



IES Maestre de Calatrava
Paseo de la Universidad,1
13005-Ciudad Real

F.P. INFORMÁTICA

**CURSO 2024-25
Marzo - Junio
Convocatoria: Ordinaria**

**Técnico Superior en
Desarrollo de Aplicaciones
Multiplataforma**

**WATCHIVE: APLICACIÓN DE GESTIÓN DE
SERIES Y PELICULAS**

**Nombre Alumno : José Ángel Aguilar Serrano
DNI : 05987833J**

ÍNDICE

INTRODUCCION	4
Objetivo.....	4
Contexto.....	5
Estado del arte	5
Metodología de desarrollo a utilizar.....	7
Estudio de viabilidad técnica y económica del proyecto.....	8
PLAN DE NEGOCIO	9
Análisis de Mercado.....	10
Tendencias del mercado	11
Idea de negocio.....	11
Propuesta de Valor.....	12
Tipo de mercado	12
La Competencia: Océano y lienzo estratégico.....	14
Encaje Problema-Solución	17
Validación.....	18
Producto Minimo Viable	19
Futuras ampliaciones tras el PMV.....	19
Modelo Canvas.....	20
RECURSOS MATERIALES Y PERSONALES	21
FASE Y SECUENCIACIÓN DE LAS ACTIVIDADES DEL PROYECTO	21
Análisis De Requisitos. Diagrama De Casos De Uso	21
Capa de persistencia: Diagrama E/R de la base de datos.....	23
Capa De Negocio: Diagrama De Clases (Uml)	25
Capa De Presentación. Interfaces Gráficas	26
Pruebas De Software.....	38
IMPLEMENTACIÓN E INTEGRACIÓN	39

Tecnologías	39
Herramientas Usadas.....	40
Detalles De La Implementación	40
EVALUACIÓN DEL PROYECTO.....	46
CONCLUSIÓN.....	47
Dificultades encontradas y soluciones aportadas	47
Desviaciones.....	47
Propuestas de mejora del proyecto.....	47
BIBLIOGRAFIA.....	48

INTRODUCCION

Objetivo

El objetivo principal de este proyecto es el desarrollo de una aplicación de escritorio que permita a los usuarios llevar el control personalizado de sus contenidos audiovisuales consumidos, tanto películas como series, ofreciendo además funcionalidades inteligentes para la recomendación de contenido.

En la plataforma los usuarios además podrán valorar el contenido visionado y expresar el estado emocional que sentían el día que decidieron visionar el contenido elegido.

Hablemos ahora de los objetivos específicos del proyecto, destacamos:

- Facilitar el registro y seguimiento de series y películas a los usuarios.
- Permitir valoraciones personales, siendo estas la puntuación dada por el usuario y la emoción asociada por el mismo al contenido.
- Mejorar la experiencia de usuario mediante recomendaciones basadas en emociones y preferencias registradas.
- Ofrecer una herramienta visual, intuitiva y sencilla, que permita una experiencia fluida.
- Practicar e integrar los conocimientos técnicos adquiridos en el ciclo: bases de datos, programación orientada a objetos, interfaces gráficas, etc.
- Creación de dos roles diferenciados:
 - Usuario: Puede registrar películas y series a sus listas, filtrar el contenido y recibir recomendaciones.
 - Administrador: Puede gestionar los usuarios que acceden a la aplicación
- Crear un sistema de aprendizaje sencillo para personalizar las recomendaciones de cada usuario y que estas evolucionen a medida que los hábitos de consumo del usuario también lo hacen.
- Hacer uso de una API para acceder a información.

Hemos pensado en desarrollar una aplicación sencilla pero cuya funcionalidad sea cómoda y amigable para el usuario, pensando en darle escalabilidad a futuro, de este modo, llevaremos a cabo un proyecto que podrá resolverse de manera adecuada dentro de los márgenes de tiempo establecidos, aportando una mayor calidad a la que sería al elegir un proyecto mucho más ambicioso. Haremos uso de la API gratuita de The Movie Database (TMDb), evitando así la necesidad de almacenar grandes volúmenes de datos audiovisuales, lo que reduce enormemente el coste de almacenamiento y el tiempo de desarrollo

Contexto

En un mundo donde el consumo de contenidos digitales ha crecido exponencialmente durante los últimos años para satisfacer las necesidades de consumidores que desean ver más y más entretenimiento de manera rápida e inmediata, los usuarios se enfrentan a la sobrecarga de información.

Desde principios de la década de los 2010, plataformas como Netflix han ido poco a poco ocupando el tiempo libre de los usuarios, hecho que se vio enormemente potenciado por la pandemia que azotó al mundo en el año 2019. Desde entonces, los consumidores han incrementado su deseo de consumir contenido digital de una manera abrumadora especialmente para ellos mismos.

Múltiples plataformas de streaming, miles de títulos disponibles, y poca capacidad de organizar toda esa información.

Además, las recomendaciones que las plataformas ofrecen a los usuarios suelen girar en torno al contenido que se puede visionar en dicha plataforma, ignorando así en muchas ocasiones opciones que pueden encajar mejor con los usuarios.

Este proyecto surge como una solución de valor añadido que ayuda a los usuarios a recordar qué han visto, cómo se sintieron, y qué podrían ver a continuación, todo desde una plataforma que centraliza y personaliza su experiencia audiovisual en base a sus gustos, personalidades y preferencias.

Para crear una experiencia personalizada haremos uso de las emociones. Los usuarios consumen diferente dependiendo de cuál sea su estado de ánimo, nuestro objetivo es mejorar dichas recomendaciones en base a cómo se siente la persona a la hora de querer consumir alguna serie o película.

Estado del arte

Como hemos comentado anteriormente, el crecimiento en los últimos años del contenido digital está haciendo que los usuarios deban afrontar una sobrecarga de opciones a la hora de elegir qué consumir. Una encuesta realizada a más de 2000 suscriptores estadounidenses reveló que una persona promedio dedica alrededor de 110 horas al año solo a navegar por los catálogos de streaming, sin decidirse por un contenido. (Saples, 2024)

Tanto es el tiempo que invertimos que algunos hacen referencia a este cansancio como “fatiga de decisión” que lleva a los usuarios a directamente no elegir nada o tomar decisiones impulsivas que muchas veces afectan negativamente a su experiencia. (Maslansky, 2023)

En el panorama actual, existen diversas plataformas que permiten a los usuarios, registrar, valorar y descubrir contenido audiovisual. A continuación, vamos a realizar un breve análisis de algunas de las más relevantes:

IMDb (Internet Movie Database)

Base de datos en línea que ofrece información detallada sobre películas, series, actores y otros profesionales del cine y la televisión.

Características principales:

- Valoraciones y reseñas de usuarios
- Listas personalizadas
- Información técnica y artística detallada

Limitaciones:

- Enfoque más orientado a la información técnica que a la experiencia personal del usuario.

FilmAffinity

Plataforma española de recomendación de cine y series.

Características principales:

- Valoraciones y críticas de usuarios
- Sistema de “almas gemelas” que conecta usuarios con gustos similares.
- Listas personalizadas y seguimiento de contenido visto

Limitaciones:

- Las recomendaciones se basan en afinidades de puntuaciones, sin considerar otros apartados del usuario.

MyAnimeList

Plataforma especializada en anime y manga, que permite a los usuarios llevar un registro de series vistas y leer reseñas. Como dato adicional, esta fue la mayor fuente de inspiración para este proyecto.

Características principales:

- Listas personalizadas de anime/manga vistos o por ver.
- Valoraciones y reseñas.
- Foros y clubes para discusión temática.

Limitaciones

- Enfoque limitado al anime y manga

Además de las limitaciones individuales de cada plataforma, como podemos apreciar ninguna de ellas hace uso de los estados de ánimo del usuario para llevar a cabo sus recomendaciones, esto nos da una ventaja diferenciadora.

El eje principal de WatchHive es la experiencia emocional del usuario.

Analicemos entonces, nuestras características diferenciadoras:

- Registro emocional: Permite al usuario asociar emociones específicas al contenido que decide visualizar, creando un historial emocional personalizado.
- Recomendaciones basadas en emociones: El sistema sugiere contenido que se alinea con el estado emocional del usuario. Además, aprenderá de cada usuario cual es el género que el usuario prefiere visionar cuando siente una emoción concreta.
- Al hacer uso de la API de The Movie Database (TMDb) reducimos la carga de mantenimiento y actualización del contenido.

Metodología de desarrollo a utilizar

Se ha optado por una metodología ágil basada en SCRUM, adaptada al trabajo individual y colaborando con la tutora del proyecto. Las fases han sido organizadas en iteraciones con tareas definidas por objetivos semanales, esto permite llevar a cabo una planificación flexible que nos ayudará a mejorar el producto progresivamente mediante la retroalimentación que recibamos en cada entrega.

Esta metodología también nos facilita la identificación de errores o desviaciones del proyecto a la vez que nos brinda una validación continua de las funcionalidades que vamos implementando a lo largo del desarrollo.

Estudio de viabilidad técnica y económica del proyecto

En este apartado vamos a realizar un breve análisis acerca de cuán viable técnica y económicamente desarrollar un proyecto como WatchHive.

Si nos basamos en las herramientas y tecnologías aprendidas a lo largo del ciclo el proyecto es completamente viable, no requiere de infraestructuras complejas ni de licencias de ningún software de pago. La base de datos puede funcionar localmente o si en un futuro se deseara escalar, podría usarse un servidor.

Económicamente hablando el coste que puede conllevar realizar una aplicación de este tipo es extremadamente bajo ya que hablamos de software libre y gratuito (Visual Studio Community, MySQL Workbench 8.0, API REST TMDb). Dado que el desarrollo lo llevará a cabo una única persona el coste de personal se ve reducido a 0.

Si en un futuro se desease que el proyecto genere una rentabilidad económica podríamos plantear la integración de anuncios dentro de la aplicación en espacios delimitados y reservados exclusivamente para el uso de banners comerciales.

Además, si el proyecto llegase a escalar más allá de lo esperado se podría plantear un modelo de negocio donde las propias plataformas puedan promocionar sus estrenos más recientes en su ecosistema pagando para que este aparezca mas arriba en las páginas de series y películas más populares, **sin afectar nunca a las recomendaciones para los usuarios las cuales siempre se basarán exclusivamente en sus gustos y emociones personales.**

FASES DEL PROYECTO

El desarrollo del proyecto ha sido estructurado en varias fases diferenciadas, siguiendo una metodología ágil que permite una planificación flexible, iterativa y centrada en la mejora continua. Cada fase ha sido diseñada para abordar aspectos específicos del desarrollo, desde el análisis inicial de requisitos hasta las pruebas finales.

Estas fases nos permiten distribuir adecuadamente el trabajo y los recursos, también facilitar el control del avance, la detección temprana de errores y la validación progresiva del proyecto. A continuación, se detallan cada una de las etapas en una tabla, incluyendo las actividades realizadas y los plazos aproximados de ejecución.

Vamos a marcarnos como fecha de inicio el día 17 de marzo, y como fecha de entrega tenemos el día 1 de junio, contamos por lo tanto con aproximadamente 11 semanas para desarrollar el proyecto.

Fase	Actividades	Duración aproximada
1. Planificación	Definir objetivos, análisis de requisitos, anteproyecto.	1 semana
2. Diseño	Modelo de base de datos, diagramas de clases, interfaces, estructura del proyecto.	1 semana
3. Implementación	Desarrollo de las funcionalidades principales, conexión a base de datos, interfaz, pruebas con API.	4 semanas
4. Pruebas y mejoras	Test de funcionalidad, corrección de errores y mejoras de UI.	1 semana
5. Documentación	Memoria técnica, manual de usuario, plan de negocio	2 semanas
6. Presentación	Presentaciones, grabación de videos y entrega final	1 semana

PLAN DE NEGOCIO

El siguiente apartado de la memoria recoge todo lo referente al módulo de empresas impartido durante el 2º curso del ciclo. Posteriormente trataremos los apartados técnicos del proyecto en el resto del documento.

Resumen del proyecto

El proyecto WatchHive tiene como objetivo el desarrollo de una aplicación orientada a la gestión personalizada de contenido audiovisual, principalmente películas y series. Es una herramienta que permite a los usuarios organizar y registrar de forma intuitiva el contenido que han visto, añadir valoraciones personales, asociar emociones a cada experiencia y recibir recomendaciones personalizadas basadas en sus gustos.

Brainstorming

Para buscar la idea nos hicimos la siguiente pregunta: ¿qué tareas comunes en el día a día de una persona consumen a veces más tiempo del que deberían y podríamos evitar haciendo uso de una aplicación?

La respuesta que más nos llamó la atención y que haría que cualquier persona a día de hoy pudiera sentirse identificada fue: buscar una película o una serie para ver.

Así se nos ocurrió la idea de crear una aplicación que permita organizar el contenido visionado y además ofrezca al usuario recomendaciones en base a los estados de ánimo en los que normalmente se encuentra al consumir un género en específico de contenido.

Análisis de Mercado

Problema detectado

Actualmente, el consumo de contenido audiovisual se encuentra en su época dorada gracias a las plataformas de streaming (Netflix, HBO, Prime Video, Disney+, etc.). Sin embargo, los usuarios no disponen de una herramienta centralizada que permita:

- Llevar a cabo un seguimiento unificado de todo el contenido visualizado
- Asignar emociones personales a la visualización de cada título para poder recibir recomendaciones personalizadas según estados de ánimo del usuario.
- Gestionar listas de contenido pendiente o favorito de forma independiente a cada servicio.

Esto genera desorganización, frustración e infoxicación, especialmente entre usuarios que alternan constantemente entre plataformas.

Tendencias del mercado

Saturación de Contenido y Plataformas

- **Cifras clave:** El usuario promedio en 2024 está suscrito a 3 o más servicios de streaming (Netflix, HBO Max, Disney+, Prime Video, etc.).
- **Problema:** Esto genera fatiga de elección. El exceso de contenido provoca que los usuarios pasen más tiempo buscando qué viendo algo.
- **Oportunidad:** Herramientas como WatchHive, que permiten organizar, recordar y clasificar lo visto, ayudaría a reducir esta carga cognitiva.

Personalización y Autoanálisis Digital

- El usuario moderno valora experiencias que **se adaptan a sus gustos y emociones**.
- Crece el interés por apps que permiten **rastrear estado emocional, productividad o hábitos** (como Notion, Daylio o MoodPath).
- En este caso, **asociar emociones a lo que se ha visto** permite al usuario comprender sus patrones de consumo emocional, lo que puede derivar en recomendaciones más empáticas y no solo algorítmicas.

Crecimiento del consumo multiplataforma

- Hoy en día se consume contenido en móviles, tablets, PCs, Smart TVs, etc.
- Una solución multiplataforma que centralice la experiencia tiene mucho más valor.
- WatchHive puede aprovechar esta fragmentación, **dando cohesión y continuidad** al historial del usuario, esté donde esté.

Idea de negocio

Crear una **aplicación de escritorio**, con visión a multiplataforma (desarrollo posterior para dispositivos portátiles como móviles, tablets, etc.) que permita a los usuarios:

- Organizar películas/series.
- Llevar un historial de visionado.
- Puntuar y asociar emociones al contenido.
- Recibir recomendaciones personalizadas en base al punto anterior.

Propuesta de Valor

Ofrecemos una herramienta inteligente, personalizable y emocionalmente consciente que permite a los usuarios llevar un control detallado de las películas y series que ven, asociándolas a sus emociones, puntuaciones y fechas, para mejorar su experiencia audiovisual y su autoconocimiento.

Tipo de mercado

El producto está dirigido principalmente a usuarios finales (consumidores individuales), por lo tanto, estamos hablando de un **mercado B2C (Business to Consumer)**. Los usuarios utilizarán la plataforma para obtener recomendaciones personalizadas, esto implica varias cosas:

- Alto volumen de usuarios
- Ciclo de compra corto (uso inmediato)
- Necesidad de una experiencia de usuario muy optimizada

Nos ubicamos en un **Mercado de nicho emergente** dentro del **sector digital**, ya que, aunque se relaciona con grandes sectores es un nicho específico aún poco explotado:

- Recomendaciones de contenido basadas en emociones.
- Interacción personalizada e inteligente (en auge debido a la IA)

TAM estimado

El mercado global del streaming de contenido audiovisual y plataformas de recomendación está valorado en +150 mil millones de dólares y se espera que continúe creciendo con la expansión del entretenimiento digital, el machine learning y la inteligencia artificial.

Además, si consideramos plataformas que personalizan la experiencia del usuario mediante algoritmos, IA emocional y análisis de comportamiento, también se solapan sectores de EdTech y HealthTech.

SAM estimado:

Nos centramos inicialmente en usuarios hispanohablantes (con posibilidad de escalar a futuro a más idiomas) interesados en películas y series (plataformas tipo Netflix, HBO, etc.), principalmente entre 16 y 45 años.

Este grupo se estima en 30-40 millones de personas (considerando España y Latinoamérica), con alta tasa de consumo de contenido audiovisual y creciente interés en personalización.

SOM estimado:

Con una estrategia de marketing eficaz, alianzas estratégicas y fidelización, podríamos alcanzar aproximadamente el 1-2% del SAM en los primeros 2-3 años, es decir:

300.000 a 800.000 usuarios activos.

Segmento de clientes

El segmento está compuesto principalmente por:

- Consumidores frecuentes de contenido audiovisual, de entre 16 y 45 años, con presencia activa en varias plataformas de streaming.
- Usuarios organizados o con perfiles analíticos que desean mantener un control y registro del contenido que ven.
- Personas con poco tiempo y que buscan recomendaciones personalizadas.

Mapa De Empatía

Nombre: Iván, 21 años

¿Qué piensa y siente?

- Quiere aprovechar su tiempo libre para ver algo que encaje con su estado emocional.
- Se frustra cuando las plataformas no entienden lo que quiere.
- Busca experiencias personales y significativas, no solo entretenimiento.

¿Qué oye?

- Recomendaciones de amigos sobre series o películas
- Opiniones de influencers o críticas en redes sociales
- Publicidad de plataformas de streaming que prometen "lo mejor para ti"

¿Qué ve?

- Interfaces abrumadoras con cientos de títulos
- Recomendaciones genéricas o repetitivas
- Plataformas que muestran solo lo más popular o reciente
- Reseñas o valoraciones sin contexto emocional.

¿Qué dice y hace?

- "No sé qué ver hoy".
- "Me apetece algo que me haga reír" o "No estoy para dramas ahora"
- Consulta trailers, reseñas o rankings
- Cambia de contenido a los 10 minutos si no conecta con él
- Comparte lo que ve en redes cuando le impacta emocionalmente

Usuarios finales

- Personas que disponen de varias plataformas de streaming
- Usuarios con intereses emocionales que buscan analizar sus patrones de consumo en base a sus emociones
- Pequeñas comunidades o foros de recomendación que quieren compartir sus listas y puntuaciones entre amigos

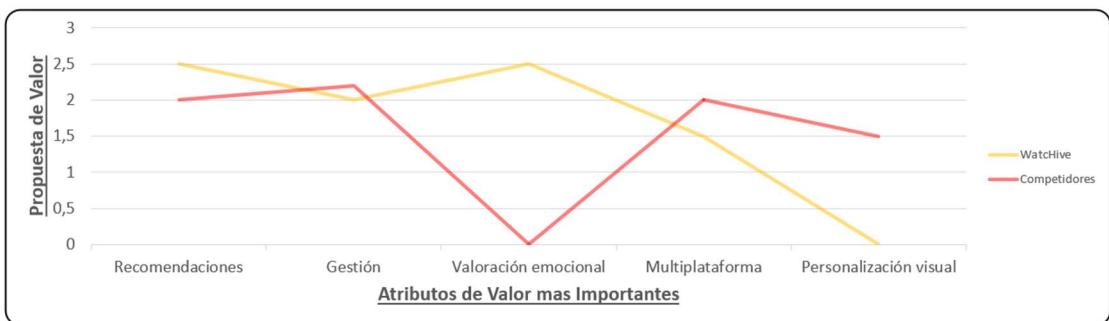
La Competencia: Océano y lienzo estratégico

Fijémonos en algunos competidores que podríamos encontrarnos.

Herramientas similares ya existentes en el mercado:

Plataforma	Características	Limitaciones
Letterboxd	Red social para cinéfilos, listas, reseñas, puntuaciones.	Solo películas, no series.
Trakt.tv	Seguimiento de películas y series, integración con Kodi	Requiere apps externas para aprovecharlo.
TV Time	Seguimiento de series y películas, calendario de estrenos.	Poca personalización emocional.
IMDb Watchlist	Registro simple de vistos o deseados.	No ofrece análisis o seguimiento detallado.

Como podemos observar, ya existe una oferta de este tipo de servicios en el mercado, eso nos sitúa en un **Océano Rojo**, las reglas ya están definidas, la competencia es alta. Sin embargo, nuestro proyecto cuenta con una serie de ventajas que el resto de competidores no poseen.



Nuestra ventaja competitiva:

- Integración emocional.
- Seguimiento cruzado y neutro entre plataformas.
- Enfoque en el usuario, no en el marketing de las plataformas

Este enfoque distinto que hemos conseguido brindar al proyecto nos facilitará la integración en el mercado gracias a que nos **diferenciaremos** de los competidores ofreciendo una herramienta específica que ellos no poseen: el sistema de recomendación en base a las emociones.

Análisis DAFO y CAME

Análisis DAFO

Fortalezas	Debilidades
<ul style="list-style-type: none"> • Aplicación ligera y gratuita • Integración con API externa (TMDb) • Gestión de emociones y organización única 	<ul style="list-style-type: none"> • Limitado a entorno de escritorio (WPF) al inicio • Falta de sincronización multiplataforma
Oportunidades	Amenazas
<ul style="list-style-type: none"> • Mercado en crecimiento • Posible integración con redes sociales • Evolución hacia app móvil/web 	<ul style="list-style-type: none"> • Competencia con plataformas más grandes • Dependencia de una API externa (TMDb) • Cambios en licencias de uso de contenidos

Análisis CAME

Mantener Fortalezas	Corregir Debilidades
<ul style="list-style-type: none"> • Aplicación ligera y gratuita • Interfaz intuitiva y sencilla 	<ul style="list-style-type: none"> • Hacer una versión para web y dispositivos móviles
Explotar Oportunidades	Afrontar Amenazas
<ul style="list-style-type: none"> • Explorar colaboración con bloggers/canales de series • Posible monetización a través de anuncios y tratos con plataformas de streaming 	<ul style="list-style-type: none"> • Establecer backup plan si falla la API TMDb

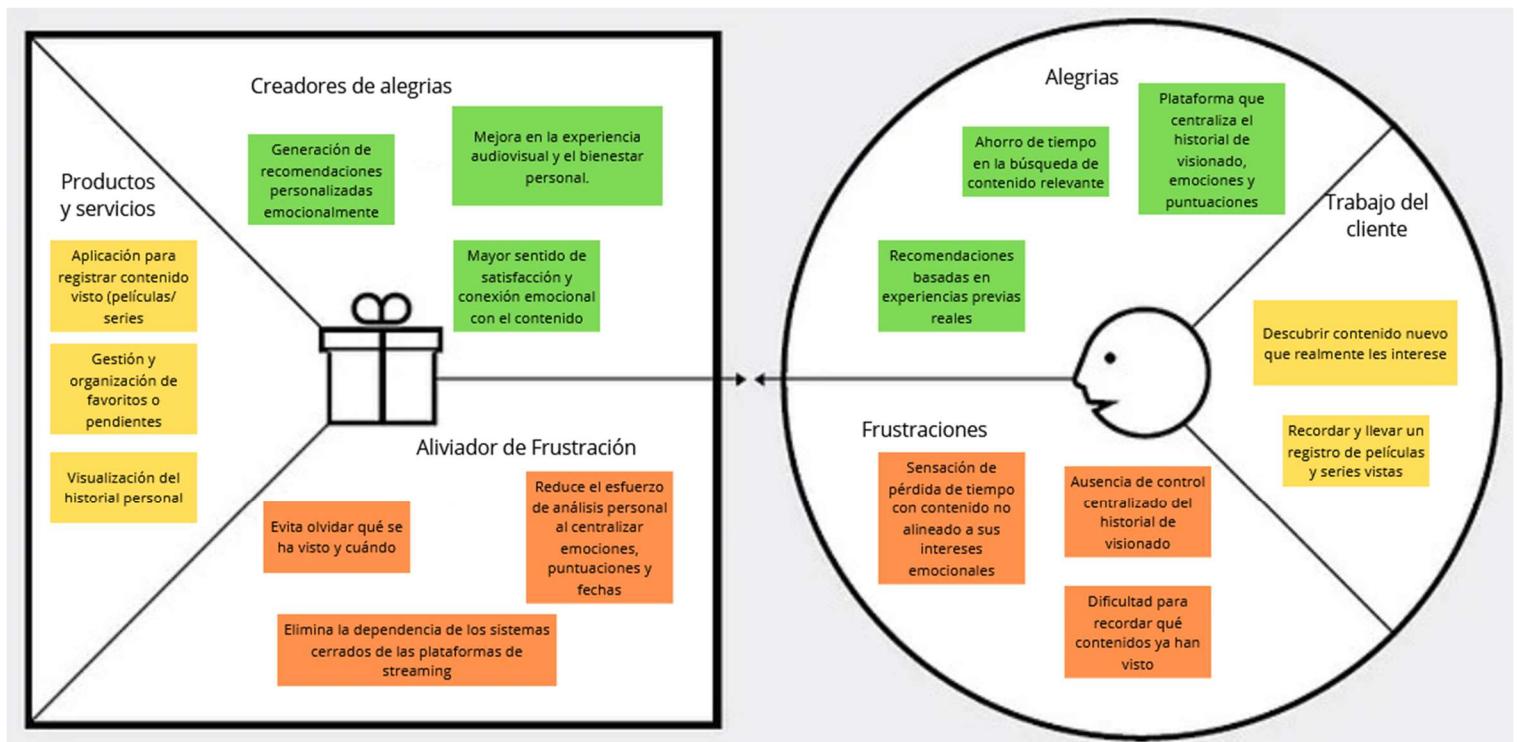
Encaje Problema-Solución

- **Productos y servicios**
 - Plataforma web para registrar contenido visto (películas/series).
 - Asignación de emociones, puntuación y fecha al contenido visualizado.
 - Sistema de recomendaciones basado en emociones y preferencias.
 - Visualización del historial personal.
 - Gestión y organización de favoritos o pendientes.
- **Creadores de Alegría**
 - Generación de recomendaciones personalizadas emocionalmente.
 - Mejora en la experiencia audiovisual y el bienestar personal.
 - Mayor sentido de satisfacción y conexión emocional con el contenido.
 - Posibilidad de analizar la evolución emocional del usuario a través del tiempo.
- **Aliviador de frustraciones**
 - Evita olvidar qué se ha visto y cuándo.
 - Elimina la dependencia de los sistemas cerrados de las plataformas de streaming.
 - Reduce el esfuerzo de análisis personal al centralizar emociones, puntuaciones y fechas.
 - Mejora la experiencia de selección de contenido.
- **Alegrías**
 - Mayor autoconocimiento emocional a través del contenido que ven.
 - Recomendaciones basadas en experiencias previas reales.
 - Plataforma que centraliza el historial de visionado, emociones y puntuaciones.
 - Ahorro de tiempo en la búsqueda de contenido relevante.
 - Satisfacción al revivir y compartir experiencias audiovisuales significativas.
- **Frustraciones**
 - Dificultad para recordar qué contenidos ya han visto.
 - Falta de herramientas personalizadas que analicen emociones y gustos.
 - Recomendaciones impersonales o poco precisas en plataformas de streaming.
 - Sensación de pérdida de tiempo con contenido no alineado a sus intereses emocionales.
 - Ausencia de control centralizado del historial de visionado.

- **Trabajo del cliente**

- **Recordar y llevar un registro de películas y series vistas.**
- **Valorar y puntuar el contenido audiovisual.**
- **Expresar cómo se han sentido con lo que han visto.**
- **Encontrar recomendaciones que se ajusten a sus gustos y estado emocional.**

Descubrir contenido nuevo que realmente les interese.



Validación

Para validar la necesidad y el encaje de la solución propuesta, se pueden llevar a cabo las siguientes acciones:

- **Interacción en redes o foros** como Reddit (*r/television*, *r/movies*) para recibir feedback de usuarios activos.
- Encuestas a usuarios de plataformas como Netflix o HBO preguntando si usan alguna app para registrar lo que ven.
- Pruebas de concepto (PMV) compartido con un grupo reducido de usuarios para medir el interés real.

Producto Mínimo Viable

El objetivo del Producto Mínimo Viable (PMV) es que validemos de forma rápida y económica la propuesta que estamos haciendo. Principalmente, buscamos comprobar si los usuarios:

- Están dispuestos a usar una plataforma con este enfoque.
- Encuentran valor en las recomendaciones basadas en sus estados de ánimo
- Mejoran su experiencia de visualización gracias a la personalización

Funcionalidades clave del PMV

El PMV se construye con solo las características esenciales para validar la hipótesis central:

- **Login de usuario**
- **Pantalla de recomendaciones en base al estado de ánimo**
- **Sistema de puntuación de 1 a 5 puntos**, con el que se pretende ajustar más aun las recomendaciones personalizadas. Por ejemplo, si un genero es muy elegido, pero tiene una puntuación muy baja para una emoción en concreto, este no se tendrá tan en cuenta como otros que se vean menos, pero tengan puntuaciones más altas.
- **Vista de detalle del contenido**: Poster, descripción, fecha de estreno, etc.
- **Gestión de lista de “visto” y de “pendientes.”**

Tecnología usada para el PMV

- **Frontend**: Aplicación de escritorio (WPF)
- **Backend**: C# con conexión a una base de datos local MySQL.
- **Base de datos**: MySQL.
- **API externa**: TheMovieDB para obtener datos visuales y sinopsis.

Futuras ampliaciones tras el PMV

Si las pruebas resultan exitosas nos centraremos en mejorar el rendimiento de la aplicación y en hacer que esta no dependa de una API externa para que este todo mucho mas centralizado. Ampliaremos y mejoraremos el sistema de recomendación y daremos a los usuarios la capacidad de personalizar su perfil. Planteamos también la posibilidad de ofrecer a los usuarios comentar reseñas en los contenidos que lo deseen para que estas puedan compartirse con el resto de usuarios. Por supuesto ampliaremos la plataforma para que utilice otros idiomas a parte del español, centrándonos sobre todo en dar soporte al inglés.

Canvas del modelo de negocio



RECURSOS MATERIALES Y PERSONALES

Para el desarrollo de WatchHive se han utilizado recursos materiales accesibles y habituales en entornos de desarrollo académico. El equipo principal que se ha usado en el desarrollo es un ordenador portátil personal que cuenta con un procesador Intel Core i7, 16 GB de memoria RAM y sistema operativo Windows 11 y por supuesto acceso a conexión estable a internet y herramientas de software libre y de desarrollo como Visual Studio 2022 Community, MySQL 9.1 Command Line Client, MySQL Workbench 8.0 CE, GitHub y la API de The Movie Database (TMDb).

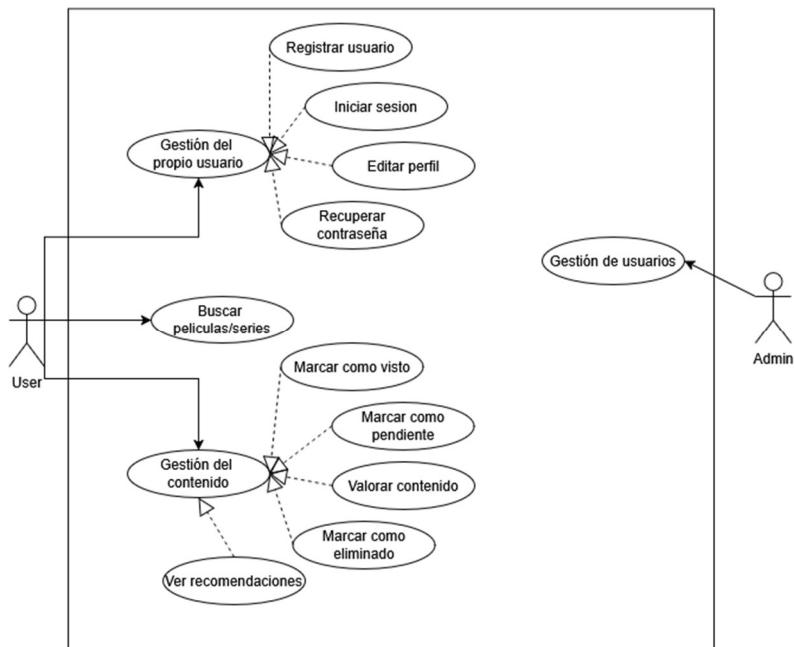
A nivel de personal, el proyecto ha sido desarrollado íntegramente por el alumno José Ángel Aguilar Serrano, encargado tanto del diseño como de la programación, en un periodo de tiempo de aproximadamente 3 meses dedicando una media de unas 3 horas al día.

La planificación temporal fue clave para permitir una gestión eficiente del tiempo y de los recursos que se tenían a disposición.

FASE Y SECUENCIACIÓN DE LAS ACTIVIDADEDES DEL PROYECTO

A modo introductorio aclarar que todos los diagramas se encuentran al final de cada uno de los puntos, de modo que el inicio es una pequeña explicación teórica de lo que se ve reflejado en la imagen.

Análisis De Requisitos. Diagrama De Casos De Uso



El análisis de requisitos funcionales lo representaremos con un diagrama de casos de uso UML justo al final de este punto, pero antes vamos a identificar las funcionalidades principales de la plataforma y los actores que interactuarán con ella.

Actores principales

- **Usuario (User):** Es el usuario que se registra en la plataforma, que la utiliza para buscar contenidos, recibir recomendaciones, etc. Es en quién principalmente centraremos el desarrollo de la aplicación.
- **Administrador (Admin):** Representa la administración del sistema, su principal funcionalidad es la gestión de usuarios. Planteamos en cierto momento que también gestionase el contenido, sin embargo, al ser este proporcionado por la API no le encontramos sentido a dar esta funcionalidad. **Queda planteada la situación como posible objetivo a la hora de escalar la aplicación.**

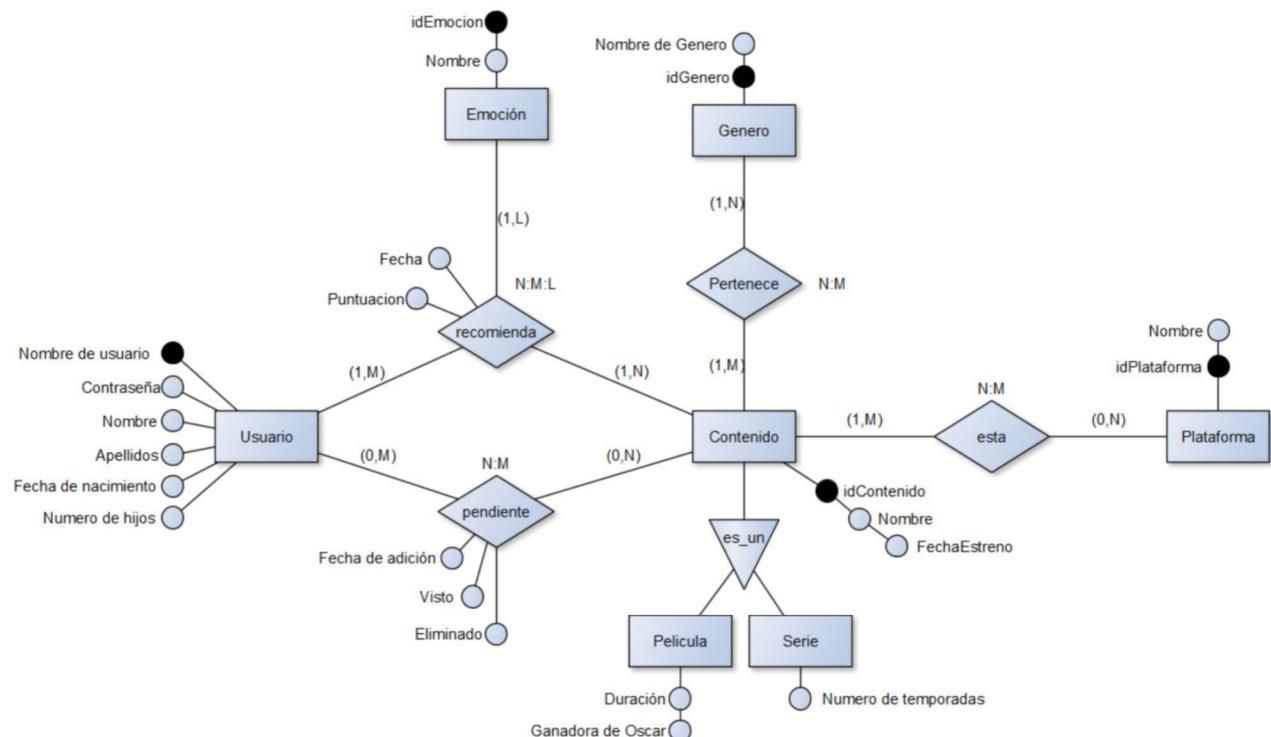
Casos de uso principales

- Gestión del propio usuario: Esto es la gestión que hace el usuario de su propia cuenta: registrarse, iniciar sesión, cambiar la contraseña, etc.
- Buscar películas y series: Es la funcionalidad que permite al usuario explorar el contenido haciendo uso de la API externa que hemos utilizado (TMDB)
- Gestión del contenido: Es la funcionalidad principal de cara al usuario, esta le permite agregar el contenido a su lista de “vistos” o “pendientes”, dependiendo de si ya ha visto el contenido o quiere anotarlo para verlo en un futuro.
- Gestión de usuarios (Admin): Este caso es el que corresponde a la funcionalidad del administrador, aquí contemplamos funcionalidades como eliminar usuarios, añadirlos y modificarlos.

Capa de persistencia: Diagrama E/R de la base de datos.

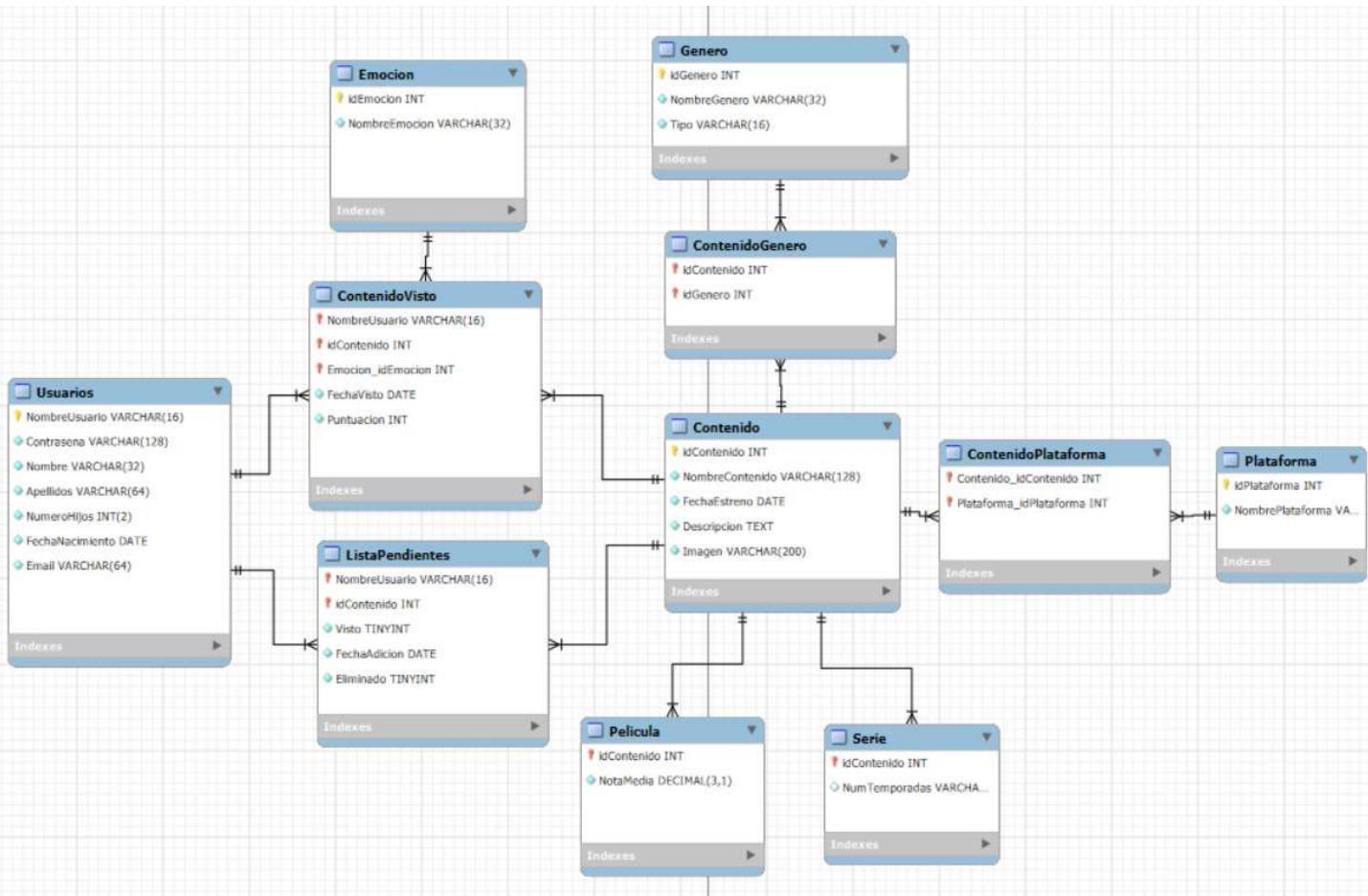
El modelo entidad-relación (E/R) que presentamos representa la estructura lógica temprana de la base de datos de la aplicación, centrada en la gestión personalizada del contenido audiovisual y los estados de ánimo del usuario. Dado que es una capa crítica para asegurar la persistencia de la información vamos a detenernos a explicar algunas de las entidades y relaciones más relevantes que se pueden apreciar en el diagrama:

- **Usuario:** recoge los datos personales del usuario, incluye información útil para poder perfeccionar el sistema de recomendaciones a futuro. La clave primaria será el propio nombre de usuario, por lo que este no podrá existir previamente.
- **Contenido:** Representa las películas y series, diferenciadas mediante una relación de especialización, de esta forma dividimos los atributos de cada una y facilita la escalabilidad.
- Tanto Genero como Emoción serán entidades clave para desarrollar el sistema de recomendaciones
- **Recomienda:** Es una relación ternaria entre Usuario, Contenido y Emoción, nos permitirá conocer los gustos del usuario y con qué puntuaciones valora el contenido que ve.
- **Pendiente:** Representa la lista de contenido pendiente de ver del usuario.



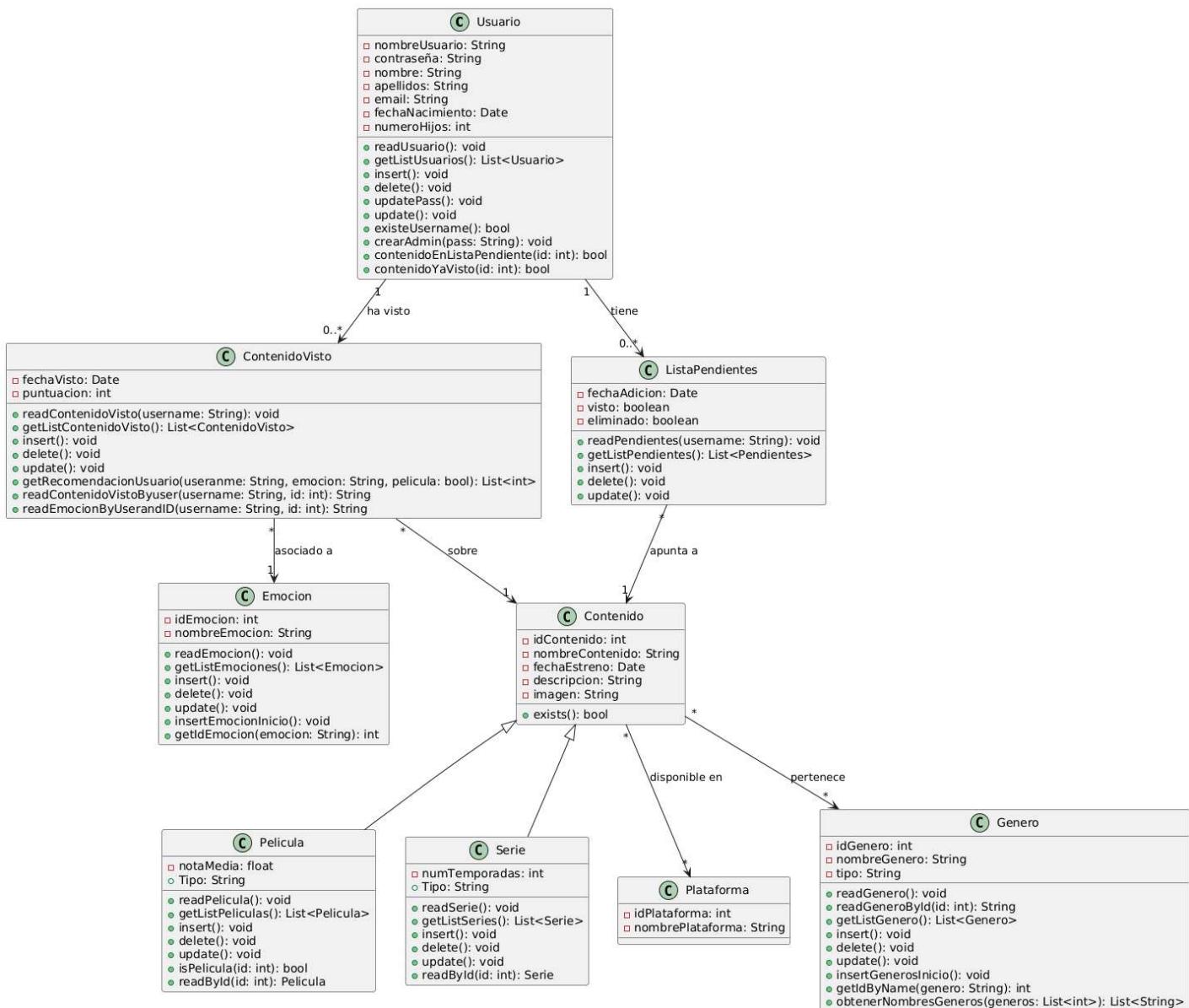
Hemos optado por una estructura relacional normalizada, sin datos redundantes para garantizar eficiencia en las consultas. La diferenciación entre película y serie mediante una jerarquía permite mantener una única tabla de contenidos base y evitar duplicidades.

A continuación, mostramos una imagen de cómo pasamos a tablas este diagrama de entidad relación, para que se pueda apreciar el diseño final de la base de datos estando ya normalizada y creada en MySQL.



Consideramos importante mencionar que en esta parte del proyecto **surgieron desviaciones**, estas serán tratadas en el correspondiente apartado “**Desviaciones**” que se puede encontrar más adelante en este mismo documento.

Capa De Negocio: Diagrama De Clases (Uml)



En este punto se presenta el diseño de la capa de negocio a través de un diagrama de clases UML, para crearlo, nos hemos ayudado de la herramienta PlantUML. Este modelo representa la estructura lógica del sistema, alineando las entidades principales de la base de datos con clases orientadas a objetos, y definiendo claramente sus atributos y métodos.

Las clases reflejan los elementos centrales de la aplicación: usuarios, contenidos, emociones, géneros, listas de visualización y plataformas.

Hacemos uso de la relación de herencia para establecer la relación entre Contenido y Películas y Series, lo hacemos de esta manera para poder reutilizar los atributos comunes y crear una arquitectura más limpia.

Cada clase incorpora operaciones específicas, añadiendo métodos a más concretos a algunas de ellas para que encajen con el comportamiento que esperamos obtener al desarrollar la aplicación. Un ejemplo muy claro de esto es la clase Usuario, que tiene métodos para registrar uno nuevo, otro para acceder a las listas de contenido de un usuario concreto, etc.

Hemos tenido en cuenta que el diseño de las clases debe ser consecuente con la integración de la API de TMDb, que es la encargada de proporcionar los datos del contenido audiovisual.

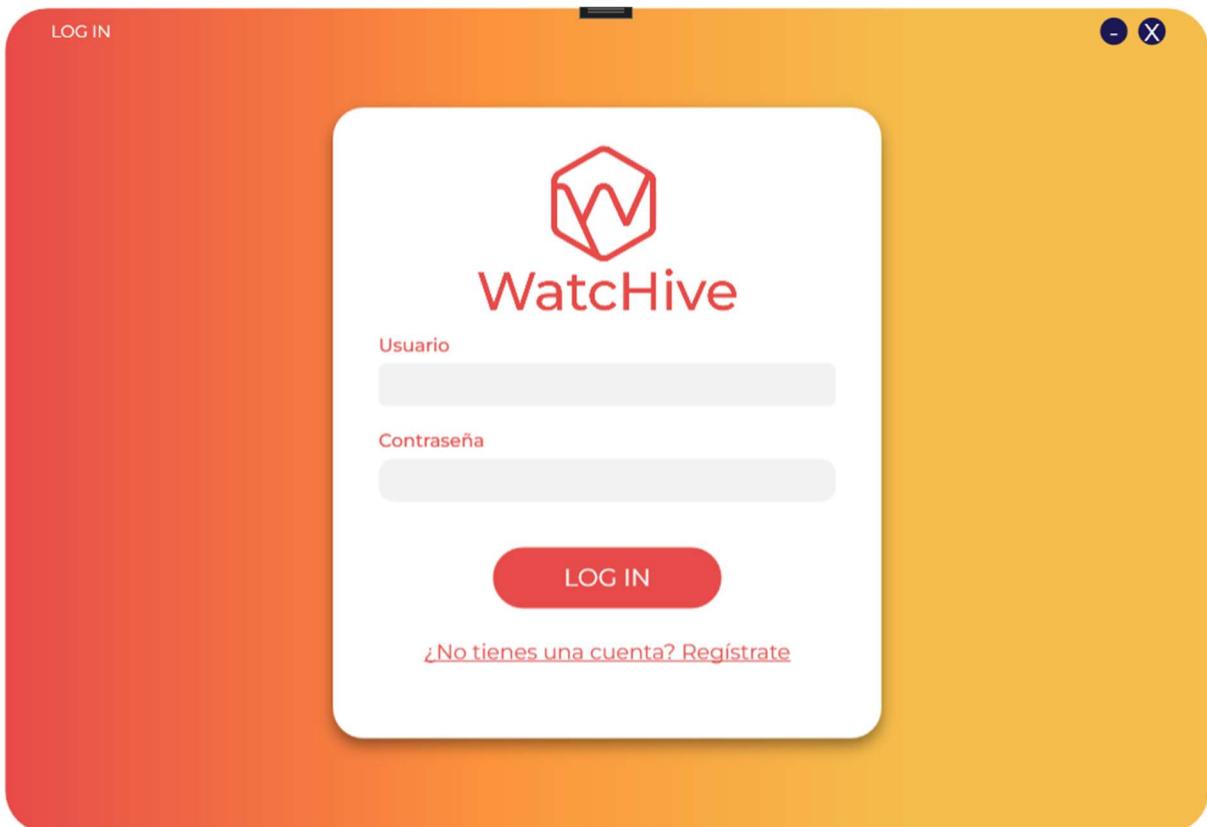
El resultado creemos que es un modelo con cohesión, que puede extenderse y que está preparado para futuras mejoras en la aplicación.

Capa De Presentación. Interfaces Gráficas

Vamos a recoger en este apartado el apartado visual de la aplicación. Para ello nos vamos a servir de capturas de pantalla hechas directamente a la aplicación WatchHive, haremos un pequeño resumen de qué funcionalidad se puede encontrar en dicha vista y explicaremos algunas decisiones de diseño si procede. Finalmente, aclarar que vamos a intentar mostrar las capturas de pantalla intentando seguir el flujo que seguiría un usuario al abrirla.

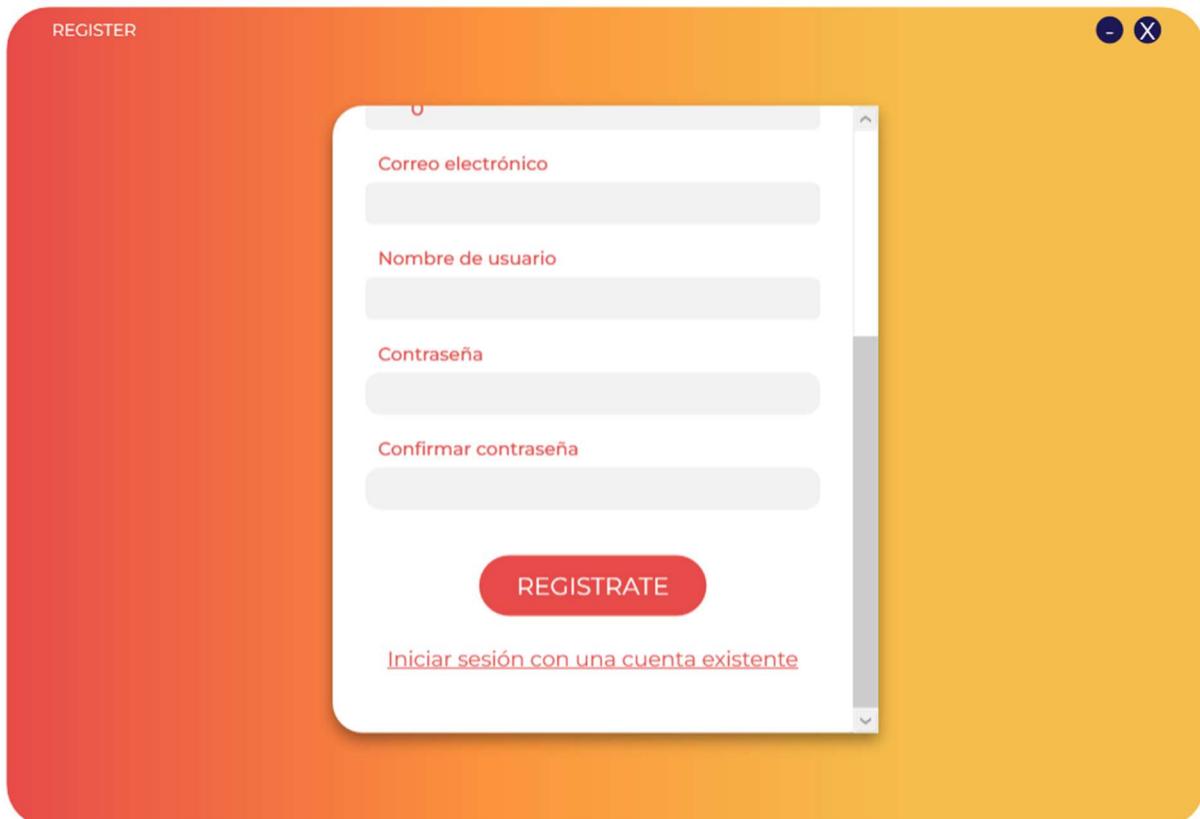
Pantalla de Inicio de sesión

En esta pantalla podemos ver un pequeño formulario de inicio de sesión, si el usuario aun no esta registrado hará click en el texto que se encuentra cerca del borde inferior el cual dice “¿No tienes una cuenta? Regístrate” y se le redirigirá a un formulario de registro.



Pantalla Formulario de Registro de usuario

Como ya hemos mencionado, el usuario accede a esta pantalla desde la ventana de inicio de sesión. Esta pantalla es muy similar a la anterior, con la diferencia de que el formulario abarca todos los atributos necesarios para la creación de un nuevo usuario. Para registrarse el usuario debe llenar todos los campos de manera correcta y pulsar el botón “REGISTRATE”. Si ha accedido a esta ventana por error, puede volver al inicio de sesión haciendo click en el texto que se encuentra justo debajo del botón.

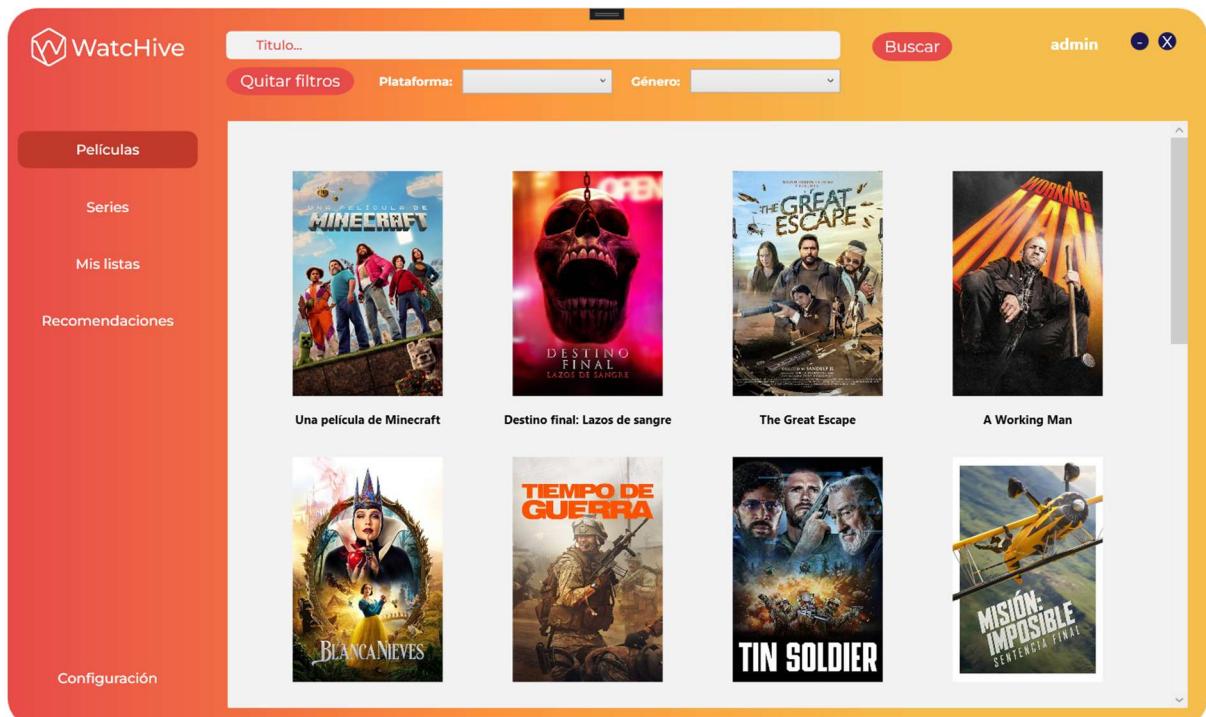


Pantalla principal

Dentro de esta pantalla vamos a ver que podemos acceder a distinto contenido utilizando los botones de la izquierda. Dependiendo de en que vista nos encontremos, el usuario tendrá acceso a una u otras funcionalidades.

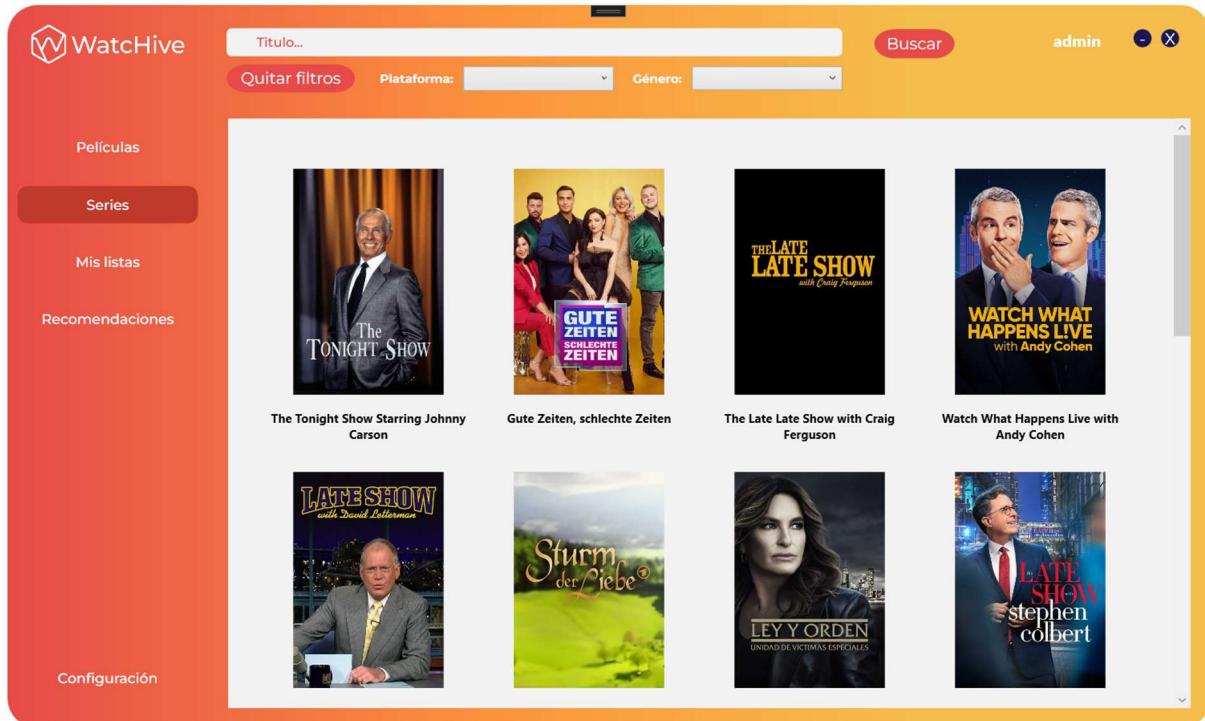
Vista “Películas”

El usuario accede a esta vista haciendo click en el botón “Películas” que se ubica a la izquierda de la pantalla. La función principal de esta vista es mostrar las películas mas populares obtenidas desde la API de TMDb, el usuario puede hacer click en ellas para ver la información de cada título y añadirlo a sus listas si así lo desease. Además, en la parte superior de la pantalla se ubica la barra de búsqueda por títulos y los filtros de Plataforma y Género. Estos filtros se detallan más adelante en el documento.



Vista “Series”

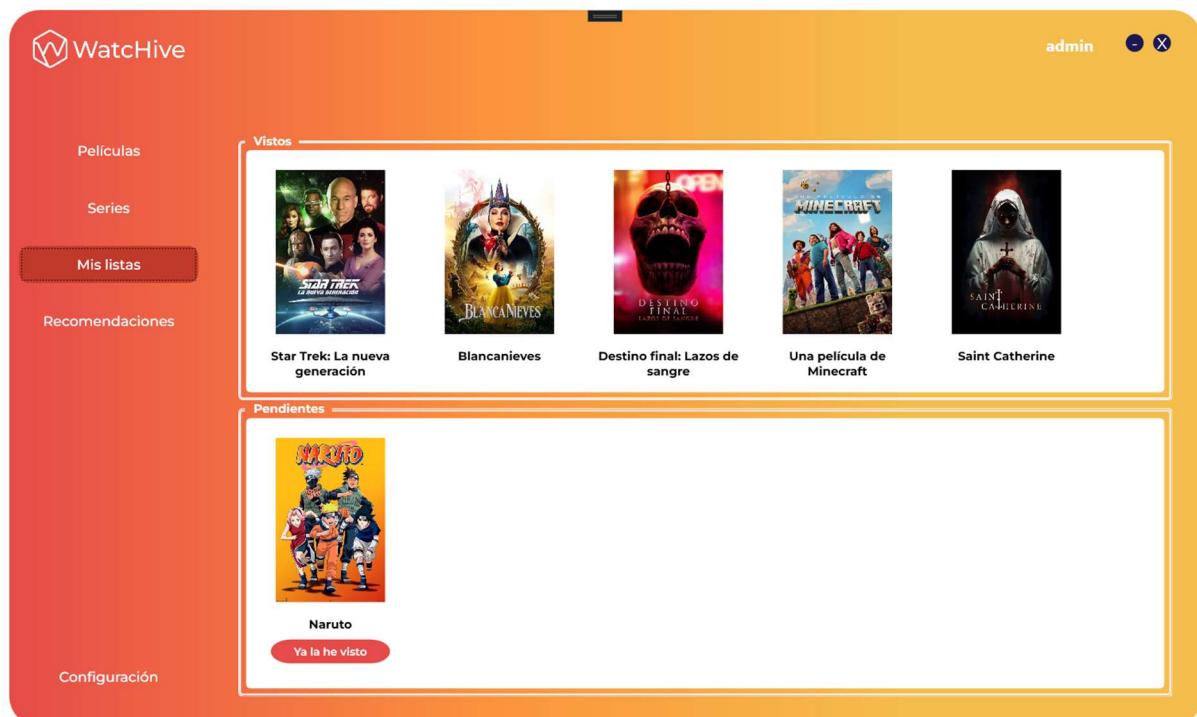
El usuario accede a esta vista haciendo click en el botón “Series” que se ubica a la izquierda de la pantalla. La función de esta vista es la misma que para Películas, pero en ella el usuario podrá ver y buscar series de televisión.



Vista “Mis listas”

El usuario accede a esta vista haciendo click en el botón “Mis listas” que se ubica a la izquierda de la pantalla. En esta vista es donde se encuentran las listas a las que el usuario puede añadir los distintos contenidos que le interesen. La lista “Vistos” es la encargada de mostrar al usuario los contenidos que ya ha visionado, mientras que “Pendientes” es donde el usuario puede encontrar los contenidos que le han llamado la atención pero que aún no ha tenido tiempo de ver. Debajo de los títulos de la lista de Pendientes aparecerá un botón que permite al usuario cambiar de lista los contenidos y marcarlos ahora si como Vistos, al pulsarlo, el título cambiará después de que el usuario introduzca una serie de datos.

En esta vista los filtros de la parte superior de la pantalla se ocultan, pues su funcionalidad esta ligada a la API y esta ventana hace uso de los datos que vamos almacenando en nuestra base de datos.



Pestaña de detalles desde “Mis listas”

Esta pestaña le aparecerá al usuario al hacer click en la imagen o el título de alguno de los contenidos que tenga en sus listas. Si el usuario hace click en un contenido de la lista Vistos, verá los detalles del título además de la fecha en la que lo vió y como se sentía en el momento en que lo vió. Si el usuario hiciese click en un contenido de la lista de Pendientes vería exactamente lo mismo pero estos campos no mostrarían datos hasta que no se marce el contenido como Visto.

A movie poster for "Snow White and the Huntsman". It features a woman with a crown and red lips holding a red apple, with another woman in a yellow dress standing behind her. The background is a forest scene with a castle in the distance.

Blancanieves

Genero: Fantasía - Familia
Nota media: 4,3
Fecha Visto: 21/05/2025 0:00:00
Te sentías: Aburrido/a

Tras la desaparición del benévolos Rey, la Reina Malvada dominó la otrora bella tierra con una vena cruel. La princesa Blancanieves huye del castillo cuando la Reina, celosa de su belleza interior, intenta matarla. En lo profundo del oscuro bosque, se topa con siete enanos mágicos y un joven bandido llamado Jonathan. Juntos, luchan por sobrevivir a la implacable persecución de la Reina y aspiran a recuperar el reino en el proceso.

A movie poster for "Naruto". It features several characters from the anime, including Naruto Uzumaki in the foreground, Sasuke Uchiha, Sakura Haruno, and others, all in their signature ninja attire.

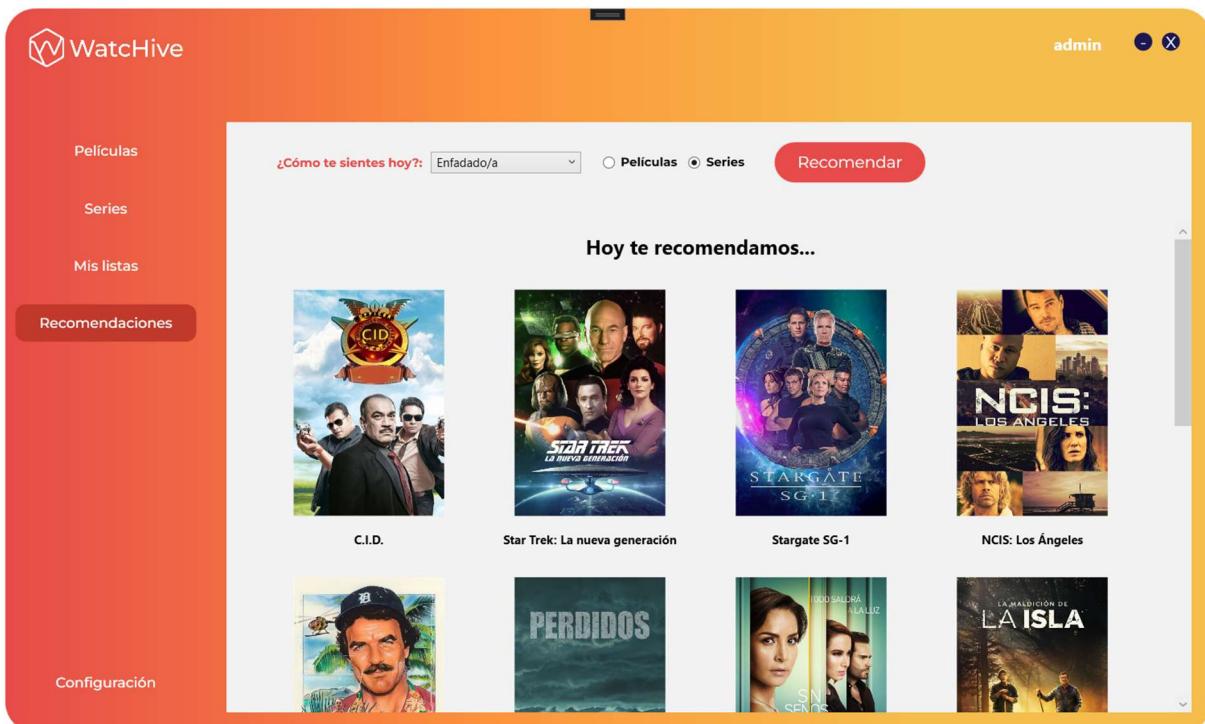
Naruto

Genero: Animación - Acción y Aventura - Ciencia Ficción y Fantasía
Número de temporadas: 1
Fecha Visto:
Te sentías:

Naruto es un joven aprendiz de ninja marginado y temido en la Villa Oculta de la Hoja tanto por su carácter hiperactivo y gamberro como por el terrible poder sellado en su interior. Acompañado por el intenso Sasuke y la ingeniosa Sakura, Naruto comienza su aprendizaje decidido a convertirse en maestro Hokage, la más alta distinción entre los ninjas, para ganarse el reconocimiento de los suyos.

Vista de Recomendaciones

El usuario accede a esta vista haciendo click en el botón “Recomendaciones” que se ubica a la izquierda de la pantalla. La función de esta vista es como bien indica su nombre la de ofrecer las recomendaciones individualizadas a cada usuario en base a su estado de ánimo, el cual se indica desde el desplegable que se encuentra en la parte superior de la vista. El usuario puede alternar entre recibir recomendaciones de Series o de Películas marcando la opción que prefiera justo al lado del desplegable y pulsando el botón de Recomendar.



Pestaña de detalles desde el resto de pantallas.

Si el usuario hace click en la imagen o el titulo de un contenido desde una vista que no sea “Mis listas” la ventana que se mostrará es la que se ve a continuación. Es una estructura prácticamente idéntica a la anterior,sin embargo, aparecen dos botones en la parte inferior de la pantalla, cada uno de ellos sirve para añadir el contenido a la lista correspondiente según que botón pulsemos. Además la fecha de visionado y la emoción que se sintió no aparecen en esta ventana.



Star Trek: La nueva generación

Genero: Ciencia Ficción y Fantasía - Acción y Aventura - Drama - Misterio
Número de temporadas: 1

Star Trek: La nueva generación o Viaje a las estrellas: La nueva generación en Uruguay (Star Trek: The Next Generation) (algunas veces abreviada ST:TNG o TNG) es una serie de ciencia ficción estadounidense dentro del universo de Star Trek. Situada en el siglo XXIV, 21 años después de Star Trek: la serie original, el programa cuenta con una nueva tripulación y una nueva nave Enterprise. Su estreno fue el 28 de septiembre de 1987, contando con más de 27 millones de televidentes el episodio piloto "Encuentro en Farpoint". Con un total de 178 episodios (el mayor

[Agregar a vistos](#) [Agregar a pendientes](#)

Formulario para agregar contenido a Vistos

Si desde cualquier view o ventana el usuario quiere añadir un contenido concreto a su lista de Vistos, le aparecerá esta pequeña pantalla con un pequeño formulario a llenar. En él se deben indicar la fecha de visionado del contenido, el estado de ánimo y una puntuación. Estos dos últimos datos son esenciales y todos ellos obligatorios para poder realizar el posterior cálculo de recomendaciones para cada usuario. El botón Aceptar insertará el contenido en la base de datos y cerrará la pestaña, mientras que el botón Cancelar la cerrará sin intentar realizar inserciones.

¿Cuándo viste el contenido?

21/05/2025

¿Cómo te sentías ese día?

Feliz

Puntuación:

1 2 3 4 5

Cancelar

Aceptar

Formulario de recuperación de contraseña

El usuario accede a esta vista haciendo click en el botón “Configuración” que se ubica a la izquierda de la pantalla principal. Inicialmente solo puede interactuar con el campo de texto que se encuentra más arriba y con el botón Confirmar contraseña. Hasta que no introduzca la contraseña actual del usuario, no se activarán el resto de elementos de la página que permiten terminar el proceso de cambio de contraseña.



Introduce tu contraseña actual:

Confirmar contraseña

Nueva contraseña:

Repetir nueva contraseña:

Cancelar

Confirmar

A diagram illustrating a password recovery form. It features a large rounded rectangle with a double-line border (top orange, bottom yellow) containing several input fields and buttons. Inside, there's a red-bordered area at the top left. The text "Introduce tu contraseña actual:" is in red, followed by a white input field. Below it is a dark blue button with the text "Confirmar contraseña" in white. Further down, there are two more input fields: one for "Nueva contraseña:" and one for "Repetir nueva contraseña:", both in red text. At the bottom right are two rounded rectangular buttons: a red one labeled "Cancelar" and a dark blue one labeled "Confirmar".

Vista de Administrador.

A esta pantalla únicamente tiene acceso el usuario del administrador del sistema. En esta pantalla muy parecida a la pantalla principal de la aplicación el administrador encontrará una tabla con todos los datos de los usuarios registrados en la aplicación y un formulario que le permitirá insertar nuevos usuarios y modificar los ya existentes. Es una sencilla pantalla de gestión. En la zona inferior izquierda está el botón Entrar como usuario para entrar a lo que sería la aplicación normal utilizando el usuario del administrador, por si se desean realizar pruebas.

The screenshot shows the 'Vista de Administrador' (Administrator View) of the WatchHive application. The interface has a header with the WatchHive logo and the title 'Vista de Administrador'. On the left, there's a sidebar with a red gradient background containing a button labeled 'Entrar como Usuario' (Log in as User). The main area has a yellow gradient background. At the top, there's a navigation bar with tabs: 'Usuarios' (selected), 'Citas', 'Historial', and 'Configuración'. Below the tabs is a search bar. The central part of the screen contains a table with user data:

Nombre de usuario	Nombre	Apellidos	Nacimiento	Hijos	Correo electrónico
admin	admin	admin	2/19/2000 12:00:00 AM	0	admin@admin.com
joseagi	jose angel	aguilar serrano	5/14/2025 12:00:00 AM	0	joseagiser@gmail.com

Below the table is a large input field with a placeholder 'Introduzca el nombre del paciente'. To the right of the table is a form for adding new users:

Nombre:

Apellidos:

Fecha de nacimiento: Seleccione una fecha (15)

Número de hijos: 0

Correo electrónico:

Nombre de usuario:

Contraseña:

At the bottom of the main area are three buttons: 'Añadir' (Add), 'Eliminar' (Delete), and 'Modificar' (Modify).

Pruebas De Software

Para asegurar la calidad del proyecto, se han ido realizando distintas pruebas de funcionalidad y de integración. La mayoría de pruebas han sido realizadas por mi mismo, sin embargo, he contado en dos ocasiones con la ayuda de una tercera persona ajena al proyecto para que me proporcionase ayuda realizando él la función de un usuario que no conoce el desarrollo.

Lo principal durante estas pruebas fue asegurar el correcto funcionamiento de las funcionalidades principales de la aplicación: registro de nuevos usuarios, inicio de sesión de usuarios existentes, el sistema de recomendaciones, el filtrado de los contenidos y la función de las recomendaciones.

Gracias a estas pruebas encontramos algunos errores pequeños que pudieron solucionarse a tiempo, algunos ejemplos de errores encontrados son:

- Filtrado del contenido desde “Mis listas” o “Recomendaciones”: Era una funcionalidad que nunca se planteó y tal y como estaba implementada la lógica, si el usuario utilizaba los filtros situados en la parte superior de la pantalla desde estas vistas lo que haría la aplicación sería cargar la última ventana que hubiese abierto el usuario (entre Películas o Series) y proceder a aplicar el filtro en dicha pantalla.
- Error al registrar usuarios al insertar un valor no válido en el campo “Fecha de nacimiento”
- Algunos estados de ánimo no proporcionaban recomendaciones si no existían registros previos por parte del usuario. Esto tiene relación con como maneja los géneros la API de TMDb, lo trataremos con más detalle en el apartado de **Detalles de la implementación**.

IMPLEMENTACIÓN E INTEGRACIÓN

Tecnologías

Para desarrollar el proyecto se han seleccionado tecnologías modernas y accesibles que hemos tratado también a lo largo del grado y que encajan con los requisitos funcionales del producto.

Vamos a detallar a continuación en una tabla las principales tecnologías utilizadas

Categoría	Tecnología / Herramienta	Descripción
Lenguaje de programación	C# (.NET)	Lenguaje utilizado para el desarrollo de toda la lógica de negocio y la interfaz gráfica
Entorno de desarrollo	Visual Studio 2022 Community	IDE empleado para programar, compilar y depurar el proyecto de forma eficiente
Base de datos	MySQL	Sistema de gestión de base de datos relacional para almacenar y estructurar la información
Interfaz Gráfica	Windows Forms	Tecnología integrada en .NET utilizada para la creación de las ventanas e interacción visual
API externa	TMDb (The Movie Database)	Fuente de datos para obtener información actualizada sobre películas, series y metadatos
Control de versiones	GitHub	Sistema de control de versiones distribuido para gestionar el código fuente del proyecto
Gestor de dependencias	NuGet	Utilizado para la gestión de librerías adicionales necesarias para el correcto funcionamiento
Modelado y documentación	Draw.io y PlantUML	Herramientas empleadas para diseñar diagramas (casos de uso, clases, E/R, etc.).

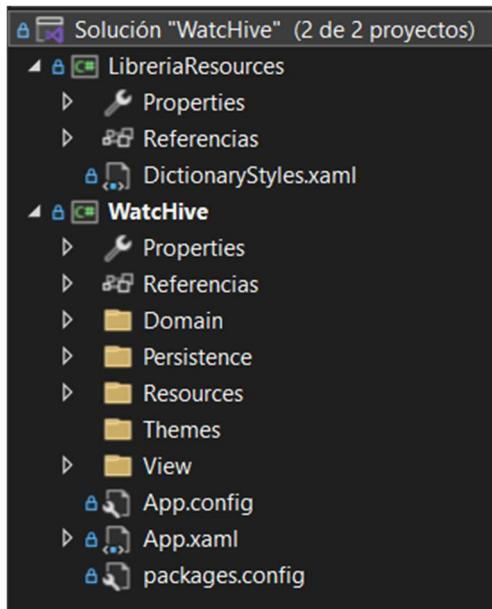
Herramientas Usadas

Hemos hecho uso de diversas herramientas de apoyo que han facilitado tanto la programación como la gestión del trabajo y el diseño de los componentes. La siguiente tabla recoja dichas herramientas.

Herramienta	Propósito
Visual Studio 2022 Community	Entorno de desarrollo principal para programar en C# y gestionar el proyecto
MySQL Workbench	Diseño y gestión de la base de datos relacional
Draw.io	Creación de diagramas UML, E/R y de casos de uso
PlantUML	Generación de diagramas de clases UML mediante lenguaje textual.
Postman	Pruebas de peticiones HTTP a la API de TMDb
Canva	Apoyo en la creación visual de interfaces y recursos gráficos

Detalles De La Implementación

Estructura general del código

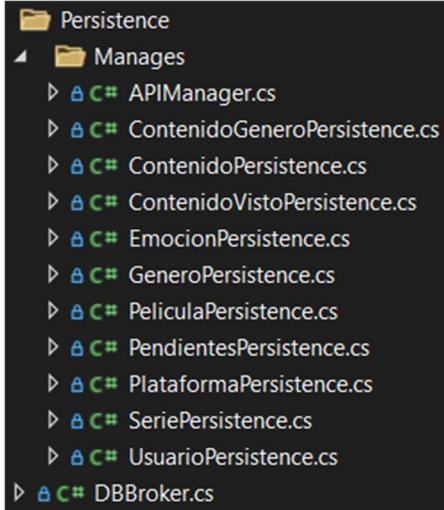


La solución esta compuesta por dos proyectos.

- **Watchive:** es el proyecto principal que implementa toda la lógica de negocio, persistencia de datos y la interfaz de usuario
- **LibreriaResources:** biblioteca compartida de recursos visuales: estilos y diccionarios de recursos XAML.

Dentro del **proyecto principal** utilizamos el patrón **MVC** (Modelo Vista Controlador) para separar en paquetes las distintas funciones a implementar.

La carpeta **Domain** contiene las clases de entidad que representan los objetos de la base de datos (Usuarios, Contenido, Película, etc.). Dentro de esta carpeta también se encuentran objetos que creamos para poder deserializar la respuesta de la API.



La carpeta **Persisten** ce implementa la lógica de acceso a datos. Aquí se ubican las clases responsables de interactuar con la base de datos y la API de TMDb. Contiene la clase **DBBroker** que es la clase encargada de realizar la conexión con la base de datos y los métodos base que utilizan el resto de clases **Manager**. La subcarpeta **Manages** contiene todas las clases Manager que se encargan del aspecto del acceso a datos correspondiente a una entidad. Esta carpeta sigue el patrón DAO (Data Access Object), favoreciendo el desacoplamiento entre la lógica de negocio y la capa de datos.

Finalmente, la carpeta **View** aloja las ventanas y controles visuales (UserControls y Windows) que forman la interfaz de usuario.

Conexión con la base de datos.

La conexión con la base de datos en la aplicación WatchHive se gestiona a través de la clase **DBBroker**. Esta clase actúa como interfaz centralizada para todas las operaciones de acceso a datos, cumpliendo el rol de una capa de persistencia desacoplada del resto de la lógica de negocio. La clase sigue el patrón **Singleton**, lo que garantiza que exista una única instancia activa de conexión a base de datos durante el ciclo de vida de la aplicación. Esto se logra a través del método estático **obtenerAgente()**.

```
39 referencias
public static DBBroker obtenerAgente()
{
    if (DBBroker._instancia == null)
    {
        DBBroker._instancia = new DBBroker();
    }
    return DBBroker._instancia;
}
```

Método leer(string sql)

```
22 referencias
public List<Object> ...leer(String sql)
{
    List<Object> resultado = new List<object>();
    List<Object> fila;
    int i;
    MySql.Data.MySqlClient.MySqlDataReader reader;
    MySql.Data.MySqlClient.MySqlCommand com = new MySql.Data.MySqlClient.MySqlCommand(sql, DBBroker.conexion);
    conectar();
    reader = com.ExecuteReader();
    while (reader.Read())
    {
        fila = new List<Object>();
        for (i = 0; i <= reader.FieldCount - 1; i++)
        {
            fila.Add(reader[i].ToString());
        }
        resultado.Add(fila);
    }
    desconectar();
    return resultado;
}
```

Método **modifier(string sql)**: Ejecuta sentencias Insert, Update o Delete

```
19 referencias
public int ...modifier(String sql)
{
    MySql.Data.MySqlClient.MySqlCommand com = new MySql.Data.MySqlClient.MySqlCommand(sql, DBBroker.conexion);
    int resultado;
    conectar();
    resultado = com.ExecuteNonQuery();
    desconectar();
    return resultado;
}
```

El DBBroker es utilizado por las distintas clases de la carpeta Persistence/Manages, como UsuarioPersistence, ContenidoPersistence, etc. **Cada una de estas clases implementa las operaciones específicas para su entidad y delega la ejecución SQL al DBBroker**

Integración con la API de TMDB

Uno de los pilares de esta aplicación es su integración con **The Movie Database (TMDB)**, una plataforma que proporciona información actualizada y estructurada sobre películas, series, géneros y plataformas de visualización. Esta integración permite que la experiencia de usuario sea mucho más rica y dinámica, ya que no se trabaja con datos estáticos locales, sino con información viva obtenida de una fuente externa de confianza.

La lógica de consumo de la API se encuentra centralizada en la clase APIManager, ubicada en el espacio de nombres WatchHive.Persistence.Manages. Esta clase encapsula todos los métodos necesarios para realizar peticiones HTTP hacia los distintos endpoints que ofrece TMDB, usando **HttpClient** como cliente HTTP y la librería **Newtonsoft.Json** para deserializar las respuestas JSON en clases DTO personalizadas.

Algunos de los métodos implementados permiten obtener películas y series populares, buscar contenido por título, filtrar por géneros o proveedores, o recuperar listas de géneros y plataformas de visualización disponibles en la región del usuario. Cada uno de estos métodos construye dinámicamente la URL de la petición en base a los parámetros requeridos (como la clave de API, idioma, géneros, etc.) y realiza la llamada de forma asíncrona.

```
public async Task<List<TMDBMovie>> GetPopularMoviesAsync()
{
    string url = $"{_baseUrl}/movie/popular?api_key={_apiKey}&language=es-ES&page=1";
    HttpResponseMessage response = await _httpClient.GetAsync(url);

    if (response.IsSuccessStatusCode)
    {
        string jsonResponse = await response.Content.ReadAsStringAsync();
        var tmdbResponse = JsonConvert.DeserializeObject<TMDBMovieResponse>(jsonResponse);
        return tmdbResponse.results;
    }

    return new List<TMDBMovie>();
}
```

Por ejemplo, el método **GetPopularMoviesAsync()** permite recuperar un listado de películas populares en el momento actual. Este realiza una solicitud, procesa la respuesta y transforma los resultados en una lista de objetos TMDBMovie.

Además, se incluyen métodos más avanzados como **GetMoviesByGenresAsync** o **GetSeriesByProviderAsync**, que permiten realizar filtros más precisos según los gustos o el contexto de uso del usuario. Estos métodos combinan múltiples parámetros en la URL para enviar consultas más específicas a TMDb, lo que permite personalizar la recomendación de contenido en función de criterios concretos.

Recomendaciones en base a emociones

Antes de nada, vamos a aclarar un punto que quedó abierto anteriormente. Los géneros que maneja la API presentan una peculiaridad entre series y películas. Algunos géneros si que se encuentran en ambas categorías como puede ser, por ejemplo, ‘Misterio’ que puede usarse para filtrar tanto películas como series, sin embargo, cada una de las categorías cuenta con **géneros específicos y equivalentes entre sí**. Es decir, el género ‘Aventura’ es exclusivo de las películas, pero en las series encontramos el género ‘Acción y Aventura’.

Para implementar esta función hemos desarrollado un sistema de aprendizaje sencillo. A diferencia de los clásicos sistemas de recomendación que se basan únicamente en histórico de visualización o popularidad, aquí se plantea una experiencia más empática con el usuario, permitiéndole descubrir contenido en función de su estado emocional actual. Resumiéndolo y en pocas palabras: La aplicación aprende de cada usuario cual es su contenido favorito para cada estado de ánimo, basándose en las veces que ha consumido un mismo género sintiéndose de una manera concreta y en las puntuaciones que le ha asignado. Aclarar que esta funcionalidad no tiene en cuenta los contenidos que el usuario agrega a su lista de pendientes, solo a los contenidos que ya ha visto.

Cuando un usuario nuevo se registra, al no existir registros previos de sus gustos se utilizará un diccionario estándar para cada una de las categorías.

```
public Dictionary<string, List<int>> emocionGenerospelis = new Dictionary<string, List<int>>
{
    { "Feliz", new List<int> { 35, 10751 } },           // Comedia, Familia
    { "Triste", new List<int> { 18, 10749 } },          // Drama, Romance
    { "Enfadado/a", new List<int> { 28, 53 } },         // Accion, Thriller
    { "Ansioso/a", new List<int> { 9648, 27 } },        // Misterio, Terror
    { "Aburrido/a", new List<int> { 12, 14 } },          // Aventura, Fantasa
    { "Relajado/a", new List<int> { 16, 10402 } },       // Animacion, Musica
    { "Motivado/a", new List<int> { 99, 80 } },          // Documental, Crimen
    { "Indefinida", new List<int> { 99, 80 } },
};

public Dictionary<string, List<int>> emocionGenerosseries = new Dictionary<string, List<int>>
{
    { "Feliz", new List<int> { 35, 10751 } },           // Comedia, Familia
    { "Triste", new List<int> { 18, 10749 } },          // Drama, Romance
    { "Enfadado/a", new List<int> { 10759, 9648 } },      // Accion, Thriller
    { "Ansioso/a", new List<int> { 9648, 18 } },         // Misterio, Terror
    { "Aburrido/a", new List<int> { 10765, 10759 } },       // Aventura, Fantasa
    { "Relajado/a", new List<int> { 16, 10751 } },       // Animacion, Musica
    { "Motivado/a", new List<int> { 99, 80 } },          // Documental, Crimen
    { "Indefinida", new List<int> { 99, 80 } },
};
```

Esto es importante ya que, cuando el usuario registre su primer contenido en su lista de vistos pueden ocurrir dos casos, supongamos que el usuario registra una película:

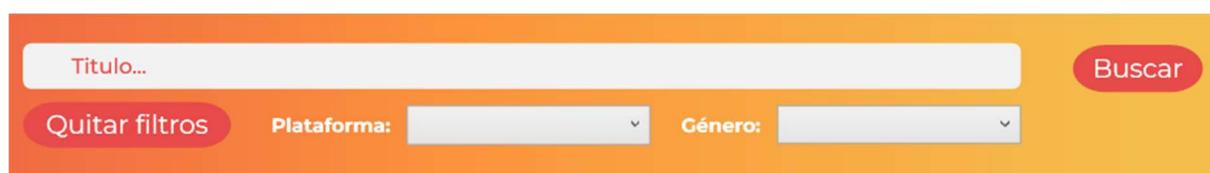
1. Uno o todos los géneros del contenido se pueden utilizar para filtrar tanto películas como series, por lo tanto, las recomendaciones para series del mismo género ya estarán personalizadas, pues existen registros que lo asocian con emociones del usuario.
2. El o los géneros son exclusivos de la categoría Películas y por lo tanto estos registros **NO se tendrán en cuenta para futuras recomendaciones** ya que no existen registros que se asocien a ningún género de Series.

El método que se encarga de realizar el calculo de las recomendaciones se encuentra en la clase **ContenidoVistoPersistence** y es el siguiente

```
2 referencias
internal List<int> recomendacionUsuario(string useranme, string emocion, string tipo)
{
    List<int> listaGenerosRecomendaciones = new List<int>();
    string query = "select g.idGenero, AVG(cv.Puntuacion) AS promedio_puntuacion," +
        " COUNT(*) AS veces_visto," +
        " AVG(cv.Puntuacion) * LOG(COUNT(*) + 1) AS relevancia " +
    "FROM ContenidoVisto cv " +
    "JOIN Emocion e ON cv.Emocion_idEmocion = e.idEmocion " +
    "JOIN ContenidoGenero cg ON cv.idContenido = cg.idContenido " +
    "JOIN Genero g ON cg.idGenero = g.idGenero WHERE cv.NombreUsuario = '" +
    +useranme+ "' AND e.NombreEmocion = '" + emocion + "' " +
    "AND (g.Tipo = '" + tipo + "' OR g.Tipo ='both'))" +
    " GROUP BY g.idGenero " +
    "ORDER BY relevancia DESC " +
    "LIMIT 3";
    List<object> lvistos = DBBroker.obtenerAgente().leer(query);
    foreach (List<object> fila in lvistos)
    {
        int idGenero = Convert.ToInt32(fila[0]);
        listaGenerosRecomendaciones.Add(idGenero);
    }
    return listaGenerosRecomendaciones;
}
```

La consulta que se puede ver en la imagen de arriba es la encargada de las recomendaciones, esta obtiene las emociones que tenia el usuario, los géneros asociados al contenido que vio y las puntuaciones que el usuario asignó. Se calcula una media de puntuación para cada género y luego se aplica la siguiente **fórmula $AVG(cv.Puntuacion) * \log(COUNT(*)) + 1$** para calcular una **relevancia ponderada**, que mezcla cantidad de visionados con valoración, filtrando todo por un Usuario, una emoción y un tipo de contenido (serie, película, o ambos, para identificar en qué tipo de contenido se puede encontrar el género).

Filtros de búsqueda



En este apartado queremos aclarar como hemos tenido que adaptar la funcionalidad de los filtros. Dado que la API posee un número limitado de llamadas cada 10 segundos, y que no existe un endpoint que nos permita filtrar por más de uno de estos criterios, optamos por limitar la función de filtrado.

Para limitarlo lo que hicimos fue que el usuario solo puede usar 1 filtro por cada búsqueda, es decir, puede filtrar o bien por título, por género o por plataforma en la que ver el contenido, para evitar que esto sea confuso para el usuario y que piense que alguno de los filtros que esta usando no se esta aplicando bien hicimos lo siguiente: Cada vez que el usuario haga click en alguno de los desplegables o en la barra de búsqueda, el resto de filtros se mostrarán vacíos en la interfaz para mostrarle al usuario que ya no se están aplicando y no se tendrán en cuenta en la lógica de filtrado. Esta funcionalidad fue la mejor solución que pudimos dar para evitar quedarnos sin llamadas a la API.

Uso de LINQ

Hemos utilizado numerosas LINQ en el proyecto, un lugar donde es muy evidente su utilización es en la vista del administrador, en la barra de búsqueda, esta utiliza una LINQ para buscar a los usuarios dentro del DataGrid.

```
var filteredList = listaUsuarios.Where(usuario =>
    usuario != null && (
        usuario.username != null && usuario.username.ToLower().Contains(searchText) ||
        (usuario.nombre != null && usuario.nombre.ToLower().Contains(searchText)) ||
        (usuario.apellidos != null && usuario.apellidos.ToLower().Contains(searchText)) ||
        (usuario.fechaNacimiento != null && usuario.fechaNacimiento.ToString("yyyy-MM-dd").Contains(searchText)) ||
        (usuario.password != null && usuario.password.ToLower().Contains(searchText)) ||
        (usuario.numHijos.ToString().Contains(searchText)) ||
        (usuario.email != null && usuario.email.ToLower().Contains(searchText))
    )
).ToList();
```

The screenshot shows a Windows application window. At the top, there is a search bar containing the text "jos". Below the search bar is a DataGrid table with the following columns: Nombre de usuario, Nombre, Apellidos, Nacimiento, Hijos, and Correo electronico. The table contains one row of data: "joseagi", "jose angel", "aguilar serrano", "5/14/2025 12:00:00 AM", "0", and "joseagiser@gmail.com".

Nombre de usuario	Nombre	Apellidos	Nacimiento	Hijos	Correo electronico
joseagi	jose angel	aguilar serrano	5/14/2025 12:00:00 AM	0	joseagiser@gmail.com

EVALUACIÓN DEL PROYECTO

Hemos cumplido todos los requisitos presentados al principio de este documento y hemos llevado a cabo un desarrollo poniendo en práctica todo lo aprendido a lo largo del ciclo formativo. Consideramos que la calidad del producto final es buena, conforme a lo esperado, y la estructura que hemos desarrollado es altamente escalable y mantenible a largo plazo. La aplicación final es sencilla e intuitiva para el usuario final y cumple con las funcionalidades que este esperaría encontrar en una aplicación como Watchive. En lo personal, el proyecto me ha demostrado las numerosas capacidades que he adquirido a lo largo de estos dos años de estudio y me encuentro realmente satisfecho con el resultado.

CONCLUSIÓN

Dificultades encontradas y soluciones aportadas.

Las principales dificultades que nos encontramos tuvieron relación con manejar las respuestas de la API, fue en una fase muy temprana del proyecto, pero tras consultar a fondo la documentación que ofrece la propia API y realizar algunas pruebas no encontramos más problemas asociados a esto.

Otra dificultad y de hecho un fallo en el desarrollo fue no tener en cuenta la creación de una clase “Helper” que contenga métodos que van a ser usados en múltiples ventanas. Esto ha llevado a que algunos fragmentos de código tengan duplicados.

Desviaciones

Siendo breves, una de las principales desviaciones que se han presentado durante el desarrollo es la eliminación de la tabla Plataformas y su relación con la tabla Contenidos. Tomamos esta decisión debido a que en muchas ocasiones el contenido cambia de plataforma tras cierto tiempo o incluso puede cambiar de plataforma si la aplicación se usase en un idioma distinto al español. Por lo tanto, decidimos obtener estos datos directamente desde la API ya que es mucho más dinámico y cualquier cambio en lo referente a la plataforma de streaming para un contenido concreto se verá inmediatamente reflejado en la aplicación.

Propuestas de mejora del proyecto

El proyecto es funcional en su estado actual pero por supuesto se podrían ampliar sus funcionalidades para ofrecer una versión mucho más pulida y completa al usuario. Algunas de las mejoras que creemos que podríamos implementar son las siguientes:

- La primera es pulir el código, creando una clase Helper que centralice los métodos que van a ser usados por más de una clase.
- Ampliación de la funcionalidad social de la aplicación, ofreciendo a los usuarios la capacidad de dejar comentarios en los contenidos que hayan visto.
- Mejora del filtrado del contenido, podría solucionarse utilizando una versión de pago de la API sin restricción de llamadas.
- Permitir más personalización al usuario.

BIBLIOGRAFIA

- Staples, J. (2024, December 19). *Why it's so hard to find something to watch lately*. New York Post. <https://nypost.com/2024/12/19/lifestyle/why-its-so-hard-to-find-something-to-watch-lately/>
- Maslansky, U. (2023, May 9). *Decision fatigue is real — Here's how it affects your work and life*. Forbes. <https://www.forbes.com/sites/forbescoachescouncil/2023/05/09/decision-fatigue-is-real-heres-how-it-affects-your-work-and-life/>
- The Movie Database. (s.f.). *TMDb*. Recuperado el 22 de mayo de 2025, de <https://www.themoviedb.org/?language=es>
- CampusMVP. (s.f.). *Introducción rápida a LINQ con C#*. Recuperado el 22 de mayo de 2025, de <https://www.campusmvp.es/recursos/post/introduccion-rapida-a-linq-con-c-sharp.aspx>
- YouTube. (2023, 24 enero). *WPF. Recursos. Estilos [04-25] | Aprende C# desde cero hasta lo avanzado* [Video]. <https://www.youtube.com/watch?v=jttZvtveoMg&t=2700s>