UNIR FP
Informática y Comunicaciones

Técnico Superior en DAM – DAW - ASIR

Título del Trabajo Fin de Estudios

Trabajo fin de estudio presentado por:	Ivan Nuñez Rodriguez Ivan Serrano Nuñez
Tipo de trabajo:	Catalogo VideoJuegos Web
Tutor/a:	Damian Sualdea Soy
Fecha:	4/4/2025

Resumen

El presente proyecto tiene como objetivo el desarrollo de un sistema inteligente de gestión de videojuegos que permita a los usuarios explorar títulos, calificarlos, marcar su estado de juego y recibir recomendaciones personalizadas mediante Inteligencia Artificial. Se ha implementado un backend robusto, estructurado en capas, utilizando PostgreSQL como sistema de base de datos y desplegado en un servidor remoto mediante Docker Compose. La aplicación se alimenta de datos reales extraídos desde la API pública de IGDB. Actualmente se ha completado la integración con esta API, así como la configuración de Firebase y el servidor VPS. El frontend está en proceso de desarrollo con Angular y se prevé su finalización en próximas fases.

Palabras clave: videojuegos, recomendaciones, IA, IGDB, Firebase

Abstract

This project aims to develop an intelligent video game management system, allowing users to explore, rate, organize, and receive personalized game recommendations powered by Artificial Intelligence. A structured backend has been implemented using PostgreSQL as the database system and deployed on a remote server via Docker Compose. The system connects with IGDB's public API to obtain real game data. Firebase has been set up for future integration. The frontend, developed with Angular, is currently under construction and expected to be completed in upcoming stages.

Keywords: videogames, recommendations, AI, IGDB, Firebase

Índice de contenidos

Índice de figuras

Índice de tablas

1. Introducción

1.1. Justificación

En la actualidad, el mundo del videojuego se ha convertido en una industria en constante crecimiento, con una oferta inmensa y diversa de títulos que dificultan al usuario tomar decisiones informadas sobre qué jugar. A partir de esta necesidad nace la idea de desarrollar una plataforma inteligente que ayude a los usuarios a gestionar su colección de videojuegos, descubrir nuevos títulos y recibir recomendaciones personalizadas.

Este Trabajo Fin de Ciclo se enmarca dentro del ámbito del desarrollo de aplicaciones multiplataforma, integrando conocimientos de programación, bases de datos, desarrollo web y tecnologías emergentes como la inteligencia artificial y el uso de APIs externas. La aplicación propuesta combina aspectos técnicos avanzados con una interfaz intuitiva y funcional para resolver un problema real y actual.

1.2. Objetivos

Objetivo general
Diseñar e implementar un sistema inteligente de gestión y recomendación de videojuegos que permita a los usuarios interactuar con una plataforma conectada a fuentes de datos externas y dotada de funcionalidades personalizadas mediante IA.
Objetivos específicos

- Desarrollar una base de datos relacional en PostgreSQL para gestionar usuarios, videojuegos y sus relaciones.
- Diseñar una API RESTful en ASP.NET Core para gestionar la lógica de negocio.
- Integrar la API de IGDB para obtener información detallada de los videojuegos.
- Conectar el backend con Weaviate para ofrecer recomendaciones personalizadas mediante IA.
- Crear un frontend responsive en Angular (en desarrollo) para la interacción con los usuarios.
- Implementar un entorno de despliegue con Docker y alojamiento en Oracle Cloud VPS.
- Aplicar buenas prácticas de desarrollo como uso de DTOs, seguridad con JWT y separación por capas.
-

2. Módulos formativos aplicados en el trabajo

Este Trabajo Fin de Ciclo ha permitido aplicar de forma transversal los conocimientos adquiridos en los diferentes módulos del ciclo formativo de **Desarrollo de Aplicaciones Multiplataforma (DAM)**. A continuación, se detallan los módulos implicados y su relación con el proyecto:

- **Programación:** Desarrollo del backend utilizando C# y ASP.NET Core. Se han aplicado principios de programación orientada a objetos, estructuración en capas y uso de DTOs.
- **Bases de Datos:** Diseño e implementación de la base de datos en PostgreSQL. Uso de claves primarias, foráneas, índices, y control de integridad referencial.
- **Lenguajes de marcas y sistemas de gestión de información:** Uso de JSON y XML en la comunicación con APIs y en la estructura de datos para respuestas RESTful.
- **Entornos de desarrollo:** Utilización de Visual Studio Code, Git y Docker como herramientas principales durante la implementación.
- **Desarrollo de interfaces:** (En curso) Aplicación Angular como interfaz de usuario con vistas para exploración, detalles y gestión de videojuegos.
- **Acceso a datos:** Uso de ADO.NET para la gestión de datos entre backend y base de datos PostgreSQL.
- **Sistemas informáticos y redes:** Configuración del servidor Oracle Cloud con Docker Compose para el despliegue de servicios.
- **Inglés técnico:** Interpretación y uso de documentación oficial en inglés, especialmente para la integración con la API de IGDB y Weaviate.
- **Formación y orientación laboral / Empresa e iniciativa emprendedora:** Aplicación de planificación, organización y presentación de un proyecto real, incluyendo hoja de ruta y entregas intermedias.

3. Herramientas y lenguajes utilizados

Durante el desarrollo del proyecto se han utilizado diversas herramientas, tecnologías y lenguajes de programación. A continuación, se detallan los principales:

Tabla 1 Descripcion lenguajes utilizado

Herramienta / Lenguaje	Descripción
C# / ASP.NET Core	Lenguaje y framework utilizados para implementar la lógica del backend y la API RESTful. Permite el desarrollo de aplicaciones modulares y escalables.
PostgreSQL	Sistema de gestión de bases de datos relacional elegido por su estabilidad, rendimiento y compatibilidad con herramientas modernas.
Angular	Framework frontend basado en TypeScript. Se está utilizando para construir la interfaz de usuario responsive.
Docker / Docker Compose	Herramienta de virtualización ligera para el empaquetado y despliegue del proyecto en contenedores. Facilita la portabilidad y escalabilidad.
Oracle Cloud VPS	Plataforma donde se ha configurado el entorno de producción del proyecto, desplegando backend y base de datos.
Firebase	Plataforma de Google utilizada para el hosting futuro del frontend y posibles integraciones como autenticación o Firestore.
IGDB API	Fuente de datos externa para obtener información actualizada sobre videojuegos (títulos, imágenes, fechas, etc.).
Weaviate	Motor de búsqueda vectorial con IA utilizado para ofrecer recomendaciones personalizadas basadas en aprendizaje automático.
Git / GitHub	Sistema de control de versiones y plataforma colaborativa para el control y seguimiento del código fuente.
Visual Studio Code	Editor de código utilizado para el desarrollo del backend y la configuración del proyecto.
Figma / Xcalidraw	Herramientas de prototipado utilizadas para diseñar wireframes y visualizar la interfaz gráfica de la aplicación.

4. Metodologías utilizadas

El desarrollo del proyecto se ha llevado a cabo aplicando metodologías ágiles, especialmente **Scrum**, adaptada al contexto educativo y de trabajo en equipo reducido.

Enfoque aplicado:

- **Planificación iterativa:** Se establecieron entregas por fases semanales, priorizando la arquitectura, la funcionalidad y la integración progresiva.
- **Trello como tablero Kanban:** Se utilizó para organizar tareas por columnas (pendiente, en progreso, finalizado), facilitando la visualización del flujo de trabajo.
- **Reuniones periódicas entre miembros del equipo:** Se realizaron sesiones de seguimiento para revisar el estado de avance, identificar bloqueos y redefinir prioridades.
- **Documentación viva**: A través de Google Docs se registraron entregas intermedias, avances técnicos y coordinación con el tutor.

Justificación de la elección
Scrum y Kanban permiten una mejor adaptación al cambio y una entrega incremental del producto. Esta metodología ha resultado especialmente útil al tratarse de un proyecto con múltiples tecnologías, integración de servicios externos, y despliegue en entornos reales.

5. Componentes del equipo y aportaciones realizadas por cada alumno

El equipo está formado por dos integrantes, quienes han trabajado de forma colaborativa en todas las fases del proyecto, dividiendo las tareas según fortalezas personales y áreas de conocimiento.

Integrantes del equipo:

- **Iván Núñez Rodríguez**
- **Iván Serrano Núñez**

Aportaciones individuales:

Iván Núñez Rodríguez

- Liderazgo en la estructuración del backend en ASP.NET Core.
- Diseño del modelo relacional en PostgreSQL.
- Integración de la API de IGDB.
- Conexión con Weaviate para el sistema de recomendaciones.
- Redacción y coordinación documental en Google Docs.

Iván Serrano Núñez

- Desarrollo y diseño de la interfaz gráfica (frontend en Angular, en curso).
- Creación de prototipos y wireframes en Figma/Xcalidraw.
- Configuración del entorno de despliegue con Docker y Oracle Cloud VPS.
- Organización del flujo de trabajo mediante Trello.
- Preparación de entregas intermedias y diseño visual de la memoria.
- Participación activa en decisiones técnicas y pruebas funcionales.

Ambos integrantes han colaborado en las reuniones con el tutor, el desarrollo de funcionalidades clave y la documentación del proyecto, compartiendo responsabilidades técnicas y organizativas.

5.1. Estudio de mercado

El auge del sector de los videojuegos ha dado lugar a una gran variedad de plataformas para descubrir, calificar y gestionar juegos. Sin embargo, pocas de ellas combinan en un mismo entorno la posibilidad de **gestionar una colección personal**, **recibir recomendaciones personalizadas mediante IA**, y **consultar información oficial de los juegos desde fuentes como IGDB**.

5.1.1. Tabla comparativa de aplicaciones actualmente en el mercado

Tabla 2 Comparativa de aplicaciones actualmente en el mercado

Característica	IGDB	Steam	Backloggery	SmartGameCatalog (propuesta)
Información de juegos	✅ Sí	✅ Sí	❌ No	✅ Sí (vía IGDB API)
Calificación de juegos	❌ No	✅ Sí	✅ Sí	✅ Sí
Gestión de colección	❌ No	✅ Limitada	✅ Sí	✅ Sí (con estados y favoritos)
Recomendaciones IA	❌ No	❌ No	❌ No	✅ Sí (con Weaviate)
Interfaz personalizada	❌ No	✅ Sí (estática)	❌ No	✅ Sí (Angular)
Integración multiplataforma	✅ Parcial	✅ Parcial	❌ No	✅ Sí

5.1.2. Análisis DAFO de nuestra solución

Ilustración 1 Análisis DAFO

5.1.3. Subapartado 1.2

Tras el análisis de la competencia y del mercado, se concluye que la mayoría de las plataformas actuales se centran únicamente en la visualización o gestión parcial de videojuegos, dejando de lado una experiencia más personalizada e inteligente.

La solución propuesta destaca por integrar las siguientes ventajas clave:

- **Recomendaciones personalizadas con IA**, adaptadas al historial del usuario y puntuaciones anteriores.
- **Gestión completa de la colección personal**, con estados definidos y favoritos.
- **Visualización enriquecida** de los videojuegos gracias a la integración con IGDB.
- **Backend escalable y preparado para el despliegue en entornos reales**, usando tecnologías modernas como Docker, PostgreSQL y Oracle Cloud.
- **Interfaz en desarrollo centrada en la experiencia del usuario**, con diseño responsive y modular en Angular.

Esta propuesta busca ir más allá del simple listado de juegos, generando un espacio personal donde cada jugador pueda descubrir, organizar y valorar su experiencia lúdica de forma intuitiva y potente.

5.2. Modelo de datos

El modelo de datos se ha diseñado siguiendo una estructura relacional, con el objetivo de garantizar la integridad, escalabilidad y eficiencia del sistema. Se ha optado por PostgreSQL como sistema gestor de base de datos, utilizando claves primarias, foráneas e índices para optimizar las consultas y relaciones.

El modelo contempla entidades clave como usuarios, juegos, calificaciones, estados, favoritos y recomendaciones, lo que permite una gestión completa y personalizada de la experiencia del usuario dentro de la plataforma.

Entidades principales

- **Users**: Información personal, credenciales y rol del usuario.
- **Games**: Información de los videojuegos obtenida a través de la API de IGDB.
- **Ratings**: Calificaciones y reseñas asignadas por los usuarios.
- **GameStatuses**: Estado de cada juego según el usuario (Wishlist, Owned, Playing, Completed, Abandoned).
- **Favorites**: Juegos marcados como favoritos por el usuario.
- **Recommendations**: Sugerencias generadas mediante IA basadas en las preferencias del usuario.

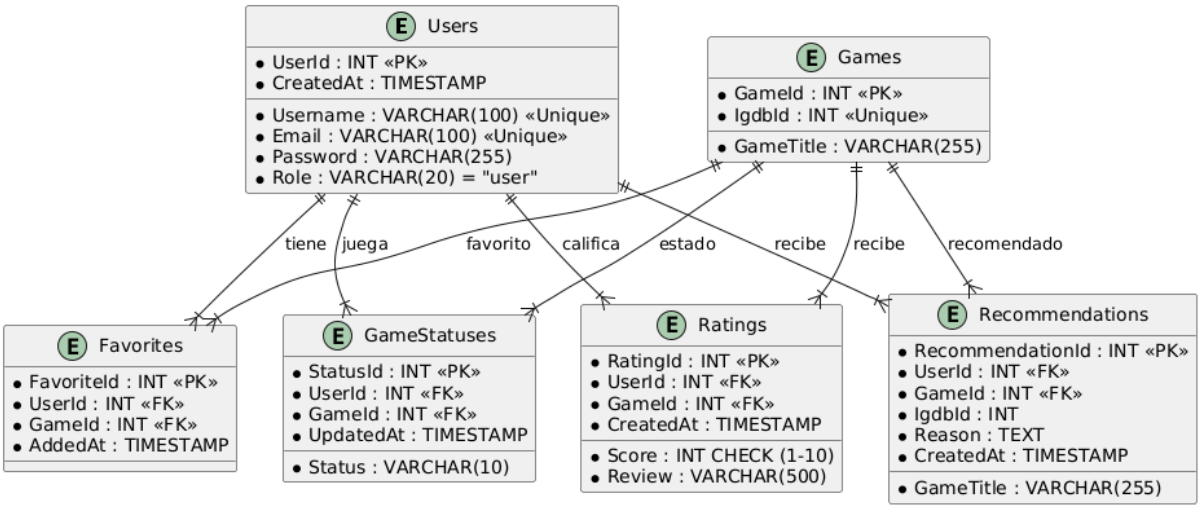

Ilustración 2 Diagrama E/R

5.3. Diagramas UML

Los diagramas UML permiten representar visualmente la estructura y comportamiento del sistema, facilitando su comprensión y análisis. En este proyecto se han elaborado dos tipos principales: **diagrama de clases** y **diagramas de casos de uso**.

5.3.1. Diagrama de clases

Clases principales:

- User: almacena los datos del usuario, como nombre, email, rol, etc.
- Game: representa un videojuego con su título y código de IGDB.
- Rating: calificación y reseña del usuario sobre un juego.
- GameStatus: estado de un juego (jugando, completado, abandonado...).
- Favorite: asociación entre usuario y juegos favoritos.
- Recommendation: recomendaciones generadas con información de título y motivo.

Relaciones destacadas:

- Un User puede tener muchas Ratings, Favorites, GameStatuses y Recommendations.
- Un Game puede estar relacionado con múltiples usuarios en diferentes contextos

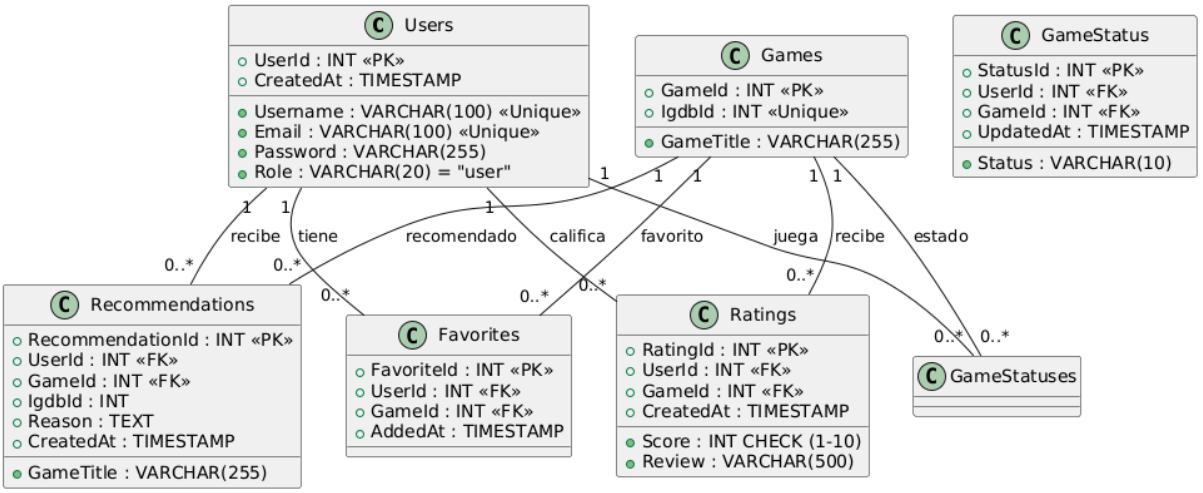

Ilustración 3 Diagrama de clases

5.3.2. Clasificación de usuarios

El sistema diferencia usuarios en tres roles principales:

- **Administrador:** con permisos de mantenimiento, auditoría y gestión general del sistema.
- **Usuario estándar:** puede calificar, guardar estados, recibir recomendaciones y gestionar su perfil.
- **Desarrollador/técnico:** con acceso a funciones internas para pruebas, conexión de IA, y gestión de datos.

Esta segmentación permitirá aplicar control de acceso mediante anotaciones (@PreAuthorize) en el backend y una interfaz personalizada para cada tipo de usuario.

Se ha utilizado el enfoque **Persona UX** para definir arquetipos de usuario, facilitando el diseño del frontend y la lógica de recomendación personalizada.

5.3.3. Caso de uso 01

Actor: Usuario registrado

Descripción: El usuario busca un juego en la plataforma y asigna una puntuación (1 a 10) junto con una reseña opcional.

Flujo básico:

- El usuario accede a su cuenta.
- Busca un juego usando filtros o búsqueda libre.
- Selecciona el juego y accede a su ficha.
- Introduce una puntuación y un comentario.
- La información se guarda en la base de datos y queda disponible para futuras recomendaciones.

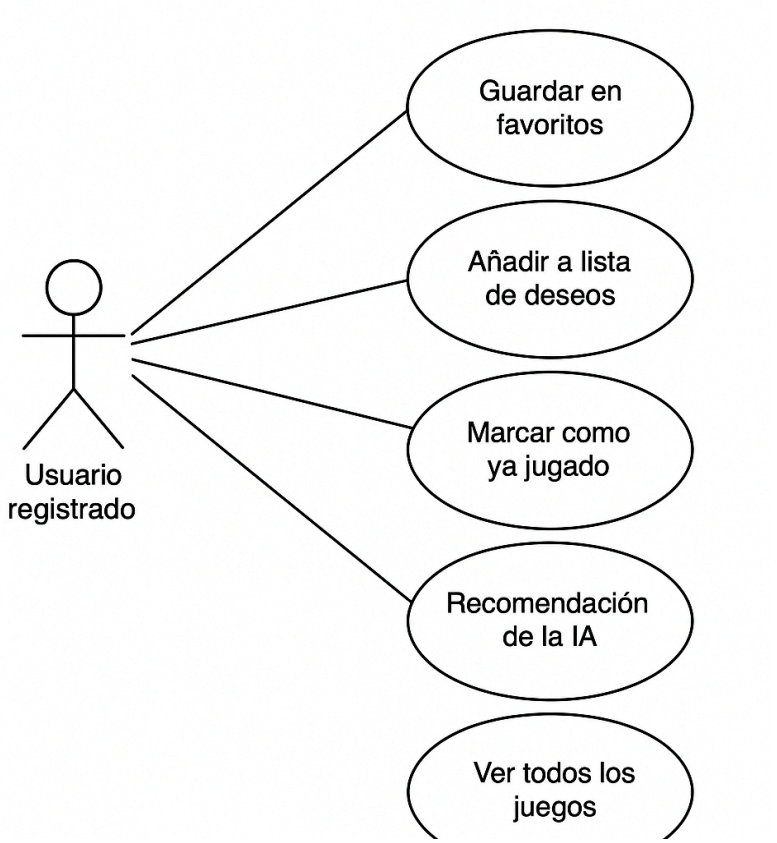

Ilustración 4 Caso de uso

5.4. Diseño de interfaces

5.4.1. Wireframes

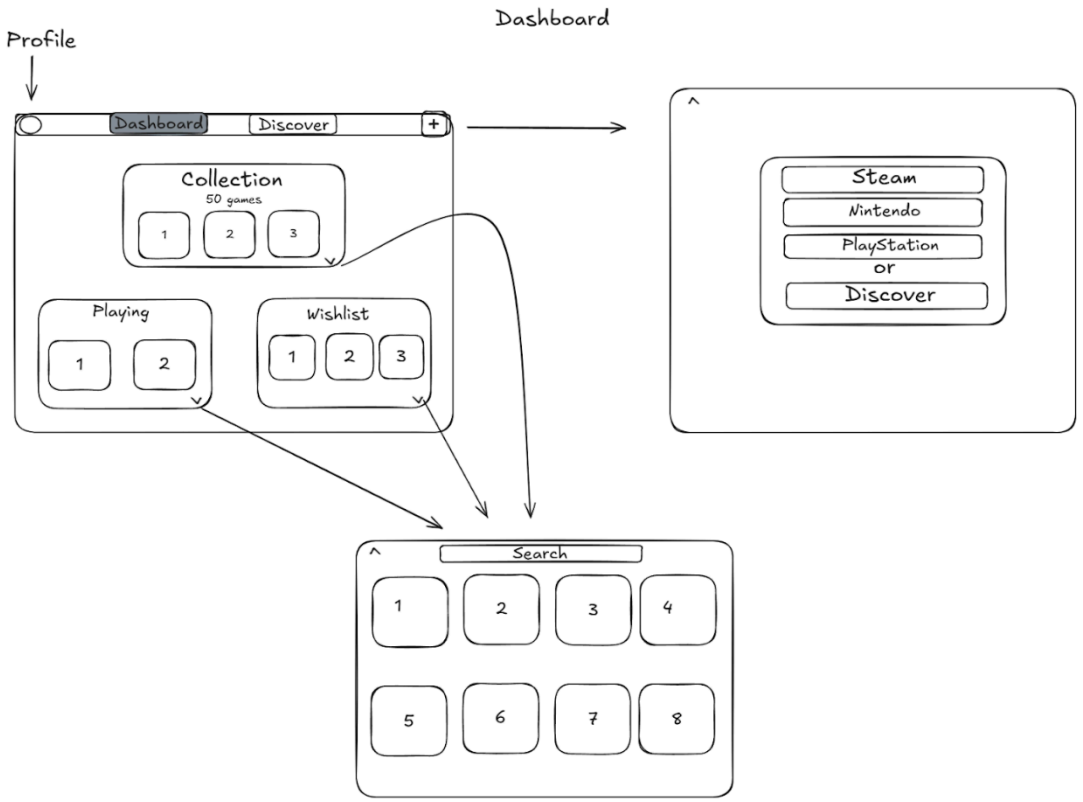

Ilustración 5 Wireframes pantalla principal

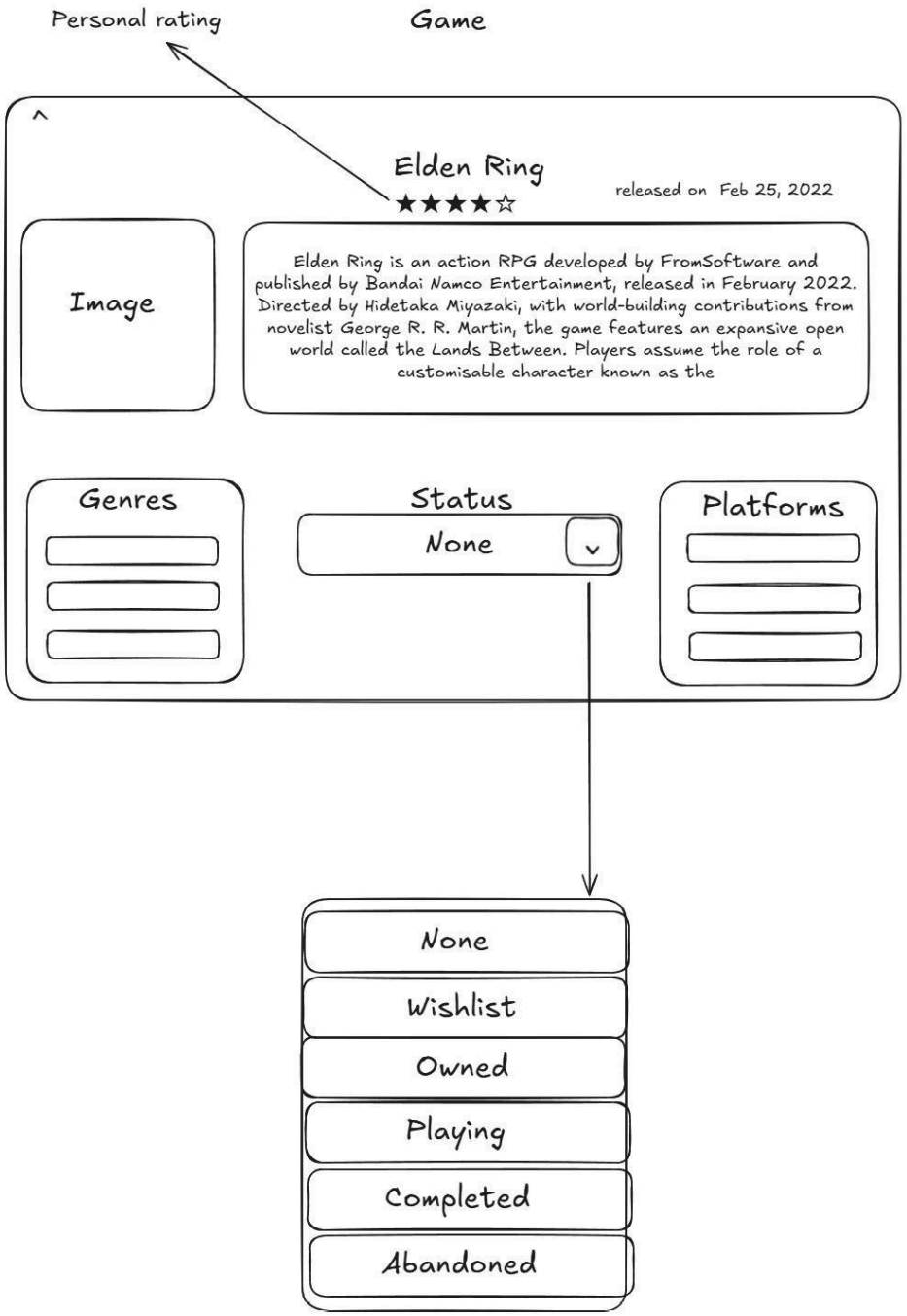

Ilustración 6 Wireframes vista detalles

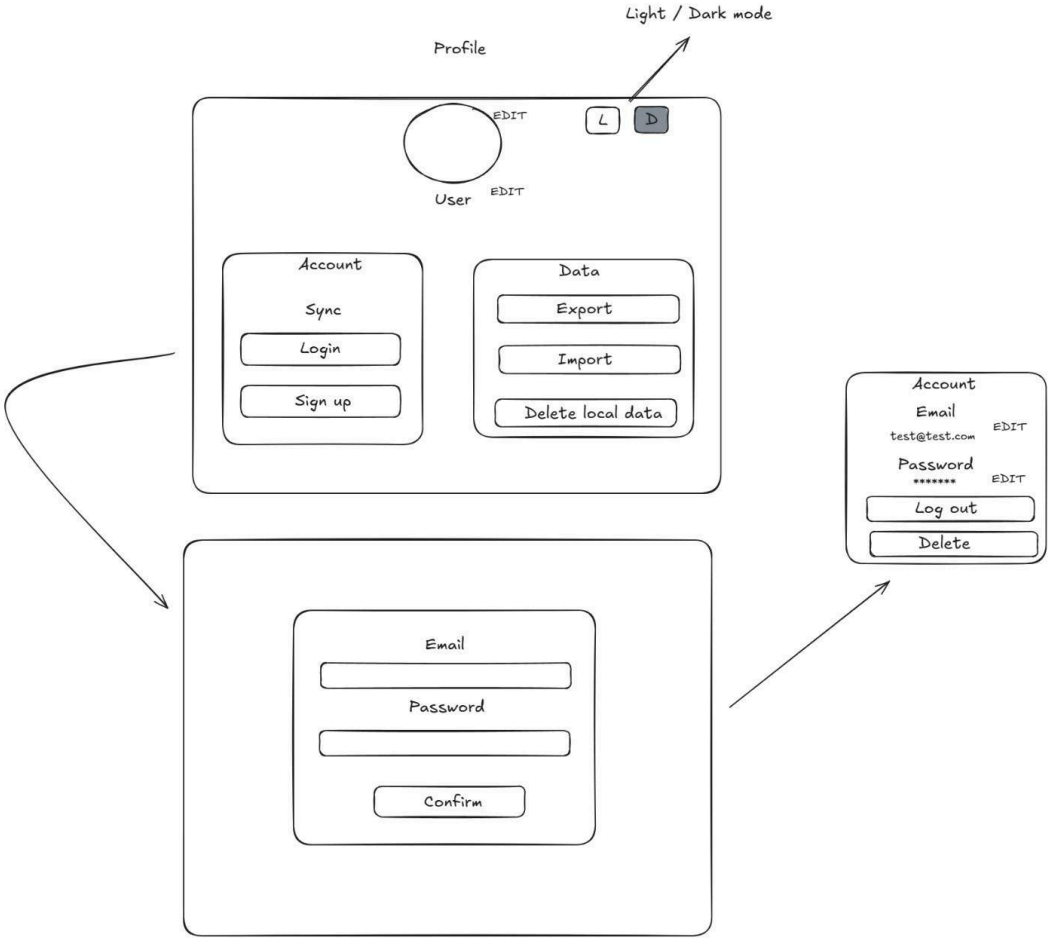

Ilustración 7 Wireframes vista perfil de usuario

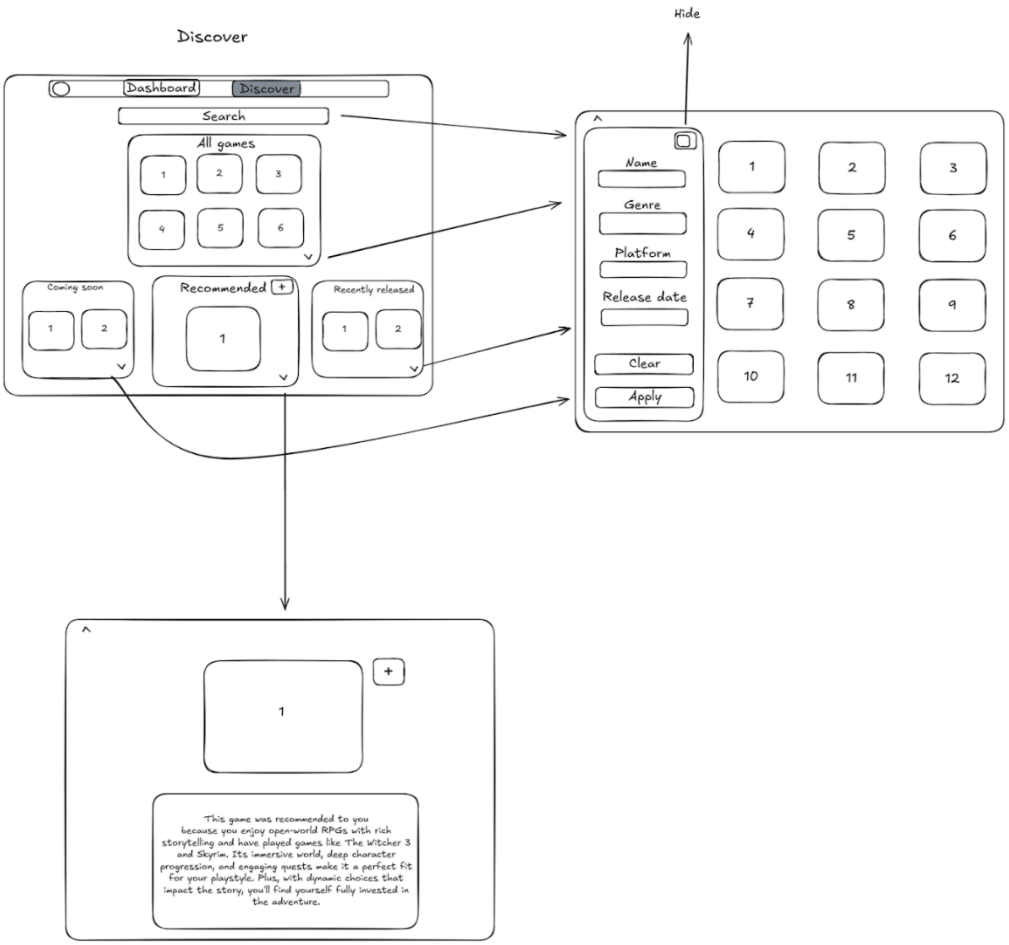

Ilustración 8 Wireframes vista discover

5.4.2. Prototipo de interfaz de alta definición

Ilustración 6 Prototipo de interfaz de alta definición

5.4.3. Paleta de colores

Explicación de la elección de colores

Ilustración 7 Paleta de colores

5.4.4. Logotipo

Explicación de la elección de logotipo

Ilustración 8 Logotipo en positivo

Logotipo en negativo

Ilustración 9 Logotipo en negativo

5.5. Planificación temporal y trabajo en equipo

La correcta organización del tiempo y la asignación de tareas han sido elementos clave para el avance del proyecto. Se ha trabajado con herramientas de planificación como Excel y Trello para mantener el control del progreso y permitir una visualización clara del estado de cada fase.

5.5.1. Presupuesto temporal de tareas

A continuación, se presenta una tabla resumen del reparto de tareas estimado y su duración aproximada:

Tabla 3 Presupuesto temporal de tareas

Tareas principales	Subtareas	Duración estimada (h)	Responsable(s)
Planificación del Proyecto	Definir objetivos, recursos y fases del proyecto	8	Ambos integrantes
Diseño del Backend	Estructura de carpetas, DTOs, API, seguridad JWT	30	Iván Núñez
Base de Datos	Diseño DER, script SQL, conexión con backend	20	Iván Núñez
Integración de IGDB	Conexión API, pruebas, modelado de datos	12	Iván Núñez
Diseño de Interfaces	Wireframes, UI, usabilidad	16	Iván Serrano
Frontend Angular	Componentes, vistas, integración API	32	Iván Serrano (en curso)
Sistema de Recomendación	Configuración de Weaviate, integración IA	15	Iván Núñez
Despliegue y Docker	Configuración de contenedores y VPS	10	Iván Núñez
Documentación	Redacción de memoria, presentaciones y entregas	18	Ambos integrantes

Este presupuesto ha sido útil como punto de partida para medir el rendimiento real y ajustar la carga de trabajo de forma dinámica.

5.5.2. Organización de tareas y tiempos finales

Para la organización diaria del proyecto se ha utilizado Trello con el siguiente esquema:

- **Pendiente:** tareas por iniciar o en espera de revisión.
- **En progreso:** tareas activas durante la semana.
- **Finalizado:** tareas completadas y revisadas.

Además, se generó un diagrama de Gantt (ver Anexo A) a partir del documento diagrama_gantt_horas.xlsx que refleja visualmente la evolución del proyecto desde su inicio hasta la actualidad, incluyendo:

- Fase de análisis y diseño.
- Fase de desarrollo backend.
- Fase de integración con IA y despliegue.
- Fase actual: desarrollo de frontend e integración.

5.5.3. Trabajo en equipo

El proyecto ha sido desarrollado en colaboración entre los dos integrantes, compartiendo tanto tareas técnicas como organizativas. La coordinación ha sido constante mediante reuniones semanales y el uso compartido de documentación en Google Docs y GitHub.

También se han mantenido registros de aportaciones al repositorio y distribución de código mediante Git, lo cual garantiza la trazabilidad del trabajo individual.

6. Conclusiones

6.1. Análisis de desviaciones temporales y de tareas

Durante el desarrollo del proyecto se han producido algunas desviaciones respecto al presupuesto inicial, principalmente en el área de frontend. El diseño y desarrollo de la interfaz gráfica ha requerido más tiempo del previsto inicialmente debido a la integración progresiva de componentes dinámicos y la adaptación a criterios de usabilidad.

Área	Tiempo estimado (h)	Tiempo real (h)	Desviación	Comentario
Backend y base de datos	60	58	-2	Dentro de lo previsto
Recomendaciones IA	15	18	+3	Ajustes en integración con Weaviate
Diseño de interfaces	16	22	+6	Cambios visuales y validación con usuarios
Frontend Angular	32	En curso	—	En proceso
Documentación y entregas	18	20	+2	Más revisión de formato y estructura

6.2. Conclusiones generales del proyecto

6.2.1. Evaluación global del proyecto.

El proyecto ha cumplido con los objetivos principales establecidos en el anteproyecto. Se ha logrado construir una base sólida, integrando tecnologías modernas y planteando un producto funcional con alto potencial de ampliación. Las integraciones con IGDB y Weaviate aportan un valor añadido significativo respecto a otros sistemas.

6.2.2. Reflexión sobre el proceso de aprendizaje y desarrollo.

El trabajo en equipo, la planificación y el contacto continuo con tecnologías reales han sido fundamentales para el éxito del proyecto. Hemos reforzado competencias técnicas clave (API REST, JWT, despliegue en la nube, IA) y también habilidades blandas como la coordinación, la documentación y la presentación de resultados.

6.2.3. Recomendaciones para futuros proyectos similares.

- Planificar tiempos extra para diseño de interfaz y pruebas con usuarios.
- Documentar desde el primer día, tanto a nivel técnico como visual.

- Priorizar la modularidad para facilitar ampliaciones futuras.
- Realizar validaciones tempranas con posibles usuarios para mejorar la UX.

6.3. Limitaciones y prospectiva

6.3.1. Posibles mejoras y ampliaciones del proyecto.

- Implementación de filtros y búsqueda avanzada por género, año, plataforma, etc.
- Añadir funcionalidades sociales como compartir listas de juegos o rankings personales.
- Guardado en la nube del historial de actividad.

6.3.2. Nuevas líneas de investigación o desarrollo que podrían derivarse del proyecto.

- Uso de aprendizaje automático para personalizar recomendaciones más allá de las puntuaciones.
- Implementación de sistemas de reputación entre usuarios (gamificación).
- Aplicación móvil complementaria.

6.3.3. Sugerencias para la implementación en entornos reales.

El sistema puede integrarse fácilmente como extensión de tiendas de videojuegos, comunidades gamers o plataformas educativas, adaptando su lógica de recomendaciones a otros dominios (cine, libros, música…).

7. Referencias bibliográficas

PostgreSQL Global Development Group. (2024). *PostgreSQL Documentation*.

Recuperado de https://www.postgresql.org/docs/

Microsoft. (2024). *ASP.NET Core Documentation*.

Recuperado de https://learn.microsoft.com/en-us/aspnet/core/

Docker. (2024). *Documentación oficial de Docker y Docker Compose*.

Recuperado de https://docs.docker.com/

IGDB. (2024). *API Reference*.

Recuperado de https://api-docs.igdb.com/

Weaviate. (2024). *Weaviate: Vector Database*.

Recuperado de https://weaviate.io/

Firebase. (2024). *Firebase Documentation*.

Recuperado de https://firebase.google.com/docs

Angular. (2024). *Angular Official Guide*.

Recuperado de https://angular.io/docs/

Anexo A. Diagramas de GANTT

Anexo B. Código fuente de la solución y pruebas

ANEXO C. MANUAL DE INSTALACIÓN - DESPLIEGUE

8. Requisitos previos

Antes de iniciar la instalación, asegúrese de contar con los siguientes elementos:

- **Docker Desktop (Windows)**
  Descargar Docker Desktop
- **.NET SDK 9.0.202 (x64)**
  Descargar SDK .NET
- **Node.js v23.10.0**
  Descargar Node.js

---

9. IntelliSense (opcional)

Para evitar problemas con autocompletado e instalaciones en el entorno de desarrollo:

```
npm install --ignore-scripts
```

Añadir un archivo .dockerignore para excluir carpetas innecesarias del contenedor:

```
node_modules
dist
```

---

10. Crear un nuevo backend (referencia inicial)

Desde un contenedor de .NET SDK:

```
docker run --rm -v ${PWD}/app:/app -w /app mcr.microsoft.com/dotnet/sdk:9.0 dotnet new webapi
```

Crear los archivos base:

```
touch docker-compose.yml
touch Dockerfile
```

---

11. Ejecución

Para levantar todos los servicios con Docker Compose:

```
docker-compose up --build
```

---

12. Accesos por defecto

- **API (Backend ASP.NET):**
  http://localhost:5000
- **Frontend (a través de Nginx):**
  http://localhost:80
- **PgAdmin (gestor de base de datos):**
  http://localhost:5050
  Usuario: admin@example.com
  Contraseña: admin

Cadena de conexión:

```
Host=db;Port=5432;Database=gametracker;Username=postgres;Password=pass
```

- **Weaviate (sistema de recomendaciones por IA):**
  http://localhost:8080/v1

Anexo D. Documentación de la API

En caso de que se implemente una API

Anexo E. Otros anexos de interés

Puedes crear los adjuntos que consideres necesarios – de acuerdo con el tutor de proyecto – según la necesidad y naturaleza del proyecto.