AVMASTER

Proyecto Fin de Ciclo (PFC)

Desarrollo de Aplicaciones Multiplataforma

Martín García Ramos

Versión: Final marzo 2024

Índice

1.- Descripción del proyecto y ámbito de implantación:

1.A. - ¿Qué has desarrollado y por qué?

Se ha desarrollado una aplicación web que tiene como objetivo ser un lugar seguro y privado en el que poder registrar anónimamente los contenidos audiovisuales que te gustan, así como ver detalles interesantes de los mismos.

La idea surgió de la necesidad de tener siempre a mano un blog de notas para apuntar donde te habías quedado en esa serie o hace 3 meses. Si eres un ferviente seriéfilo o cinéfilo, más pronto que tarde empezarás a tener una duda real sobre qué series y qué películas te gustaron en el pasado. Evidentemente en un corto periodo de tiempo puedes recordarlas pero no en un plazo de 1, 2, 5, 10 o más años donde unos títulos se te entremezclarán con otros y donde muchos de esos contenidos simplemente los olvidarás, para luego un día decir: ¿Cómo se llamaba aquella película/serie que me gustó tanto?

Con ese propósito nace AVMaster, una plataforma web, con la que el usuario puede interactuar de manera intuitiva e ir gestionando de manera automática todos los contenidos que le gustan para luego poderlos filtrar como se desee.

1.B. - ¿Quién lo usará?

En principio, cualquier persona puede, aunque en realidad esta plataforma tiene un interés especial en el colectivo de los seriéfilos y cinéfilos dedicados, aquellas personas que en sus ratos libres no conciben un mejor plan que ponerse a ver solos o en compañía contenidos audiovisuales.

En ese sentido todo el branding de la web está diseñado con la finalidad de atraer al público de todas las diversas edades, así como ayudándoles a localizar fácilmente cada una de las opciones con las que pueden interactuar.

1.C. - ¿Qué tecnologías has usado?

Las tecnologías usadas fueron varias:

- **HTML:** Es un lenguaje de marcado utilizado para crear y estructurar páginas web.
- **CSS:** Es un lenguaje utilizado para definir el estilo y la presentación de documentos HTML.
- **JavaScript:** Es un lenguaje de programación utilizado principalmente para agregar interactividad y dinamismo a páginas web.
- **React:** Es una biblioteca de JavaScript utilizada para construir interfaces de usuario interactivas y reutilizables para aplicaciones web.
- **Bootstrap:** Es un marco de trabajo de código abierto utilizado para diseñar y desarrollar interfaces web responsivas y estilizadas.

1.D. - ¿Se prevén cambios en el futuro?

Uno de los cambios que serían más urgentes sería el de ajustar correctamente el centrado de los elementos de manera responsiva, a la hora de mostrar el listado de ítems. Es lo único(Junto con el encuadre de la barra lateral en la página de detalles)que he intentado hacer y no he conseguido realizar para que quedara correctamente establecido.

Por otro lado, uno de los objetivos iniciales que se preveían para la aplicación era que en la página de detalles de la serie o la película también saliera su tráiler, así que esta sería una mejora sencilla a implementar.

Además, en esa misma página de detalles también iba a tener las fotografías de los actores, directores, etc, que estuvieran disponibles en la API de themoviedb.com, pero se terminó descartando al complicarse innecesariamente, también se podría incluir en una segunda iteración.

Por otro lado, originalmente en el proyecto también se planteó que en vez de solo mostrarse el listado de favoritos, el usuario pudiera listar por muchos más atributos e incluso pudiera hacer sus propios subconjuntos, creando por ejemplo: “Series que dejé de ver”, de esa manera la aplicación mejoraría notablemente su usabilidad e interés por parte de su público potencial; lamentablemente, no se terminó descartando por su complejidad, pero se podría atajar en una posible segunda iteración del proyecto.

2. – Temporalización del proyecto y fases de desarrollo

2.A - PLANIFICACIÓN:

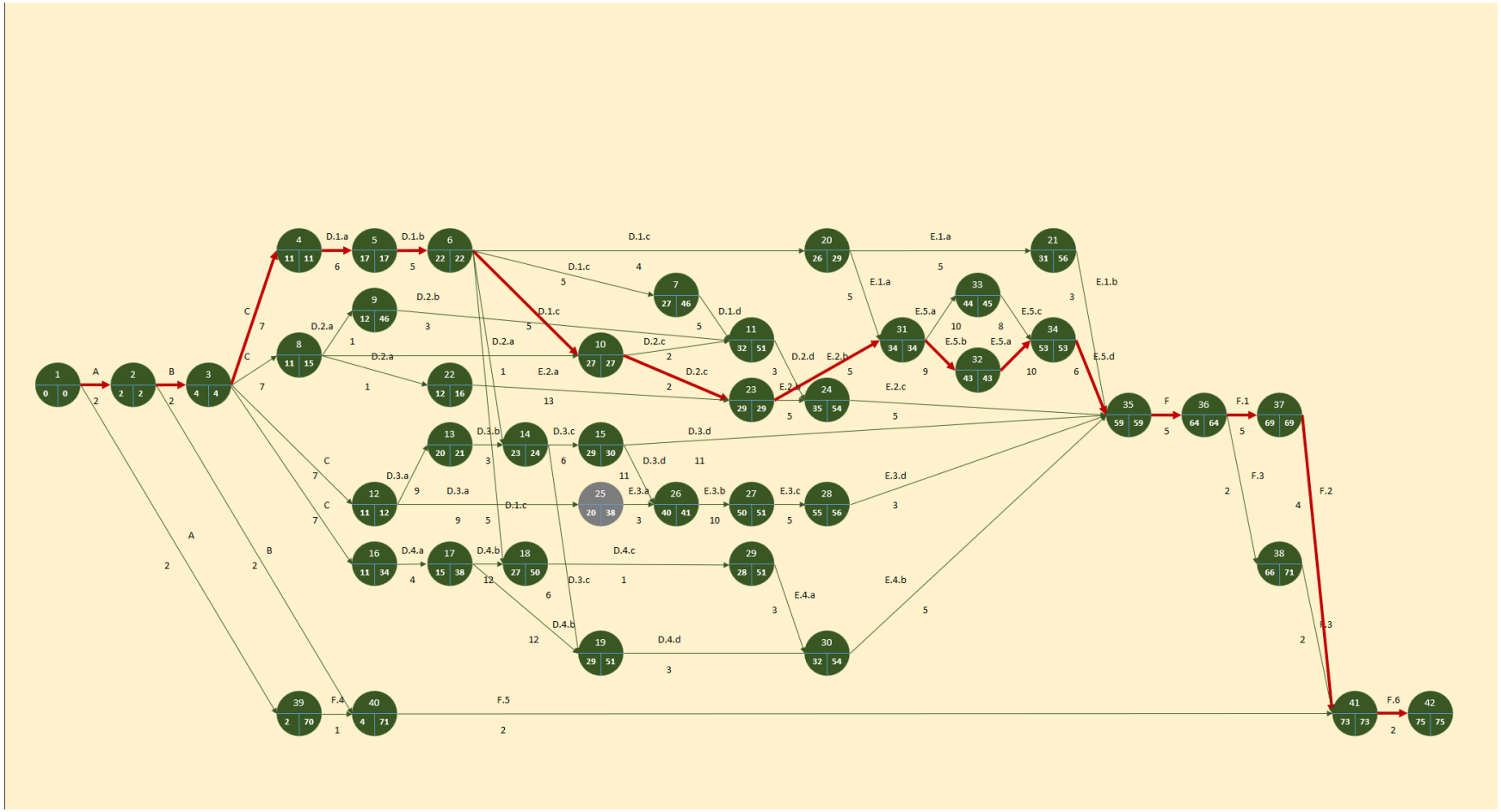
Planificación y descripción de las actividades en los diagramas PERT y GANT. Debido a que se permiten iteraciones dinámicas, estos diagramas se han localizado en el Excel adjunto a este pdf.
No obstante dejo aquí algunas capturas para dejar constancia de ello:

2.A.1. PERT:

Para ver en mejor detalle mirar el Excel adjunto.

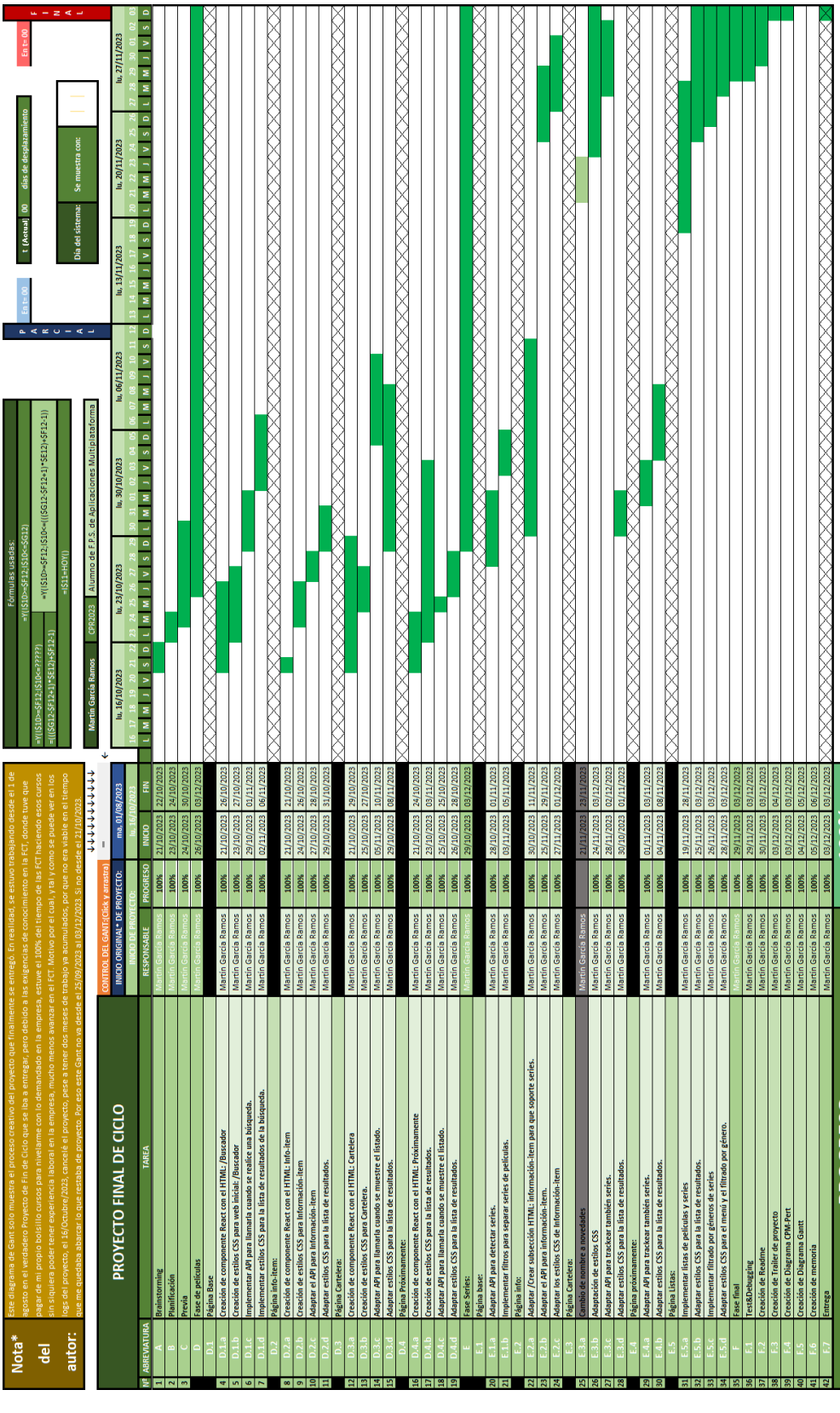
Nº	ABREVIATURA	TAREA	FECHA INICIO	FECHA FIN	DURACIÓN TOTAL (En días)
1	A	Brainstorming	21/10/2023	22/10/2023	2,0
2	B	Planificación	23/10/2023	24/10/2023	2,0
3	C	Previa	24/10/2023	30/10/2023	7,0
	D	Fase de películas	26/10/2023	03/12/2023	
	D.1	Página Base			
4	D.1.a	Creación de componente React con el HTML: /Buscador	21/10/2023	26/10/2023	6,0
5	D.1.b	Creación de estilos CSS para web inicial: /Buscador	23/10/2023	27/10/2023	5,0
6	D.1.c	Implementar API para llamarla cuando se realice una búsqueda.	29/10/2023	01/11/2023	4,0
7	D.1.d	Implementar estilos CSS para la lista de resultados de la búsqueda.	02/11/2023	06/11/2023	5,0
	D.2	Página info-item:			
8	D.2.a	Creación de componente React con el HTML: Info-item	21/10/2023	21/10/2023	1,0
9	D.2.b	Creación de estilos CSS para Información-item	24/10/2023	26/10/2023	3,0
10	D.2.c	Adaptar el API para Información-item	27/10/2023	28/10/2023	2,0
11	D.2.d	Adaptar estilos CSS para la lista de resultados.	29/10/2023	31/10/2023	3,0
	D.3	Página Cartelera:			
12	D.3.a	Creación de componente React con el HTML: Cartelera	21/10/2023	29/10/2023	9,0
13	D.3.b	Creación de estilos CSS para Cartelera.	25/10/2023	27/10/2023	3,0
14	D.3.c	Adaptar API para llamarla cuando se muestre el listado.	05/11/2023	10/11/2023	6,0
15	D.3.d	Adaptar estilos CSS para la lista de resultados.	29/10/2023	08/11/2023	11,0
	D.4	Página Próximamente:			
16	D.4.a	Creación de componente React con el HTML: Próximamente	21/10/2023	24/10/2023	4,0
17	D.4.b	Creación de estilos CSS para la lista de resultados.	23/10/2023	03/11/2023	12,0
18	D.4.c	Adaptar API para llamarla cuando se muestre el listado.	25/10/2023	25/10/2023	1,0
19	D.4.d	Adaptar estilos CSS para la lista de resultados.	26/10/2023	28/10/2023	3,0
	E	Fase Series:	29/10/2023	03/12/2023	
	E.1	Página base:			
20	E.1.a	Adaptar API para detectar series.	28/10/2023	01/11/2023	5,0
21	E.1.b	Implementar filtros para separar series de películas.	03/11/2023	05/11/2023	3,0
	E.2	Página info:			
22	E.2.a	Adaptar /Crear subsección HTML: información-item para que soporte serie	30/10/2023	11/11/2023	13,0
23	E.2.b	Adaptar el API para información-item.	25/11/2023	29/11/2023	5,0
24	E.2.c	Adaptar los estilos CSS de Información-item	27/11/2023	01/12/2023	5,0
	E.3	Página Cartelera:			
25	E.3.a	~~Cambio de nombre a novedades~~	~~21/11/2023~~	~~23/11/2023~~	3,0
26	E.3.b	Adaptación de estilos CSS	24/11/2023	03/12/2023	10,0
27	E.3.c	Adaptar API para trackear también series.	28/11/2023	02/12/2023	5,0
28	E.3.d	Adaptar estilos CSS para la lista de resultados.	30/10/2023	01/11/2023	3,0
	E.4	Página próximamente:			
29	E.4.a	Adaptar API para trackear también series.	01/11/2023	03/11/2023	3,0
30	E.4.b	Adaptar estilos CSS para la lista de resultados.	04/11/2023	08/11/2023	5,0
	E.5	Página Listas:			
31	E.5.a	Implementar listas de películas y series	19/11/2023	28/11/2023	10,0
32	E.5.b	Adaptar estilos CSS para la lista de resultados.	25/11/2023	03/12/2023	9,0
33	E.5.c	Implementar filtrado por géneros de series	26/11/2023	03/12/2023	8,0
34	E.5.d	Adaptar estilos CSS para el menú y el filtrado por género.	28/11/2023	03/12/2023	6,0
35	F	Fase final	29/11/2023	03/12/2023	5,0
36	F.1	Test&Debugging	29/11/2023	03/12/2023	5,0
37	F.2	Creación de Readme	30/11/2023	03/12/2023	4,0
38	F.3	Creación de Trailer de proyecto	03/12/2023	04/12/2023	2,0
39	F.4	Creación de Diagrama CPM-Pert	03/12/2023	03/12/2023	1,0
40	F.5	Creación de Diagrama Gantt	04/12/2023	05/12/2023	2,0
41	F.6	Creación de memoria	05/12/2023	06/12/2023	2,0
42	F.7	Entrega	03/12/2023	03/12/2023	1,0
					204,0 / 43

Nº	ABREVIATURA	TAREA	PREDECESOR PARA INICIARSE
1	A	Brainstorming	—
2	B	Planificación	A
3	C	Previa	B
	D	Fase de películas	
	D.1	Página Base	
4	D.1.a	Creación de componente React con el HTML: /Buscador	C
5	D.1.b	Creación de estilos CSS para web inicial: /Buscador	'D.1.a'
6	D.1.c	Implementar API para llamarla cuando se realice una búsqueda.	'D.1.b'
7	D.1.d	Implementar estilos CSS para la lista de resultados de la búsqueda.	'D.1.c'
	D.2	Página info-item:	
8	D.2.a	Creación de componente React con el HTML: Info-item	C
9	D.2.b	Creación de estilos CSS para Información-item	'D.2.a'
10	D.2.c	Adaptar el API para Información-item	'D.1.c' & 'D.2.a'
11	D.2.d	Adaptar estilos CSS para la lista de resultados.	'D.1.d' & 'D.2.b' & 'D.2.c'
	D.3	Página Cartelera:	
12	D.3.a	Creación de componente React con el HTML: Cartelera	C
13	D.3.b	Creación de estilos CSS para Cartelera.	'D.3.a'
14	D.3.c	Adaptar API para llamarla cuando se muestre el listado.	'D.1.c' & 'D.3.b'
15	D.3.d	Adaptar estilos CSS para la lista de resultados.	'D.3.c'
	D.4	Página Próximamente:	
16	D.4.a	Creación de componente React con el HTML: Próximamente	C
17	D.4.b	Creación de estilos CSS para la lista de resultados.	'D.4.a'
18	D.4.c	Adaptar API para llamarla cuando se muestre el listado.	'D.1.c' & 'D.4.b'
19	D.4.d	Adaptar estilos CSS para la lista de resultados.	'D.4.b' & 'D.3.c'
	E	Fase Series:	
	E.1	Página base:	
20	E.1.a	Adaptar API para detectar series.	'D.1.c'
21	E.1.b	Implementar filtros para separar series de películas.	'E.1.a'
	E.2	Página info:	
22	E.2.a	Adaptar /Crear subsección HTML: información-item para que soporte serie	'D.2.a'
23	E.2.b	Adaptar el API para información-item.	'D.2.c' & 'E.2.a'
24	E.2.c	Adaptar los estilos CSS de Información-item	'D.2.d' & 'E.2.b'
	E.3	Página Cartelera:	
25	E.3.a	~~Cambio de nombre a novedades~~	'D.3.a'
26	E.3.b	Adaptación de estilos CSS	'D.3.d' & 'E.3.a'
27	E.3.c	Adaptar API para trackear también series.	'E.3.b'
28	E.3.d	Adaptar estilos CSS para la lista de resultados.	'E.3.c'
	E.4	Página próximamente:	
29	E.4.a	Adaptar API para trackear también series.	'D.4.c'
30	E.4.b	Adaptar estilos CSS para la lista de resultados.	'D.4.d' & 'E.4.a'
	E.5	Página Listas:	
31	E.5.a	Implementar listas de películas y series	E.1.a' & 'E.2.b'
32	E.5.b	Adaptar estilos CSS para la lista de resultados.	'E.5.a'
33	E.5.c	Implementar filtrado por géneros de series	'E.5.a'
34	E.5.d	Adaptar estilos CSS para el menú y el filtrado por género.	'E.5.b' & 'E.5.c'
35	F	Fase final	D.3.d' & 'E.1.b' & 'E.2.c' & 'E.3.d' & 'E.4.b' & 'E.5.d'
36	F.1	Test&Debugging	'F'
37	F.2	Creación de Readme	'F.1'
38	F.3	Creación de Trailer de proyecto	'F.1'
39	F.4	Creación de Diagrama CPM-Pert	'A'
40	F.5	Creación de Diagrama Gantt	'B' & 'F.4'
41	F.6	Creación de memoria	'F.2' & 'F.3' & 'F.5'
42	F.7	Entrega	'F.6'


1 0 0
2 2 2
3 4 4
4 11 11
5 17 17
6 22 22
7 27 46
8 11 15
9 12 46
10 27 27
11 32 51
12 11 12
13 20 21
14 23 24
15 29 30
16 11 34
17 15 38
18 27 50
19 29 51
20 26 29
21 31 56
22 12 16
23 29 29
24 35 54
25 20 38
26 40 41
27 50 51
28 55 56
29 28 51
30 32 54
31 34 34
32 43 43
33 44 45
34 53 53
35 59 59
36 64 64
37 69 69
38 66 71
39 2 70
40 4 71
41 73 73
42 75 75
A
B
C
D.1.a
D.1.b
D.1.c
D.1.d
D.2.a
D.2.b
D.2.c
D.2.d
D.3.a
D.3.b
D.3.c
D.3.d
D.4.a
D.4.b
D.4.c
D.4.d
E.1.a
E.1.b
E.2.a
E.2.b
E.2.c
E.3.a
E.3.b
E.3.c
E.3.d
E.4.a
E.4.b
E.5.a
E.5.b
E.5.c
E.5.d
F
F.1
F.2
F.3
F.4
F.5
F.6

2.A.2. GANTT:

Para ver en mejor detalle el Gantt mirar el Excel adjunto.



2.B. - RETOS ENCONTRADOS:

Esencialmente, el tener que desarrollar el proyecto en una tecnología desconocida en el momento del inicio del proyecto, provocó que a lo largo del desarrollo se tuvieran que rehacer diversas secciones varias veces y que el proyecto fuera más laberíntico de lo que ocurriría si se tuviera una planificación correcta.

Además por otro lado, al usar una tecnología tan nueva como ReactJS, he tenido que enfrentarme a diversos problemas con el software externo, uno de ellos por ejemplo fue detectar porqué se producía el error que se indica a continuación cuando se accedía a la aplicación tras haber seleccionado otra pestaña en el navegador y volver a la pestaña donde está la aplicación.

Uncaught runtime errors:

×

ERROR

Cannot redefine property: googletag

TypeError: Cannot redefine property: googletag

at Function.defineProperty (<anonymous>)

at <anonymous>:1:196

at <anonymous>:1:524

Tras muchas pruebas y concluir, que el error no se debía a mi propio proyecto, hallé la solución en inhabilitar la extensión de navegador: Adblocker Stands. Por lo visto, es un error conocido desde Marzo de 2024.

3. – Recursos de Hardware y software

Al tratarse de una web, los únicos requisitos de hardware necesarios son aquellos que se demandan para ejecutar debidamente un navegador como Chrome o Firefox, estos son:

3.A. - Requisitos de Hardware:

3.A.1. Requisitos del navegador:

3.A.1.a. Mínimos:

- **Procesador:** Procesador Intel Pentium 4 o AMD Athlon 64.
- **Memoria RAM:** 1 GB.
- **Espacio en disco duro:** 100 MB de espacio libre.
- **Resolución de pantalla:** 1280 x 1024.

3.A.1.b. Recomendados:

- **Procesador:** Procesador Intel Core i3 o AMD Ryzen 3, o superior.
- **Memoria RAM:** 4 GB o más para un rendimiento óptimo.
- **Espacio en disco duro:** 500 MB de espacio libre o más para la instalación y almacenamiento en caché.
- **Resolución de pantalla:** 1920 x 1080 o superior para una experiencia de navegación de alta definición.

3.A.2. Requisitos para ejecutar el servidor de ReactJS:

3.A.2.a. Requisitos Mínimos:

- **Procesador:** Procesador compatible con arquitectura x86 o x64.
- **Memoria RAM:** 2 GB o más.
- **Espacio en disco duro:** Al menos 100 MB de espacio libre para la instalación de Node.js y ReactJS, aunque el espacio requerido dependerá del tamaño de tus proyectos y las dependencias que utilices.

3.A.2.b. Requisitos Recomendados:

- **Procesador:** Procesador de al menos 2 núcleos, preferiblemente con capacidad para multihilos (Hyper-Threading o similar) para mejorar el rendimiento en tareas de compilación y ejecución.

- **Memoria RAM:** 4 GB o más para manejar proyectos de tamaño mediano a grande de manera eficiente.
- **Disco duro:** Un disco SSD (Solid State Drive) puede mejorar significativamente los tiempos de compilación y carga del servidor debido a su velocidad de acceso más rápida en comparación con un disco duro HDD (Hard Disk Drive).

3.B. - Requisitos software:

Cualquier navegador debería de ser capaz de ejecutar la web indistintamente, aunque esta fue testeada con Chrome y Firefox, así que cualquier navegador que use el motor web: Blink o el motor Gecko, así como tener el intérprete de JavaScript V8 o superior.

3.A.1. Requisitos del navegador:

3.A.1.a. Mínimos:

- **Sistema operativo:** Windows 7 o posterior, macOS X 10.10 o posterior, Ubuntu 14.04 o posterior, Debian 8 o posterior, openSUSE 13.3 o posterior, Fedora Linux 24 o posterior.

3.A.1.b. Recomendados:

- **Sistema operativo:** La última versión disponible es recomendada para garantizar la seguridad y la compatibilidad con las últimas tecnologías web.

3.A.2. Requisitos para ejecutar el servidor de ReactJS:

3.A.2.a. Requisitos Mínimos:

- **Node.js:** ReactJS se ejecuta sobre Node.js, por lo que necesitarás instalar Node.js en tu sistema. Se recomienda tener al menos la versión LTS (Long Term Support) para garantizar la estabilidad y compatibilidad. Puedes obtener Node.js desde su sitio web oficial: nodejs.org.

3.A.2.b. Requisitos Recomendados:

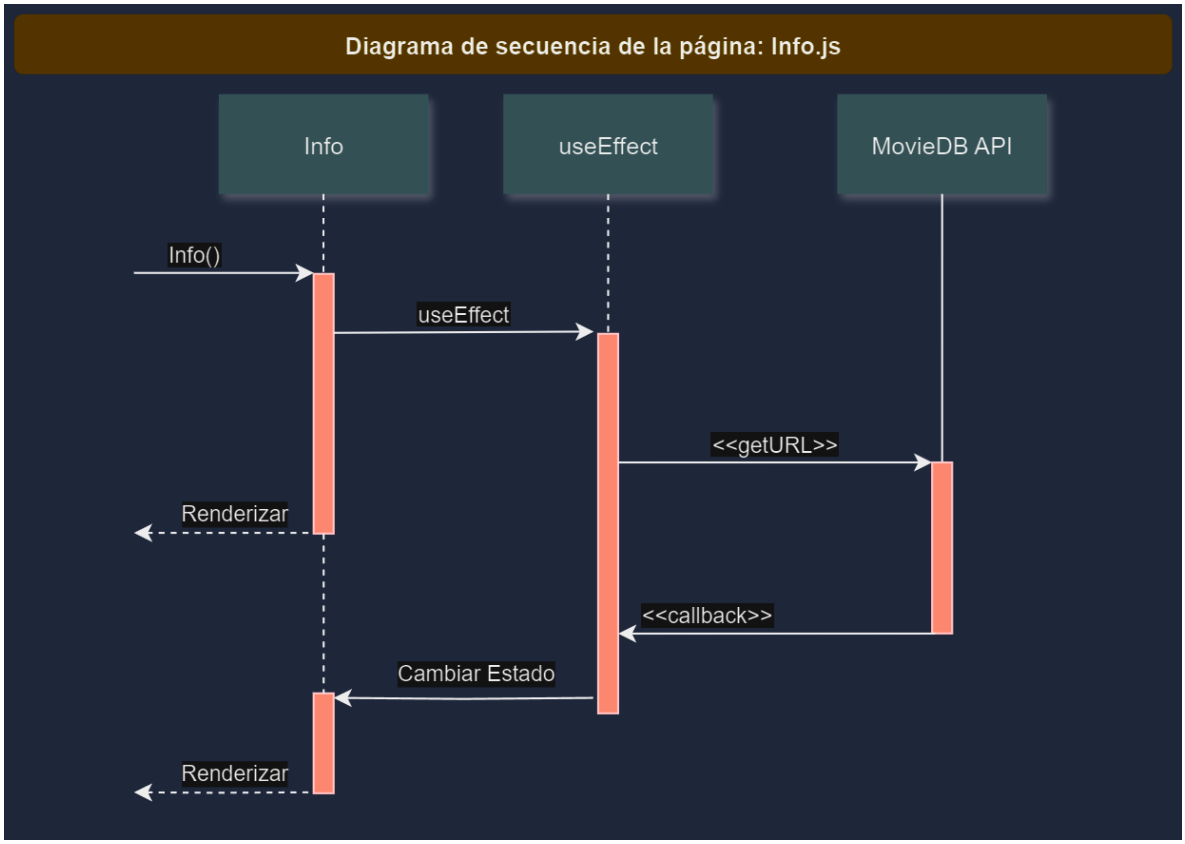
- **Gestor de paquetes npm:** npm es el gestor de paquetes de Node.js y se utiliza para instalar y administrar las dependencias de los proyectos de ReactJS. Se recomienda tener la última versión de npm instalada junto con Node.js.
- **Editor de código:** Se recomienda utilizar un editor de código con características que faciliten el desarrollo en ReactJS, como Visual Studio Code, Atom o Sublime Text.

- **Sistema de control de versiones:** Utilizar un sistema de control de versiones como Git para gestionar el código fuente de tus proyectos.

4. – Arquitectura software y de sistemas

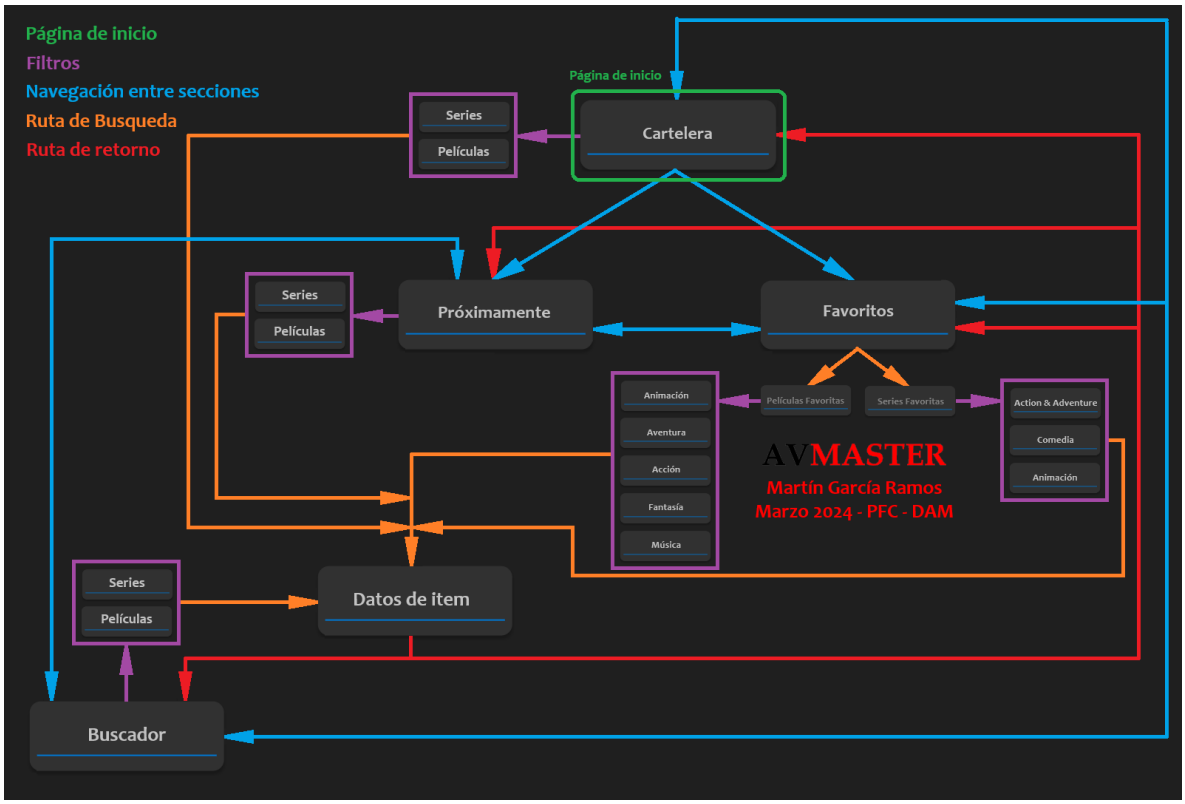
Mi proyecto al ser una web en ReactJS, no tiene “clases” o una orientación a objetos persé, tiene algo parecido a lo que se les denomina componentes y helpers; que en función a su propósito se le puede subdividir en helpers, pages, etc.…

4.A. - Diagrama de Secuencia:



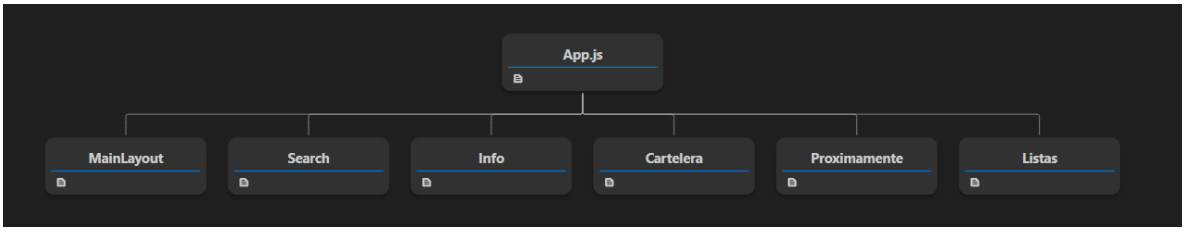
En este diagrama se puede ver cómo se envía Info(), a su vez Info llama al useEffect y este hará una petición asíncrona para obtener los datos de la película. Y mientras tanto, el Info devolverá el renderizado de la páginas de información.
Cuando el MovieDB devuelva los datos, el useEffect hará una petición para cambiar el estado de la variable película y por ende se vuelve a realizar una petición para volver a renderizar la página de Info con los nuevos datos actualizados.

4.B. - Diagrama de Navegación:



4.C. - Diagrama de Jerarquía de componentes en cada página:

El proyecto consta de una aplicación que se reparte sus funcionalidades en los siguientes pages y layout, siendo llamados estos desde App.js mediante el Route:



El **MainLayout** determina las características de la barra superior de la aplicación.

El **Search** por su parte, determina las propiedades del recuadro de búsqueda superior en su index que a su vez conecta con el componente ListadoPeliculas, que es el que se encarga de listar cada uno de los ítems, bien sean películas o series que se estén devolviendo en este caso como resultado de la búsqueda.

El **Info** por su parte, se encarga de obtener información sobre una película o serie de televisión desde una API, y luego renderiza esta información utilizando el componente DatosDePelicula.

El **Cartelera** muestra una lista de películas o series dependiendo del tipo seleccionado por el usuario, haciendo solicitudes a una API y mostrando los resultados utilizando componentes personalizados. El usuario puede cambiar entre películas y series seleccionando el tipo deseado; conectando a su vez con el componente DatosDePelicula para mostrar los datos del item seleccionado según proceda.

En **Próximamente**, gracias a su index, se muestra una lista de películas o series próximamente disponibles, haciendo solicitudes a una API y mostrando los resultados utilizando el componente de ListadoPelículas que también se reutiliza en Search y Cartelera. El usuario puede cambiar entre películas y series seleccionando el tipo deseado.

En **Listas** se encarga de la presentación general de las listas de favoritos y las opciones de filtrado, mientras que el ItemFavorito se encarga de representar cada elemento favorito individual y proporcionar opciones para interactuar con ellos, como eliminarlos de la lista.

En todas las páginas, siempre se puede volver a la página de inicio iterando con los respectivos botones del **MainLayout**.

5. – Descripción de datos

Componentes:

CountryFlag:

Parámetro Country String

FavouriteItem:

Parámetro favourite Object con los campos Type,Id,Poster, name

Parámetro deleteFavourite Función que recibe como parámetro un objeto favourite.

MovieData:

Parámetro movie Object con los campos:

Vote_average

Popularity

budget

revenue

id

name

poster_path

genres

backdrop_path

production_countries

spoken_languages

runtime (Sólo en caso de películas)

status(Sólo en el caso de las series)

realease_date (Sólo en caso de película)

first_air_date(Sólo en caso de serie)

last_air_date(Sólo en caso de serie)

title(Sólo en caso de película)

name(Sólo en caso de serie)

original_title(Sólo en caso de película).

original_name(Sólo en caso de serie)

created_by(Sólo en caso de series)

overview

number_of_seasons (Sólo en caso de serie)

number_of_episodes (Sólo en caso de serie)

episode_run_time (Sólo en caso de serie)

seasons

Parámetro type String

Parámetro actors Array de objetos (Sólo en caso de películas)

Crew

Cast

MovieList

Parámetro props.movies Array de movies con el objeto movie de MovieData.

SearchResultItem

Parámetro props.dateFormat

Parámetro props.onClick

Parámetro props.img

Parámetro props.title

Parámetro props.date

Helpers:

countryFlags.js

FavouriteManager.js

getFavourites()

isFavorite()

Parámetro favourite

insertFavourite()

Parámetro favourite

deleteFavourite()

Parámetro favourite

fetchHelper.js

getURL()

Parámetro endpoint

Parámetro params

- mapHelpers.js
  - mapMovie()
    - Parámetro movie
  - mapSeries()
    - Parámetro series
  - mapSeriesComingSoon()
    - Parámetro series

Layout:

- MainLayout.js

Pages:

- Billboard
- ComingSoon
- Favourites
- Info
- Search

Debido a que la aplicación depende en gran medida de las restricciones de la API a la que se llama, en caso de necesitarse mayor información sobre las restricciones de los campos a cargar se puede consultar más a detalle en el siguiente enlace:
https://developer.themoviedb.org/docs/search-and-query-for-details

6. – Documentación adjunta

6.A. GITHUB:

https://github.com/AGuyLearningCode/pfc_2023

6.B. DIAGRAMAS:

Están adjuntos en el zip que se entregó en classroom, así como también están subidos en la carpeta Docs del proyecto en el mismo repositorio de GitHub.

6.C. TRAILER:

En el siguiente enlace está disponible en alta calidad.

https://drive.google.com/file/d/1vjVgibddjwg5xZZsF7gHrEj4gn_Wn5ks/view?usp=drive_link

No obstante, también está subido al repositorio de GitHub una versión comprimida.