

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: 3](#_Toc162302085)

[1.A. - ¿Qué has desarrollado y por qué? 3](#_Toc162302086)

[1.B. - ¿Quién lo usará? 3](#_Toc162302087)

[1.C. - ¿Qué tecnologías has usado? 3](#_Toc162302088)

[1.D. - ¿Se prevén cambios en el futuro? 4](#_Toc162302089)

[2. – Temporalización del proyecto y fases de desarrollo 5](#_Toc162302090)

[2.A - PLANIFICACIÓN: 5](#_Toc162302091)

[2.A.1. PERT: 5](#_Toc162302092)

[2.A.2. GANTT: 8](#_Toc162302093)

[2.B. - RETOS ENCONTRADOS: 9](#_Toc162302094)

[3. – Recursos de Hardware y software 10](#_Toc162302095)

[3.A. - Requisitos de Hardware: 10](#_Toc162302096)

[3.A.1. Requisitos del navegador: 10](#_Toc162302097)

[3.A.2. Requisitos para ejecutar el servidor de ReactJS: 10](#_Toc162302098)

[3.B. - Requisitos software: 11](#_Toc162302099)

[3.A.1. Requisitos del navegador: 11](#_Toc162302100)

[3.A.2. Requisitos para ejecutar el servidor de ReactJS: 11](#_Toc162302101)

[4. – Arquitectura software y de sistemas 13](#_Toc162302102)

[4.A. - Diagrama de Secuencia: 13](#_Toc162302103)

[4.B. - Diagrama de Navegación: 14](#_Toc162302104)

[4.C. - Diagrama de Jerarquía de componentes en cada página: 14](#_Toc162302105)

[5. – Descripción de datos 17](#_Toc162302106)

[6. – Documentación adjunta 20](#_Toc162302107)

[6.A. GITHUB: 20](#_Toc162302108)

[6.B. DIAGRAMAS: 20](#_Toc162302109)

[6.C. TRAILER: 20](#_Toc162302110)

# 1.- Descripción del proyecto y ámbito de implantación:

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

Se ha **creado** una **plataforma** en **línea** **con** **la** **finalidad** de **proporcionar** un **espacio** seguro y privado **para** la **administración** de **manera** **anónima** de **tus** **videos** **favoritos** y **descubrir** **información** **relevante** **sobre** **ellos.**

La idea nació por la conveniencia de contar siempre con un bloc de notas para recordar en qué episodio de esa serie lo habías dejado o poder retomarla luego de varios meses sin seguirla. Si disfrutas en exceso viendo series y películas, es inevitable que llegue un momento en que te preguntes cuáles fueron tus favoritas en el pasado Es evidente que en un lapso breve de tiempo podrás recordar muchos de tus títulos favoritos, pero a lo largo de los años, algunos títulos se mezclarán entre sí y muchos de ellos simplemente se olvidarán, llevándote a preguntarte algún día:

¿Cuál era el nombre de esa película o serie que tanto disfruté?  
  
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 cualquiera puede hacerlo, aunque en realidad esta plataforma interesa especialmente a un dedicado grupo de amantes de las series y el cine, para aquellas personas que deseen tener un control sobre uno de sus hobbies favoritos y un registro al alcance de la mano, interactivo, fácil e intuitivo.

Con esto en mente, toda la marca del sitio web está diseñada para atraer a audiencias de todas las edades y ayudarlos a encontrar fácilmente todas 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 más importantes será el correcto ajuste del centrado de elementos al mostrar una lista de elementos. Esto junto con el diseño de la barra lateral en la página de detalles son las dos tareas que no he conseguido llevar a buen puerto.

Por otro lado, otro de los objetivos originales planeados para la aplicación era que el tráiler también se mostrara en la página de detalles de la serie o película, por lo que esta mejora sería fácil de implementar en próximas actualizaciones.

Además, en la misma página de detalles, también quería incluir fotos de actores, directores, etc. que se encontraban disponibles en la API themoviedb.com, pero que finalmente fueron abandonadas porque se volvieron extremadamente complejas. También puede incluirse en la segunda iteración.  
  
Por otro lado, el proyecto también planteaba inicialmente que en lugar de simplemente mostrar una lista de favoritos, el usuario podría listarlos según muchos más atributos e incluso crear sus propios subgrupos, por ejemplo: “Series que he dejado de ver”., la aplicación aumenta significativamente la usabilidad y el interés de la audiencia potencial; Desafortunadamente, se abandonó debido a su complejidad, pero se podría estudiar su implementación en una segunda iteración del proyecto.

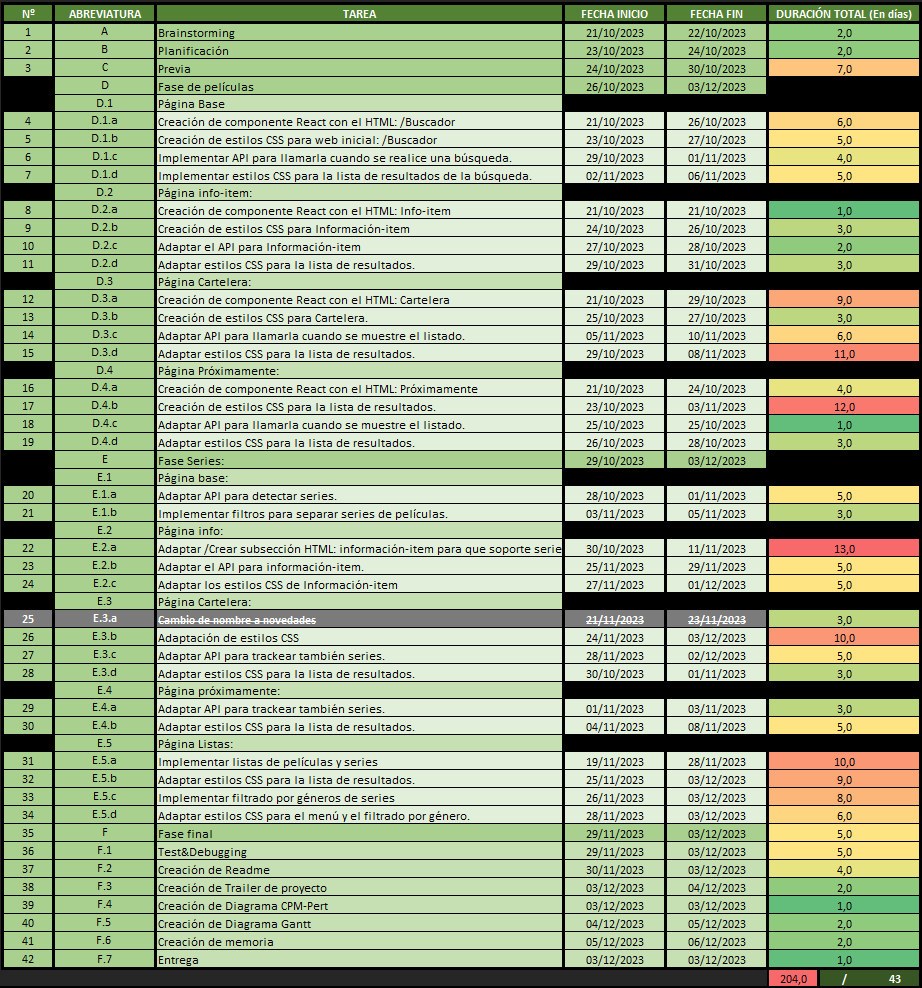
# 2. – Temporalización del proyecto y fases de desarrollo

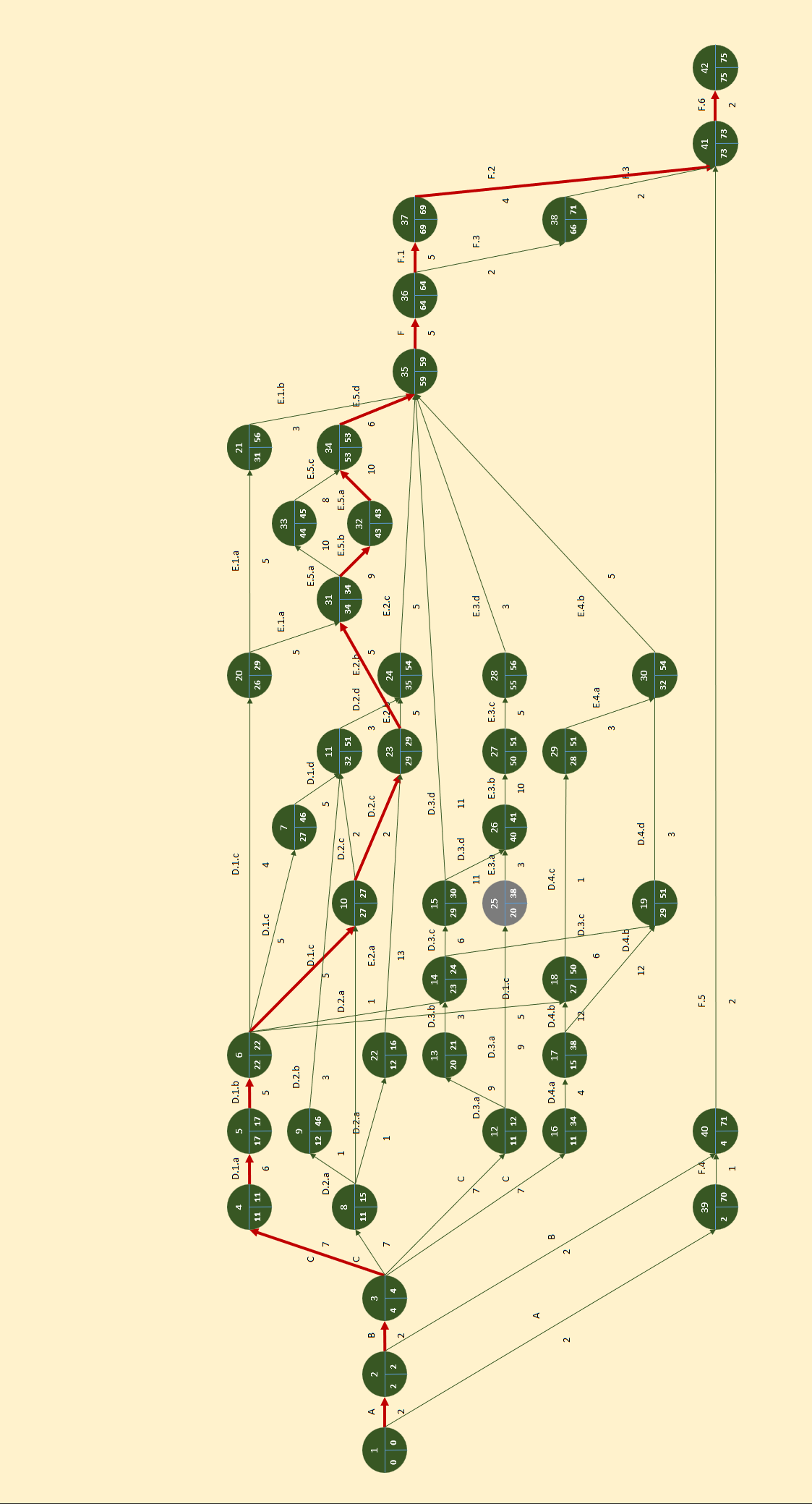
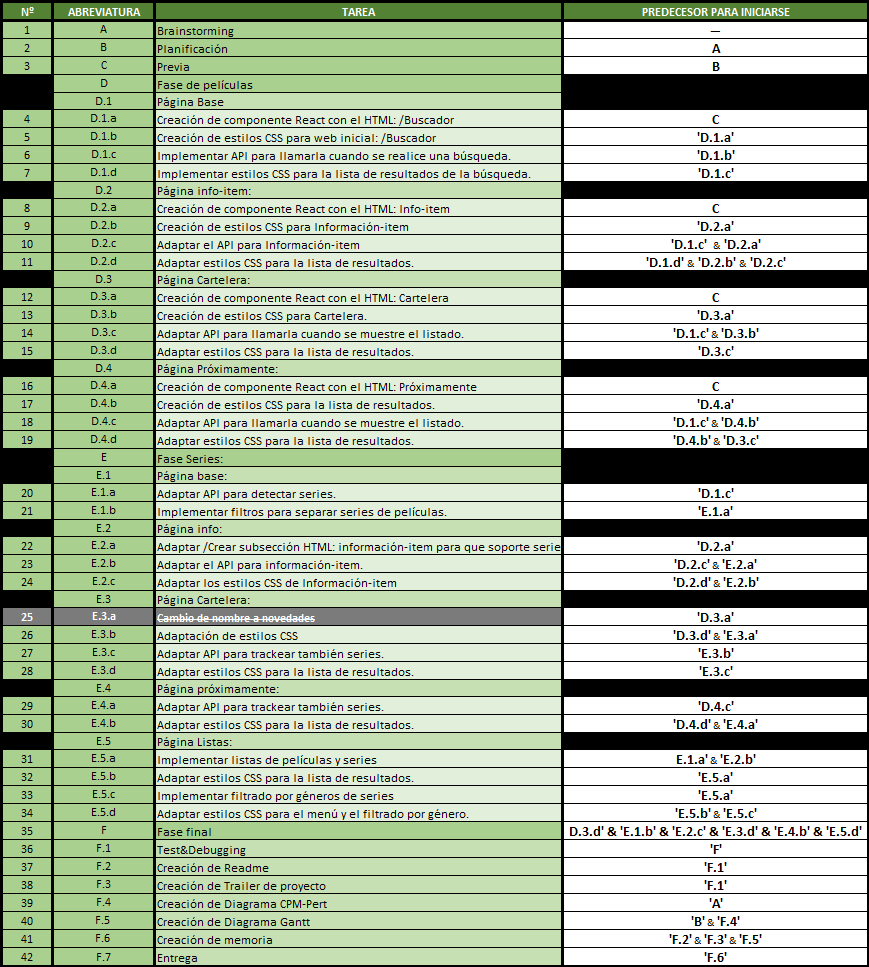
## 2.A - PLANIFICACIÓN:

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:

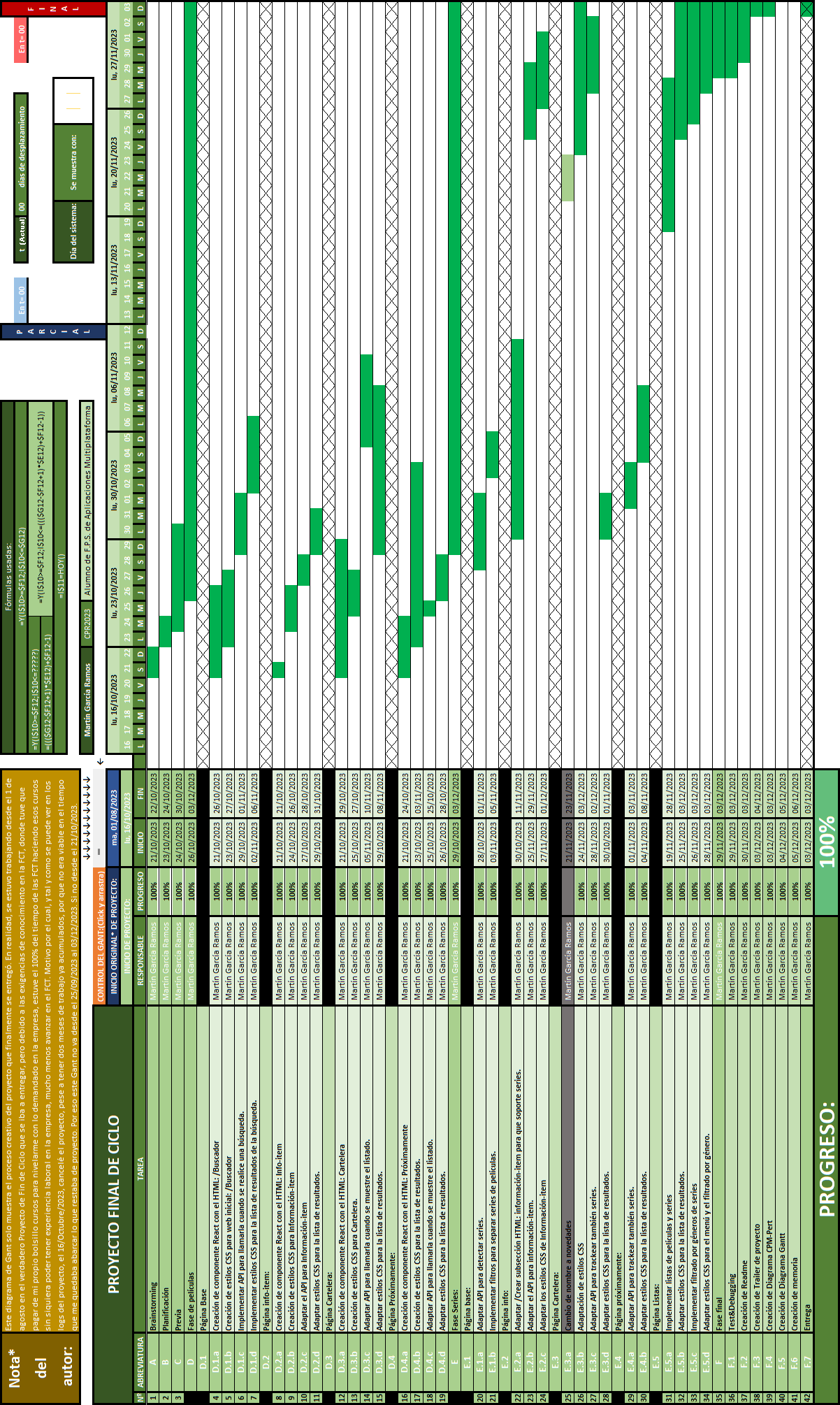
Para ver en mejor detalle mirar el Excel adjunto.





### 2.A.2. GANTT:

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



## 2.B. - RETOS ENCONTRADOS:

La falta de familiaridad con la tecnología utilizada al comienzo del proyecto resultó en la necesidad de rehacer varias secciones en repetidas ocasiones durante el desarrollo, lo cual contribuyó a que el proyecto se volviera más complicado de lo esperado si se hubiera planificado adecuadamente.

También, al trabajar con una tecnología innovadora como ReactJS, me he encontrado con varios desafíos relacionados con el software externo. Uno de estos problemas incluye la identificación de la causa de un error específico que surgía al intentar acceder a la aplicación después de seleccionar otra pestaña en el navegador y regresar a la pestaña donde se encuentra 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***

Después de realizar múltiples pruebas y confirmar que el problema no estaba relacionado con mi proyecto, logré encontrar la solución al desactivar la extensión del navegador: Adblocker Stands. Al parecer, se trata de un fallo que se descubrió en 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:

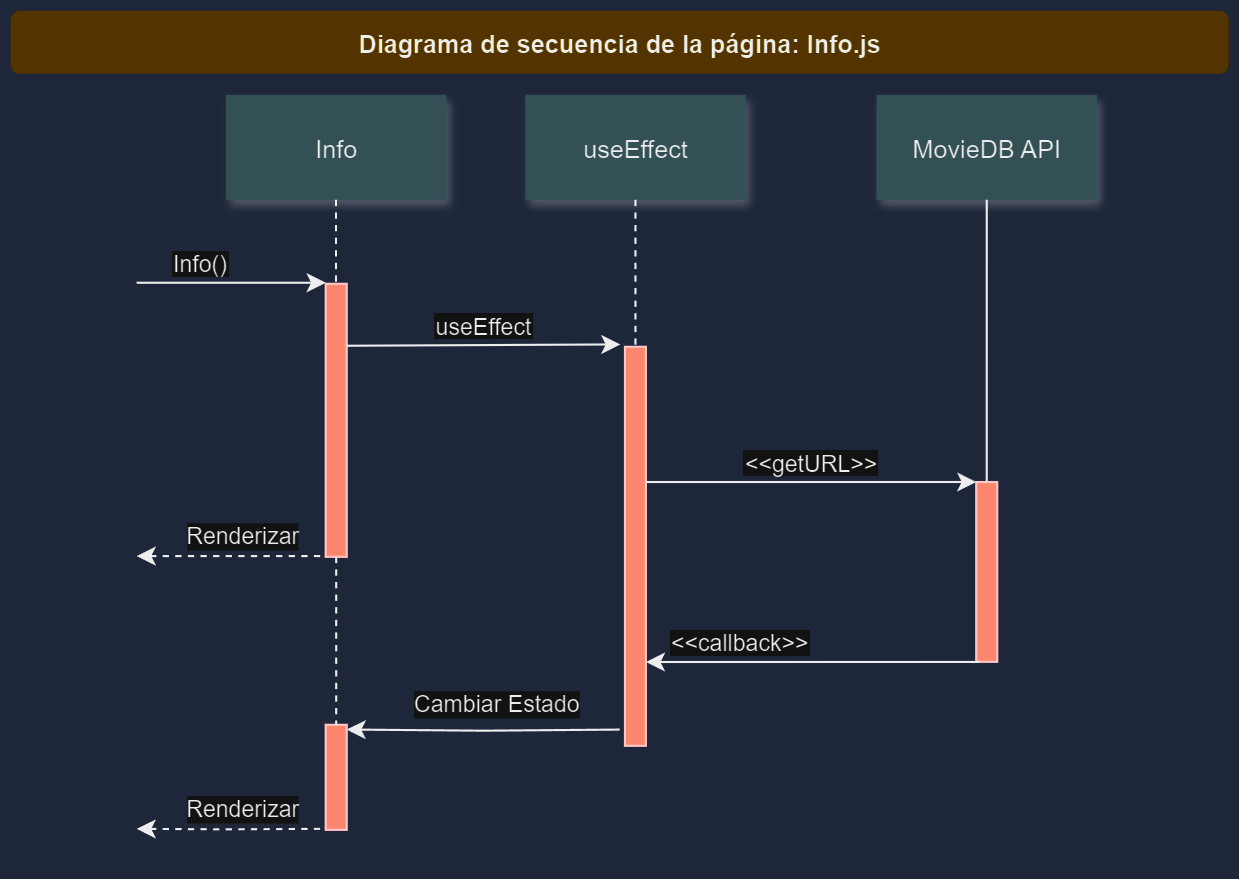
* **Gestor de paquetes npm:** npm es el gestor de paquetes de Node.js y se utiliza para instalar y administrar las dependencias de los proyectos de ReactJS. Se recomienda tener la última versión de npm instalada junto con Node.js.
* **Editor de código:** Se recomienda utilizar un editor de código con 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

En mi proyecto, que está desarrollado en ReactJS, no se utilizan "clases" ni sigue una estructura de programación orientada a objetos en sentido estricto. En su lugar, se utilizan componentes y helpers, los cuales se dividen según su funcionalidad en helpers, pages, entre otros...

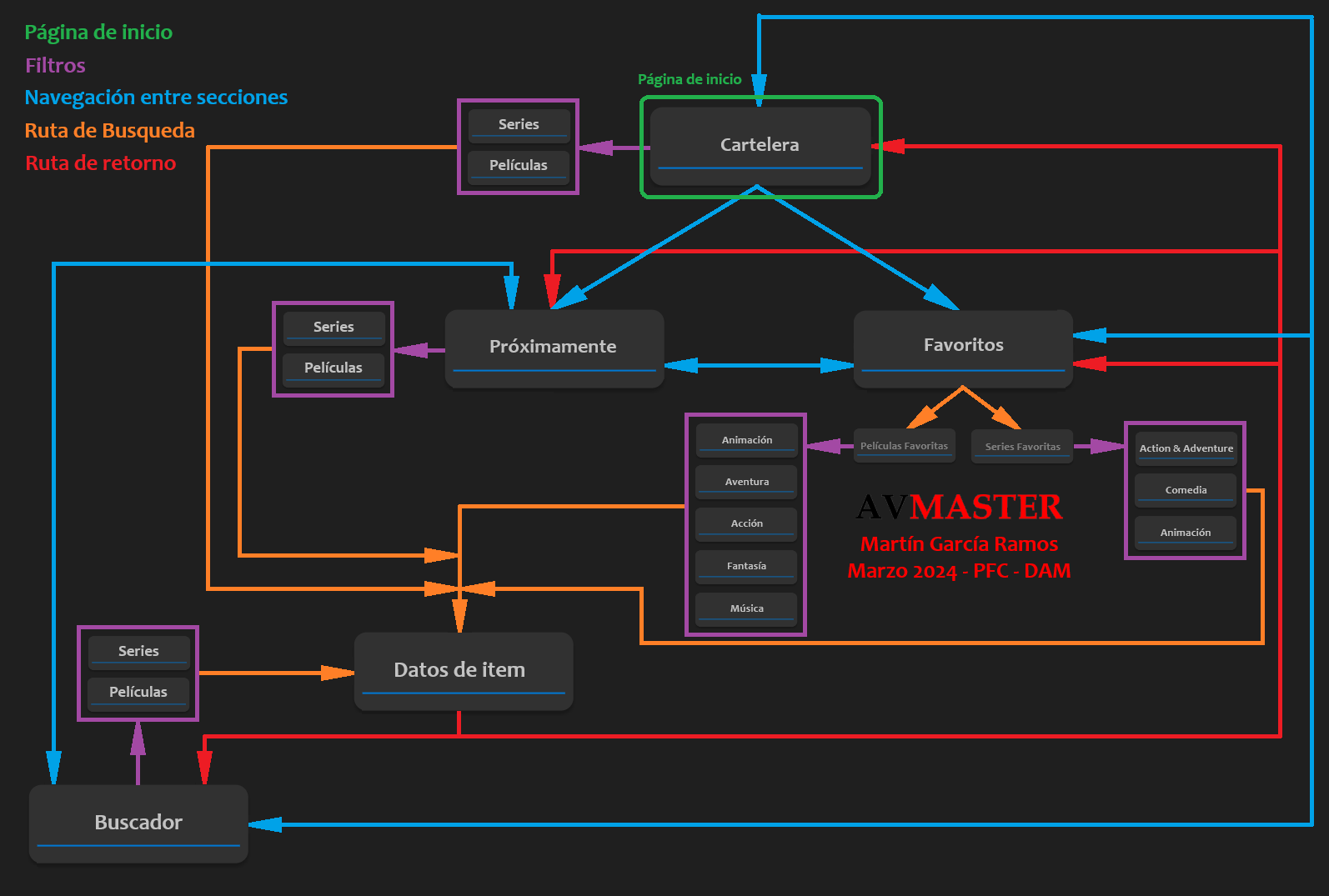
## 4.A. - Diagrama de Secuencia:

En la siguiente representación gráfica se muestra el flujo de la función Info() que, a su vez, activa el useEffect para realizar una solicitud asíncrona y obtener la información de la película. Mientras tanto, la Info generará el renderizado de las páginas de información.



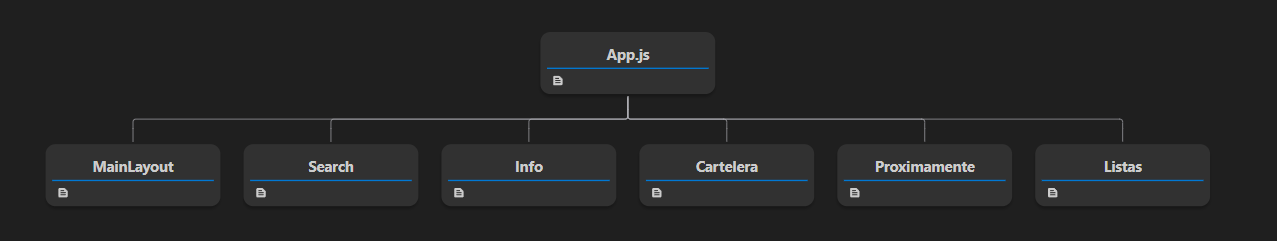
Una vez que MovieDB envíe los datos, el useEffect realizará una solicitud para actualizar el estado de la variable película, lo que provocará que la página de Info se vuelva a renderizar con la información actualizada.)

## 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:

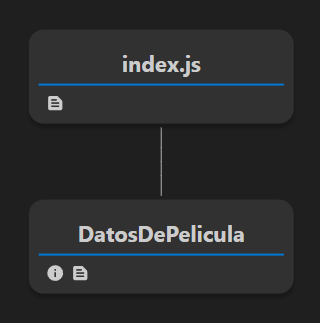


El layout **MainLayout** determina las características de la barra superior de la aplicación.

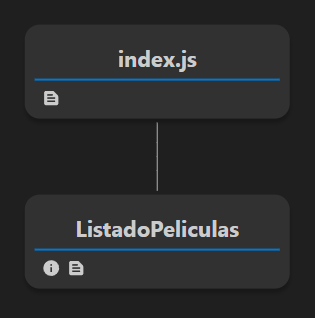
El componente **Search** por su parte, determina las propiedades del recuadro de búsqueda superior en su index que a su vez conecta con el componente ListadoPeliculas, que es el que se encarga de listar cada uno de los ítems, bien sean películas o series que se estén devolviendo en este caso como resultado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



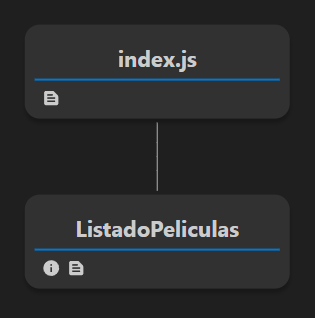
El componente **Info** por su parte, se encarga de obtener información sobre una película o serie de televisión desde una API, y luego renderiza esta información utilizando el componente DatosDePelicula.



El componente **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 el componente **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.