



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA



Escola Tècnica
Superior d'Enginyeria
Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Diseño e implementación de una aplicación web para una red social de películas

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: Héctor Lendrino Muñoz

Tutor: Miguel Rebollo Pedruelo

2016-2017

Agradecimientos

Agradezco todo el apoyo recibido del profesorado de la escuela técnica superior de ingeniería informática como a mis familiares y amigos.

A mi profesor y tutor Miguel Rebollo por todo el tiempo y recursos dedicados.

A mi madre y a mi padre por todo el cariño y ánimos en estos últimos años de carrera.

A la familia 2.0 y en especial a Fati por todo el apoyo que me has dado.

Resumen

Estudio y desarrollo de una aplicación web de temática cinematográfica y utilizando el conjunto de tecnologías MEAN, MongoDB para la gestión de la base de datos, Express que facilita la comunicación en la aplicación, AngularJS para una mayor comodidad para la parte del cliente y Node.JS que ofrece las principales funcionalidades para el proyecto.

Palabras clave: API, MEAN stack, HTML5, CSS3, Javascript, MongoDB

Resum

Estudi i desenvolupament d'una aplicació web de temàtica cinematogràfica i utilitzant el conjunt de tecnologies MEAN, MongoDB per a la gestió de la base de dades, Express que facilita la comunicació en l'aplicació, AngularJS per a una major comoditat per a la part del client i Node.JS que oferix les principals funcionalitats per al projecte.

Palabras clave: API, MEAN stack, HTML5, CSS3, Javascript, MongoDB

Abstract

Study and development about web application with a cinematographic theme and using MEAN stack, MongoDB for the management of the database, express that makes it easier the communication on the application, AngularJS for greater convenience for the customer and the last one Node.JS for the main functionalities in the project.

Keywords: API, MEAN stack, HTML5, CSS3, Javascript, MongoDB

Índice de contenido

Índice de tablas.....	9
Índice de figuras	10
1. Introducción.....	12
1.1 Motivación	12
1.2 Objetivos	12
1.3 Estructura del proyecto	13
2. Entorno	15
2.1 Introducción	15
2.2 Entorno de realización.....	15
2.3 Sistemas similares	16
2.3.1 DCine	16
2.3.2 IMDb.....	17
2.3.3 Filmaffinity.....	18
2.3.4 MovieHaku	19
2.4 Análisis de sistemas similares	20
2.4.1 Análisis cuantitativo	20
2.4.2 Análisis cualitativo	21
2.5 Síntesis.....	22
2.6 Tecnología a emplear	22
3. Especificación de requisitos	24
3.1 Introducción	24
3.1.1 Propósito	24
3.1.2 Ámbito del sistema	24
3.1.3 Definiciones, acrónimos y abreviaturas	24
3.2 Descripción general	25
3.2.1 Perspectiva de la aplicación.....	25
3.2.2 Características de los usuarios	25
3.2.3 Funciones de la aplicación.....	25
3.2.4 Restricciones	26
3.2.5 Suposiciones y dependencias	26
3.2.6 Requisitos futuros	27



3.3	Requisitos específicos	27
3.3.1	Requisitos específicos.....	27
3.3.2	Atributos del sistema.....	29
3.3.3	Modelo de negocio canvas	30
4.	Diseño.....	31
4.1	Arquitectura del sistema.....	31
4.2	Arquitectura de tres capas	31
4.2.1	Capa de persistencia	31
4.2.2	Capa de negocio	32
4.2.3	Capa de presentación	33
5.	Implementación y evaluación	37
5.1	Tecnología utilizada.....	37
5.1.1	HMTL	37
5.1.2	CSS.....	37
5.1.3	Javascript.....	38
5.1.4	MongoDB.....	38
5.1.5	Express	38
5.1.6	AngularJS	38
5.1.7	Node.JS.....	39
5.2	Preparación del entorno	39
5.2.1	ATOM	39
5.2.2	Estructura de ficheros	39
5.2.3	TheMovieDB API.....	42
5.2.4	Programación, inicialización y pruebas	43
6.	Conclusiones	50
6.1	Resumen	50
6.2	Dificultades y soluciones	50
6.3	Cambios futuros.....	51
6.4	Valoración personal	51
7.	Referencias.....	53
7.1	Internet	53
7.2	Bibliográficas	54

Índice de tablas

Tabla 1 – Tabla de análisis cuantitativo.....	20
Tabla 2 – Tabla de análisis cualitativo.....	21
Tabla 3 – Requisito específico RF01.....	27
Tabla 4 – Requisito específico RF02.....	27
Tabla 5 – Requisito específico RF03.....	28
Tabla 6 – Requisito específico RF04.....	28
Tabla 7 – Requisito específico RF05.....	28
Tabla 8 – Requisito específico RF06.....	28
Tabla 9 – Requisito específico RF07.....	28
Tabla 10 – Requisito específico RF08.....	29

Índice de figuras

Ilustración 1 – Cabecera de DCine.....	16
Ilustración 2 – Que ofrece DCine.....	16
Ilustración 3 – Cabecera IMDb.....	17
Ilustración 4 – Que ofrece IMDbPro.....	17
Ilustración 5 – Cabecera FilmAffinity.....	18
Ilustración 6 – Diseño adaptativo FilmAffinity.....	18
Ilustración 7 – Cabecera MovieHaku.....	19
Ilustración 8 – Que ofrece MovieHaku.....	19
Ilustración 9 – Diagrama de casos de uso.....	26
Ilustración 10 – Diagrama de flujo.....	31
Ilustración 11 – Esquema base de datos.....	32
Ilustración 12 – Diagrama de flujo usuario registrado.....	32
Ilustración 13 – Diagrama de flujo de la búsqueda de una película.....	33
Ilustración 14 – Menú principal de la aplicación.....	33
Ilustración 15 – Ventana de registro.....	34
Ilustración 16 – Ventana de identificación.....	34
Ilustración 17 – Ventana del usuario.....	35
Ilustración 18 – Ficha técnica de la película.....	35
Ilustración 19 – Flujo de ventanas.....	36
Ilustración 20 – Logo HTML5.....	37
Ilustración 21 – Logo CSS3.....	37
Ilustración 22 – Logo JavaScript.....	38
Ilustración 23 – Logo MongoDB.....	38
Ilustración 24 – Logo Express.....	38
Ilustración 25 – Logo AngularJS.....	38
Ilustración 26 – Logo NodeJS.....	39

Ilustración 27 – Logo Atom.....	39
Ilustración 28 – Estructura del proyecto.....	39
Ilustración 29 – Conexión a la base de datos local.....	40
Ilustración 30 – Archivo package.js.....	40
Ilustración 31 – Archivo server.js.....	41
Ilustración 32 – TheMovieDb.....	42
Ilustración 33 – Clave API.....	42
Ilustración 34 – JSON de ejemplo.....	42
Ilustración 35 – Script para cargar angular y el controlador.....	43
Ilustración 36 – Modelo de usuario.....	44
Ilustración 37 – Archivo .bat.....	45
Ilustración 38 – Esperando conexión BD.....	45
Ilustración 39 – Inicio Node Shell.....	45
Ilustración 40 – Conexión establecida.....	45
Ilustración 41 – Ventana principal con el buscador.....	46
Ilustración 42 – Ventana de registro.....	46
Ilustración 43 – Ventana de registro exitoso.....	47
Ilustración 44 – Ventana de perfil.....	47
Ilustración 45 – Buscando una película.....	48
Ilustración 46 – Ventana de detalles de una película.....	48
Ilustración 47 – Lista de mis películas.....	49



1. Introducción

1.1 Motivación

Vivimos en un mundo rodeado de tecnología en el cual la facilidad para acceder a un contenido de entretenimiento es cada vez mayor. Actualmente se genera a diario una gran cantidad de películas y series de televisión para todos los públicos y es habitual el comentar un capítulo o una película tras el visionado de ella.

Por desgracia pocos son los sitios web donde puedes llevar el seguimiento de tus películas a la vez que hay una comunidad activa y en castellano donde poder “charlar” sobre el último capítulo de la serie de moda o la película más esperada del año.

Por ello se han pensado en el desarrollo de una aplicación web en la que los usuarios puedan realizar lo anteriormente comentado.

1.2 Objetivos

Este proyecto se centra principalmente en la creación de una aplicación web de películas donde poder llevar un seguimiento de las películas.

Los objetivos principales del proyecto son:

- Aprender nueva tecnología para desarrollar, en un futuro, nuevas aplicaciones.
- Aprender cómo desarrollar un proyecto de estas características, mediante el estudio de aplicaciones similares y la aplicación de estándares como el IEE830 (Especificación de requisitos software).
- Aprender como estructurar y organizar un proyecto de estas características teniendo en cuenta las limitaciones temporales.
- Desarrollar un aplicación

1.3 Estructura del proyecto

Los puntos en los que está estructurado el proyecto son los siguientes:

Capítulo 2 - Entorno de desarrollo

Se centrará en un análisis de aplicaciones similares para averiguar las bases de las aplicaciones de este tipo y averiguar cómo destacar del resto.

Capítulo 3: Especificación de requisitos

Utilización del estándar IEEE830 para realizar una especificación formal y con una buena base antes de empezar el desarrollo.

Capítulo 4 - Diseño

Antes de ponerse con el desarrollo de la aplicación se realizara un diseño con bocetos para poder evitar posibles errores en la implementación.

Capítulo 5 - Implementación

En este punto se desarrolla la aplicación teniendo en cuenta los puntos anteriores junto a una explicación de la tecnología utilizada.

Capítulo 6 - Conclusiones

Se extraen las conclusiones, los problemas encontrados a lo largo del desarrollo junto a las soluciones encontradas, los cambios que se podrían aplicar en un futuro y una valoración personal de lo que ha supuesto este proyecto.



2. Entorno

2.1 Introducción

En 2003 el americano Dale Dougherty acuñó el término web 2.0, cuya principal misión en la participación de los usuarios.

La Real Academia Española define una red social como una plataforma digital de comunicación global que pone en contacto a un gran número de personas.

En 1991, Col Needham, el desarrollador de una de las webs de películas más importantes (IMDb) publicó una serie de herramientas, shell scripts, para consultar las listas que usuarios creaban para ver las listas sobre actores y actrices y sus películas. Este acontecimiento fue la creación de webs sobre películas.

Las webs siguieron evolucionando, la llamada web 1.5 tienen su aparición en 1997 y su principal característica es que son páginas web construidas a partir de una o varias bases de datos, donde los usuarios apenas tenían participación. En esta época IMDb ya ofrecía una interfaz para la consulta de películas y listas mediante correo electrónico.

En 2003 aparece la web 2.0 gracias a Dale Dougherty donde la interacción con el usuario es cada vez mayor. IMDb ya contaba en esta época con los comentarios, las votaciones y las listas para ver que películas eran las más valoradas. La participación del usuario empezaba a ser importante.

Actualmente vivimos en la época de las redes sociales, cualquier aplicación web tiene que permitir al usuario relacionarse con otras personas obligando a cualquier aplicación web a tener características para la interacción de usuarios. No poner ninguna de estas características supone un riesgo bastante elevado. Por ejemplo MovieHaku otra web de películas se define como “Una red social de cine en la que podrás encontrar información, recibir recomendaciones y conocer nuevos amigos” donde las opiniones de la gente son más importantes que los datos técnicos de la película.

2.2 Entorno de realización

Este proyecto une la interacción del usuario junto a la necesidad de entretenimiento, en este caso una web 2.0 de películas.

Analizar sistemas similares es uno de los principales objetivos de este apartado para poder realizar un correcto desarrollo, conseguir un valor extra para destacar del resto y convertir el proyecto en un modelo de negocio rentable.

2.3 Sistemas similares

2.3.1 DCine

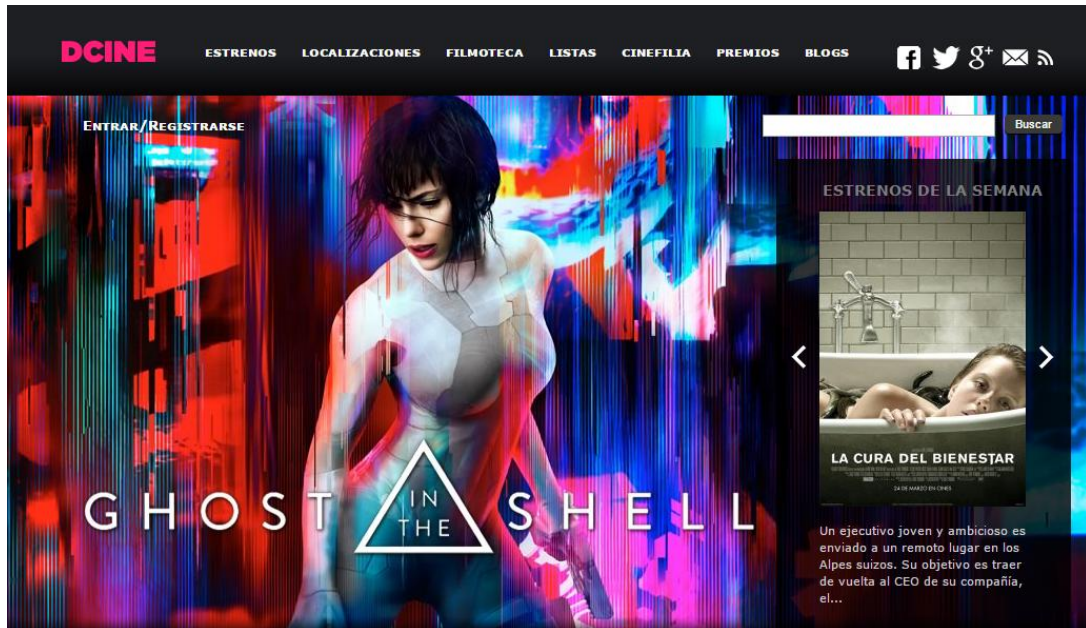


Ilustración 1 - Cabecera Dcine

Dirección web: <http://www.dcine.org/>

Idioma: Castellano

DCine nació en 2004 fundada por dos amigos amantes del cine, definen la web como “un juguete con el que poder divertirnos” quieren que los usuarios sean los dueños y protagonistas casi en absoluto de los contenidos.



Ilustración 2 - Qué ofrece Dcine

No solo se centra en mostrar información sobre los últimos estrenos o sobre las películas más valoradas, quiere hacerse un hueco y diferenciarse del resto de redes sociales. Ofrece al usuario la posibilidad de descubrir las localizaciones de rodaje, escenas y frases memorables de sus películas favoritas.

2.3.2 IMDb

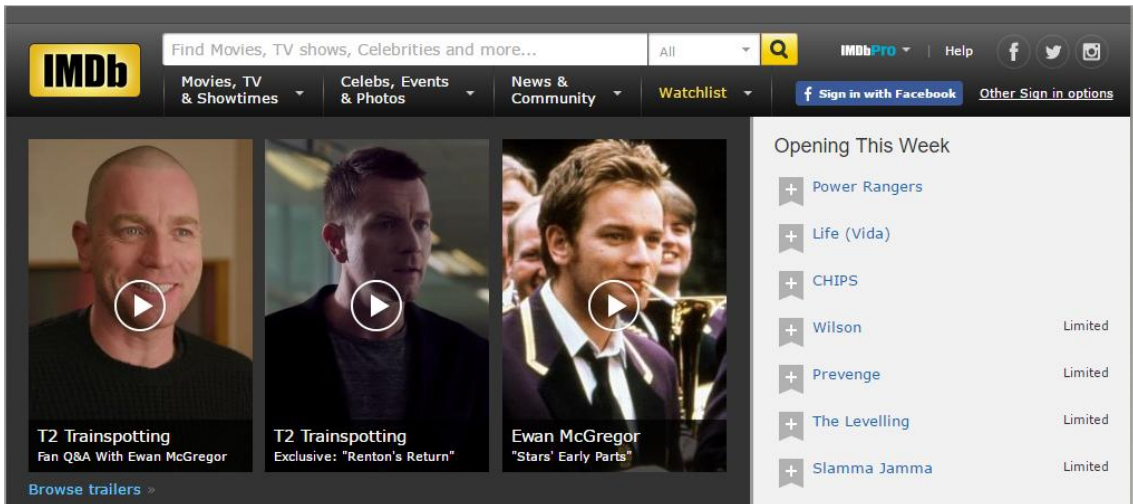


Ilustración 3 - Cabecera IMDb

Dirección web: <http://www.imdb.com/>

Idioma: Inglés

IMDb (Internet Movie Database) cuyo significado en español es Base de datos de películas en Internet, creada en 1990 y comprada por Amazon en 1998.

Los usuarios pueden obtener información sobre películas y series, próximos estrenos y la programación de canales de televisión. Algunas de estas opciones no están disponibles en todos los países.

Cuenta con una aplicación para el móvil, un diseño web adaptativo y una versión de pago mensual denominada IMDb Pro diseñada para los usuarios que se dedican a la industria.

IMDbPro

Industry professionals rely on IMDbPro every day

Check filmographies & credits

Access profile pages for more than 4 million people and cast and crew listings for more than 2 million titles

Showcase yourself on IMDb & Amazon

Manage your photos and the credits you are Known For on IMDbPro, IMDb, and Amazon Video

Find industry contacts & talent representation

Stay connected with contact details for over 200,000 industry professionals and companies

Build shareable lists

Leverage powerful advanced search tools to easily create lists of titles and people

Track in-development & production titles

Keep up-to-date on over 25,000 projects not available on IMDb

Monitor industry trends

Check STARmeter & MOVIEmeter rankings for every person and title on IMDb

Ilustración 4 - Qué ofrece IMDbPro

2.3.3 Filmaffinity



Ilustración 5 - Cabecera Filmaffinity

Dirección web: <http://www.filmaffinity.com/es/main.html>

Idioma: Castellano e inglés

Filmaffinity fue creada en el año 2002, ofrece gran información de documentales, cortometrajes, series y películas.

En 2016 comenzó a internacionalizarse, para ajustarse mejor a los usuarios de distintos países, aparte de Filmaffinity España tienen sede en USA, UK, México, Argentina, Chile y Colombia.

No dispone de una aplicación móvil pero si tiene un diseño web adaptativo para los navegadores de los móviles.

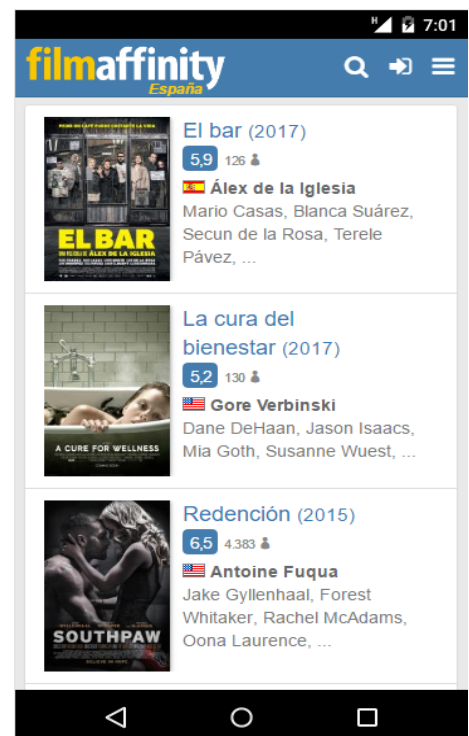


Ilustración 6 - Diseño adaptativo Filmaffinity

2.3.4 MovieHaku



Ilustración 7 - Cabecera MovieHaku

Dirección web: <http://moviehaku.com/>

Idioma: Castellano

MovieHaku desarrollada por 37pixels, empresa de 3 miembros dedicada al desarrollo web.

Se definen como “red social de cine que combina la participación de sus usuarios con la información de personas y películas para crear relaciones relevantes.”

El objetivo principal es tener una gran base de datos de películas y que la participación de los usuarios no se limite a críticas y discusiones.



Ilustración 8 - Que ofrece MovieHaku

2.4 Análisis de sistemas similares

2.4.1 Análisis cuantitativo

	DCine	IMDb	Filmaffinity	MovieHaku
Aplicación móvil	✗	✓	✗	✗
Buscador	✓	✓	✓	✓
Autocompletar búsqueda	✗	✓	✓	✓
Cartelera	✗	✗	✓	✗
Conexión segura	✗	✓	✗	✗
Diseño web adaptativo	✗	✓	✓	✗
Información de los cines	✗	✗	✓	✗
Listas	✓	✓	✓	✓
Próximos estrenos	✓	✓	✓	✓
Redes sociales (Facebook, Twitter...)	✓	✓	✓	✓
Subscripción mensual	✗	✓	✗	✗
Tiempo de carga con publicidad(segundos)*	1.72(±0.34)*	4.86(±0.43)*	4.57(±0.83)*	1.68(±0.31)*
Tiempo de carga sin publicidad(segundos)*	1.19(±0.23)*	3.80(±0.26)*	2.81(±0.08)*	1.23(±0.23)*
Tiempo de búsqueda(segundos)*	2.82(±0.19)*	1.72(±0.14)*	1.34(±0.10)*	2.02(±0.16)*

*Los tiempos se muestran como la media ± la desviación

Tras la creación de la tabla de análisis cuantitativo, se puede observar que la aplicación a desarrollar va a tener un buscador, listas de películas, próximos estrenos y otras redes sociales. IMDb y FilmAffinity son las aplicaciones más completas y por tanto será webs de referencia a la hora de diseñar la aplicación. Lo único negativo de estas web es su gran tiempo de carga debido a la gran cantidad de publicidad.

Por último destacar que IMDb es la única con una subscripción mensual, la cual, permite a los usuarios obtener características exclusivas orientadas a un tema más profesional y nada que ver con el entretenimiento que la aplicación busca.

2.4.2 Análisis cualitativo

	DCine	IMDb	Filmaffinity	MovieHaku
Facilidad para buscar información	★	★★	★★★★	★★★
Intuitivo	★★	★	★★★★	★★★
Registro sencillo	★★★	★★★★	★★	★

Tras probar las distintas webs seleccionadas se ha llegado a las conclusiones de la tabla de análisis cualitativo donde 4 estrellas es la máxima puntuación y por el contrario una estrella es la menor.

En el análisis cuantitativo se ha comentado qué IMDb es una de las aplicaciones referentes pero evaluando sus características cualitativas se ha llegado a la conclusión de que es una aplicación más profesional, orientada a ofrecer mucha información sobre las películas, equipos de producción, lugares de rodaje, etc. Para registrarse se debe introducir una gran cantidad de datos y toda la información se muestra en pantalla generando al usuario nuevo una dificultad elevada para encontrar la información que busca. El idioma es otro de los problemas ya que IMDb está completamente en inglés.

Filmaffinity es la clara ganadora del análisis, idioma en castellano, fácil de encontrar la información lo único que falla es la gran cantidad para el registro.

2.5 Síntesis

Tras realizar un estudio de mercado, analizando las aplicaciones similares, se ha detectado algunas características que comparten todas las webs, como son el registro de usuarios, interacción con otras redes sociales y una lista con los próximos estrenos. También cabe destacar que solo una de las webs analizadas tiene versión para teléfonos inteligentes.

El tema de la publicidad puede perjudicar los tiempos de carga como se muestra en la tabla de características cualitativas, puede convertirse en un cuello de botella y ralentizar demasiado la aplicación.

En conclusión, las características comunes necesarias en la nueva web, la aplicación móvil es una característica a tener en cuenta pero se ha decidió utilizar, en un principio, un buen diseño adaptativo para evitar que los usuarios gasten memoria de sus dispositivos para instalar la aplicación.

2.6 Tecnología a emplear

La tecnología a emplear va a ser distinta según la capa en la que nos encontremos:

JavaScript se ha convertido en la herramienta más potente para el desarrollo web y el conjunto de herramientas MEAN (MongoDB, ExpressJS, AngularJS, NodeJS) ha superado a LAMP (Linux, Apache, MySQL, PHP) debido a su gran integración, facilidad y cubriendo todas las áreas del proyecto (Base de datos, servidor y cliente) además de una gran cantidad de ejemplos y tutoriales.

Para la capa de navegador usaremos HTML Y CSS.



3. Especificación de requisitos

3.1 Introducción

3.1.1 Propósito

En este apartado se identificará y clasificará los requisitos del futuro sistema para poder analizar las necesidades funcionales del mismo.

Esta aplicación web va dirigida a cualquier usuario que quiera información sobre películas desde el título original al reparto y equipo de producción.

3.1.2 Ámbito del sistema

El nombre de la aplicación es: “HighLight Movie”.

El objetivo principal de la aplicación web es mostrar información de cualquier película estrenada o por estrenar junto a la participación del usuario para llevar un seguimiento de sus películas, ya sea película vista, favorita, pendiente de ver o directamente descartada.

Otros objetivos a desarrollar en un futuro, para la evolución de la aplicación, serían el poder obtener información de los cines, obtener ofertas de su cine favorito, obtener información sobre los distintos festivales y premios de cine.

La meta de este sistema es llegar a ser un modelo de negocio rentable, obtener beneficios y convertirse en una de las aplicaciones de cine más utilizadas.

3.1.3 Definiciones, acrónimos y abreviaturas

- **API (*Application Programming Interface*):** Es el conjunto de funciones y subrutinas que nos ofrece la web TheMovieDB para obtener toda la información sobre películas, estrenos y listas.
- **Sistema:** Referente a la aplicación web desarrollada.
- **Usuario premium:** Usuario con una suscripción de pago y accede a características únicas.
- **Modelos:** Es la estructura creada para el usuario que será almacenada en la base de datos y contendrá toda su información.

3.2 Descripción general

3.2.1 Perspectiva de la aplicación

El sistema es una aplicación web independiente, es decir, requiere la utilización de una API externa para obtener la información de las películas. Desarrollada de manera individual y con una descripción detallada en este mismo documento.

3.2.2 Características de los usuarios

En el sistema existen dos tipos de usuarios, el usuario no registrado cuyas funciones son básicas y el usuario registrado con acceso a todas las funcionalidades del sistema.

3.2.3 Funciones de la aplicación

Para facilitar el uso de las funcionalidades, cada una de ellas estará acompañada de una etiqueta que la identificara como caso de uso.

Para un usuario no registrado (UNR):

- Buscar información sobre cualquier película(UNR-CU01)
- Ver la cartelera de su país(UNR-CU02)
- Ver la información de los cines(UNR-CU03)
- Registrarse en la aplicación(UNR-CU04)

Para un usuario registrado (UR).

- Buscar información sobre cualquier película(UR-CU01)
- Ver la cartelera de su país(UR-CU02)
- Ver la información de los cines(UR-CU03)
- Identificarse en la aplicación(UR-CU04)
- Agregar amigos(UR-CU05)
- Llevar un registro de sus películas(UR-CU06)
 - Película vista(UR-CU06-1)
 - Película favorita(UR-CU06-2)
 - Película pendiente(UR-CU06-3)
 - Película descartada(UR-CU06-4)
- Votar una película(UR-CU07)
- Añadir un comentario(UR-CU08)
- Editar mis datos (UR-CU09)



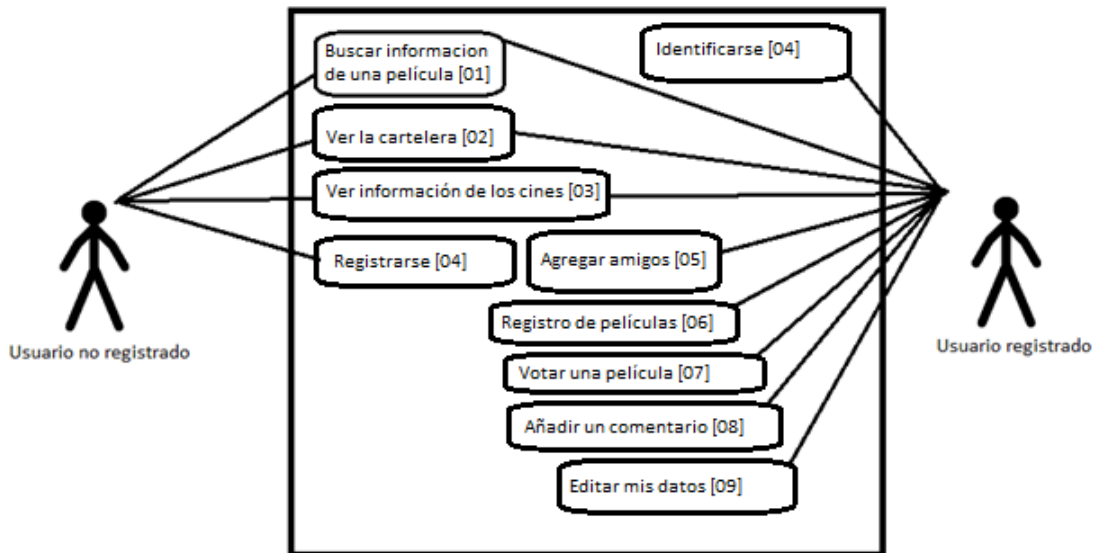


Ilustración 9 - Diagrama de casos de uso

En la ilustración 9 podemos observar el diagrama con los dos actores de nuestra aplicación.

3.2.4 Restricciones

Será necesario un ordenador con acceso a internet, con el que desarrollar el sistema y mantener el servidor activo.

En el tema de seguridad hay que tener en cuenta que el cifrado de los datos de autenticación de los usuarios es necesario.

3.2.5 Suposiciones y dependencias

La utilización de una API para la obtención de información de películas es una dependencia directa, la modificación en las subrutinas que generan las fichas de información puede hacer que se dejen de mostrar correctamente en la aplicación. Una continua revisión en esta dependencia puede evitar futuros fallos.

3.2.6 Requisitos futuros

Algunas mejoras que podría recibir el sistema son:

- Mantenimiento de la aplicación y la actualización de las versiones de los programas utilizados en su desarrollo, conforme vayan sacando nuevas versiones para evitar el software obsoleto.
- Implementación de un modelo freemium, con suscripción de pago con características exclusivas para los usuarios a cambio de la obtención de beneficios.
- La creación de una propia base de datos de películas para ofrecerla a otras aplicaciones web como API.
- La integración de la aplicación con sistemas de compra de películas, como por ejemplo Amazon o Netflix, para facilitar al usuario la compra de sus películas.

3.3 Requisitos específicos

3.3.1 Requisitos específicos

En este punto se explicarán con un mayor nivel de detalle los requisitos para que el diseño pueda realizarse de la mejor manera posible. Cada uno de los requisitos contiene su nombre, un número de referente (RFxx), el caso de uso, una descripción y una prioridad.

Nombre	Diseño web adaptativo
Referencia	RF01
Caso de uso	-
Prioridad	Alta
Descripción	El sistema debe ser correctamente visible desde cualquier dispositivo con un navegador.

Nombre	Darse de alta en la web
Referencia	RF02
Caso de uso	UNR-CU04 UR-CU05
Prioridad	Alta
Descripción	Poder formar parte del sistema como usuario registrado.

Nombre	Buscar una película y acceder a su ficha
Referencia	RF03
Caso de uso	UNR-CU01 UR-CU02
Prioridad	Alta
Descripción	Obtener la información de cualquier película.

Nombre	Añadir una película a cualquiera de las listas
Referencia	RF04
Caso de uso	UR-Cu06
Prioridad	Media
Descripción	Añadir una película a la lista de vistas, favoritas, pendientes o descartadas.

Nombre	Eliminar una película de cualquiera de las listas
Referencia	RF05
Caso de uso	UR-Cu06
Prioridad	Media
Descripción	Poder eliminar una película de la lista de vistas, favoritas, pendientes o descartadas.

Nombre	Configurar mi perfil de usuario
Referencia	RF06
Caso de uso	UR-CU09
Prioridad	Media
Descripción	Poder editar los datos de mi perfil, como puede ser el nombre, el email o cambiar la contraseña.

Nombre	Poder votar una película
Referencia	RF07
Caso de uso	UR-CU07
Prioridad	Baja
Descripción	Poder dar una nota a una película entre 0 – 10 estrellas.

Nombre	Poder añadir un comentario a una película
Referencia	RF08
Caso de uso	UR-CU08
Prioridad	Media
Descripción	Escribir una opinión sobre cualquier película y que sea público al resto de la comunidad.

3.3.2 Atributos del sistema

- Disponibilidad: Garantiza que cuando el usuario quiera el sistema estará disponible.
- Interacción: Los usuarios serán capaces de comunicarse con el resto de la comunidad.
- Fiabilidad: El sistema tendrá que pasar por una serie de test unitarios para garantizar el correcto comportamiento de sus funcionalidades.
- Mantenibilidad: El sistema funciona correctamente y está en constante mantenimiento.
- Portabilidad. El usuario podrá acceder al sistema desde cualquier dispositivo que disponga de un navegador y acceso a internet.
- Seguridad: Los datos de los usuarios se encuentran almacenados, protegidos y cifrados.



-

3.3.3 Modelo de negocio canvas

Canvas es una tabla donde se puede mostrar los principales aspectos de un modelo de negocio. Se ha realizado el canvas para este sistema con el objetivo de aclarar las ideas del proyecto para convertirlo en un futuro un negocio rentable:

- Socios clave
 - Cines (Incluyendo promociones propias para los usuarios)
- Actividades clave
 - Comunidad activa (Interacción con otros usuarios)
 - Control de películas (Registro de tus películas favoritas)
 - Información sobre cines cercanos (Sin necesidad de ir al cine)
- Recursos clave
 - Web (Accesible desde cualquier dispositivo)
 - Publicidad
- Propuestas de valor
 - Comunidad activa en castellano (Actualmente hay muy pocas)
 - Eventos cinematográficos (Nacionales e internacionales con toda la información)
- Relaciones con clientes
 - Soporte técnico (Resolver cualquier duda o problema)
 - Creación de una gran y agradable comunidad
- Segmentos de cliente
 - Gente que le gusta el cine
 - Gente que quiera llevar el control de sus películas
- Canales
 - Otras redes sociales
 - Internet
- Estructura de costes
 - Desarrollo y mantenimiento de la web
 - Marketing online
- Fuentes de ingresos
 - Publicidad
 - Modelo Freemiun (Características exclusivas y sin publicidad)

La competencia es bastante fuerte pero tras analizar el canvas se puede hacer un hueco en el mercado.

4. Diseño

4.1 Arquitectura del sistema

En la ilustración 10 el diagrama muestra el flujo que va a seguir la aplicación:

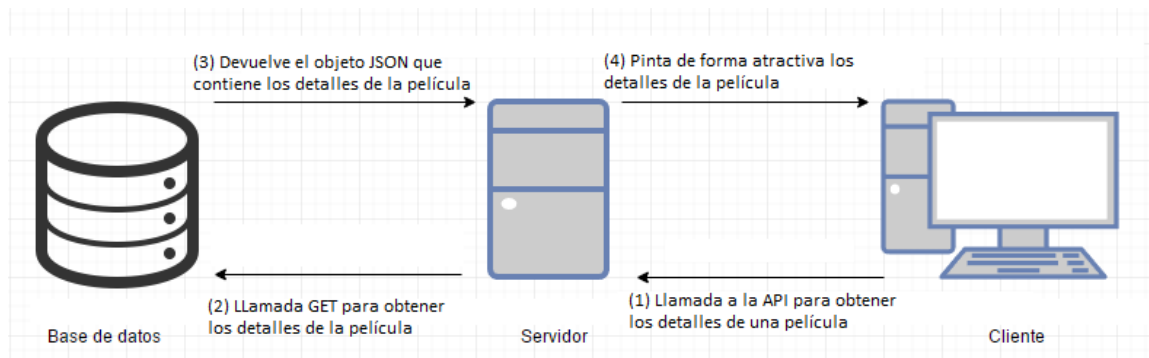


Ilustración 10 - Diagrama de flujo

4.2 Arquitectura de tres capas

El sistema está basado en el modelo Cliente/Servidor. Consiste en que los clientes realizan peticiones de información y el servidor es el encargado de responder.

Se utiliza este sistema para que un cliente no pueda acceder a datos a los que no está autorizado ya que los accesos, recursos y la integridad de los datos son controlados por el servidor.

Utilizaremos la arquitectura de tres capas ya que se pretende conseguir un sencillo desarrollo y futuro mantenimiento de la aplicación.

4.2.1 Capa de persistencia

Es la encargada de los datos, será donde se gestione todo lo relacionada a la base de datos y a la creación, edición y borrado de datos. La base de datos está compuesta por las tablas del diagrama de clases que se puede observar en la ilustración 11

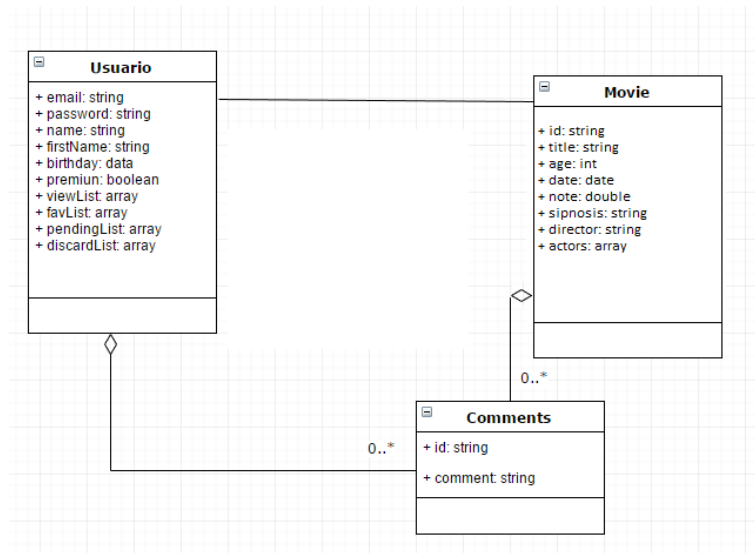


Ilustración 11 - Esquema base de datos

4.2.2 Capa de negocio

Esta capa se ocupa de comunicarse con la capa de persistencia para almacenar o recuperar datos y con la capa de presentación para atender las peticiones y presentar los resultados. En la ilustración 12 se puede observar el flujo que va a seguir un usuario no registrado para darse de alta en la web.

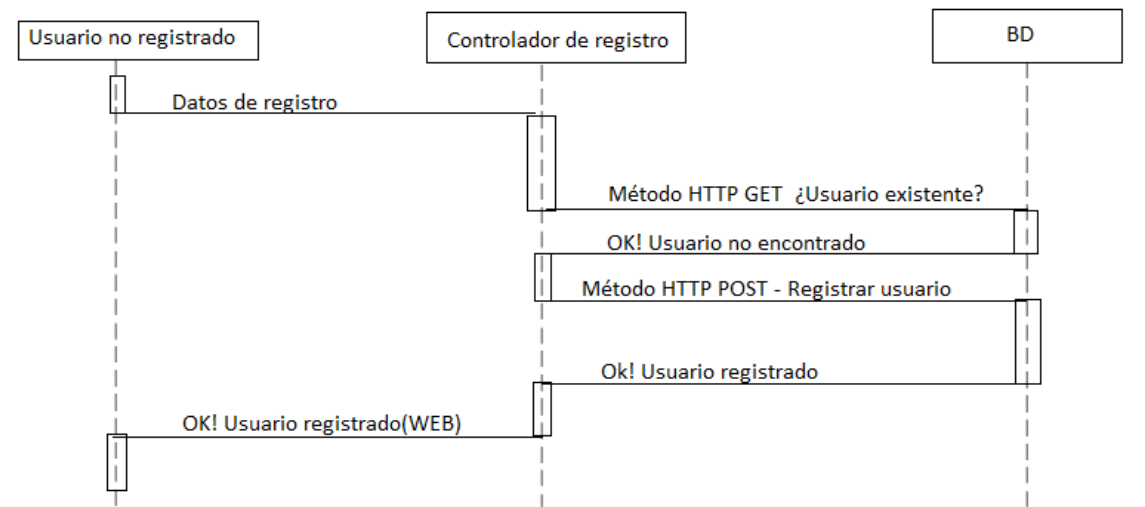


Ilustración 12 - Diagrama de flujo usuario registrado

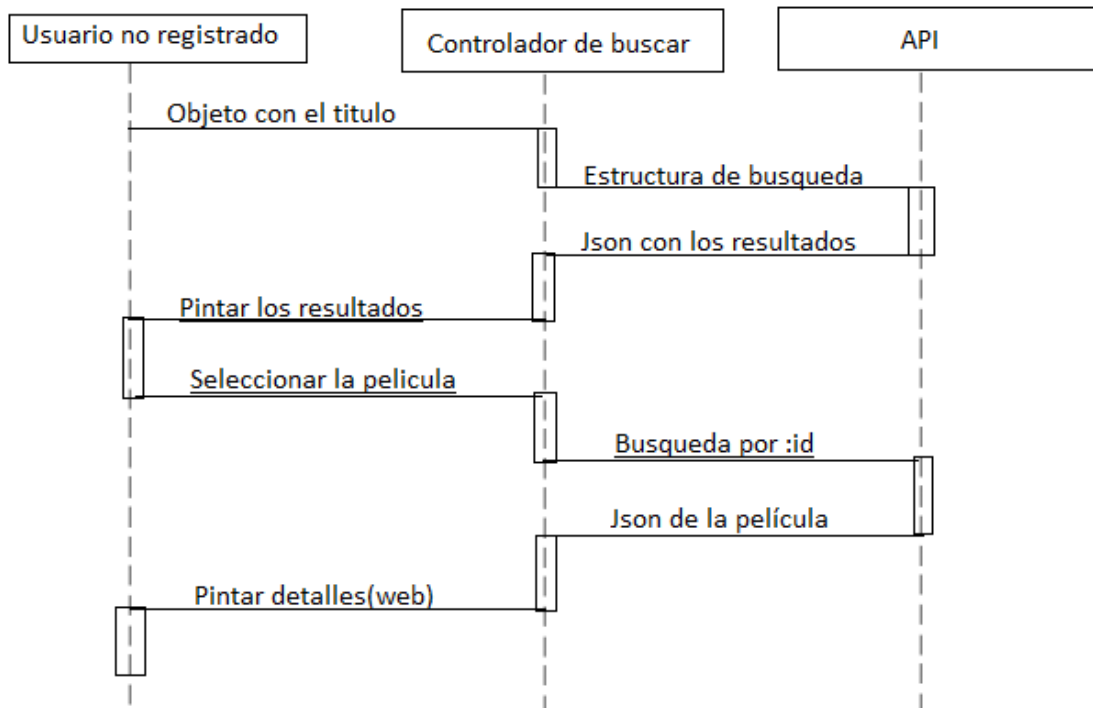


Ilustración 13 - Diagrama de flujo de la búsqueda de una película

En la ilustración 13 podemos ver el diagrama de flujo de la búsqueda de una película y sus detalles. El usuario buscará una película por su título, el controlador lo recogerá y lo enviará a la API, devolviendo los datos en forma de objeto JSON el cual el controlador se encargará de tratar y pintar correctamente en la página correspondiente (html).

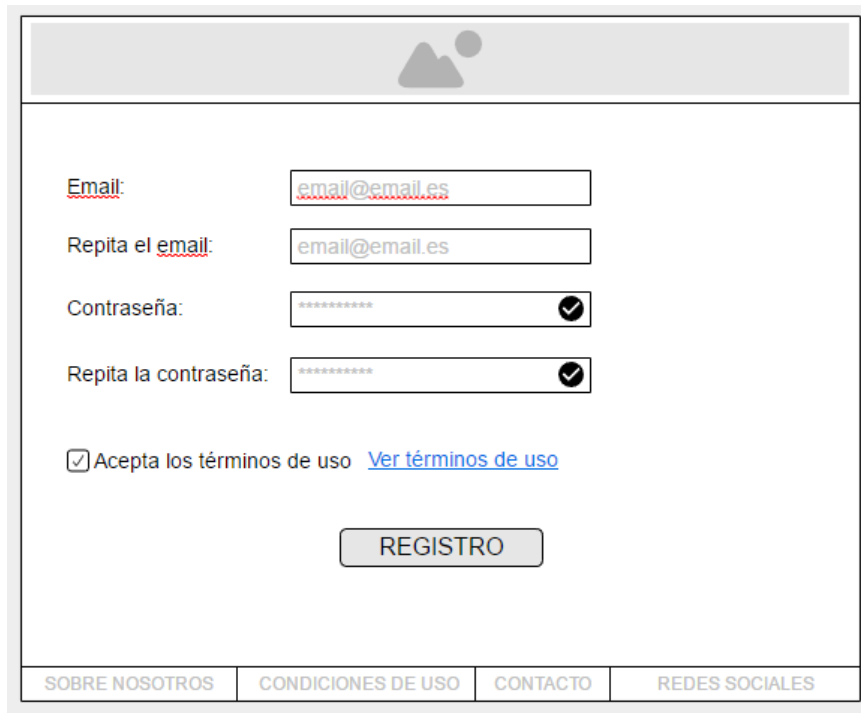
4.2.3 Capa de presentación

La representación de la información para el usuario final, interactuando con él y comunicándose únicamente con él a nivel de aplicación.

Las ilustraciones 14 – 18 muestran los bocetos de la aplicación para orientar el desarrollo de la capa de presentación:



Ilustración 14 - Menú principal de la aplicación



The registration form is contained within a light gray border. At the top center, there is a placeholder for a profile picture, represented by a gray silhouette of a person's head and shoulders. Below this, the form consists of several input fields and a checkbox. The 'Email' field contains the text 'email@email.es'. The 'Repita el email' field also contains 'email@email.es'. The 'Contraseña' field contains a series of asterisks and a checkmark icon. The 'Repita la contraseña' field also contains asterisks and a checkmark icon. Below these fields is a checkbox labeled 'Acepta los términos de uso' with a blue link 'Ver términos de uso' next to it. At the bottom center of the form is a button labeled 'REGISTRO'. At the very bottom of the page, there is a horizontal navigation bar with four links: 'SOBRE NOSOTROS', 'CONDICIONES DE USO', 'CONTACTO', and 'REDES SOCIALES'.

Ilustración 15 - Ventana de registro



The login form is contained within a light gray border. At the top center, there is a placeholder for a profile picture, represented by a gray silhouette of a person's head and shoulders. Below this, the form features a large black circular icon representing a user profile. Underneath the icon are two input fields: the first contains 'email@email.es' and the second contains a series of asterisks. Below the password field is a checkbox labeled 'Recuérdame'. At the bottom center of the form is a button labeled 'ENTRAR'. At the very bottom of the page, there is a horizontal navigation bar with four links: 'SOBRE NOSOTROS', 'CONDICIONES DE USO', 'CONTACTO', and 'REDES SOCIALES'.

Ilustración 16 - Ventana de identificación

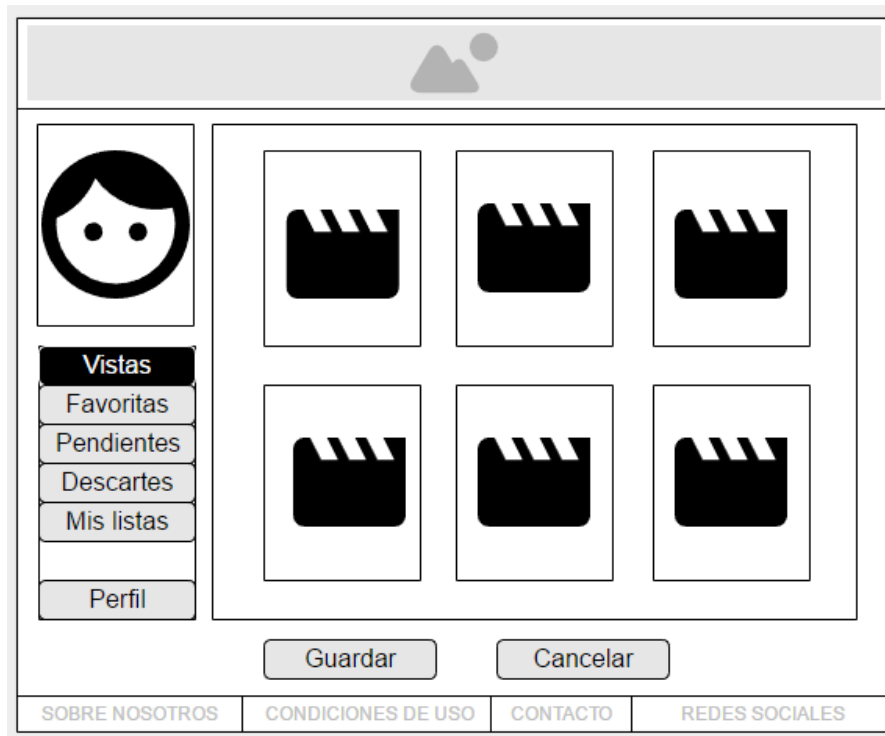


Ilustración 17 - Ventana del usuario



Ilustración 18 - Ficha técnica de la película

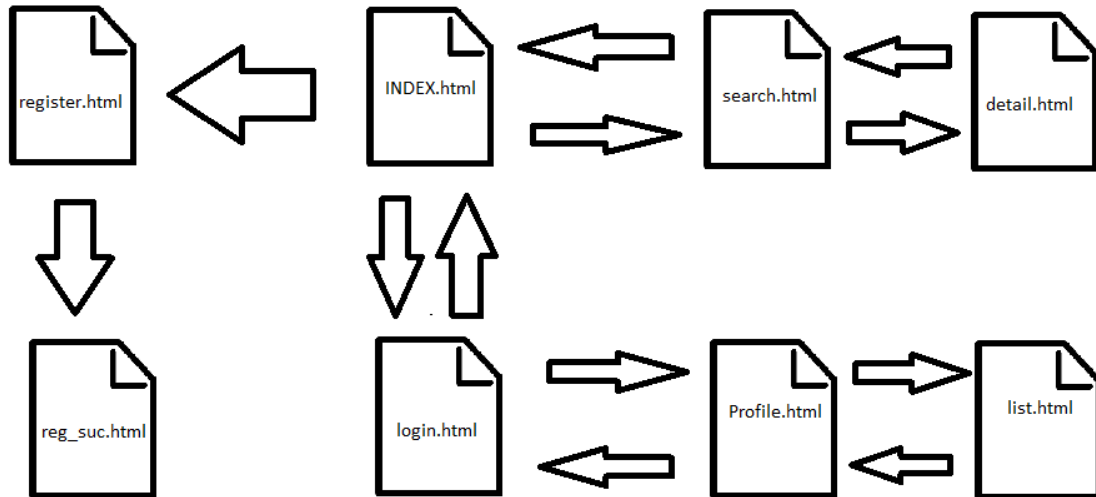


Ilustración 19 - Flujo de ventanas

Los bocetos servirán de guía a la hora de desarrollar la aplicación como puede ser la forma de estructurar la página principal (Ilustración 14), las vistas de registro y login correspondientes a las ilustraciones 15 y 16.

Para finalizar podemos observar cómo quedaría el flujo de ventanas (Ilustración 19) y junto al capítulo 4.2 Arquitectura en tres capas ha facilitado el proceso de inicio de la aplicación.

5. Implementación y evaluación

5.1 Tecnología utilizada

Las tecnologías a utilizar son, el cuadro de herramientas de trabajo MEAN, acrónimo inglés para MongoDB, Express, AngularJS y Node.JS, un conjunto software utilizado para el desarrollo web dinámicas basadas en JavaScript. Y para el apartado de front-end utilizaremos HTML5 y CSS3.

5.1.1 HTML



Ilustración 20 - Logo HTML5

El lenguaje más importante y conocido para la creación de páginas web, dándoles una estructura y un contenido. Basado en etiquetas, breves instrucciones de apertura y cierre.

HTML5 es la última versión de HTML, la cual incluye una nueva estructura que facilita la forma de generar contenido, etiquetas como <article>, <footer>, <header>, <nav> y <section> facilitan mucho la estructura de la información en la aplicación.

También trae mejoras sobre la etiqueta input, pudiendo indicar si es un url, un email, un entero, etc.

También destacar que al ser la última versión incluirá la corrección de errores en etiquetas y comportamientos que podrían afectar al correcto funcionamiento de la aplicación

5.1.2 CSS



Ilustración 21 - Logo CSS3

Al igual que HTML no es considerado un lenguaje de programación sino un lenguaje de estilo, separa el contenido de la forma y permite un mayor control sobre la apariencia.

CSS3 es la última versión de CSS, incluye nuevos efectos, con una mayor facilidad de aplicación y conseguir que la parte visual de nuestros documentos sean más llamativos como son los bordes o el sombreado.

Y la corrección de errores de antiguas versiones.

5.1.3 Javascript



Ilustración 22 - Logo JavaScript

JavaScript es un lenguaje de programación orientado a objetos utilizado principalmente para la creación de web dinámicas del lado del cliente y en ocasiones del lado del servidor.

El aumento de las APIs lo ha convertido en uno de los lenguajes de programación más utilizados en el mundo y el adecuado para desarrollar la aplicación.

5.1.4 MongoDB



mongoDB

Ilustración 23 - Logo MongoDB

Base de datos NoSQL, almacena los datos en una estructura llave-valor. Básicamente los valores de estas llaves, llamadas colecciones, son estructura tipo JSON también llamados documentos.

Creado para ofrecer escalabilidad, rendimiento y gran disponibilidad.

5.1.5 Express

express

Ilustración 24 - Logo Express

Módulo de NodeJS de alto nivel de desarrollo web que ofrece un conjunto de características para las aplicaciones web.

5.1.6 AngularJS



ANGULARJS

Ilustración 25 - Logo Angular

Es un framework modelo-vista-controlador de javascript para el desarrollo web front-end, es decir para la parte del cliente. Facilita la manipulación del DOM (Modelo de Objetos de Documento).

Permite una crear, de forma sencilla, una gran variedad de efectos consiguiendo que la cantidad de código disminuya considerablemente, a la vez que incrementa la capacidad y la facilidad para mantener dicho código.

5.1.7 Node.JS



Ilustración 26 - Logo NodeJS

Entorno en tiempo de ejecución multiplataforma encargado del funcionamiento de la capa del servidor, basado en el lenguaje de programación JavaScript y orientado a eventos asíncronos.

5.2 Preparación del entorno

5.2.1 ATOM

Editor de texto desarrollado por Github, la principal característica es, según los creadores, que es totalmente “hackeable”, es decir, la capacidad de configuración y modificación es muy elevada.



Ilustración 27 - Logo Atom

Una ventaja muy importante para este proyecto es la gran integración con NodeJS, que facilita la utilización de librerías.

5.2.2 Estructura de ficheros

```
1 - app/  
  ---- models/  
  ----- user.js  
  ---- routes.js  
2 - config/  
  ----- database.js  
3 - node_modules/  
4 - public/  
  ----- controllers/  
  ----- css/  
  ----- images/  
  ----- core.js  
  ----- index.html  
5 - package.json  
6 - server.js
```

Ilustración 28 - Estructura del proyecto

La primera carpeta que nos encontramos según la ilustración número 25 es la app.

Contiene el archivo de rutas y los modelos, en el caso de este proyecto, solo es necesario el modelo del usuario (archivo user.js) ya que el resto de modelos como podría ser la película nos son necesarios ya que la base de datos nos la proporciona la API.

El modelo de usuario guardará todos los datos relacionados con él como su nombre, sus apellidos, su fecha de nacimiento, su email, su contraseña, etcétera.

Por otro lado nos encontramos con el fichero routes.js es el encargado de realizar todos los métodos GET, POST y DELETE de HTTP, ya sea para comprobar la autenticación (Método GET), crear un nuevo usuario (Método POST) o borrar una película de la lista (Método DELETE).



La segunda carpeta es la de config, para este proyecto solo necesitamos el archivo database.js y es el encargado de realizar la conexión con la base de datos que para este proyecto está en local, en la siguiente ilustración podemos observar como realiza la conexión.

```
1 module.exports = {
2   url : 'mongodb://localhost:27017/HLM_DB'
3 }
```

Ilustración 29 - Conexión a la base de datos local

La tercera carpeta es node_modules, se crea la primera vez que se enciende la aplicación y es la encargada de almacenar todas las librerías y dependencias utilizadas por el proyecto.

La cuarta es public, contiene todos los archivos para el frontend de nuestra aplicación, es la que el cliente se descarga y a la que puede tener acceso, por tanto no es muy conveniente dejar ahí archivos que puedas perjudicar la integridad y la seguridad de la aplicación.

Contiene todos los archivos .html y sus respectivos .css (Carpeta css/), controladores (Carpeta controllers/) y las imágenes utilizadas para la web (Carpeta images/).

Por último nos quedan los archivos más importantes el package.json y el server.js

```
1 {
2   "name"      : "HLM-Project",
3   "version"   : "0.0.0",
4   "description" : "Proyecto final de carrera",
5   "main"      : "server.js",
6   "author"    : "Héctor Lendrino Muñoz",
7   "dependencies" : {
8     "express" : "~4.7.2",
9     "mongoose" : "latest",
10    "morgan" : "~1.2.2",
11    "body-parser": "~1.5.2",
12    "method-override": "~2.1.2"
13  }
14 }
```

Ilustración 30 - Archivo package.json

Muestra una información sobre el proyecto e incluye todas las dependencias y sus versiones que van a ser utilizadas por el proyecto.


```

1 // set up =====
2 var express = require('express');
3 var app     = express();           // Crear nuestra aplicacion con express
4 var mongoose = require('mongoose'); // mongoose para mongodb
5 var port    = process.env.PORT || 8080; // Establecer el puerto de conexión
6 var database = require('./config/database'); // Cargar la configuracion de la base de datos
7 var morgan = require('morgan');     // registro por consola
8 var bodyParser = require('body-parser'); // Información de HTML POST (express4)
9 var methodOverride = require('method-override'); // Simular los metodos DELETE and PUT (express4)
10 // configuracion =====
11 mongoose.connect(database.url);     // Conectar a la base de datos local
12 app.use(express.static(__dirname + '/public'));
13 app.use(morgan('dev'));             // Mostrar cada log en consola
14 app.use(bodyParser.urlencoded({ 'extended': 'true' }));
15 app.use(bodyParser.json());
16 app.use(bodyParser.json({ type: 'application/vnd.api+json' }));
17 app.use(methodOverride());
18 // routes =====
19 require('./app/routes.js')(app);
20 // listen =====
21 app.listen(port);
22 console.log("App listening on port " + port);

```

Ilustración 31 - Archivo server.js

Dentro de este archivo podemos distinguir 3 partes, la declaración de las dependencias, la configuración de cada una de ellas y por último el arranque de la aplicación.

Una vez configurado todo el proyecto y antes de empezar con la programación debemos instalar las dependencias.

Hay dos formas de hacerlo, la primera es ejecutando en el Shell de node el siguiente comando:

npm install

Buscará en el archivo package.json (ilustración 30) e instalará todas las dependencias, en la carpeta node_modules (creada automáticamente con la ejecución del comando).

La otra opción es con el comando:

node server.js

Instalará todas las dependencias y si no hay errores, en dependencias como en el código de programación, iniciará la aplicación, quedando la aplicación escuchando a cualquier petición de usuarios (línea 21 de la ilustración 31).



5.2.3 TheMovieDB API



Ilustración 32 - TheMovieDB

Para la obtención de información, se va a utilizar un servicio web, conocido como API, que permite realizar una búsqueda sencilla de información sobre una gran base de datos de películas.

En el caso de este proyecto, usaremos la API que ofrece The Movie DB. Para poder trabajar se requiere una cuenta, el registro es totalmente gratuito. Una vez registrados y autenticados necesitaremos una clave API, que la web nos ofrece en Settings/API. Para la realización de este proyecto usaremos la siguiente:

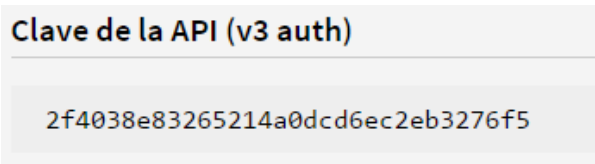


Ilustración 33 - Clave API

Además la web nos ofrece un ejemplo de uso con nuestra clave:

https://api.themoviedb.org/3/movie/550?api_key=2f4038e83265214a0dcd6ec2eb3276f5

Si utilizamos el anterior link en el navegador, nos muestra un objeto JSON que podremos tratar en el controlador de javascript para luego poder pintar la información en el HTML.

```
{
  "adult": false,
  "backdrop_path": "/87hTDIay2N2qWyX4Ds7ybXi9h8I.jpg",
  "belongs_to_collection": null,
  "budget": 63000000,
  "genres": [
    {
      "id": 18,
      "name": "Drama"
    }
  ],
  "homepage": "http://www.foxmovies.com/movies/fight-club",
  "id": 550,
  "imdb_id": "tt0137523",
  "original_language": "en",
  "original_title": "Fight Club",
  "overview": "A ticking-time-bomb insomniac and a slippery soap salesman channel primal male aggression into a shocking new form of therapy. Their concept catches on, with underground \"fight clubs\" forming in every town, until an eccentric gets in the way and ignites an out-of-control spiral toward oblivion.",
  "popularity": 7.709882,
  "poster_path": "/adw6Lq9FiC9zjVEpOqfq03ituwp.jpg",
  "production_companies": [
    {
      "name": "Regency Enterprises",
      "id": 508
    },
    {
      "name": "Fox 2000 Pictures",
      "id": 711
    },
    {
      "name": "Taurus Film",
      "id": 20555
    },
    {
      "name": "Linson Films",
      "id": 54050
    },
    {
      "name": "Atman Entertainment",
      "id": 54051
    },
    {
      "name": "Knickerbocker Films",
      "id": 54052
    }
  ],
  "production_countries": [
    {
      "iso_3166_1": "DE",
      "name": "Germany"
    },
    {
      "iso_3166_1": "US",
      "name": "United States of America"
    }
  ],
  "release_date": "1999-10-15",
  "revenue": 100853753,
  "runtime": 139,
  "spoken_languages": [
    {
      "iso_639_1": "en",
      "name": "English"
    }
  ],
  "status": "Released",
  "tagline": "Mischief. Mayhem. Soap.",
  "title": "Fight Club",
  "video": false,
  "vote_average": 8.199999999999999,
  "vote_count": 8240
}
```

Ilustración 34 - JSON de ejemplo

5.2.4 Programación, inicialización y pruebas

Una vez configurada la estructura de la aplicación hace falta darle la forma al proyecto. Podemos distinguir, el diseño de los archivos html y sus correspondientes controladores (parte de cliente) y los controladores internos y la conexión a la base de datos (parte servidor).

Por un lado están los archivos html, son los archivos que el usuario ve (modificados por los css) y con los que interactúa. Su principal función es mostrar la información (recibida de los controladores) que el usuario está solicitando. Para que la conexión con los controladores sea correcta hay que incluir cargar angular (Para nuestro proyecto se usará la versión 1.0.8) y el archivo javascript (.js) que contendrá el controlador:

```
<script src="//ajax.googleapis.com/ajax/libs/angularjs/1.0.8/angular.min.js"></script>
<script src="filmController.js"></script>
```

Ilustración 35 - Script para cargar angular y el controlador

Y por otro, los controladores el funcionamiento principal es, recoger los datos de la solicitud generada por el usuario, comunicarse con el servidor o con la API (dependiendo de cual se la tarea a realizar) y una vez recibe los datos (que pueden ser lo que ha solicitado el usuario o mensaje de error) pues se encarga de enviarlos a los archivos html.

Angular facilita mucho la función de los controladores. La utilización de angular en html es bastante sencilla. La siguiente etiqueta:

```
<body ng-app="app">
```

Indica que la etiqueta body y todo su contenido va a ser una aplicación AngularJS y con la etiqueta:

```
<div class="container" ng-controller="filmCtrl">
```

Indicamos que va a ser un controlador de tipo AngularJS y su archivo contendrá la siguiente estructura:

```
var app = angular.module('app', [])
app.controller('filmCtrl', function($scope, $http, $location) {
//Contenido del controlador filmCtrl
});
```



En la parte de servidor y la base de datos:

Para la creación de modelos y establecer la conexión con la base de datos local se utiliza el módulo Mongoose. La conexión se establece en el archivo `server.js` (ilustración 31 líneas 4 y 11).

```
var mongoose = require('mongoose');  
mongoose.connect(database.url);
```

El contenido de `database.url` se encuentra en el fichero `config/database.js` y contiene:

```
module.exports = { url: 'mongodb://localhost:27017/HLM_DB' }
```

La utilización de un fichero distinto es para mayor facilidad a la hora de tratar la conexión cuando en un futuro la base de datos de la aplicación se mantenga en un host de pago y por tanto no local.

Una vez establecida la conexión entre base de datos y NodeJs podemos crear los modelos, esta aplicación solo requiere un modelo, el usuario.

```
1 var mongoose = require('mongoose');  
2  
3 module.exports = mongoose.model('User', {  
4   name      : String,  
5   lastname  : String,  
6   email     : String,  
7   password  : String,  
8   birth     : date  
9 });
```

Ilustración 36 - Modelo de usuario

Todos los atributos de usuario que se quieran almacenar, se deberán añadir en el modelo usuario.

Para las rutas se ha creado el fichero `routes.js` (ilustración 31 línea 19):

```
require('./app/routes.js')(app);
```

El archivo contendrá las rutas con los siguientes métodos (HTTP):

- GET – Obtener datos (Ver mi lista de películas favoritas)
- POST – Crear un nuevo elemento (Crear un nuevo usuario)
- DELETE – Eliminar un elemento (Eliminar una película de mi lista)
- UPDATE – Cambiar un elemento (Cambiar mi email o contraseña)

Los pasos para arrancar la aplicación son los siguientes:

Paso 1 – Arrancar la base de datos desde un .bat

```
1 @echo off
2 cd "C:\mongo\bin"
3 mongod.exe -dbpath="C:\Users\Hector\Desktop\Proyecto\MongoDB\db"
4 exit
```

Ilustración 37 - Archivo .bat

Las líneas 1 (@echo off) y 4(exit) de la ilustración 32 indican donde comienza y acaba nuestro archivo .bat, la línea número 2 es para colocarnos en el directorio donde está instalado mongo y la línea 3 para ejecutar mongo con la base de datos de nuestro proyecto.

Para ejecutar el .bat simplemente daremos doble click sobre él. Abrirá un Shell que muestra que la base de datos está esperando una conexión.

```
2017-06-26T20:42:18.105+0200 I NETWORK [thread1] waiting for connections on port 27017
```

Ilustración 38 - Esperando conexión BD

Acto seguido abrimos la terminal de node nos colocamos en la carpeta raíz del proyecto y ejecutamos el comando: **node server.js**

```
C:\Users\Hector\Desktop\Proyecto\HighlightMovie>node server.js
App listening on port 8080
```

Ilustración 39 - Inicio Node Shell

Automáticamente nos pone que la aplicación está escuchando en el puerto 8080 y si miramos en el Shell de MongoDB observamos que se ha iniciado una conexión.

```
2017-06-26T20:42:18.105+0200 I NETWORK [thread1] waiting for connections on port 27017
2017-06-26T20:49:03.648+0200 I NETWORK [thread1] connection accepted from 127.0.0.1:62018 #1 (1 connection now open)
2017-06-26T20:49:03.653+0200 I NETWORK [conn1] received client metadata from 127.0.0.1:62018 conn1: { driver: { name:
"nodejs", version: "2.2.26" }, os: { type: "Windows_NT", name: "win32", architecture: "x64", version: "10.0.14393" },
platform: "Node.js v6.10.3, LE, mongodb-core: 2.1.10" }
```

Ilustración 40 - Conexión establecida

Con esto podemos comprobar que nuestra aplicación web está funcionando, para ello debemos situarnos en el navegador y en la URL introducir: <http://localhost:8080/>

La ilustración 39 nos muestra la ventana principal de nuestra aplicación, con el buscador, el botón de login para identificarnos en la web si ya estamos registrados o por el contrario el botón de registro (Para ir a la ilustración 40) en la cual deberemos introducir nuestros datos y darle al botón de “Dame de alta” si todos los datos son introducidos correctamente nos saldrá la ventana de la ilustración 41 y ya podremos acceder a nuestra cuenta y a nuestro perfil (Ilustración 42).

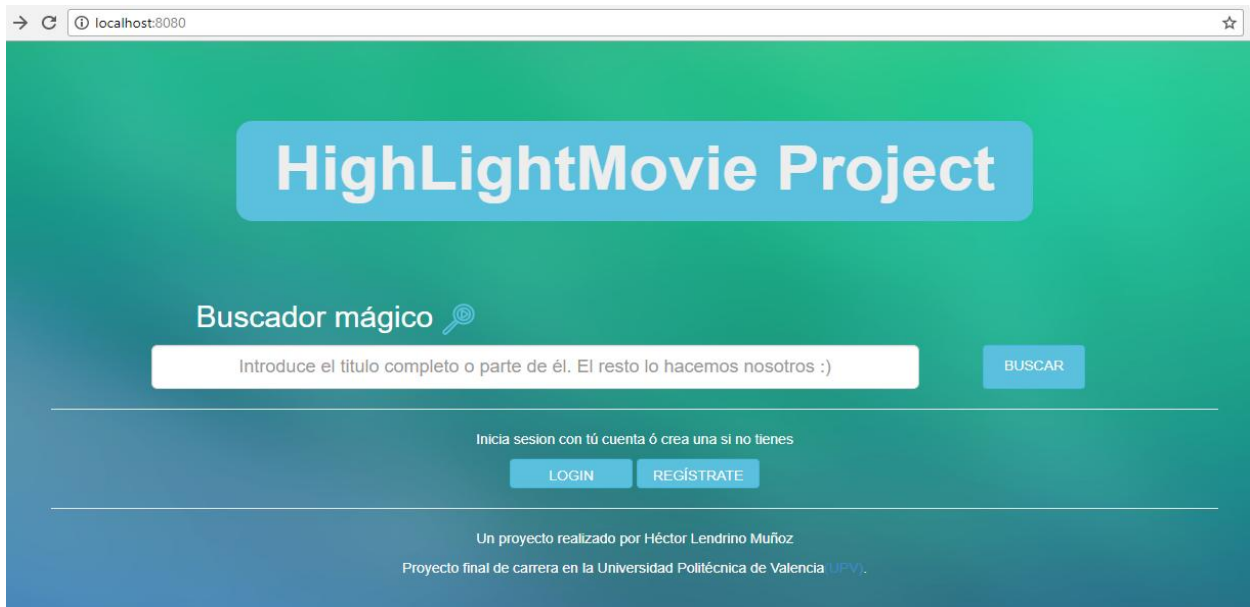


Ilustración 41 – Ventana principal con el buscador

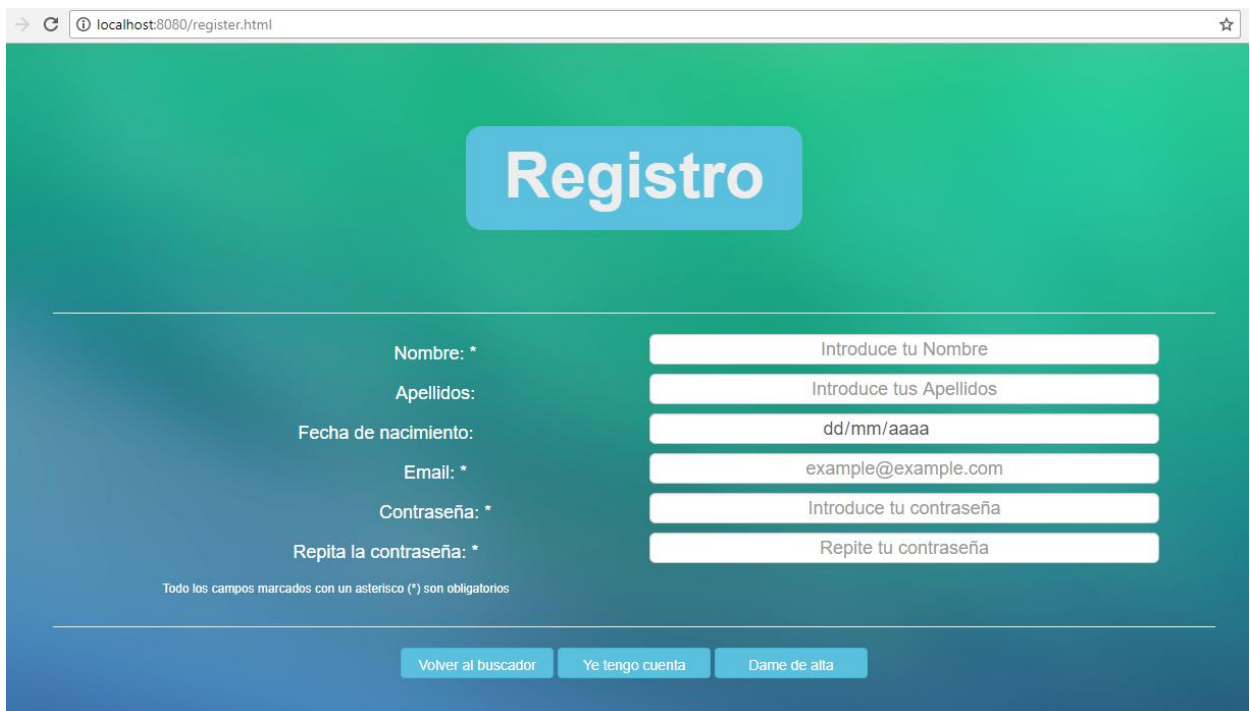


Ilustración 42 - Ventana de registro

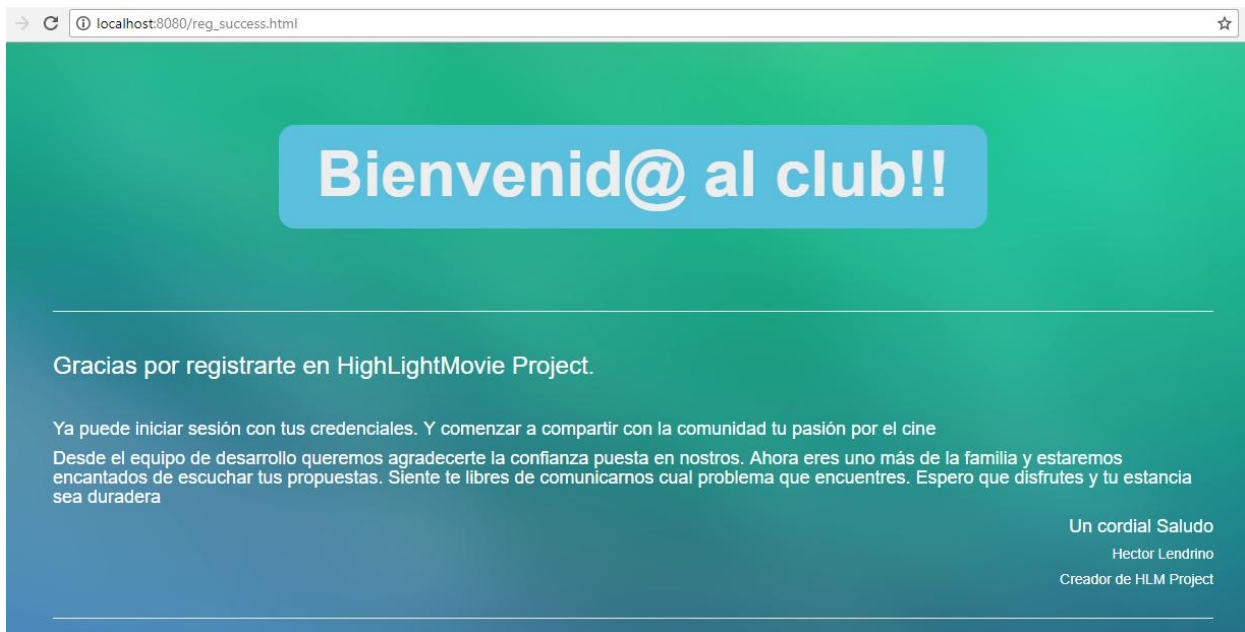


Ilustración 43 - Venta de registro exitoso

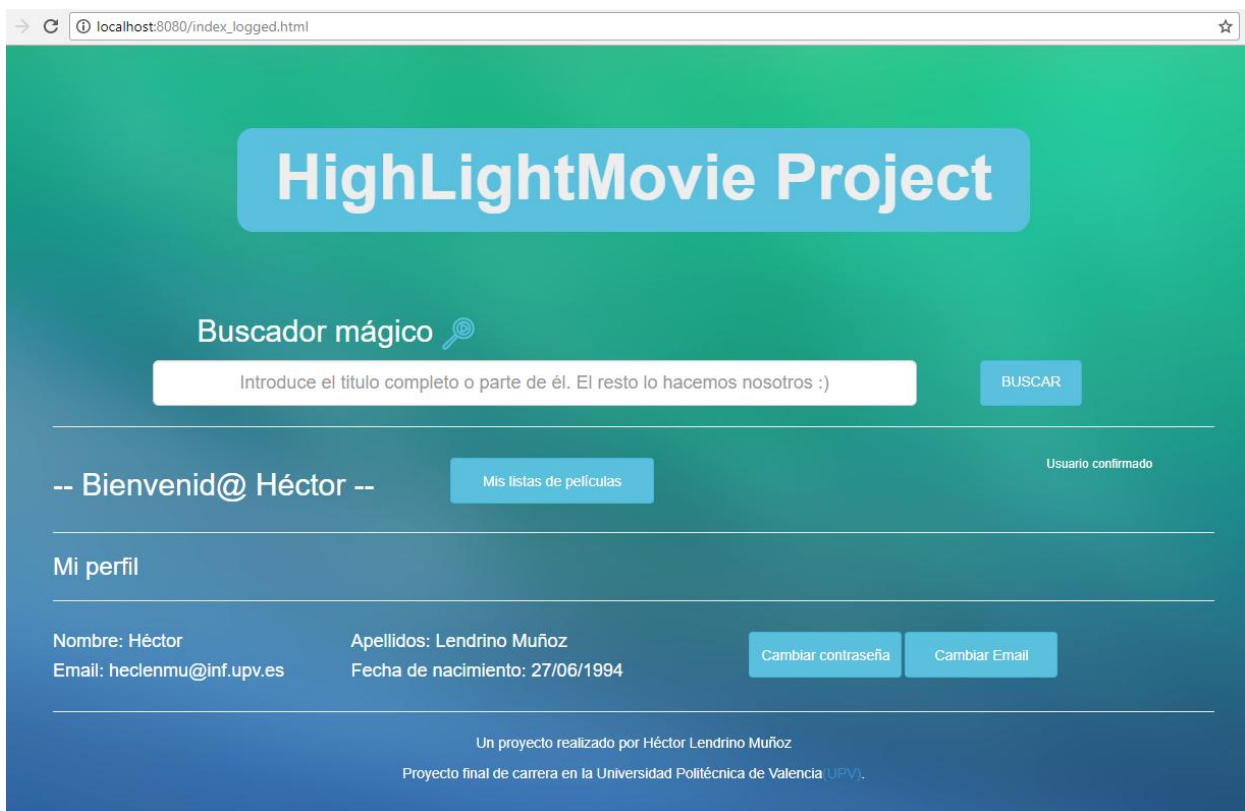
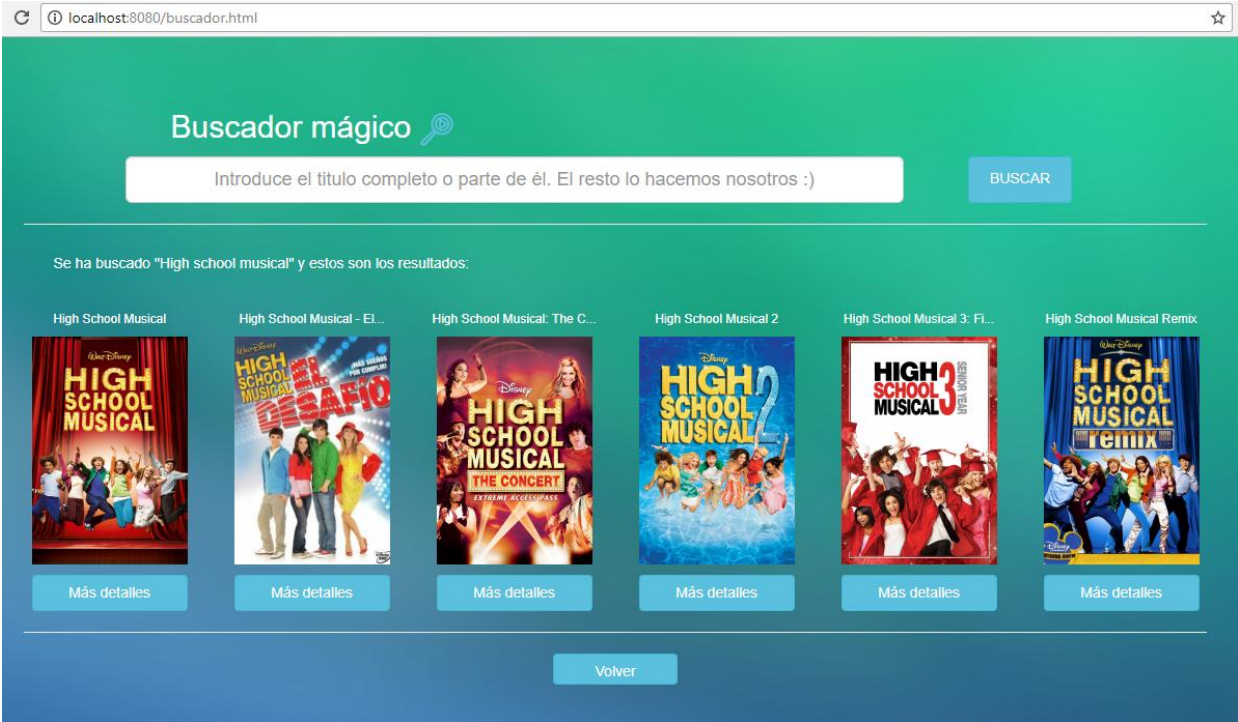


Ilustración 44 - Ventana de perfil

Diseño e implementación de una aplicación web para una red social de películas



The screenshot shows a web browser window with the address bar displaying 'localhost:8080/buscador.html'. The page has a green header with the text 'Buscador mágico' and a magnifying glass icon. Below this is a search input field containing the text 'Introduce el título completo o parte de él. El resto lo hacemos nosotros :)' and a blue 'BUSCAR' button. The main content area has a light green background and displays the search results for 'High school musical'. It lists six items with their respective posters: 'High School Musical', 'High School Musical - El... (El Paseo)', 'High School Musical: The C...', 'High School Musical 2', 'High School Musical 3: Fi...', and 'High School Musical Remix'. Each item has a 'Más detalles' button below its poster. At the bottom of the results section is a blue 'Volver' button.

Ilustración 45 - Buscando una película



The screenshot shows a web browser window with the address bar displaying 'localhost:8080/details.html'. The page has a green header with the title 'High School Musical 2' and a 'Trailer' button with a YouTube icon. A blue 'Volver' button is in the top right. The main content area features a large movie poster for 'High School Musical 2' on the left. To the right of the poster, the following information is displayed: 'Título original: High School Musical 2', 'Nota media: ★★★★★', 'Autor: 2007', 'Director: Kenny Ortega', and 'Autor: Peter Barsocchini'. Below this is a 'Sinopsis:' section with the text: 'El curso ha terminado y Troy Bolton, la superestrella del equipo de baloncesto de East High School, la inteligente Gabriella Montez y el resto de los Wildcats se preparan para disfrutar del verano más emocionante de sus vidas. Pero todo cambia cuando Sharpay Evans, la reina de los escenarios, utiliza los contactos de su "papá" para conseguir un trabajo a Troy en su súperexclusivo club de campo...'. Below the synopsis are four icons: a red flag, a play button, a heart, and a star. The 'Reparto' section follows, showing five actor portraits with their names: Zac Efron, Vanessa Hudgens, Ashley Tisdale, Lucas Grabeel, and Corbin Bleu. A blue button at the bottom right says 'Ver equipo y reparto completo'.

Ilustración 46 - Ventana de detalles de una película

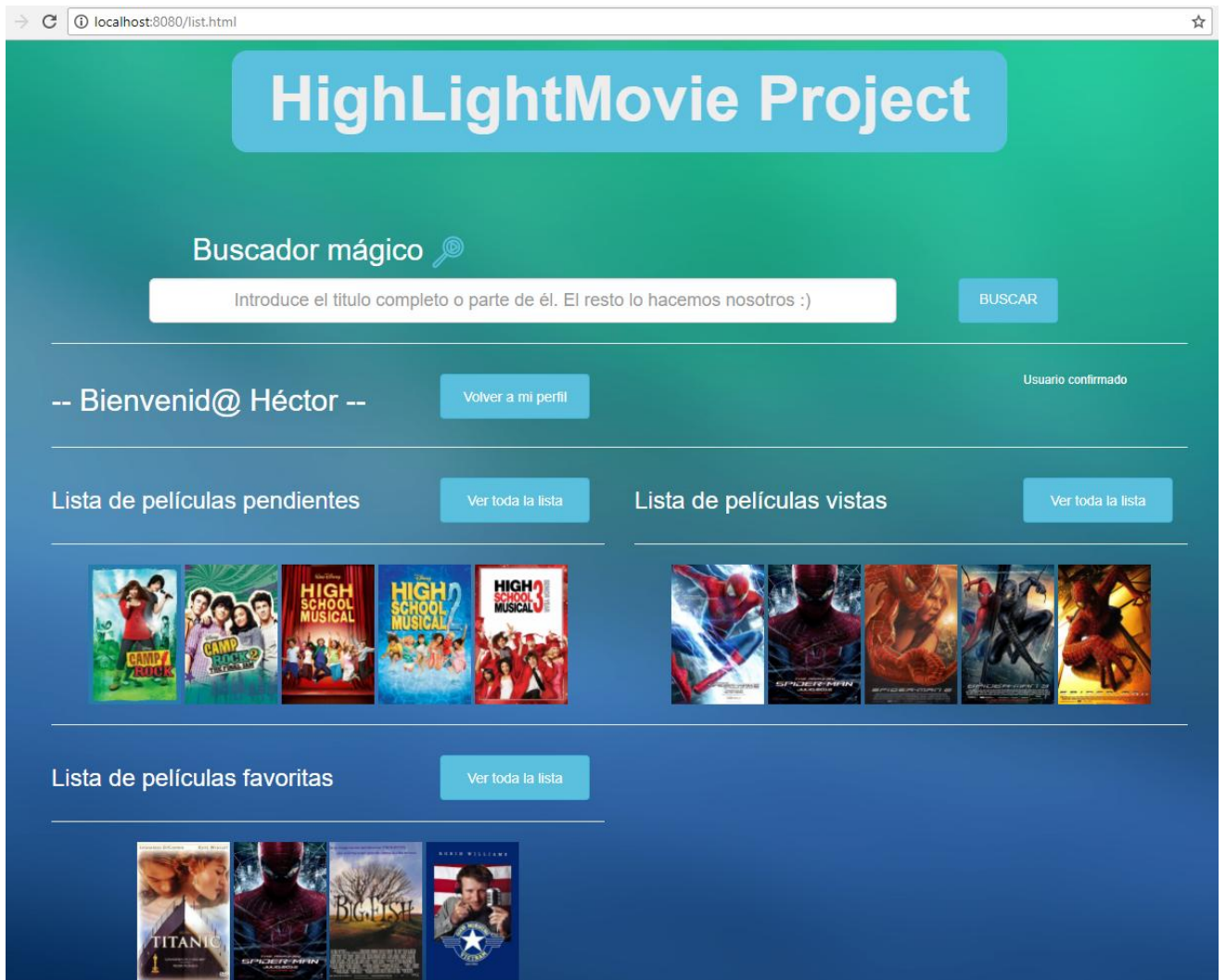


Ilustración 47 - Lista de mis películas

Tanto si somos usuarios registrados como si no, podremos buscar información sobre cualquier película, como se muestra en las ilustraciones 43, 44 y 45. El registro no es necesario si solo se quiere acceder a la información de las películas.

6. Conclusiones

6.1 Resumen

En resumen el proyecto ha consistido en analizar aplicaciones similares como IMDB o FilmAffinity, diseñar la aplicación e implementarla aprendiendo la tecnología más potente para el desarrollo web. MEAN que contiene una gran integración entre sus distintas herramientas lo que facilita el trabajo junto a HTML5 y CSS3 para la capa de navegador.

6.2 Dificultades y soluciones

A continuación se describen las dificultades más relevantes que se han encontrado en el desarrollo del proyecto.

El primer problema fue la gran cantidad de sistemas similares encontrados provocando la realización de una selección de los sistemas más conocidos y que mayor relación tenían con el proyecto a desarrollar centrándose sobre todo en los más conocidos.

Para el diseño de la aplicación, se han dedicado una gran cantidad de horas a pensar cuál sería el diseño más óptimo tanto para la aplicación web como para una futura aplicación para teléfonos inteligentes. Pensando siempre en un diseño optimizado y atractivo para el usuario final.

Al principio del proyecto se pensaba que la base de datos y su estructura iban a ser un gran problema por una posible complejidad en su diseño pero al final ha resultado ser bastante sencilla gracias a la utilización de MongoDB y la API.

Por último, el problema más relevante está directamente relacionado con la implementación del proyecto, por un lado el aprender a utilizar de manera correcta las tecnologías y por otro la limitación temporal por entregar el proyecto. Juntos han provocado que la implicación de horas sobre este proyecto haya sido mucho mayor de lo que en un principio se pensó provocando que no todos los objetivos se hayan cumplido.

6.3 Cambios futuros

Debido a la limitación temporal y los problemas encontrados hay ciertos aspectos del desarrollo de la aplicación que se han quedado pendientes por tanto hay que incluirlos en cambios futuros. Además el proyecto puede ser modificado y mejorado en diferentes aspectos.

La migración de la aplicación a otros sistemas operativos. Ya que en un principio está pensada como aplicación web pero puede ser desarrollada como aplicación Android, iOS y Microsoft.

La utilización de un sistema freemium para ofrecer características exclusivas dentro de la aplicación también sería un posible cambio. Ofrecer características exclusivas a cambio de una cuota mensual no muy elevada, tal vez sería de uno 1.99€ al mes.

La participación de las compañías y los críticos de cine sería un empuje importante a la aplicación, la creación de ofertas personalizadas por los propios cines y las críticas de gente experta en el campo generaría un beneficio muy elevado a la aplicación.

6.4 Valoración personal

Este proyecto final de carrera ha supuesto la adquisición de una gran cantidad de conocimientos.

Por un lado tenemos los conocimientos profesionales, que seguro podrán ser aplicados en la vida laboral como profesional del sector, como son asentar algunos conocimientos aprendidos durante la carrera sobretodo relacionados con el desarrollo software. También se ha comprobado la complejidad que conlleva la realización de un proyecto, desde el análisis hasta la implementación y pruebas.

El tiempo que lleva el estudio de una nueva tecnología a utilizar y más si solo es desarrollado por una persona y no por un equipo.

Y por otro lado en lo personal, la posibilidad de hacer un proyecto relacionado con uno de tus hobbies, el mundo del cine, ha mejorado la experiencia. El descubrimiento de nuevas páginas web relacionadas con el sector para obtener información tanto para uso y disfrute personal como para la utilización dentro de la aplicación.

En general ha sido una gran experiencia que ayuda a consolidar los conocimientos adquiridos a la vez que te ayuda a enfocar tu vida profesional.



7. Referencias

7.1 Internet

- [1] DCINE. *Acerca de Dcine*.
<http://www.dcine.org/acerca-de-dcine> [Acceso: Mayo 2017]
- [2] FILMAFFINITY. *FAQ FilmAffinity*.
<http://www.filmaffinity.com/es/faq.php> [Acceso: Mayo 2017]
- [3] IMDB. *IMDb web*.
<http://www.imdb.com/> [Acceso: Mayo 2017]
- [4] MDBPRO. *IMDbPro web*.
<https://secure.imdb.com/signup/index.html?r=/> [Acceso: Mayo 2017]
- [5] MOVIEHAKU. *MovieHaku Ayuda*.
<http://moviehaku.com/ayuda/> [Acceso: Mayo 2017]
- [6] MOVIEHAKU. *MovieHaku Tour*.
<http://moviehaku.com/tour/> [Acceso: Mayo 2017]
- [7] 37PIXELS. *37 Pixels web*.
<http://37pixels.net/> [Acceso: Mayo 2017]
- [8] APP MOQUPS. *Creación de bocetos*.
<https://app.moqups.com/> [Acceso: Mayo 2017]
- [9] HTML. *HTML versión 5*.
<https://developer.mozilla.org/es/docs/HTML/HTML5> [Acceso: Mayo 2017]
- [10] CSS. *CSS versión 3*.
<https://developer.mozilla.org/es/docs/Web/CSS/CSS3> [Acceso: Junio 2017]
- [11] JAVASCRIPT. *JavaScript web*.
<https://developer.mozilla.org/es/docs/Glossary/JavaScript> [Acceso: Junio 2017]
- [12] EXPRESS JS. *Web express js*.
<http://expressjs.com/es/> [Acceso: Junio 2017]
- [13] MONGO DB. *Web mongoDb*.
<https://www.mongodb.com/es> [Acceso: Junio 2017]
- [14] ANGULARJS. *Web AngularJS*.
<https://angularjs.org/> [Acceso: Junio 2017]
- [15] NODEJS. *Web NodeJS*.
<https://nodejs.org/es/> [Acceso: Junio 2017]



[16] ANGULARJS (V. 1.0.8). *Versión AngularJS versión 1.0.8*
<https://code.angularjs.org/1.0.8/> [Acceso: Junio 2017]

7.2 Bibliográficas

1. DEUSTO. Alexander Osterwalder & Yves Pigneur. Generación de modelos de negocio. Un manual para visionarios, revolucionarios y retadores, 2013
2. ALTARIA. Manuel López Quintero. Node.js, Javascript del lado del servidor: manual práctico avanzado. 2015.
3. JOHN WILEY & SONS INC. Pedro Teixeira. Professional Node.js: Building Javascript Based Scalable Software. 2013.