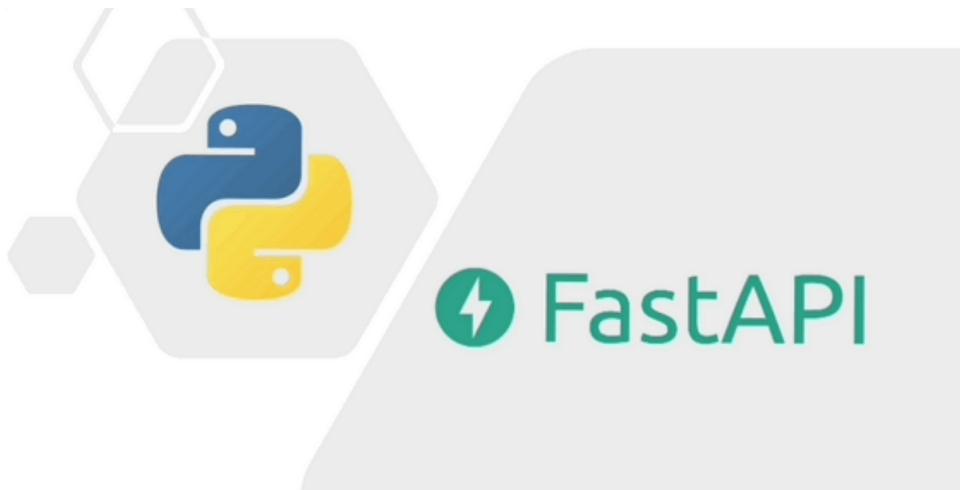


Figura 6.6: **Logo de FastAPI**. FastAPI es una potente biblioteca de Python que permite construir APIs de forma rápida, sencilla y robusta. https://miro.medium.com/v2/resize:fit:1200/1*gTztqjO7u5-GVx2cowVPsA.png.

Para mantener un entorno limpio y controlado, se utiliza un **entorno virtual** de Python, el cual permite aislar las dependencias del proyecto del resto del sistema. Esto evita conflictos entre bibliotecas y facilita la portabilidad del entorno de desarrollo.

El backend incluye dos scripts bash:

- **Script de instalación de Python y pip:** actualiza los paquetes del sistema e instala Python 3 junto con `pip` (el gestor de paquetes de Python) y `venv` (módulo para entornos virtuales).
- **Script de inicialización del entorno:** crea y activa un entorno virtual si no existe, instala todas las dependencias necesarias para el backend (incluyendo FastAPI, Uvicorn, LangChain y otras bibliotecas auxiliares), genera un archivo `requirements.txt` con la lista de dependencias, y lanza el servidor FastAPI en segundo plano, redirigiendo su salida a un archivo de log.

Dentro del backend se encuentra el archivo principal `main.py`, la carpeta `app/` que contiene los distintos módulos del sistema, y un archivo de entorno(`.env`) adicional para gestionar claves necesarias para el correcto funcionamiento de ciertas funcionalidades, cuya obtención y uso está debidamente documentada en el `README` del repositorio de GitHub.

A continuación, se describe con mayor detalle la estructura y funcionalidad de la carpeta `app/`.

Capítulo 6: Implementación

6.5.1. DB

La carpeta `db` contiene un único archivo llamado `mongodb`, el cual se encarga de gestionar la conexión con la base de datos MongoDB.

En él se utiliza la biblioteca `pymongo` para establecer la conexión con el servidor local de MongoDB (`localhost:27017`). Una vez conectados, se selecciona la base de datos `LangGames` y se almacena la colección `Usuarios` en una variable que se puede importar fácilmente desde otros módulos del backend.

Esta estrategia favorece la reutilización y centralización del acceso a la base de datos, evitando duplicación de código y facilitando su mantenimiento.

6.5.2. Utils

Esta carpeta contiene dos archivos con funciones auxiliares esenciales para el funcionamiento del backend.

- **jwt:** Este archivo se encarga de generar **tokens JWT (JSON Web Tokens)**, que permiten autenticar de forma segura a los usuarios [26]. Cada vez que un usuario inicia sesión correctamente, se le genera un token que contiene su correo electrónico (u otro dato identificativo) y una fecha de expiración. Este token se firma digitalmente usando una clave secreta, de modo que el servidor pueda verificar su validez en futuras peticiones sin necesidad de mantener sesiones abiertas. Así se garantiza una comunicación segura y sin estado entre el cliente y el servidor.
- **Utilidades Varias:** Aquí se agrupan funciones y configuraciones de utilidad. Por ejemplo:
 - Se cargan las claves necesarias para acceder a las APIs externas desde un archivo `.env`.
 - Se definen listas con los modelos de lenguaje disponibles para su uso.
 - `limpiar_respuesta()`: una función que extrae y limpia la parte JSON de las respuestas generadas por modelos de lenguaje.
 - `obtener_imagen_juego()`: consulta la API de RAWG para obtener automáticamente una imagen representativa del videojuego que se le indique.

6.5.3. Modelos

Los modelos representan la estructura de los datos con los que trabaja el sistema, facilitando la organización y el intercambio de información entre distintas partes del backend.

Capítulo 6: Implementación

El modelo principal es el del **usuario**, que incluye tanto los datos básicos como el nombre, correo y contraseña, como otros campos más específicos relacionados con el mundo de los videojuegos. Por ejemplo, se almacena información sobre las consolas que posee, la configuración de su ordenador, sus gustos personales, los juegos que ha jugado o que no le han gustado, e incluso su historial de los videojuegos recomendados por los modelos de lenguaje. Esta estructura permite personalizar al máximo la experiencia del usuario.

Además del modelo de usuario, se han definido otras clases auxiliares para manejar información concreta:

- Modelos para solicitudes de recomendación, tanto básica como personalizada.
- Un modelo para representar los próximos lanzamientos de videojuegos, incluyendo su título, fecha, plataformas y una imagen asociada.

Estas clases permiten que los distintos módulos del backend se comuniquen utilizando estructuras coherentes y bien definidas, lo que facilita la validación de datos y mejora la organización del código.

6.5.4. Gestores

Los gestores se encargan de realizar las operaciones relacionadas con los datos que maneja el sistema. En este caso, contamos con un gestor centrado en los usuarios.

El **gestor de usuarios** proporciona una serie de funciones que permiten crear un nuevo usuario, obtener sus datos, modificarlos o eliminarlos. También incluye la verificación de contraseñas para el proceso de inicio de sesión y una función para limpiar los campos opcionales del perfil del usuario. Además aplica un hash a las contraseñas para que el sistema sea más seguro.

Este archivo actúa como intermediario entre la base de datos y otras partes de la aplicación, como las rutas o los servicios. Gracias a él, se centraliza la lógica relacionada con los usuarios, facilitando el mantenimiento y la reutilización del código.

6.5.5. Servicios

Los servicios se encargan de implementar las funcionalidades centrales del sistema.

El servicio de usuario actúa como intermediario entre el gestor de usuarios y los demás servicios. Su función principal es preprocessar los datos recibidos del usuario, invocar los servicios correspondientes con dichos datos y aplicar las modificaciones necesarias en el perfil.

Capítulo 6: Implementación

El servicio de recomendación básica está diseñado para usuarios no autenticados que solicitan recomendaciones de videojuegos. Ante una petición, se consultan hasta tres modelos de lenguaje para obtener posibles respuestas. Si algún modelo falla, el sistema puede seguir con normalidad. Una vez obtenidas las respuestas, un modelo la sintetiza en una sola. Si el modelo falla, se sigue con los siguientes hasta que se obtiene una respuesta válida. Esta gestión secuencial de modelos se facilita mediante la biblioteca `LangChain`, que simplifica la invocación, manejo y conmutación entre modelos. Para acceder a los modelos, se utiliza `OpenRouter`, una API que permite interactuar con múltiples proveedores y modelos sin necesidad de gestionar varias claves de acceso. `OpenRouter` también ofrece una prueba gratuita limitada en tokens.

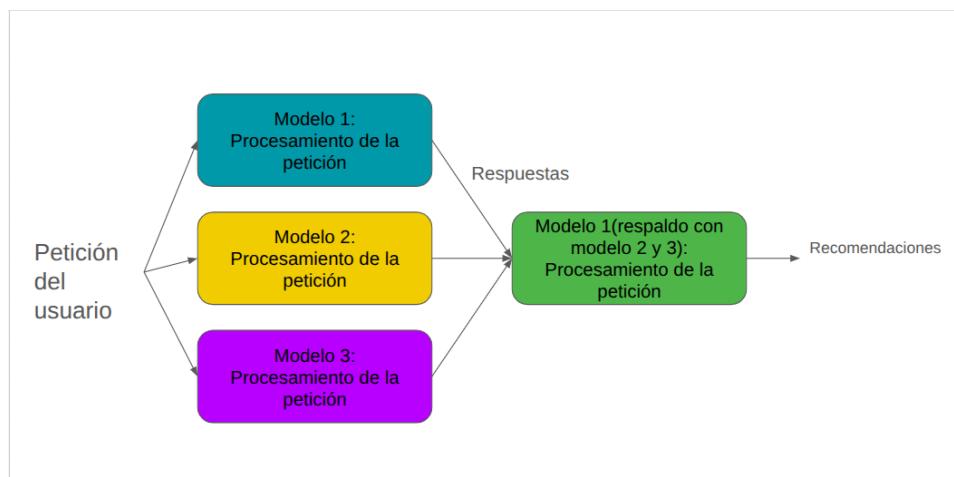


Figura 6.7: **Esquema recomendación básica.** Para generar una recomendación básica, 3 modelos de lenguaje reciben la petición, la procesan y generan sus respuestas. Finalmente un modelo, con sus correspondientes respaldo, crea una respuesta única.

El servicio de recomendación personalizada extiende la funcionalidad básica incorporando datos específicos del usuario, como su historial de videojuegos y recomendaciones previas, para evitar redundancias. Además, este servicio puede actualizar el perfil del usuario automáticamente si la petición incluye información que modifica su estado, como la venta de una consola o la adquisición de un nuevo juego.

Capítulo 6: Implementación

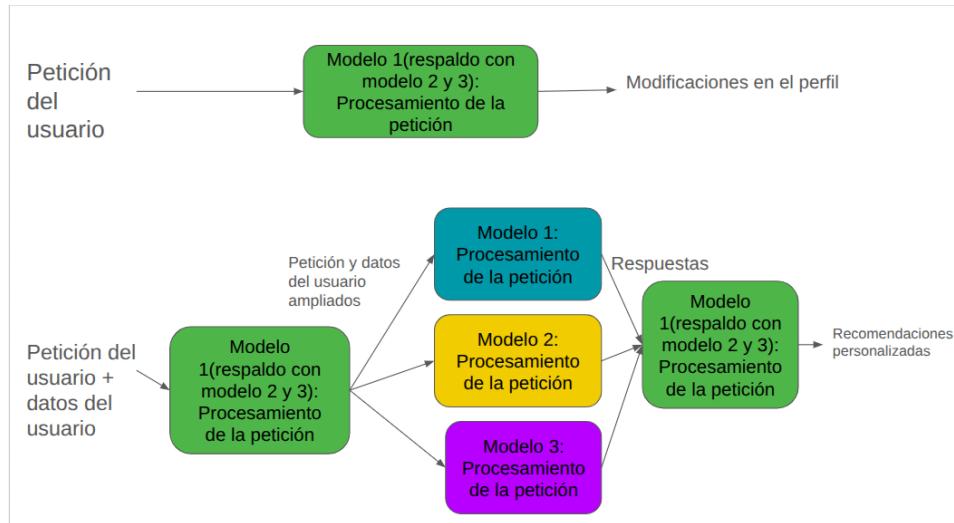


Figura 6.8: **Esquema recomendación personalizada.** Para generar una recomendación personalizada, un modelo con sus correspondientes respaldos analiza los datos del usuario y la petición para generar unos datos más ricos, a continuación 3 modelos de lenguaje reciben la petición, la procesan y generan sus respuesta. Finalmente un modelo, con sus correspondientes respaldo, crea una respuesta única. Paralelamente se analiza la petición por si implica una modificación en la información sobre videojuegos del usuario.

Por otro lado, existe un servicio dedicado exclusivamente a **actualizar la información relativa a los videojuegos del usuario** a partir del texto que éste proporciona. Este servicio utiliza modelos de lenguaje para analizar la petición y detectar los cambios necesarios, devolviendo una actualización precisa y estructurada.

Asimismo, contamos con dos servicios innovadores que mejoran significativamente la experiencia de usuario:

1. **Servicio de próximos lanzamientos:** Este servicio obtiene información sobre videojuegos próximos a salir mediante la API de RAWG, incluyendo nombres, géneros, plataformas y imágenes. Los modelos analizan estos datos para filtrar y recomendar aquellos títulos que mejor se ajustan a los gustos del usuario.
2. **Servicio de integración con Steam:** Mediante la API de Steam, este servicio recupera la lista de juegos poseídos por el usuario y el tiempo de juego asociado a cada uno, usando su ID de Steam. Basándonos en esta información, actualizamos el perfil del usuario, considerando que un juego está realmente jugado si el usuario ha dedicado más de una hora a jugarlo.

Capítulo 6: Implementación

6.5.6. Rutas

Las rutas permiten exponer las funcionalidades del sistema a través de peticiones HTTP, especificando el tipo de operación (GET, POST, PUT, DELETE), la ruta asociada (path), los posibles errores, y los datos requeridos o devueltos. Estas rutas actúan como puntos de entrada a los servicios internos de la aplicación, es decir, el controlador.

En el sistema se han definido rutas para distintas funcionalidades clave:

- **Autenticación:** Permite a los usuarios iniciar sesión mediante la verificación de credenciales. Al autenticarse correctamente, se genera y devuelve un token JWT que puede ser utilizado en peticiones protegidas.
- **Gestión de usuarios:** Incluye rutas para crear, obtener, actualizar y eliminar usuarios, así como para modificar sus datos obligatorios u opcionales, acceder a sus preferencias y sincronizar sus datos con Steam.
- **Recomendaciones:** Ofrece rutas tanto para recomendaciones básicas (sin necesidad de estar autenticado) como para recomendaciones personalizadas basadas en el perfil y el historial del usuario.
- **Próximos lanzamientos:** Permite obtener una lista de videojuegos próximos a lanzarse, filtrados en función de los gustos del usuario, utilizando datos de la API de RAWG.

Cada ruta está asociada a un módulo independiente dentro de la aplicación, lo que favorece una organización modular y facilita el mantenimiento y la ampliación del sistema.

6.5.7. Main

El archivo `main` actúa como punto de entrada de la aplicación. Su función principal es inicializar el servidor web y registrar todas las rutas disponibles en el sistema. Para ello, importa los distintos módulos de rutas definidos en la aplicación y los incluye en la instancia principal de `FastAPI` mediante el método `include_router()`.

Además, se configura el middleware de CORS (*Cross-Origin Resource Sharing*) para permitir que la aplicación frontend (por ejemplo, una interfaz en Angular corriendo en `http://localhost:4200`) pueda comunicarse con el backend sin restricciones de origen.

Este archivo es esencial para arrancar el sistema y exponer todos los servicios a través de una única instancia centralizada.

Capítulo 6: Implementación

6.6. FrontEnd: Angular

El frontend de la aplicación ha sido desarrollado utilizando **Angular**, un framework de desarrollo web moderno, mantenido por Google, que facilita la creación de aplicaciones de una sola página (SPA) con una arquitectura robusta y escalable [27].

Angular se basa principalmente en los lenguajes **HTML**, **SCSS/CSS** y **TypeScript**. Para el diseño visual y la personalización del estilo, se ha utilizado el framework de utilidades CSS **Tailwind CSS**, que permite construir interfaces de usuario de forma rápida y flexible mediante clases predefinidas.

Al igual que las otras partes cuenta con scripts, para instalar lo necesario e inicializar el frontend en segundo plano. También cuenta con un script que se utilizó para comenzar el desarrollo.

Estos scripts simplifican considerablemente el proceso de despliegue local, garantizando que cualquier desarrollador pueda ejecutar el frontend con facilidad.



Figura 6.9: **Logo de Angular**. Angular es un framework completo y muy utilizado para el desarrollo de interfaces web modernas. Fuente: https://media.lcdn.com/dms/image/v2/D4D12AQHF0nuL7dxEOA/article-cover_image-shrink_720_1280/article-cover_image-shrink_720_1280/0/1710076882649?e=2147483647&v=beta&t=z_GWbz1UXR5oIm-mSUhdCJnspOV0vronehDj9-o7xOhI.

Angular genera automáticamente gran parte de la estructura inicial del proyecto, lo cual permite centrarse desde el inicio en el desarrollo funcional y visual de la aplicación.

Uno de los elementos clave en la personalización del diseño es la configuración del sistema de estilos. Para ello se ha utilizado **Tailwind CSS**, que

Capítulo 6: Implementación

ofrece una forma muy eficiente y flexible de aplicar estilos directamente en los componentes. En este caso, se ha habilitado el modo claro y oscuro, de forma que el diseño se adapta según las preferencias del usuario. Además, se han realizado pequeños ajustes para asegurar que, incluso en campos de formularios autocompletados, los colores se mantengan coherentes con el tema activo.

La estructura principal del frontend se encuentra en la carpeta src donde reside el código fuente. Allí se definen aspectos globales como el título de la página, el ícono del navegador, la dirección base de la API y los estilos que afectan a toda la aplicación.

Dentro de esta estructura se encuentra la carpeta principal del proyecto, app, que es donde se desarrolla casi toda la funcionalidad. Esta parte incluye, por ejemplo, la definición de las rutas de navegación, que permiten moverse entre las distintas vistas como el inicio, el registro o la pantalla principal tras iniciar sesión.

También se incluye un sistema de servicios en la carpeta services, que se encargan de gestionar las conexiones con la API. Uno de estos servicios, por ejemplo, permite obtener los próximos lanzamientos de videojuegos relacionados con el usuario. Estos datos se almacenan temporalmente en memoria caché para evitar repetir llamadas innecesarias, lo que mejora la eficiencia y la velocidad de la aplicación.

Además, se han definido varias **interfaces** que actúan como plantillas para los datos que se van a utilizar, como pueden ser las recomendaciones personalizadas, los lanzamientos futuros o los datos de los usuarios. Estas interfaces permiten trabajar con los datos de forma más organizada y segura.

A continuación, se describe la carpeta de *componentes*, que contiene las diferentes piezas visuales y funcionales que forman la interfaz del sistema.

6.6.1. Componentes

En Angular, un componente es una de las piezas clave para construir interfaces. Cada uno representa una parte específica de la aplicación y cuenta con su propia estructura (HTML), comportamiento (TypeScript) y estilo (CSS o SCSS). Gracias a la modularidad que ofrece este enfoque, podemos construir interfaces complejas a partir de componentes más simples y reutilizables.

En nuestro caso, todos los componentes han sido diseñados pensando en la experiencia del usuario. Incluyen un botón para alternar entre modo claro y oscuro, se adaptan automáticamente a pantallas pequeñas (diseño responsive), e incorporan pequeñas animaciones que hacen la experiencia más agradable. Por ejemplo, se utiliza una Poké Ball giratoria durante las cargas normales, y un R2D2 para mostrar mientras carga los próximos lanzamientos.

A continuación, describimos cada componente de forma individual.

Capítulo 6: Implementación

Home

El componente Home es la página principal de la aplicación. Nada más entrar, el usuario se encuentra con una cabecera que incluye el nombre de la app, su logotipo, y dos botones: uno para iniciar sesión y otro para registrarse.

En el centro de la página hay un cuadro de texto donde el usuario puede escribir qué tipo de juego está buscando. Una vez enviada la petición, se muestran las recomendaciones en formato de tarjetas. Cada tarjeta incluye una imagen del juego, su nombre, las plataformas disponibles y una breve explicación de por qué se ha recomendado ese juego en concreto.

En la parte inferior, se encuentra una sección con un segundo logotipo y el aviso de derechos de autor.

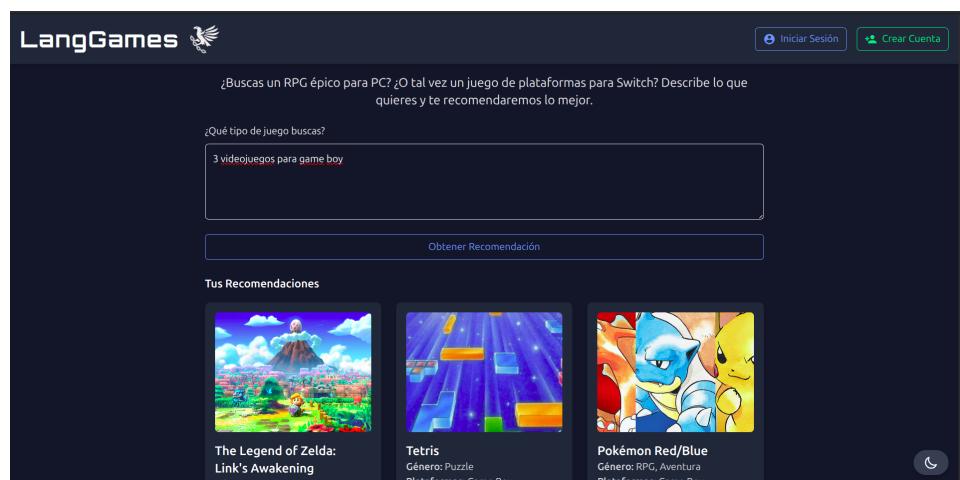


Figura 6.10: **Imagen de home 1.** Vista principal de la página con recomendaciones ya generadas.

Capítulo 6: Implementación

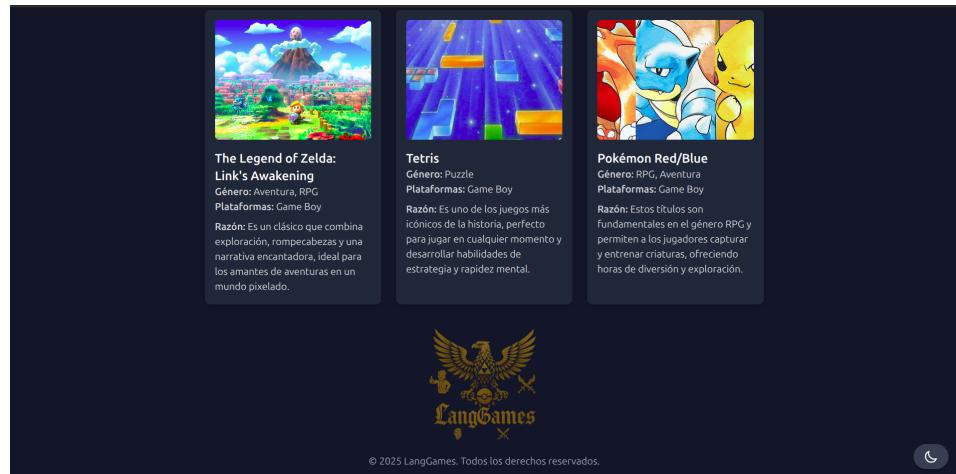


Figura 6.11: **Imagen de home 2.** Parte inferior de la página, con el segundo logotipo y el pie de página.

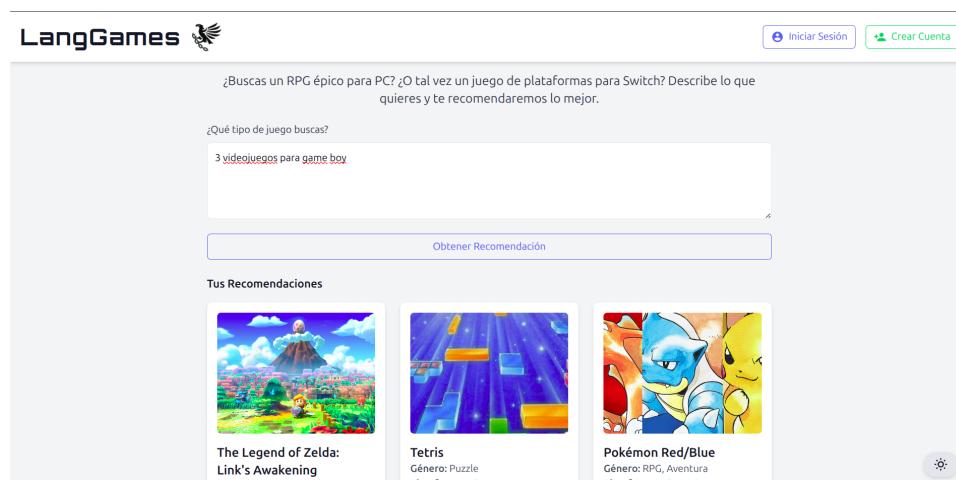


Figura 6.12: **Modo claro.** La aplicación ofrece tanto modo claro como oscuro, para mayor comodidad del usuario.

Capítulo 6: Implementación

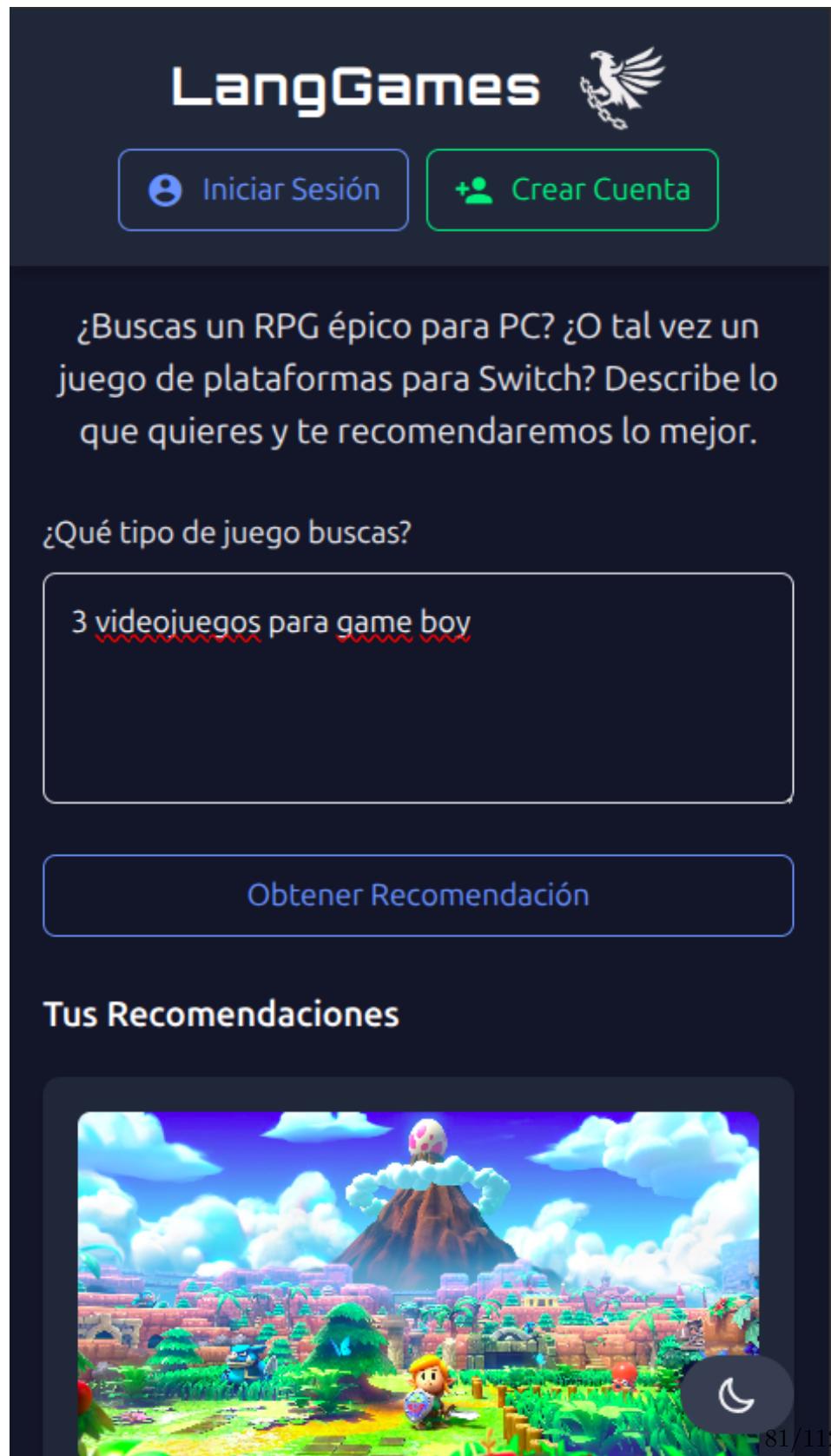
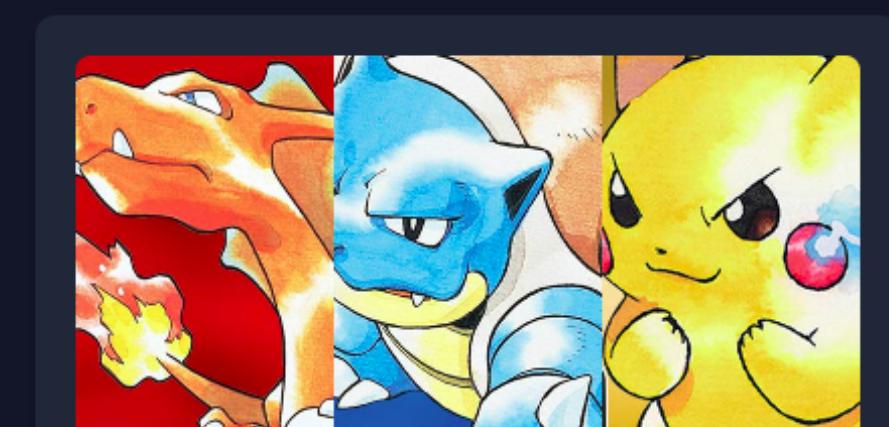


Figura 6.13: **Vista móvil 1.** Diseño adaptado para dispositivos móviles, con todos los elementos reorganizados.

Capítulo 6: Implementación

...nimiento y desarrollo habilidades de estrategia y rapidez mental.



Pokémon Red/Blue

Género: RPG, Aventura
Plataformas: Game Boy

Razón: Estos títulos son fundamentales en el género RPG y permiten a los jugadores capturar y entrenar criaturas, ofreciendo horas de diversión y exploración.



© 2025 LangGames. Todos los derechos reser



82/112

Figura 6.14: **Vista móvil 2.** Otra vista desde un dispositivo móvil, mostrando el aspecto del pie de página.

Capítulo 6: Implementación

Registro

El sistema de registro permite a los usuarios crear una cuenta nueva desde una ventana superpuesta a la pantalla principal *Home*. Para completar el registro, el usuario debe proporcionar un nombre de usuario, una dirección de correo electrónico válida y una contraseña segura.

El formulario valida en tiempo real que tanto el nombre como el correo electrónico no estén ya registrados en el sistema. En caso contrario, se muestra un mensaje de error al usuario indicando el conflicto.

En cuanto a la contraseña, esta debe cumplir con los siguientes requisitos mínimos:

- Tener una longitud mínima de 8 caracteres.
- Incluir al menos una letra y un número.

Una vez completado el formulario de manera correcta y superadas todas las validaciones, el sistema crea la cuenta de usuario y realiza el inicio de sesión de forma automática, redirigiendo al usuario a la pantalla principal.

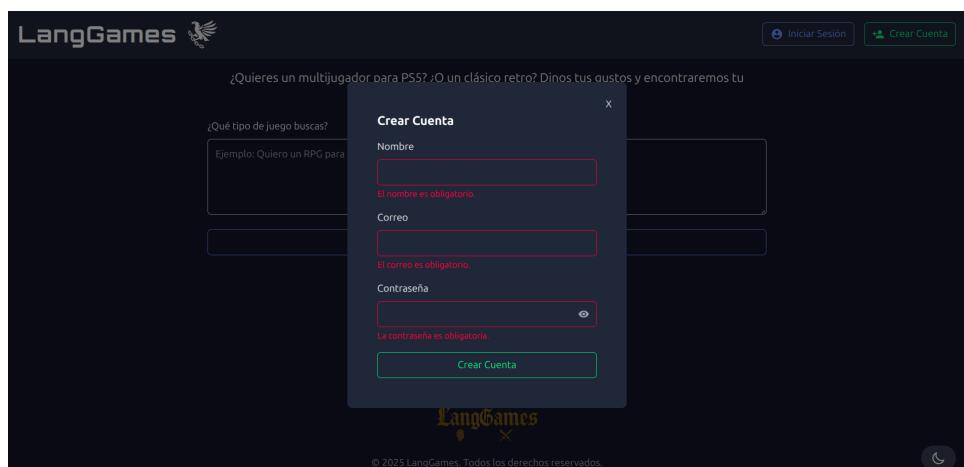


Figura 6.15: **Registro**. Formulario de creación de cuenta en la plataforma.

Login

El componente de *login* permite a los usuarios autenticarse en la plataforma introduciendo sus credenciales: correo electrónico y contraseña. El formulario verifica que los campos no estén vacíos y que el correo tenga un formato válido antes de enviar los datos.

En caso de que el usuario introduzca credenciales incorrectas o no esté registrado, se muestra un mensaje de error informativo. Si la autenticación es exitosa, se almacena el token de sesión, el rol del usuario, su nombre y su

Capítulo 6: Implementación

correo en el almacenamiento local del navegador, y se redirige automáticamente a la pantalla principal.

Además, el campo de contraseña ofrece la opción de mostrar u ocultar el texto introducido para mejorar la experiencia del usuario.

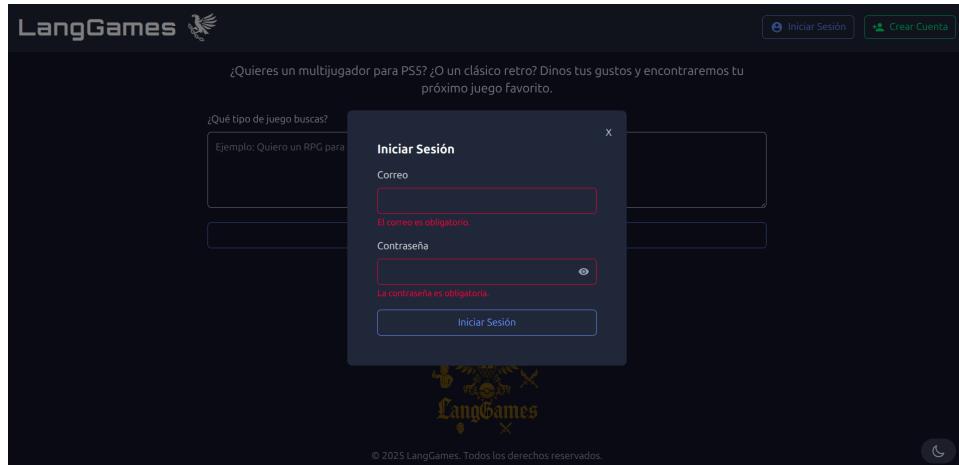


Figura 6.16: **Iniciar sesión.** Formulario para iniciar sesión en la plataforma.

Principal

Este es el componente principal de la aplicación, al que accede el usuario tras iniciar sesión o crear una cuenta.

La interfaz principal incluye una barra superior que muestra el nombre y el logotipo de la plataforma, así como un mensaje de bienvenida personalizado con el nombre del usuario. Además, se incorporan dos botones: uno para acceder a los ajustes de la cuenta y otro para cerrar sesión.

En la parte izquierda de la pantalla se muestran los cuatro próximos lanzamientos. Justo debajo, se encuentran las seis últimas recomendaciones disponibles. En el lateral derecho, hay un cuadro de texto donde el usuario puede solicitar recomendaciones, las cuales se mostrarán en la parte inferior tras ser generadas.

Capítulo 6: Implementación

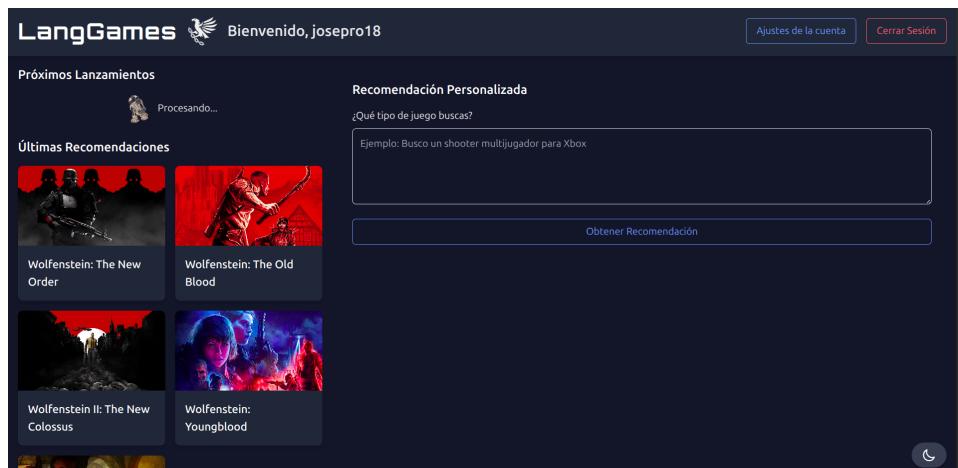


Figura 6.17: Página principal 1. Página principal de la plataforma.

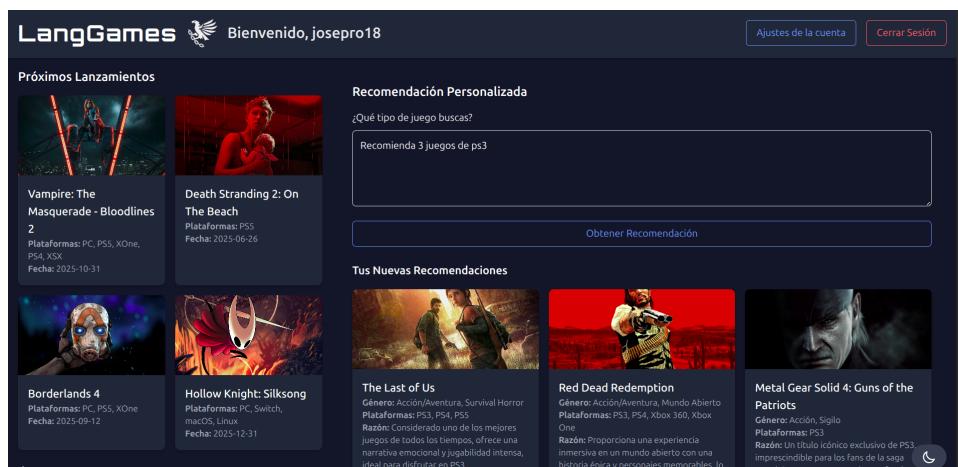


Figura 6.18: Página principal 2. Página principal de la plataforma con recomendaciones.

Ajustes de cuenta

La página de ajustes permite al usuario modificar su información personal y gestionar aspectos clave de su cuenta.

En la barra superior se encuentran dos botones: uno para vincular la cuenta con Steam mediante el ID de usuario, y otro para regresar a la página principal.

En la parte izquierda de la página hay otros dos botones. El primero permite cambiar el nombre de usuario y/o la contraseña, o eliminar la cuenta. Para cualquiera de estas acciones, es necesario introducir la contraseña actual. El segundo botón permite restablecer todos los datos relacionados con

Capítulo 6: Implementación

videojuegos almacenados en la cuenta del usuario. Debajo de estos botones se muestra un resumen con la información sobre los videojuegos del usuario.

En la parte derecha se encuentra un cuadro de texto donde el usuario puede introducir modificaciones mediante lenguaje natural, como por ejemplo indicar la adquisición de una nueva consola o la venta de un juego. El modelo de lenguaje interpreta la petición y aplica la modificación correspondiente, mostrando al usuario un mensaje con el resultado de la operación.

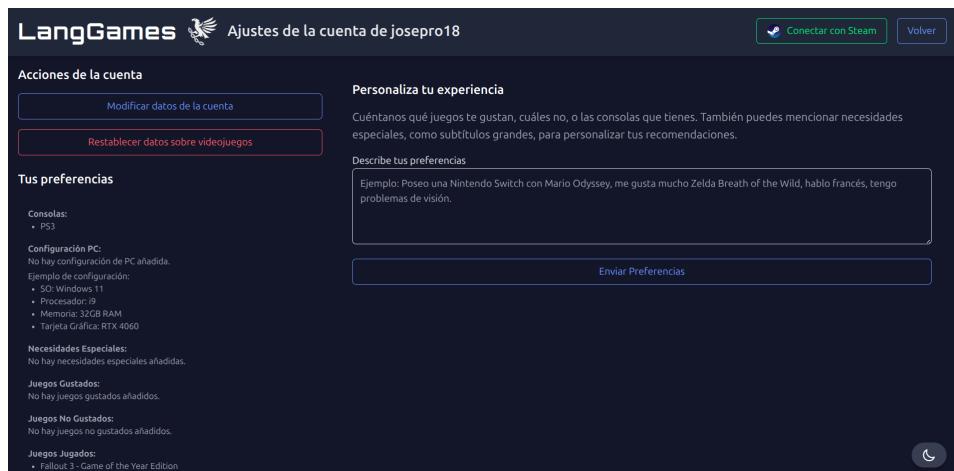


Figura 6.19: **Página de ajustes de la cuenta.** Interfaz donde el usuario puede gestionar la información de su cuenta y realizar ajustes relacionados con sus videojuegos.

Capítulo 6: Implementación

6.7. Puesta en marcha

Además, la aplicación incluye tres scripts globales: uno para ejecutar los tres scripts de instalación de los distintos componentes de la plataforma; otro para iniciar el backend y el frontend en segundo plano, almacenando la salida en un archivo de log y guardando sus PIDs para poder detenerlos posteriormente; y un tercer script que permite detener tanto el backend como el frontend que se ejecutan en segundo plano.

Capítulo 7

Pruebas

En este capítulo, se detallará el proceso de verificación del correcto funcionamiento de la plataforma a través de pruebas unitarias y de integración.

Describiremos cómo se diseñaron y ejecutaron estas pruebas para garantizar que cada módulo del sistema funcione de manera correcta y que la interacción entre los distintos componentes sea fluida y eficiente.

Asimismo se abordarán las herramientas y marcos de prueba utilizados, así como los métodos para identificar y corregir posibles errores o inconsistencias.

Igualmente se discutirán las pruebas de rendimiento, seguridad y usabilidad, evaluando cómo la plataforma responde bajo diferentes condiciones y garantizando una experiencia de usuario óptima.

Capítulo 7: Pruebas

7.1. Pruebas en la *base de datos*

Para verificar inicialmente el correcto funcionamiento de la base de datos, se utilizó `mongosh`, la consola interactiva de MongoDB. Se realizaron pruebas básicas para comprobar que el script de instalación creaba correctamente la base de datos y la colección correspondiente.

Además, se llevaron a cabo operaciones de inserción, eliminación y actualización de documentos, evaluando el comportamiento esperado del sistema. A medida que avanzaba el desarrollo del *backend* y el *frontend*, se continuó comprobando que las interacciones con la base de datos se realizaban correctamente, esta vez a través de los métodos implementados en cada uno de estos módulos.

Capítulo 7: Pruebas

7.2. Pruebas en el *backend*

Para realizar pruebas en el *backend*, se utilizó **Swagger**, una herramienta que permite visualizar y probar de forma interactiva las rutas de una API. Swagger facilita la documentación automática de las APIs RESTful, y en este caso fue generada automáticamente por **FastAPI**, el framework utilizado para el desarrollo del servidor.

Gracias a Swagger, accesible en la dirección http://localhost:8000/docs#, se pudieron ejecutar pruebas para cada uno de los endpoints definidos, especificando los campos requeridos, el cuerpo en formato JSON en caso de ser necesario, y visualizando tanto las respuestas del servidor como los posibles errores. Además el propio uvicorn cuando se ejecuta muestra que métodos ejecuta y los errores que ocurren.

Durante el desarrollo, se fue comprobando el correcto funcionamiento de cada uno de los métodos a través de esta interfaz. Esto permitió detectar y corregir errores de manera ágil, gracias a la retroalimentación inmediata proporcionada por la API. Asimismo, se realizaron pruebas introduciendo datos erróneos de forma intencionada con el objetivo de evaluar la robustez del sistema ante entradas inválidas o inconsistentes.

También se simularon escenarios de fallo como la indisponibilidad de los modelos de lenguaje —por ejemplo, debido a un exceso de tokens o a errores en el nombre del modelo— para verificar cómo respondía el sistema en situaciones excepcionales.

Posteriormente, durante el desarrollo del *frontend*, las pruebas comenzaron a realizarse principalmente desde la propia interfaz de usuario. No obstante, en ciertas ocasiones se continuó utilizando Swagger para llevar a cabo pruebas rápidas y directas sobre los endpoints.

The screenshot shows the FastAPI Swagger UI interface. At the top, it displays "FastAPI 0.1.0 OAS 3.0" and a link to "/openapi.json". The main area is titled "default" and contains a single endpoint: "GET / Readroot". Below this, under the "usuarios" section, there is a list of operations:

- "POST /usuarios/ Crearusuario" (highlighted in green)
- "GET /usuarios/porCorreo/{correo} Obtenerusuarioporcorreuta" (highlighted in blue)
- "DELETE /usuarios/porCorreo/{correo} Borrarusuarioporcorreuta" (highlighted in red)
- "PUT /usuarios/porCorreo/{correo}/Limpiar Limpiarcampousuariocorreo" (highlighted in yellow)
- "GET /usuarios/porCorreo/{correo}/obligatorios Obtenerdatosobligatorios" (highlighted in blue)
- "PUT /usuarios/porCorreo/{correo}/obligatorios Actualizardatosobligatorios" (highlighted in yellow)
- "GET /usuarios/porCorreo/{correo}/optativos Obtenerdatosoptativos" (highlighted in blue)
- "PUT /usuarios/porCorreo/{correo}/optativos Actualizardatosoptativos" (highlighted in yellow)

Figura 7.1: **Vista general de Swagger**. Vista general de la API generada por Swagger a partir del código de FastAPI.

Capítulo 7: Pruebas

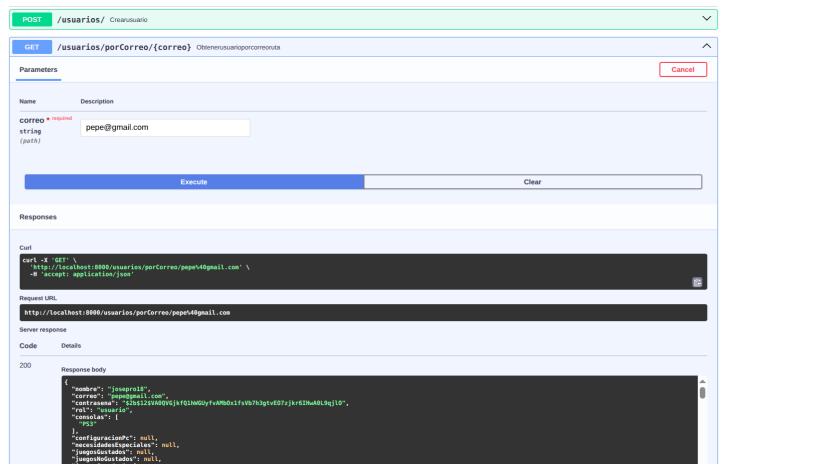


Figura 7.2: **Vista del uso de un método en Swagger.** Se muestra el método GET para obtener un usuario, al que se le pasa el correo electrónico como parámetro, devolviendo un JSON con los datos del usuario.

7.2.1. Pruebas unitarias y de integración en el *backend*

Además de las pruebas realizadas con Swagger, se han desarrollado pruebas unitarias e integradas para el módulo de *backend* utilizando la librería `unittest` junto con `mongomock`, una herramienta que simula el comportamiento de una base de datos MongoDB en memoria. Esto permitió realizar pruebas sin necesidad de conectarse a una base de datos real.

Estas pruebas se organizaron en dos ficheros separados dentro de una carpeta `test`: un archivo para gestionar usuario y otro para la autenticación. El primero contiene pruebas sobre la lógica de gestión de usuarios, mientras que el segundo verifica las rutas de autenticación expuestas por la API de FastAPI.

Las funciones principales del sistema se comprobaron de forma individual, incluyendo la creación, actualización, eliminación y verificación de usuarios. Para ello, se prepararon entornos simulados en cada prueba, creando una base de datos ficticia e inyectando los datos necesarios para comprobar el comportamiento del sistema. También se emplearon técnicas de *mocking* para simular el comportamiento de las funciones de encriptación de contraseñas (`bcrypt`) y de generación de tokens JWT, permitiendo centrarse en la lógica del sistema sin depender de funcionalidades externas.

En el caso de las rutas de autenticación, se utilizó `TestClient`, un cliente de pruebas proporcionado por FastAPI, para simular peticiones HTTP y verificar las respuestas del servidor. Se probaron casos como el inicio de sesión con credenciales correctas, contraseñas incorrectas y usuarios inexistentes, evaluando la respuesta del sistema ante distintos escenarios.

Estas pruebas automáticas permiten garantizar que los métodos clave

Capítulo 7: Pruebas

del *backend* funcionan correctamente, tanto de forma aislada como cuando interactúan entre sí. Además, al ejecutarse de manera repetida, facilitan la detección temprana de errores ante futuros cambios en el código.

Capítulo 7: Pruebas

7.3. Pruebas en el *frontend*

Para llevar a cabo las pruebas en el *frontend*, se inicializó el servidor de Angular y se accedió a la dirección local <http://localhost:4200/>. Desde esta interfaz se realizaron pruebas interactivas explorando todos los elementos visibles, accionando botones y navegando por los distintos componentes de la aplicación.

La detección de errores se realizaba principalmente a través de dos vías: por un lado, mediante los mensajes devueltos por la API en caso de errores de validación o fallos internos; por otro, mediante el uso de las herramientas de desarrollo del navegador. Estas herramientas, accesibles pulsando F12, permiten inspeccionar múltiples aspectos del comportamiento del sitio web.

En particular, se hizo uso de la **consola del navegador**, que permite visualizar errores de ejecución de Angular, advertencias, trazas de red, y mensajes de depuración. También se utilizó la pestaña de herramientas de *responsive design*, que simula el comportamiento de la interfaz en diferentes tamaños de pantalla, permitiendo verificar que la aplicación mantiene su usabilidad y legibilidad tanto en dispositivos móviles como en pantallas de escritorio.

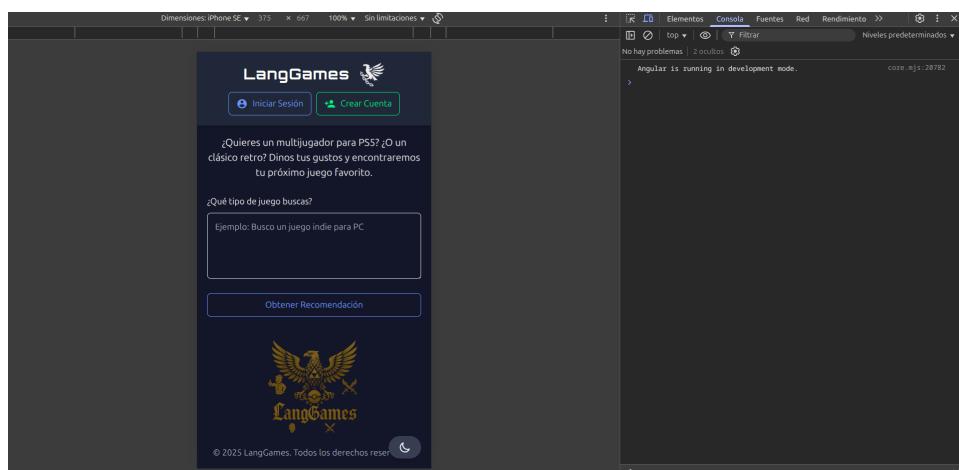


Figura 7.3: Vista de las herramientas de desarrollo del navegador. Estas herramientas permiten detectar errores de Angular y comprobar que el diseño es adaptable a distintos dispositivos.

7.3.1. Pruebas unitarias en el *frontend*

Aunque las pruebas principales del *frontend* se realizaron mediante interacción directa con la aplicación, también se desarrollaron algunas pruebas automáticas para verificar el correcto funcionamiento de ciertos componentes.

Capítulo 7: Pruebas

Se utilizó el sistema de pruebas de Angular con Karma y Jasmine [28], configurando un entorno que permite simular peticiones a la API y comprobar que las funciones responden correctamente. En concreto, se probaron aspectos como la gestión de formularios en el componente de inicio de sesión, la validación de campos, el manejo de errores y la navegación tras un inicio de sesión exitoso.

Estas pruebas demostraron ser útiles para asegurar que los servicios y componentes se comportan como se espera. Sin embargo, se decidió no extender su uso al resto de la interfaz, ya que resultaba más efectivo validar el *frontend* desde el punto de vista del usuario, observando directamente cómo se comporta la aplicación en distintos escenarios.

Capítulo 7: Pruebas

7.4. Pruebas extras

Para asegurar una mejor robustez de la plataforma, se ha realizado una instalación de la aplicación en una máquina virtual de Ubuntu nueva, lo que permitió corregir errores que no se vieron durante el desarrollo por instalar las cosas una por una.

Por ejemplo, la primera vez que se lanza angular se pide al usuario que especifique ciertas preferencias, pero al lanzarse en segundo plano el usuario no podía elegir. Para paliar este problema se realiza un primer lanzamiento de angular en primer plano después de la aplicación.

Además se ha compartido el proyecto con un compañero de informática de la universitat oberta de Catalunya para que pruebe la aplicación y de sus conclusiones.

Capítulo 7: Pruebas

7.5. Conclusión

A lo largo de este capítulo se ha descrito el proceso de pruebas llevado a cabo para verificar el correcto funcionamiento de cada uno de los módulos de la plataforma.

En primer lugar, se realizaron pruebas directas sobre la base de datos mediante `mongosh`, asegurando que las operaciones básicas de inserción, actualización y eliminación funcionaban correctamente. Posteriormente, se probó el *backend* utilizando Swagger, lo que permitió evaluar de forma precisa cada uno de los endpoints expuestos por la API, simulando tanto casos correctos como situaciones de error o fallo del sistema.

Finalmente, el *frontend* fue validado mediante interacción directa con la aplicación y mediante el uso de las herramientas de desarrollo del navegador, lo cual permitió no solo identificar errores de ejecución, sino también comprobar la adaptabilidad del diseño a distintos dispositivos.

Este enfoque integral de pruebas, que abarca desde el nivel más bajo (base de datos) hasta el más alto (interfaz de usuario), ha permitido garantizar una plataforma funcional, robusta y preparada para ofrecer una experiencia de usuario óptima. Además, se ha verificado que los distintos componentes interactúan de forma fluida, cumpliendo con los requisitos del sistema.

Capítulo 8

Conclusiones

Finalmente, en este capítulo debatiremos sobre los resultados obtenidos a lo largo del proyecto. Reflexionaremos sobre el cumplimiento de los objetivos planteados al inicio y evaluaremos la efectividad de las soluciones implementadas.

Además, se analizarán los puntos fuertes y las limitaciones de la plataforma desarrollada, así como las posibles áreas de mejora.

Del mismo modo se considerarán las implicaciones de los resultados obtenidos en el contexto de la industria de los videojuegos.

Por último, se propondrán futuras líneas de trabajo que podrían extenderse a otras áreas, basándose en los resultados y conocimientos adquiridos.

Capítulo 8: Conclusiones

8.1. Conclusiones generales

A lo largo de este proyecto se ha puesto de manifiesto la relevancia actual tanto del sector de los videojuegos como de los modelos de lenguaje natural. La creciente importancia de ambos campos genera nuevas oportunidades para desarrollar aplicaciones que ayuden a gestionar, explorar y conectar los elementos que los componen.

En este contexto, herramientas como **LangChain** y **OpenRouter** han demostrado ser especialmente potentes a la hora de integrar diferentes modelos de lenguaje de manera flexible. Gracias a ellas, es posible aprovechar el potencial de la inteligencia artificial de forma accesible, facilitando el desarrollo de nuevas soluciones o la mejora de sistemas existentes mediante funcionalidades basadas en lenguaje natural.

La plataforma desarrollada, **LangGames**, representa un primer paso hacia una solución que facilite al usuario la elección de un videojuego entre la inmensa oferta actual. La idea es ofrecer un sistema de recomendación sencillo e intuitivo, que funcione a partir de una breve descripción en lenguaje natural, y que además pueda mejorar progresivamente gracias a la información acumulada y al razonamiento de los modelos generativos de lenguajes subyacentes.

En términos generales, se puede afirmar que la mayoría de los objetivos planteados han sido alcanzados. La aplicación es capaz de recomendar videojuegos en base a preferencias expresadas por los usuarios, presenta una interfaz visualmente atractiva y ha demostrado un comportamiento robusto y razonablemente eficiente.

No obstante, durante el desarrollo surgieron ciertos obstáculos que motivaron cambios en el planteamiento inicial. Por ejemplo, no se implementó un sistema completo de usuarios administradores, ya que muchas de sus funcionalidades pudieron gestionarse directamente desde las herramientas de desarrollo. Del mismo modo, las restricciones de uso en OpenRouter, especialmente en cuanto al número de tokens disponibles, limitaron el alcance de las pruebas y la variedad de modelos evaluados.

Otro aspecto a mejorar es el tiempo de respuesta de los modelos, que resulta elevado debido al uso de una API externa y gratuita. Aunque es una limitación comprensible, puede afectar negativamente a la experiencia de usuario. Asimismo, las medidas de seguridad implementadas son limitadas, debido tanto al desconocimiento previo como a problemas técnicos encontrados al intentar añadir capas de seguridad más avanzadas.

Además, la falta de recursos económicos ha impedido realizar un despliegue en la web, lo cual habría sido ideal para validar la plataforma con usuarios reales y analizar su impacto.

Con un presupuesto adecuado, esta aplicación podría evolucionar hasta convertirse en una herramienta esencial para jugadores habituales. La incorporación de modelos más avanzados, con capacidad de búsqueda en internet

Capítulo 8: Conclusiones

precisa y una mejor integración con plataformas de videojuegos como las de Microsoft o Sony, podría situar a LangGames dentro del ecosistema digital de los videojuegos modernos.

Más allá de los videojuegos, este tipo de tecnología tiene un gran potencial de extrapolación a otros medios culturales como el cine, las series, la literatura o la música. Gracias a su flexibilidad y capacidad de comprensión del lenguaje, los modelos de IA seguirán expandiéndose y ganando protagonismo en múltiples ámbitos de la vida cotidiana. Los modelos de lenguaje han llegado para quedarse, y su papel será cada vez más central en el desarrollo de nuevas herramientas y servicios inteligentes.

Capítulo 8: Conclusiones

8.2. Conclusiones alineadas con los objetivos

A lo largo del desarrollo del proyecto, se ha logrado cumplir satisfactoriamente con la mayoría de los objetivos planteados al inicio. A continuación, se presenta un análisis detallado del grado de cumplimiento de cada uno de ellos:

- **Diseñar una interfaz clara, intuitiva y accesible:** Este objetivo se ha cumplido con creces mediante el uso del framework Angular para el desarrollo del frontend. La interfaz no solo es clara y fácil de usar, sino que también es completamente **responsive**, adaptándose a diferentes tamaños de pantalla (ordenadores, tablets, móviles). Además, se ha implementado un **modo claro y oscuro** seleccionable por el usuario, lo que mejora aún más la accesibilidad y experiencia de uso.
- **Mejorar la personalización del sistema:** La aplicación almacena de forma persistente las preferencias y elecciones del usuario, lo que permite ofrecer recomendaciones más precisas a lo largo del tiempo. Además, se ha implementado una conexión con la plataforma **Steam**, lo que amplía la capacidad de personalización al integrar información externa sobre juegos que el usuario ya posee o ha explorado.
- **Implementar y coordinar múltiples modelos de lenguaje:** Gracias a la utilización de LangChain junto con OpenRouter, se ha conseguido integrar diversos modelos de lenguaje de forma flexible y modular. Esto permite generar recomendaciones coherentes a partir de descripciones en lenguaje natural, sacando partido de las fortalezas de cada modelo según el contexto.
- **Incorporar factores adicionales en las recomendaciones:** El sistema puede considerar distintos factores adicionales, como los juegos preferidos, el idioma del usuario o incluso necesidades especiales. Estos datos se pueden introducir explícitamente mediante texto natural, y los modelos son capaces de interpretarlos para ofrecer sugerencias más afinadas.
- **Facilitar la actualización del sistema con nuevas tendencias:** Este objetivo también se ha cumplido gracias a la integración con la API de **RAWG**, que permite acceder a información actualizada sobre videojuegos, incluyendo próximos lanzamientos y novedades del sector. Esto garantiza que el sistema pueda mantenerse relevante y adaptado a la evolución del mercado.
- **Optimizar el rendimiento del sistema:** Aunque el rendimiento general es satisfactorio, el tiempo de respuesta puede resultar algo elevado debido al uso de una API externa gratuita para los modelos

Capítulo 8: Conclusiones

de lenguaje. A pesar de ello, se ha optimizado la arquitectura interna para asegurar fluidez y robustez. Una mejora futura podría consistir en utilizar servidores o APIs dedicadas o modelos alojados localmente.

- **Garantizar la privacidad y seguridad de los datos:** La seguridad se ha tenido en cuenta desde el diseño de la arquitectura. Las contraseñas de los usuarios se almacenan de forma segura mediante **hashing**, y el sistema utiliza **tokens JWT (JSON Web Tokens)** para gestionar sesiones e identificar usuarios de manera segura. Estas medidas, si bien básicas, suponen una buena base para futuros refuerzos de seguridad en un entorno de producción.

En conjunto, puede afirmarse que el proyecto cumple con los objetivos definidos, sentando unas bases sólidas sobre las que continuar construyendo. A pesar de ciertas limitaciones técnicas o presupuestarias, la plataforma desarrollada ha demostrado su viabilidad funcional y su potencial de crecimiento en futuras versiones.

Capítulo 8: Conclusiones

8.3. Alineamiento con los Objetivos de Desarrollo Sostenible (ODS)

Este proyecto se ha alineado con varios de los Objetivos de Desarrollo Sostenible definidos por la ONU [29], destacando los siguientes:

- **ODS 4: Educación de calidad**

Aunque la plataforma tiene un enfoque mayormente lúdico, la investigación y su base tecnológica puede adaptarse para apoyar experiencias educativas gamificadas, así como para recomendar videojuegos con contenido educativo, promoviendo la educación a través de medios digitales accesibles.

- **ODS 5: Igualdad de género**

La plataforma está diseñada para ser completamente inclusiva y accesible independientemente del género del usuario. No presenta ningún sesgo de género en sus recomendaciones ni en la interacción con la interfaz, fomentando así la igualdad de oportunidades en el acceso y disfrute del contenido.

- **ODS 9: Industria, Innovación e Infraestructura**

El desarrollo de esta plataforma de recomendación de videojuegos incorpora tecnologías emergentes como inteligencia artificial (LangChain), bases de datos NoSQL (MongoDB), API modernas (FastAPI) y frameworks de frontend modernos (Angular). Todo ello contribuye a fomentar la innovación tecnológica y la creación de infraestructuras digitales sostenibles y escalables.

- **ODS 10: Reducción de las desigualdades**

Se ha tenido en cuenta la diversidad de perfiles de usuario, incluyendo posibles limitaciones visuales, psicomotrices u otras necesidades específicas que puedan afectar la experiencia de juego. La plataforma está preparada para integrar mecanismos de accesibilidad y adaptar sus recomendaciones en función de las capacidades del usuario, contribuyendo a reducir desigualdades en el acceso al ocio digital.

Capítulo 8: Conclusiones

8.4. Trabajos futuros

El proyecto desarrollado representa un punto de partida sólido para seguir explorando y ampliando las posibilidades de las tecnologías involucradas. A continuación, se presentan algunas líneas de trabajo futuras que permitirían mejorar y extender la plataforma actual:

- **Despliegue en la web y pruebas con usuarios reales:** Uno de los pasos más evidentes sería llevar a cabo un **despliegue completo de la plataforma en la web**, utilizando servicios como Vercel, Firebase o servidores propios. Esto permitiría realizar pruebas con usuarios reales, obtener retroalimentación directa y validar la utilidad del sistema en un entorno práctico.
- **Ampliar el perfilado del usuario:** En futuras versiones, se podría profundizar aún más en la personalización teniendo en cuenta una gama más amplia de características del usuario, como su estado de ánimo, el tiempo disponible para jugar, su historial completo de juegos implementando conexión con otras plataformas de videojuegos o incluso datos demográficos. Esta información permitiría ofrecer recomendaciones mucho más precisas y adaptadas al contexto.
- **Mejorar la seguridad y escalabilidad:** Si bien se han implementado medidas básicas como el uso de JWT y el hash de contraseñas, se podrían integrar soluciones de **mayor robustez en seguridad**, como HTTPS, autenticación multifactor, control de roles más detallado y protección frente a ataques comunes.
- **Integrar nuevos modelos de lenguaje:** La incorporación de modelos más avanzados o especializados permitiría reducir la latencia, mejorar la calidad de las respuestas y adaptar el comportamiento del sistema a diferentes escenarios. También podría estudiarse la posibilidad de entrenar un modelo propio optimizado para el dominio de los videojuegos.
- **Extender la plataforma a otros ámbitos culturales:** El enfoque del sistema no tiene por qué limitarse a los videojuegos. La misma lógica de funcionamiento puede adaptarse fácilmente para ofrecer recomendaciones de **películas, series, libros o música**, aprovechando el poder de los modelos de lenguaje para comprender las preferencias del usuario y explorar amplios catálogos culturales.
- **Uso del proyecto como base para trabajos futuros:** Este Trabajo de Fin de Grado podría servir como **base para proyectos más ambiciosos** en cursos posteriores. Otros alumnos podrían retomarlo, mejorarlo, ampliarlo o incluso integrarlo en contextos educativos,

Capítulo 8: Conclusiones

comerciales o investigativos. La modularidad y documentación del sistema facilitan esta posibilidad.

- **Adquisición o adopción del proyecto por parte de una empresa:** Aunque este trabajo ha sido concebido con fines académicos, la idea desarrollada podría resultar de interés para empresas del sector de los videojuegos. La plataforma presenta un enfoque innovador en la recomendación personalizada basada en lenguaje natural, lo que abre la puerta a una posible **transferencia tecnológica**, ya sea mediante colaboración, adquisición o integración en soluciones comerciales ya existentes.

Estas líneas de trabajo no solo abrirían nuevas posibilidades técnicas y funcionales, sino que también permitirían consolidar la plataforma como una herramienta útil y versátil tanto en el sector del ocio digital como en otros contextos académicos, culturales y tecnológicos.

Capítulo 8: Conclusiones

8.5. Conclusiones personales

A pesar de las dificultades personales que han acompañado el desarrollo de este trabajo, la valoración final es personalmente positiva.

Ha sido muy gratificante comprobar cómo una idea inicial, casi esbozada, ha ido tomando forma paso a paso hasta convertirse en una aplicación funcional. Como persona que ha consumido videojuegos durante toda su vida, ha sido una experiencia transformadora pasar del rol de usuario al de creador, desarrollando una utilidad que potencialmente podría ser utilizada por miles de personas con los mismos intereses.

Este trabajo también evidencia de forma clara el progreso técnico alcanzado a lo largo del Grado. Se ha pasado de realizar desarrollos básicos a diseñar y construir una aplicación completa desde cero, utilizando tecnologías modernas y no abordadas en profundidad durante el plan de estudios, al menos en la mención de Computadores y Sistemas Inteligentes, como *FastAPI* para el backend, *LangChain* para la integración de modelos de lenguaje natural, *MongoDB* como base de datos NoSQL y *Angular* para el desarrollo del frontend.

Asimismo, el proceso ha potenciado significativamente las capacidades de investigación tecnológica y aprendizaje autodidacta. Se ha evolucionado desde un conocimiento superficial del concepto de API hasta la implementación de una propia, plenamente funcional, conectada con una base de datos y con una interfaz web moderna y dinámica. En este contexto, se ha adquirido experiencia práctica y avanzada en el uso de *Python*, *MongoDB* y *Angular*, consolidando una base técnica sólida que va más allá de los contenidos tratados formalmente en el Grado.

Aunque el proyecto pueda parecer sencillo o limitado desde una perspectiva externa, lo cierto es que implica un esfuerzo considerable, muchas horas de dedicación y una gran cantidad de conocimientos adquiridos y aplicados. Más allá del resultado final, lo que realmente queda es una sensación de logro y una base sólida sobre la que seguir construyendo en el futuro.

Bibliografía

- [1] Jorge Arias. “La industria de los videojuegos ya genera más ingresos que la música y el cine juntos”. En: *THE OBJECTIVE* (2023).
- [2] David Williamson et al. “Video games and the future of learning”. En: *Phi Delta Kappan* 87.2 (2005), págs. 104-111.
- [3] Aaron Van Dorn. “A profound portrait of a life online”. En: *The Lancet Child & Adolescent Health* 9.1 (2025), págs. 14-15.
- [4] LangChain Inc. *LangChain*. Último acceso: marzo de 2025. 2025. URL: <https://www.langchain.com/>.
- [5] Vasilios Mavroudis. “LangChain v0. 3”. En: *Preprints* (2024).
- [6] LangChain Inc. *LangChain Básico*. Último acceso: marzo de 2025. 2025. URL: <https://www.langchain.com/langchain>.
- [7] LangChain Inc. *RAG*. Último acceso: marzo de 2025. 2025. URL: <https://www.langchain.com/retrieval>.
- [8] LangChain Inc. *LangChain Agentes*. Último acceso: marzo de 2025. 2025. URL: <https://www.langchain.com/agents>.
- [9] LangChain Inc. *LangChain Evaluación*. Último acceso: marzo de 2025. 2025. URL: <https://www.langchain.com/evaluation>.
- [10] LangChain Inc. *LangChain LangGraph*. Último acceso: marzo de 2025. 2025. URL: <https://www.langchain.com/langgraph>.
- [11] LangChain Inc. *LangChain LangSmith*. Último acceso: marzo de 2025. 2025. URL: <https://www.langchain.com/langsmith>.
- [12] Ye Zhang et al. “Unlocking personalized anime recommendations: Lang-chain and llm at the forefront”. En: *Journal of Industrial Engineering and Applied Science* 2.2 (2024), págs. 46-53.
- [13] Johans Quintero. *Sistema de recomendación con agente LangChain*. 2024. URL: <https://github.com/johansquintero/recommender-agent>.
- [14] Daniel Yélamos Pérez et al. “Recomendación de videojuegos basado en análisis semántico y minería de opinión”. B.S. thesis. 2016.

BIBLIOGRAFÍA

- [15] Alejandro Roldán Soblechero. “Aplicación de recomendación de videojuegos”. En: (2024).
- [16] Equipo de Talent.com. *Sueldo medio informático España*. Último acceso: marzo de 2025. 2025. URL: <https://es.talent.com/salary?job=ingeniero+inform%C3%A1tico#:~:text=El%20salario%20ingeniero%20inform%C3%A1tico%20promedio,%20%E2%82%AC%2036.000%20al%20a%C3%B3n..>
- [17] Equipo de CosteEnergia.es. *Coste del kwh 2025*. Último acceso: marzo de 2025. 2025. URL: <https://www.costeenergia.es/historico/pvpc-anual>.
- [18] Leon R Yankwich. “What Is Fair Use?” En: *The University of Chicago Law Review* 22.1 (1954), págs. 203-215.
- [19] Richard Petersen. *Ubuntu 24.04 LTS Server: Administration and Reference*. surfing turtle press, 2025.
- [20] Alexey Zagalsky et al. “The emergence of github as a collaborative platform for education”. En: *Proceedings of the 18th ACM conference on computer supported cooperative work & social computing*. 2015, págs. 1906-1917.
- [21] Alexandru Boicea, Florin Radulescu y Laura Ioana Agapin. “MongoDB vs Oracle–database comparison”. En: *2012 third international conference on emerging intelligent data and web technologies*. IEEE. 2012, págs. 330-335.
- [22] Cornelia Györödi et al. “A comparative study: MongoDB vs. MySQL”. En: *2015 13th international conference on engineering of modern electric systems (EMES)*. IEEE. 2015, págs. 1-6.
- [23] Uday Patkar et al. “Python for web development”. En: *International Journal of Computer Science and Mobile Computing* 11.4 (2022), pág. 36.
- [24] Bill Lubanovic. *FastAPI*. .O'Reilly Media, Inc.”, 2023.
- [25] Malhar Lathkar. “Getting started with FastAPI”. En: *High-Performance Web Apps with FastAPI: The Asynchronous Web Framework Based on Modern Python*. Springer, 2023, págs. 29-64.
- [26] Michael Jones, Brian Campbell y Chuck Mortimore. *Json web token (jwt) profile for oauth 2.0 client authentication and authorization grants*. Inf. téc. 2015.
- [27] Simone Chiaretta. *Front-end Development with ASP. NET Core, Angular, and Bootstrap*. John Wiley & Sons, 2018.
- [28] Brad Green y Shyam Seshadri. *AngularJS*. .O'Reilly Media, Inc.”, 2013.

BIBLIOGRAFÍA

- [29] Organización de las Naciones Unidas. *Objetivos de Desarrollo Sostenible*. Sitio oficial de las Naciones Unidas sobre los ODS, último acceso: junio de 2025. 2015. URL: <https://www.un.org/sustainabledevelopment/es/objetivos-de-desarrollo-sostenible/>.

