

DPTO. INFORMÁTICA - I.E.S. LA MARISMA
MÓDULO PROYECTO
C.F.G.S.
DESARROLLO DE APLICACIONES WEB

MY THINGS:

Aplicación para gestionar colecciones.

The screenshot displays the 'MY THINGS' application interface. At the top, there's a navigation bar with 'My collection', 'My stats', and 'Releases'. A user profile section on the left shows a dog's head and an 'Add Game' button. Below this is a filter menu with checkboxes for 'Wanted', 'Owned', 'Physical', 'Digital', 'Plus', 'Now', 'Started', 'Finished', 'Completed', and 'Abandoned'. The main content area shows a list of games under the 'Games' tab. Each game entry includes a cover image, title, story and completionist hours, and status buttons (Wanted, Owned, Physical, Digital, Plus, Now, Started, Finished, Completed, Abandoned). The games listed are Moss, Marvel's Spider-Man, Judgment, and Yakuza Kiwami 2.

My collection My stats Releases Test User ▾

Games Books Films

Search something... Search Clear

Sort: Updated (New to Old) ▾ Showing 8 of 19 results.

Add Game

- ☐ Wanted
- ☐ Owned
- ☐ Physical
- ☐ Digital
- ☐ Plus
- ☐ Now
- ☐ Started
- ☐ Finished
- ☐ Completed
- ☐ Abandoned

Filter Clear

Moss

Story: 4 Hours Completionist: 6 Hours

Wanted Physical Plus Started Completed

Owned Digital Now Finished Abandoned

Marvel's Spider-Man

Story: 17 Hours Completionist: 33 Hours

Wanted Physical Plus Started Completed

Owned Digital Now Finished Abandoned

Judgment

Story: 26 Hours Completionist: 98 Hours

Wanted Physical Plus Started Completed

Owned Digital Now Finished Abandoned

Yakuza Kiwami 2

Story: 18½ Hours Completionist: 74½ Hours

Wanted Physical Plus Started Completed

Owned Digital Now Finished Abandoned

Autor/es: Angela Conde López
Fecha: 14/06/2021

Tutor: Santiago Domínguez Zapico

HOJA RESUMEN-PROYECTO

Título del proyecto: My Things: Aplicación para gestionar colecciones.	
Autor: Angela Conde López	Fecha: 14/06/2021
Tutor: Santiago Domínguez Zapico	
Titulación: C.F.G.S. Desarrollo de Aplicaciones Web	
Palabras clave: <ul style="list-style-type: none">• Castellano: aplicación, colección, cliente, servidor, API.• Inglés: application, collection, front-end, back-end, API.	
Resumen del proyecto: <p>(Español) Desarrollo de una aplicación web que permite al usuario añadir artículos a una colección guardada en una base de datos y gestionar dicha colección. La información de los artículos se obtiene de APIs o sitios web proporcionados por terceros.</p> <p>(English) Development of a web application that allows the user to add items to a collection stored in a database and manage that collection. The items information is obtained from APIs or websites provided by third parties.</p>	

INDICE

1. INTRODUCCIÓN.	6
1.1 INTRODUCCIÓN A LA MEMORIA.	6
1.2 DESCRIPCIÓN.....	6
1.3 OBJETIVOS GENERALES.	6
1.4 MOTIVACIONES PERSONALES.	6
1.5 ESTRUCTURA DE LA MEMORIA.....	8
2. ESTUDIO DE VIABILIDAD.	9
2.1 REQUISITOS DEL SISTEMA.	9
2.1.1 INTRODUCCIÓN.	9
2.1.2 REQUISITOS.....	9
2.1.3 CATALOGACIÓN Y PRIORIZACIÓN DE LOS REQUISITOS.	10
2.1.4 RESTRICCIONES.....	10
2.2 PLANIFICACIÓN DEL PROYECTO.	11
2.2.1 TAREAS DEL PROYECTO.	11
2.2.2 PLANIFICACIÓN TEMPORAL.....	12
2.3 EVALUACIÓN DE RIESGOS.	14
2.3.1 LISTA DE RIESGOS.	14
2.3.2 PLAN DE CONTINGENCIA.....	14
3. ANÁLISIS.....	15
3.1 DIAGRAMA DE CASOS DE USO.....	15
3.2 CASOS DE USO.....	16
3.3 DIAGRAMAS DE ACTIVIDAD.....	21
3.4 DIAGRAMA DE CLASES.....	22
3.5 MAPA DE NAVEGACIÓN.	23
3.6 MOCKUPS.....	23
3.7 UX.....	25
3.7.1 NAVEGACIÓN POR LA INTERFAZ.....	25
3.7.2 PALETA DE COLORES.....	25
3.7.3 ORDEN DE LOS BOTONES	25
3.7.4 COLORES DE LOS BOTONES	25
3.7.5 ACCESIBILIDAD.....	25
4. DISEÑO.....	27
4.1 SELECCIÓN DEL ENTORNO DE DESARROLLO.	27
4.2 SELECCIÓN DE LA BASE DE DATOS.....	27
4.3 SELECCIÓN DE LA API EXTERNA.	27
4.3.1 LISTADO DE APIS CANDIDATAS.....	27
4.3.2 SELECCIÓN DE LA API.....	28
4.3.3 EJEMPLO DE PETICIÓN EN GIANTBOMB.....	28
4.3.4 EJEMPLO DE PETICIÓN EN IGDB.....	28
4.4 ESTRUCTURA DE LA BASE DE DATOS.....	32
4.4.1 DIAGRAMA DE LAS TABLAS.....	32
4.4.2 DISPARADORES PARA AUDITORÍA.....	33
4.5 ARQUITECTURA FÍSICA.....	34
4.6 ARQUITECTURA DE SOFTWARE.....	35
4.6.1 CAPA DE PERSISTENCIA.....	35
4.6.1.1 MODELO.....	35

4.6.2 CAPA DE PRESENTACIÓN	36
4.6.2.1 VISTA.	36
4.6.3 CAPA DE LÓGICA	36
4.6.3.1 CONTROLADORES.....	36
4.6.3.2 ENRUTAMIENTO.....	36
5. IMPLEMENTACIÓN.....	37
5.1 TECNOLOGÍAS.	37
5.1.1 BACKEND.	37
5.1.1.1 PHP 7.4.11.	37
5.1.1.2 LARAVEL 8.22.	37
5.1.1.3 MariaDB 10.4.14.	38
5.1.1.4 BLADE.	38
5.1.2 FRONTEND.	38
5.1.2.1 HTML 5.	38
5.1.2.2 CSS 3.	38
5.1.2.3 BOOTSTRAP 4.	39
5.1.2.4 JAVASCRIPT ES6.....	39
5.2 HERRAMIENTAS.	39
5.2.1 VISUAL STUDIO CODE.	39
5.2.2 PHPMYADMIN.	40
5.2.3 COMPOSER.	40
5.2.4 NPM.	41
5.2.5 GIT / GITHUB.....	42
5.2.6 ARTISAN.	42
5.2.7 PHPUNIT.	43
5.2.8 LARAVEL DUSK.	43
5.2.9 SONARQUBE	43
5.3 ESTRUCTURA.	44
5.3.1 APP.....	44
5.3.2 CONFIG.	45
5.3.3 DATABASE.....	45
5.3.4 PUBLIC.	45
5.3.5 RESOURCES.....	46
5.3.6 ROUTES.	46
5.3.7 STORAGE.....	46
5.3.8 TESTS.....	46
5.3.9 VENDOR.	46
6. PRUEBAS.	47
6.1 PRUEBAS AUTOMATIZADAS.	47
6.1.1 PRUEBAS DE CARACTERÍSTICAS.	47
6.1.2 PRUEBAS UNITARIAS.	48
6.2 PRUEBAS MANUALES.	50
6.2.1 VISUALIZACIÓN EN DISTINTOS DISPOSITIVOS.	50
6.2.2 VISUALIZACIÓN EN DISTINTOS NAVEGADORES.....	60
6.3 ANÁLISIS ESTÁTICO.....	66
6.4 RESULTADOS OBTENIDOS.	67
7. CONCLUSIONES.	68
7.1 POSIBLES AMPLIACIONES Y MODIFICACIONES.	68
7.2 LICENCIA.....	68
7.3 VALORACIÓN PERSONAL.....	70

8. BIBLIOGRAFÍA.....	71
8.1 APIS	71
8.2 HERRAMIENTAS	71
8.3 LARAVEL	71
8.4 AUTENTICACIÓN.....	71
8.5 ORM	71
8.6 EMAIL	71
8.7 CRAWLER.....	71
8.8 OAUTH.....	71
8.9 UX Y ACCESIBILIDAD	72
8.10 TESTS Y PRUEBAS.....	72
8.11 LICENCIAS.....	72
8.12 DOCUMENTACIÓN	72
8.13 GESTIÓN DE PROYECTO.....	72
8.14 UML.....	72

1. INTRODUCCIÓN.

1.1 INTRODUCCIÓN A LA MEMORIA.

En este documento explico en qué consiste el proyecto que he desarrollado, desde la elección del mismo hasta su finalización, pasando por su planificación, diseño e implementación.

1.2 DESCRIPCIÓN.

Desarrollo de una aplicación web que permite al usuario añadir artículos a una colección guardada en una base de datos y gestionar dicha colección.

El usuario puede añadir artículos, modificar su información sobre posesión y uso del artículo o eliminar el artículo de su colección.

El usuario podrá ver una lista detallada de sus artículos con información sobre estos. Esta lista podrá ser filtrada y ordenada por varios parámetros, así como ejecutar una búsqueda sobre ella.

La información de los artículos se obtiene de APIs y/o sitios webs proporcionados por terceros.

Para la demostración práctica de esta aplicación se ha implementado únicamente la colección de videojuegos de Playstation 4. Cualquier otra colección implementada en un futuro seguirá exactamente el mismo patrón que ésta y únicamente será necesario cambiar la API de la cual se extrae la información y la consulta a realizar sobre ella.

1.3 OBJETIVOS GENERALES.

El objetivo de este proyecto es aplicar y ampliar los conocimientos adquiridos durante el ciclo formativo.

En especial centro mi proyecto en aprender a consumir datos proporcionados por servicios de terceros.

1.4 MOTIVACIONES PERSONALES.

Mi interés personal era desarrollar una aplicación en la que trabajar con APIs y ampliar mis conocimientos sobre ellas.

Las APIs están cobrando más y más fuerza en las grandes empresas, sobre todo en el sector de la banca tras la normativa europea obligatoria de servicios de pago (PSD2).

Las APIs son empleadas para el transporte de datos de forma eficiente. Pero no solamente son una gran herramienta, sino que para muchas empresas sus servicios de API constituyen una de sus mayores fuentes de financiación al vender estos datos a terceros en un mercado en el que el auge del big data está incrementando el valor de los datos de forma exponencial.

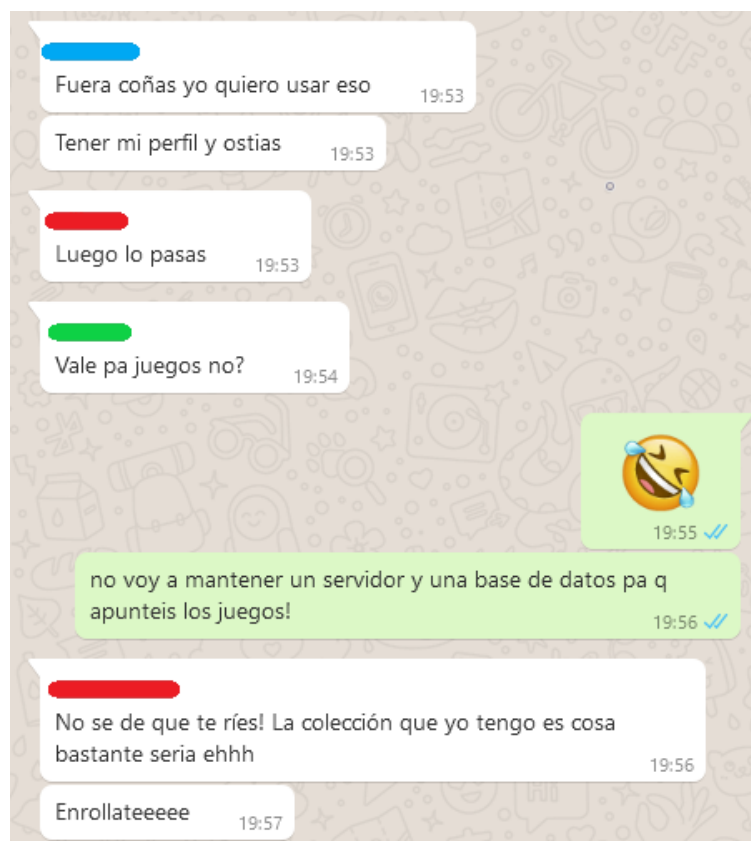
El hecho de que las APIs sean un bien muy valioso, me limitó a la hora de buscar un tema sobre el que plantear mi proyecto, ya que la mayoría de buenas APIs proporcionadas por terceros están orientadas al consumo por parte de empresas y su uso tiene un coste.

La mayoría de buenas APIs gratuitas que pude encontrar proporcionan datos sobre productos de ocio, entretenimiento y cultura.

Esto me llevó a decidir desarrollar una biblioteca de productos de ocio para uso personal.

Investigando sobre el tema llegué a la conclusión de que:

- Existen varios servicios web que permiten catalogar tus colecciones, pero están orientados al aspecto social (compartir, comentar, comparar).
- Para uso propio existe software instalable en un ordenador, así como apps para dispositivos móviles, con las limitaciones de uso que ello conlleva.
- Los usuarios que querían gestionar sus colecciones online de manera sencilla recurrían a hojas de cálculo en servicios de SaaS, como Google Docs.
- Los usuarios reaccionaban positivamente ante la idea de mi proyecto:



1.5 ESTRUCTURA DE LA MEMORIA.

Este documento está dividido en varias secciones principales:

1. Introducción:

Una breve descripción del proyecto, el objetivo del mismo y mis motivaciones para realizarlo.

2. Estudio de viabilidad:

Un estudio de los requisitos del sistema, los recursos que serán necesarios, las tareas a realizar y la planificación temporal.

3. Análisis:

Los casos de uso, diagramas de actividad, diagramas de clases, mapa de navegación, mockups, etc.

4. Diseño:

El entorno de desarrollo, la estructura de la base de datos, etc.

5. Implementación:

Una explicación de la implementación de las distintas capas del proyecto.

6. Pruebas:

Las pruebas realizadas, su propósito y sus resultados.

7. Conclusiones:

Las conclusiones finales tras completar el proyecto.

8. Bibliografía:

Los recursos bibliográficos empleados durante el proyecto.

2. ESTUDIO DE VIABILIDAD.

2.1 REQUISITOS DEL SISTEMA.

2.1.1 INTRODUCCIÓN.

La aplicación consume servicios externos proporcionados por terceros para obtener la información de los artículos.

El uso de la aplicación requiere que el usuario se registre en el sistema.

La aplicación está enfocada a usuarios que posean al menos un manejo básico de aplicaciones web ya sea en ordenadores o en dispositivos portátiles como el teléfono móvil o tablet.

2.1.2 REQUISITOS.

La interfaz debe ser diseñada para un usuario con conocimientos básicos de aplicaciones web, de forma intuitiva y fácil de usar.

La interfaz debe ser responsive para facilitar su uso en distintas plataformas.

La aplicación debe permitir el registro y autenticación de usuarios.

La aplicación debe permitir al usuario modificar sus datos personales o eliminar su cuenta.

El sistema debe poder recibir la información requerida de la fuente externa y almacenarla en la base de datos propia.

La aplicación debe poder extraer los datos almacenados en la base de datos propia y enviarlos a la vista.

La aplicación debe permitir añadir nuevos objetos a la base de datos, modificarlos y eliminarlos.

La aplicación debe permitir filtrar los objetos visibles por distintos parámetros.

La aplicación debe permitir ordenar la lista de objetos visibles por distintos parámetros.

La aplicación debe permitir ejecutar una búsqueda sobre la lista de objetos visibles.

2.1.3 CATALOGACIÓN Y PRIORIZACIÓN DE LOS REQUISITOS.

Utilizando la técnica de priorización de requisitos MoSCoW se catalogan en “Must have”, “Should have”, “Could have” y “Would like but won’t get”.

MUST:

- Registro y autenticación de usuarios.
- Recibir información de fuente externa.
- Almacenar la información externa en la base de datos propia.
- Añadir objetos nuevos.

SHOULD:

- Modificar o eliminar la cuenta de usuario.
- Editar los objetos añadidos.
- Eliminar los objetos añadidos.
- Filtrar los objetos visibles.
- Interfaz responsive.

COULD:

- Registro y autenticación mediante OAuth.
- Ver estadísticas propias de uso de los objetos de la colección del usuario.
- Ver calendario de lanzamientos próximos.
- Añadir API propia para la gestión de usuarios y los objetos de su colección.
- Permitir al usuario compartir toda o partes de su colección mediante una sección accesible de forma pública.
- Añadir enlaces monetizados para la compra de los objetos marcados como deseados.

WON'T:

- Integración de servicios de terceros, como la API de afiliados de Amazon, para obtener el precio actual del objeto de la colección.

2.1.4 RESTRICCIONES.

Para acceder a la aplicación será necesario disponer de un dispositivo con acceso a Internet, ya sea un ordenador o un teléfono móvil.

Para poder hacer uso de la aplicación será necesario registrarse y/o identificarse.

La aplicación tendrá que ser accesible desde los navegadores más utilizados, como Mozilla Firefox, Google Chrome o Microsoft Edge.

Las contraseñas de los usuarios deberán almacenarse de forma encriptada.

Las modificaciones sobre el perfil del usuario o los objetos de su colección solamente deberán poderse realizar por el dueño de los mismos debidamente autenticado.

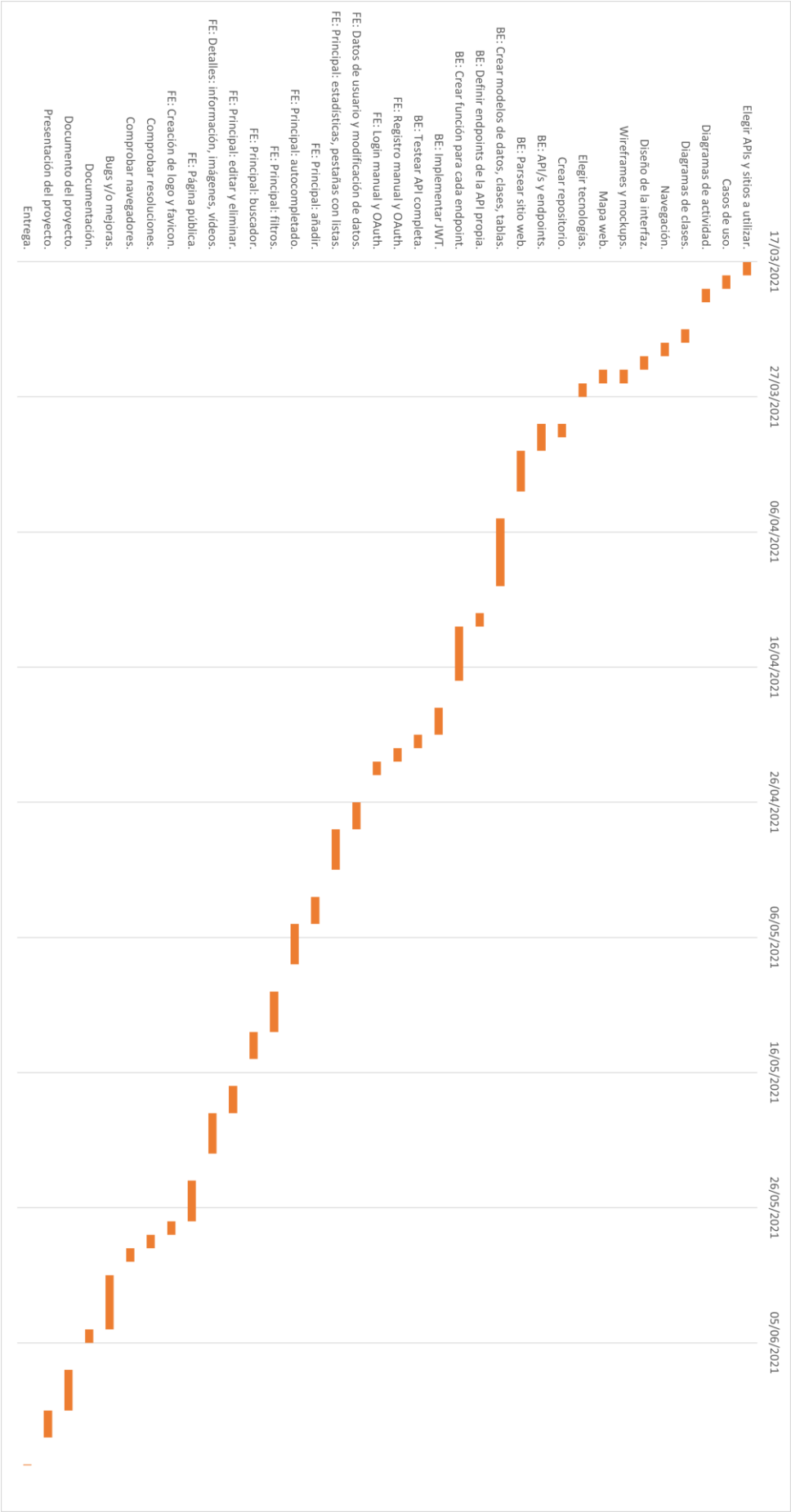
2.2 PLANIFICACIÓN DEL PROYECTO.

2.2.1 TAREAS DEL PROYECTO.

- Elegir APIs y sitios a utilizar.
- Casos de uso.
- Diagramas de actividad.
- Diagramas de clases.
- Navegación.
- Diseño de la interfaz.
- Wireframes y mockups.
- Mapa web.
- Elegir tecnologías.
- Crear repositorio.
- API/s y endpoints.
- Parsear sitio web.
- Crear modelos de datos, clases, tablas.
- *Definir endpoints de la API propia.*
- *Crear función para cada endpoint.*
- *Implementar JWT.*
- *Testear API completa.*
- Registro manual y OAuth.
- Login manual y OAuth.
- Datos de usuario y modificación de datos.
- Principal: estadísticas, pestañas con listas.
- Principal: añadir.
- Principal: autocompletado.
- Principal: filtros.
- Principal: buscador.
- Principal: editar y eliminar.
- Detalles: información, imágenes, vídeos.
- Página pública.
- Términos de uso y Política de privacidad.
- Creación de logo y favicon.
- Testeo: Comprobar resoluciones.
- Testeo: Comprobar navegadores.
- Testeo: Bugs y/o mejoras.
- Documentación.

2.2.2 PLANIFICACIÓN TEMPORAL.

TAREA	COMIENZO	FIN	DURACION
Elegir APIs y sitios a utilizar.	17/03/2021	17/03/2021	1
Casos de uso.	18/03/2021	18/03/2021	1
Diagramas de actividad.	19/03/2021	19/03/2021	1
Diagramas de clases.	22/03/2021	22/03/2021	1
Navegación.	23/03/2021	23/03/2021	1
Diseño de la interfaz.	24/03/2021	24/03/2021	1
Wireframes y mockups.	25/03/2021	25/03/2021	1
Mapa web.	25/03/2021	25/03/2021	1
Elegir tecnologías.	26/03/2021	26/03/2021	1
Crear repositorio.	29/03/2021	29/03/2021	1
BE: API/s y endpoints.	29/03/2021	30/03/2021	2
BE: Parsear sitio web.	31/03/2021	02/04/2021	3
BE: Crear modelos de datos, clases, tablas.	05/04/2021	09/04/2021	5
<i>BE: Definir endpoints de la API propia.</i>	12/04/2021	12/04/2021	1
<i>BE: Crear función para cada endpoint.</i>	13/04/2021	16/04/2021	4
<i>BE: Implementar JWT.</i>	19/04/2021	20/04/2021	2
<i>BE: Testear API completa.</i>	21/04/2021	21/04/2021	1
FE: Registro manual y OAuth.	22/04/2021	22/04/2021	1
FE: Login manual y OAuth.	23/04/2021	23/04/2021	1
FE: Datos de usuario y modificación de datos.	26/04/2021	27/04/2021	2
FE: Principal: estadísticas, pestañas con listas.	28/04/2021	30/04/2021	3
FE: Principal: añadir.	03/05/2021	04/05/2021	2
FE: Principal: autocompletado.	05/05/2021	07/05/2021	3
FE: Principal: filtros.	10/05/2021	12/05/2021	3
FE: Principal: buscador.	13/05/2021	14/05/2021	2
FE: Principal: editar y eliminar.	17/05/2021	18/05/2021	2
FE: Detalles: información, imágenes, vídeos.	19/05/2021	21/05/2021	3
FE: Página pública.	24/05/2021	26/05/2021	3
FE: Creación de logo y favicon.	27/05/2021	27/05/2021	1
Comprobar resoluciones.	28/05/2021	28/05/2021	1
Comprobar navegadores.	29/05/2021	29/05/2021	1
Bugs y/o mejoras.	31/05/2021	03/06/2021	4
Documentación.	04/06/2021	04/06/2021	1
Documento del proyecto.	07/06/2021	09/06/2021	3
Presentación del proyecto.	10/06/2021	11/06/2021	2
Entrega.	14/06/2021	14/06/2021	1



2.3 EVALUACIÓN DE RIESGOS.

2.3.1 LISTA DE RIESGOS.

El mayor riesgo es el cese de prestación del servicio de la API externa, que inutilizaría la funcionalidad de añadir nuevos objetos que no se encuentren ya en la base de datos propia. La posibilidad de ocurrir esto es muy baja, ya que se ha seleccionado una API profesional proporcionada por una de las mayores empresas de su sector.

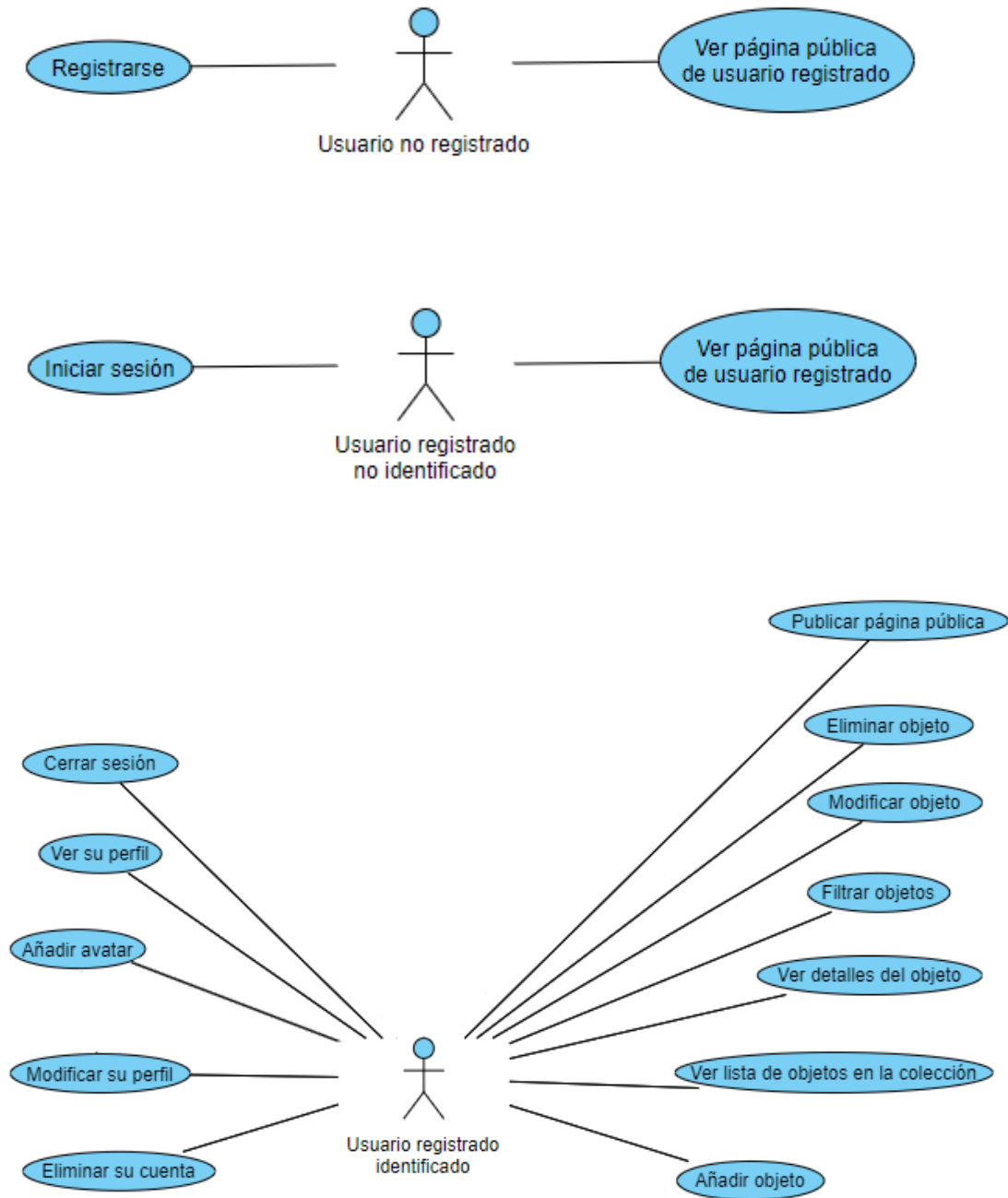
2.3.2 PLAN DE CONTINGENCIA.

Ante el cese de prestación del servicio de la API externa se ha seleccionado otra API externa de características similares y con muchos años de servicio, que podría proporcionar los datos necesarios.

El cambio se llevaría a cabo de forma rápida y sencilla y únicamente quedaría inutilizada de forma temporal la funcionalidad de añadir nuevos objetos.

3. ANÁLISIS.

3.1 DIAGRAMA DE CASOS DE USO.



3.2 CASOS DE USO.

Caso de uso	Registrarse
Descripción	Permite al usuario registrarse en la aplicación web accediendo a las funcionalidades que proporciona.
Actores	Usuario no registrado.
Precondiciones	Ninguna.
Flujo normal	<ol style="list-style-type: none"> 1. El usuario selecciona la opción "Register". 2. El usuario introduce sus datos en el formulario de registro. 3. El usuario pulsa "Register". 4. Se validan si los datos son correctos.
Postcondiciones	Un nuevo usuario es creado en la base de datos. El usuario es redirigido a la pantalla principal con la sesión ya iniciada.
Excepciones	Si el usuario ha introducido datos incorrectos, el sistema le informará y le permitirá volver a intentar el registro, manteniendo los datos correctos introducidos en los campos del formulario.

Caso de uso	Iniciar sesión
Descripción	Permite al usuario iniciar sesión en la aplicación web.
Actores	Usuario registrado.
Precondiciones	Registrarse.
Flujo normal	<ol style="list-style-type: none"> 1. El usuario selecciona la opción "Login". 2. El usuario introduce su usuario y contraseña. 3. El usuario pulsa "Login". 4. Se validan si los datos son correctos.
Postcondiciones	El usuario es redirigido a la pantalla principal con la sesión iniciada.
Excepciones	<p>Si el usuario ha introducido datos incorrectos, el sistema le informará y le permitirá volver a intentar iniciar sesión 2 veces más.</p> <p>Si el usuario introduce datos incorrectos 3 veces seguidas el sistema le hará esperar 5 minutos para volver a intentarlo.</p>

Caso de uso	Cerrar sesión
Descripción	Permite al usuario cerrar sesión en la aplicación web.
Actores	Usuario registrado.
Precondiciones	Iniciar sesión.
Flujo normal	<ol style="list-style-type: none"> 1. El usuario selecciona la opción "Logout".
Postcondiciones	La sesión se cierra y el usuario es redirigido a la pantalla de registro / inicio de sesión.
Excepciones	Ninguna.

Caso de uso	Ver su perfil
Descripción	Permite al usuario ver sus datos de perfil.
Actores	Usuario registrado.
Precondiciones	Iniciar sesión.
Flujo normal	1. El usuario selecciona la opción "My profile".
Postcondiciones	El usuario es redirigido a una nueva página con sus datos.
Excepciones	Ninguna.

Caso de uso	Modificar su perfil
Descripción	Permite al usuario modificar sus datos de perfil.
Actores	Usuario registrado.
Precondiciones	Iniciar sesión.
Flujo normal	<ol style="list-style-type: none"> 1. El usuario selecciona la opción "Edit my profile". 2. El usuario introduce sus nuevos datos en el formulario. 3. El usuario pulsa "Save changes". 4. Se validan si los datos son correctos.
Postcondiciones	<p>Los cambios se guardan en la base de datos.</p> <p>El usuario es redirigido a su página de perfil donde puede comprobar que sus datos han sido modificados.</p>
Excepciones	Si el usuario ha introducido datos incorrectos, el sistema le informará y le permitirá volver a intentar, manteniendo los datos correctos introducidos en los campos del formulario.

Caso de uso	Añadir avatar
Descripción	Permite al usuario añadir una imagen de perfil.
Actores	Usuario registrado.
Precondiciones	Iniciar sesión.
Flujo normal	<ol style="list-style-type: none"> 1. El usuario selecciona la opción "Upload avatar". 2. El usuario selecciona la imagen en su ordenador. 3. El usuario pulsa "Upload". 4. Se valida si la imagen tiene el formato y peso correcto.
Postcondiciones	<p>La imagen se guarda en el servidor.</p> <p>El nombre de la imagen se guarda en la base de datos.</p> <p>El usuario es redirigido a la página de perfil donde puede comprobar que su avatar ha sido añadido.</p>
Excepciones	Si el usuario ha introducido una imagen con formato incorrecto, el sistema le informará y le permitirá volver a intentar.

Caso de uso	Eliminar su cuenta
Descripción	Permite al usuario eliminar su cuenta.
Actores	Usuario registrado.
Precondiciones	Iniciar sesión.
Flujo normal	1. El usuario selecciona la opción "Delete my account". 2. El usuario acepta la confirmación pulsando en "Yes, delete my account permanently".
Postcondiciones	La sesión se cierra y el usuario es redirigido a la pantalla de registro / inicio de sesión. El usuario es marcado como borrado en la base de datos.
Excepciones	Ninguna.

Caso de uso	Ver página pública de usuario registrado
Descripción	Permite al usuario ver la página pública de un usuario registrado.
Actores	Usuario no registrado / Usuario registrado.
Precondiciones	Ninguna.
Flujo normal	1. El usuario accede a la URL proporcionada por el usuario registrado.
Postcondiciones	Ninguna.
Excepciones	Si la URL es incorrecta se mostrará un error 404 personalizado indicándole que compruebe que está bien escrita.

Caso de uso	Publicar página pública
Descripción	Permite al usuario publicar una página pública.
Actores	Usuario registrado.
Precondiciones	Iniciar sesión.
Flujo normal	1. El usuario selecciona la opción "Make my wishlist public".
Postcondiciones	Se cambia la condición de acceso a la página pública del usuario para permitir acceso a cualquier Usuario no registrado o Usuario registrado. Se proporciona la URL al usuario.
Excepciones	Ninguna.

Caso de uso	Ver lista de objetos de la colección
Descripción	Permite al usuario ver la lista de objetos de su colección.
Actores	Usuario registrado.
Precondiciones	Iniciar sesión.
Flujo normal	1. El usuario inicia sesión o selecciona la opción "Home".
Postcondiciones	Ninguna.
Excepciones	Si el usuario no es dueño de la colección a la que intenta acceder se mostrará un error 403 personalizado.

Caso de uso	Añadir objeto
Descripción	Permite al usuario añadir un objeto a su colección.
Actores	Usuario registrado.
Precondiciones	Iniciar sesión.
Flujo normal	<ol style="list-style-type: none"> 1. El usuario selecciona la opción "Add new". 2. El usuario busca el objeto en el buscador, lo selecciona y acepta.
Postcondiciones	<p>Si el objeto existe en la base de datos propia, se añade a la tabla intermedia de objetos del usuario.</p> <p>Si el objeto no existe en la base de datos propia, se solicita la información a la API externa, se parsea la web externa, se crea el objeto, se guarda en la base de datos propia y se añade a la tabla intermedia de objetos del usuario.</p> <p>El usuario es redirigido a la pantalla principal donde puede comprobar que el objeto ha sido añadido a su colección.</p>
Excepciones	Si el objeto no se puede encontrar ni en la base de datos propia ni en la API externa, se informará al usuario solicitándole contactar con soporte para añadir el objeto de forma manual.

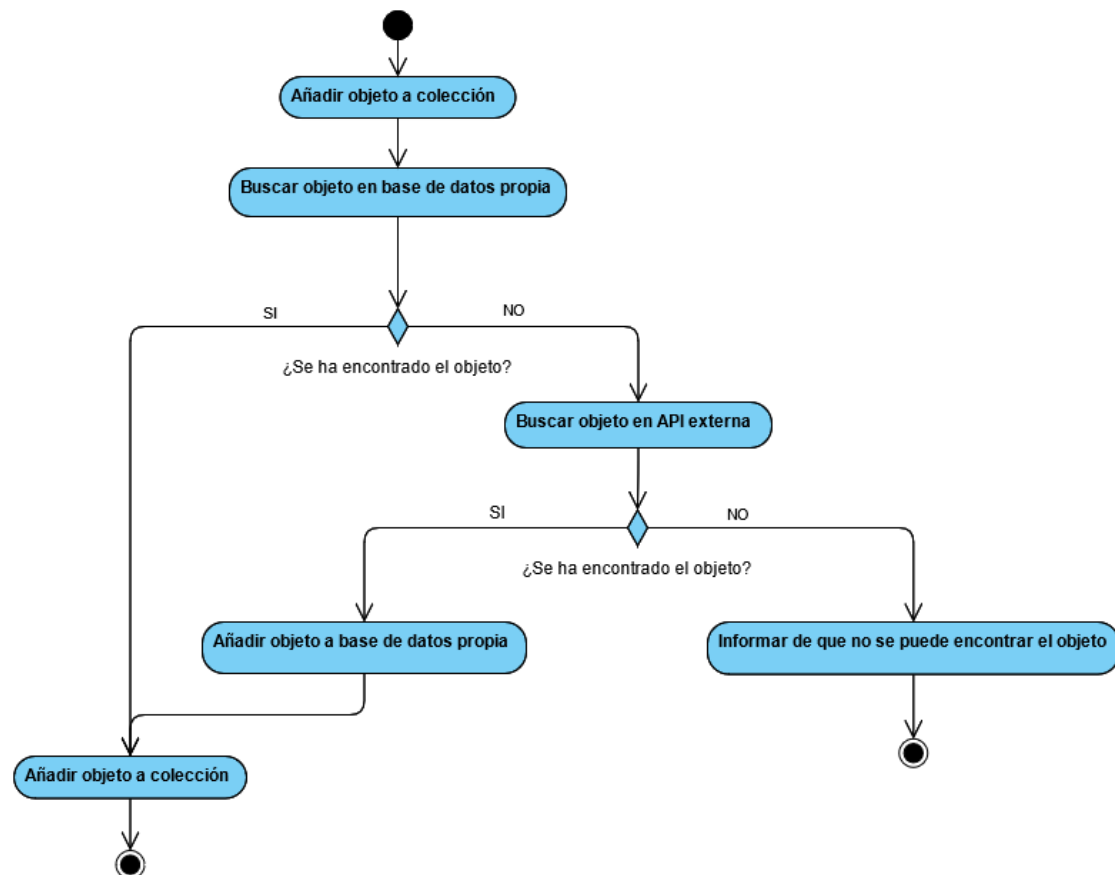
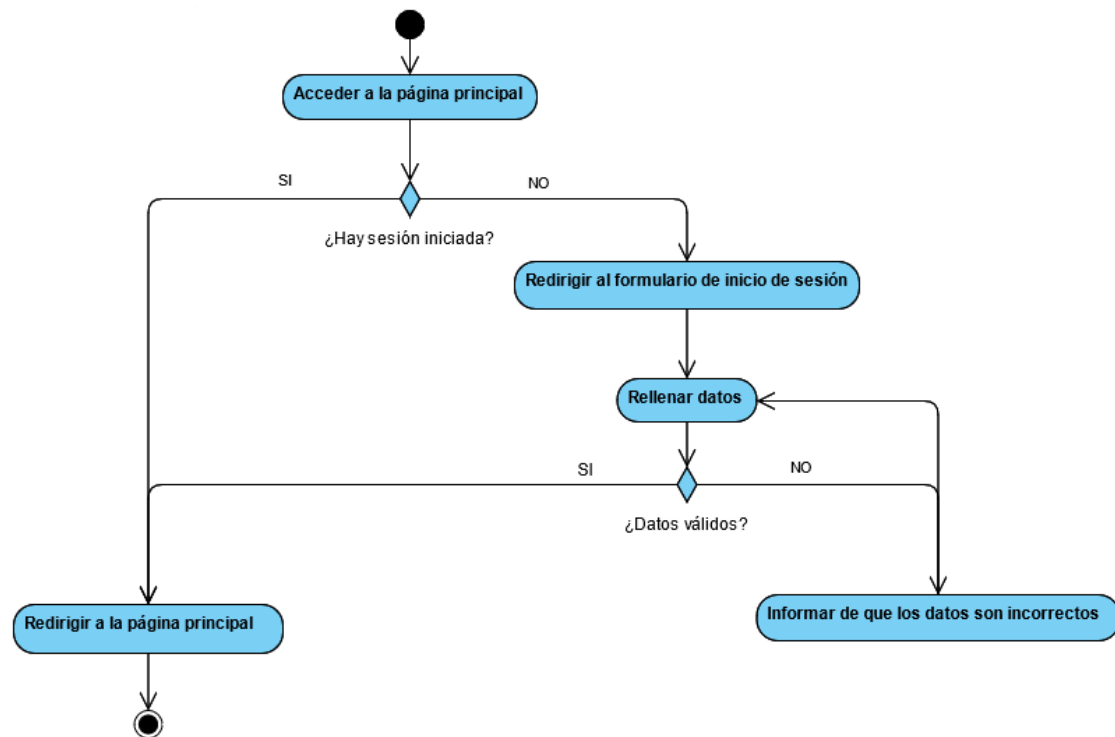
Caso de uso	Modificar objeto
Descripción	Permite al usuario modificar sus datos de uso del objeto.
Actores	Usuario registrado.
Precondiciones	<p>Iniciar sesión.</p> <p>Añadir un objeto.</p>
Flujo normal	<ol style="list-style-type: none"> 1. El usuario selecciona la opción "Edit". 2. El usuario modifica los datos de uso y pulsa "Save changes".
Postcondiciones	El usuario es redirigido a la pantalla principal donde puede comprobar que sus datos de uso del objeto han sido modificados.
Excepciones	Ninguna.

Caso de uso	Eliminar objeto
Descripción	Permite al usuario eliminar un objeto de su colección.
Actores	Usuario registrado.
Precondiciones	<p>Iniciar sesión.</p> <p>Añadir un objeto.</p>
Flujo normal	<ol style="list-style-type: none"> 1. El usuario selecciona la opción "Remove". 2. El usuario confirma eliminar el objeto pulsando "Yes, remove it from my collection".
Postcondiciones	El usuario es redirigido a la pantalla principal donde puede comprobar que el objeto ha sido eliminado de su colección.
Excepciones	Ninguna.

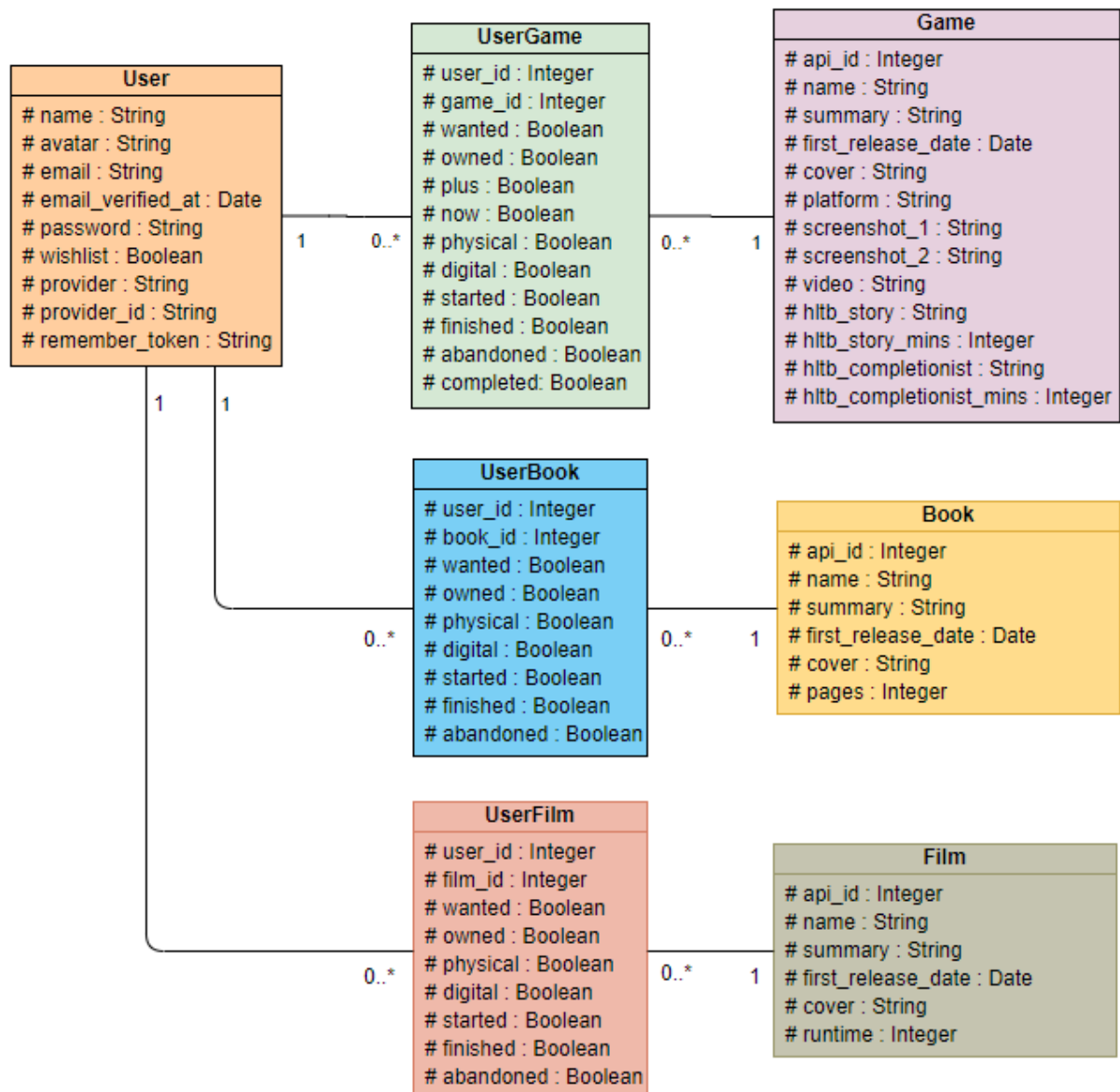
Caso de uso	Filtrar objetos
Descripción	Permite al usuario filtrar la lista de objetos de su colección.
Actores	Usuario registrado.
Precondiciones	Iniciar sesión. Añadir un objeto.
Flujo normal	1. El usuario selecciona o deselecciona los checkboxes de los filtros de los objetos que quiere ver o no ver en la lista.
Postcondiciones	La lista muestra únicamente los objetos marcados como visibles mediante los filtros.
Excepciones	Ninguna.

Caso de uso	Ordenar objetos
Descripción	Permite al usuario ordenar la lista de objetos de su colección.
Actores	Usuario registrado.
Precondiciones	Iniciar sesión. Añadir un objeto.
Flujo normal	1. El usuario selecciona de un desplegable una de las opciones de ordenación disponibles.
Postcondiciones	La lista muestra los objetos ordenados según la opción seleccionada. Si el usuario ha seleccionado filtros, la lista muestra únicamente los objetos marcados como visibles mediante los filtros.
Excepciones	Ninguna.

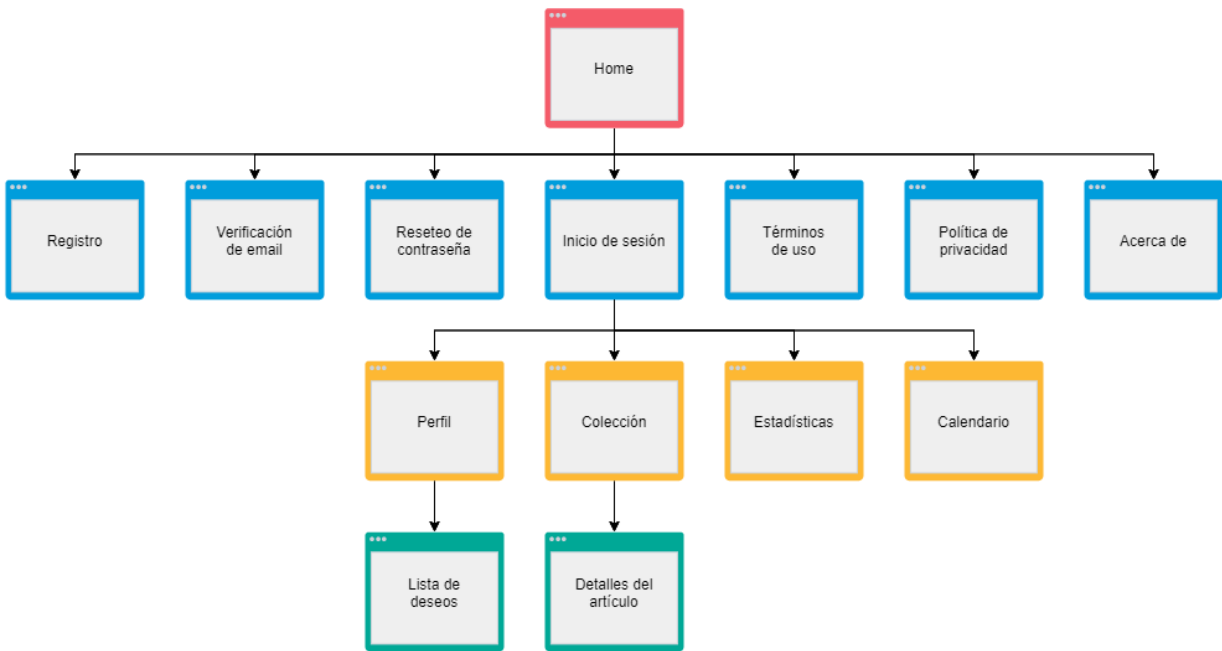
Caso de uso	Ver detalles del objeto
Descripción	Permite al usuario ver detalles de un objeto de su colección.
Actores	Usuario registrado.
Precondiciones	Iniciar sesión. Añadir un objeto.
Flujo normal	1. El usuario selecciona la opción "View details".
Postcondiciones	El usuario es redirigido a la pantalla de detalles de ese objeto.
Excepciones	Si el usuario intenta ver los detalles de un objeto que no está en su colección se mostrará un error 404 personalizado indicándole que el objeto no existe.

3.3 DIAGRAMAS DE ACTIVIDAD.

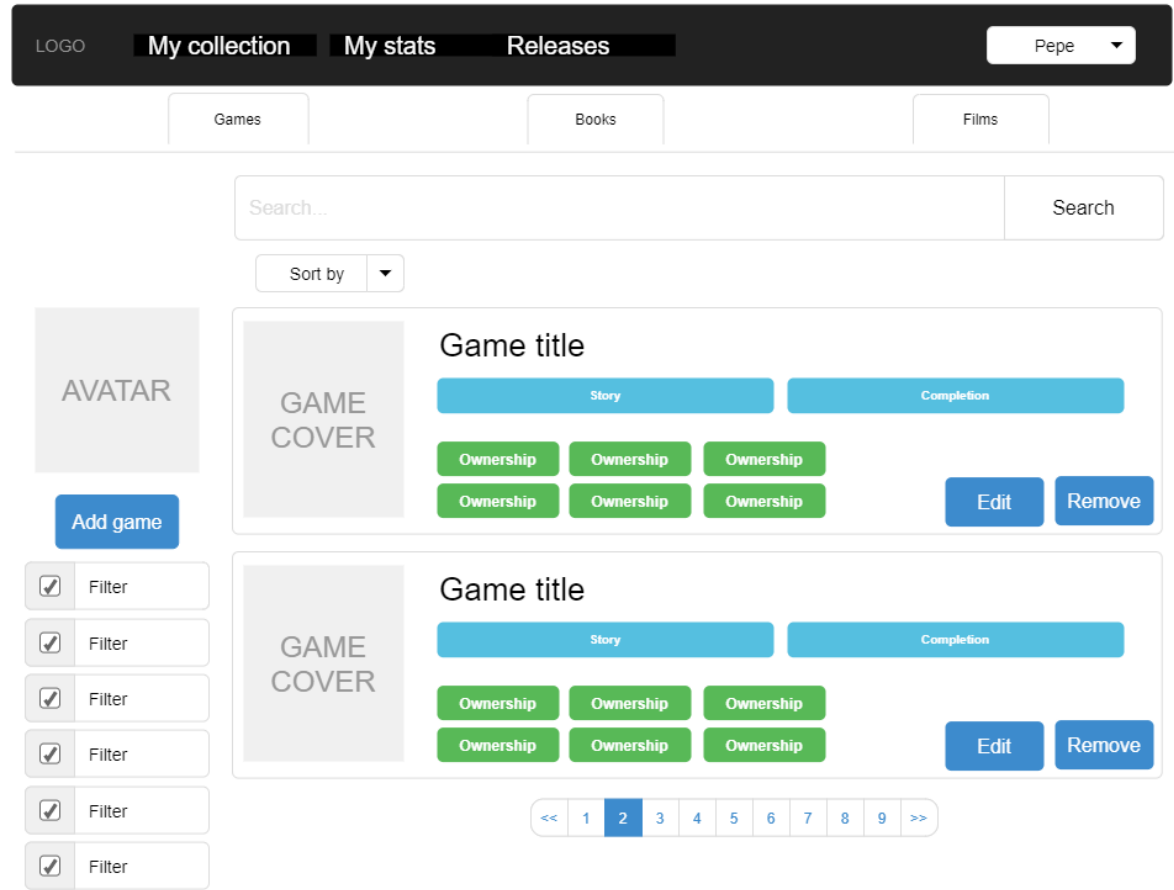
3.4 DIAGRAMA DE CLASES.




3.5 MAPA DE NAVEGACIÓN.



3.6 MOCKUPS.



LOGO

Games

Books

Films

Search...

Search

Sort by

▼

GAME COVER

Game title

Story

Completion

Ownership

Ownership

Ownership

Ownership

Edit

Remove

GAME COVER

Game title

Story

Completion

Ownership

Ownership

Ownership

Ownership

Edit

Remove

<<

1

2

3

4

5

6

7

8

9

>>

Página 24 de 73

3.7 UX

3.7.1 NAVEGACIÓN POR LA INTERFAZ

Con la finalidad de ofrecer al usuario facilidad de uso de la aplicación, se ha diseñado con una interfaz simple, minimalista e intuitiva.

Se hace uso de una barra de navegación para acceder a las distintas secciones de la aplicación.

En la sección de la colección, que consta de varios apartados para cada tipo de coleccionable, se hace uso de pestañas para separar y facilitar la navegación entre tipos.

3.7.2 PALETA DE COLORES

En cuanto a la paleta de colores se ha optado por un diseño minimalista bicolor en blanco y gris marengo para la interfaz en general, sobre la que destacan los tonos coloridos empleados para mostrar la información de los artículos de la colección del usuario o sus estadísticas.

3.7.3 ORDEN DE LOS BOTONES

Otro aspecto importante es el orden de los botones de Aceptar y Cancelar en modales y formularios.

Hay distintas pautas a seguir según distintas compañías. Así, Microsoft indica que se debe colocar en primer lugar el botón de Aceptar y en segundo lugar el de Cancelar. Por su parte, Apple y Google indican que se debe colocar el botón que inicia una acción a la derecha del todo y el que la cancela justo a su izquierda.

Al ser una aplicación diseñada para su uso en distintas plataformas y teniendo en cuenta el rápido crecimiento de usuarios que utilizan como plataforma principal un móvil Android o iPhone, se ha optado por seguir las instrucciones de Apple y Google.

3.7.4 COLORES DE LOS BOTONES

En cuanto a los colores de los botones, se ha optado por utilizar el color gris marengo principal de la interfaz para los botones en general. A los botones que inician o confirman una acción destructiva se les ha dado el color rojo como señal visual de peligro. A los botones que simplemente cierran un modal se les ha dado color gris claro para restarles importancia respecto al botón principal.

3.7.5 ACCESIBILIDAD

Para facilitar la accesibilidad a usuarios con discapacidades visuales, motoras o cognitivas se han tenido en cuenta los factores que afectan a la accesibilidad de la aplicación web, entre ellos:

1. Compatibilidad con el teclado.

La aplicación debe ser completamente navegable con el teclado. Se debe permitir saltar entre secciones o elementos, marcar, pulsar...

Además debe hacerse uso de indicadores de posición que permitan identificar el elemento en el que se encuentra el teclado.

También se debe respetar un orden lógico de navegación, por ejemplo de arriba abajo y de izquierda a derecha.

2. Puntos de referencia ARIA.

Estas son etiquetas que se agregan al contenido para definirlo claramente en la página, esto permite a los lectores de pantalla y dispositivos similares comprender el contenido y es especialmente útil con contenido dinámico.

ARIA también es útil para hacer que la navegación sea más sencilla, ya que permite a los usuarios saltar directamente a contenido específico.

3. Texto alternativo para las imágenes.

Esto permite a los lectores de pantalla "leer" la imagen.

4. Contraste de colores.

Se deben utilizar colores para los elementos que contrasten con el fondo, evitando crear una paleta de colores donde los tonos sean demasiado similares en matiz y saturación.

5. Redimensionamiento.

La mayoría de los dispositivos y navegadores permiten a los usuarios cambiar el tamaño del texto, lo que puede ser útil para las personas con discapacidad visual.

La aplicación web debe estar diseñada de forma que aumentar el nivel de zoom no dañe las funcionalidades.

6. Evitar reproducción automática de multimedia.

Puede ser muy molesto que la música o los videos comiencen cuando se carga una página, pero este es un problema aún mayor en términos de accesibilidad.

Por ejemplo, descubrir cómo apagar los medios puede ser difícil cuando se usa un lector de pantalla, mientras que otros usuarios simplemente pueden confundirse o incluso asustarse por el ruido repentino. Por lo tanto, se debe evitar incluir elementos que comiencen sin que el usuario los solicite primero.

También es mejor evitar la navegación automática, como carruseles y controles deslizantes. Esto puede ser increíblemente frustrante si el usuario necesita más tiempo para absorber toda la información antes de pasar a la siguiente diapositiva o sección.

7. Evitar CAPTCHAs.

Los captchas visuales suponen un gran problema para las personas invidentes, con baja visión o con una discapacidad de aprendizaje como la dislexia.

Por eso es preferible emplear sistemas como preguntas de lógica, textuales o que no requieran interacción por parte del usuario (como el método "Honeypot" empleado en este proyecto, con un checkbox oculto para atrapar bots).

4. DISEÑO.

4.1 SELECCIÓN DEL ENTORNO DE DESARROLLO.

El entorno de desarrollo elegido está compuesto por Windows, Apache, MariaDB y PHP por su extendido uso y soporte y en especial por su alta compatibilidad a la hora de ser desplegado.

4.2 SELECCIÓN DE LA BASE DE DATOS.

La base de datos será una base de datos relacional gestionada mediante MariaDB, sistema de gestión de bases de datos derivado de MySQL, uno de los sistemas más populares y de sencillo uso e implantación.

La elección de una base de datos de tipo relacional se debe a que las tablas de usuarios y de objetos deben relacionarse entre sí para permitir al usuario añadir objetos a su colección.

4.3 SELECCIÓN DE LA API EXTERNA.

4.3.1 LISTADO DE APIS CANDIDATAS.

A la hora de elegir API externa se han examinado las características y acotado las opciones a estas:

RAWG

Documentación: <https://api.rawg.io/docs/>

Limitaciones:

- 20000 peticiones al mes.
- Uso no comercial.
- Link a su web.

IGDB

Documentación: <https://api-docs.igdb.com/>

Limitaciones:

- 4 peticiones por segundo.
- Uso no comercial.
- Requiere cuenta de desarrollador de Twitch.

GIANTBOMB

Documentación: <https://www.giantbomb.com/api/documentation/>

Limitaciones:

- 200 peticiones por recurso por hora.
- Uso no comercial.
- Link a su web.

4.3.2 SELECCIÓN DE LA API.

La elegida ha sido IGDB, con GIANTBOMB en segundo lugar como plan de contingencia ante el cese de IGDB.

RAWG queda relegada a un tercer puesto debido a que mucha de su información avanzada es exclusiva para usuarios de pago.

Tanto IGDB como GIANTBOMB proporcionan información muy similar.

La razón para elegir IGDB sobre GIANTBOMB es el sistema por el cual se ejecuta la petición.

GIANTBOMB utiliza una petición GET con un sistema de filtrado básico por parámetros de URL.

IGDB utiliza una petición POST con consultas avanzadas en el cuerpo de la petición, lo que permite obtener toda la información necesaria con una sola petición y ofrece muchas más facilidades a la hora de escalar la aplicación en el futuro y requerir solicitar información más compleja.

4.3.3 EJEMPLO DE PETICIÓN EN GIANTBOMB.

Para buscar 3 imágenes de un juego por su título en GIANTBOMB, lo primero es hacer una petición de la ID del juego:

```
https://www.giantbomb.com/api/search/?api_key={API_KEY}&limit=1&resources=game&field_list=guid&query={JUEGO}&format=json
```

Esto devuelve la GUID (ID en formato cadena). Con esta GUID podemos realizar una petición para obtener las 3 imágenes.

https://www.giantbomb.com/api/images/{GUID}?api_key={API_KEY}&limit=3&format=json

4.3.4 EJEMPLO DE PETICIÓN EN IGDB.

Las peticiones de IGDB se efectúan mediante POST.

En la cabecera hay que indicar la ID de cliente y el Token de autorización proporcionados por el servicio de desarrolladores de Twitch.

En el cuerpo de la petición hay que escribir la consulta a realizar, en lenguaje Apicalypse:

<https://apicalypse.io/syntax/>

Además, se pueden realizar subconsultas desde esa misma petición en vez de realizar consultas sucesivas, utilizando “expanders”.

Expander

Some fields are actually ids pointing to another endpoint. The expander feature is a convenient way to go into these other endpoints and access more information from them in the same query, instead of having to do multiple queries.

Por ejemplo para buscar la ID de un juego de PS4 por su título:

Endpoint: <https://api.igdb.com/v4/games>

Cuerpo:

```
fields id;  
search "{TITULO}";  
where platforms = (48);  
limit 1;
```

Si queremos además añadir información sobre el juego, solamente hay que modificar la consulta en el cuerpo de la petición:

Endpoint: <https://api.igdb.com/v4/games>

Cuerpo:

```
fields id, cover, first_release_date, name, platforms, screenshots, summary,  
videos;  
search "{TITULO}";  
where platforms = (48);  
limit 1;
```

Esto nos devuelve la información solicitada, entre ella las IDs de la carátula, capturas de pantalla y vídeos. Para poder obtener estos recursos, en lugar de hacer otra petición solicitándolos por su ID, podemos hacer subconsultas modificando el cuerpo de la petición:

Endpoint: <https://api.igdb.com/v4/games>

Cuerpo:

```
fields id, cover.image_id, first_release_date, name, platforms,  
screenshots.image_id, summary, videos.video_id;  
search "{TITULO}";  
where platforms = (48);  
limit 1;
```

El resultado final es obtener esta información en formato JSON:

```
{
  "id": 7334,
  "cover": {
    "id": 82054,
    "image_id": "co1rba"
  },
  "first_release_date": 1427155200,
  "name": "{TITULO}",
  "platforms": [
    48
  ],
  "screenshots": [
    {
      "id": 9530,
      "image_id": "adssvwsfbaxcrucjolv9"
    },
    {
      "id": 9531,
      "image_id": "bklmdxphzoflbgatushg0"
    },
    {
      "id": 9532,
      "image_id": "vcnwnetnskfkmfkwkfd"
    },
    {
      "id": 9533,
      "image_id": "gbjnfjwerjkrj6vnyh3jfnf"
    },
    {
      "id": 9534,
      "image_id": "bklgrttrtbgatusgtnhb"
    },
    {
      "id": 9535,
      "image_id": "uqmif3sshdrbbcd0pu8l"
    }
  ],
  "summary": "{RESUMEN DE LA HISTORIA DEL JUEGO}",
  "videos": [
    {
      "id": 2830,
      "video_id": "rMxEpPCSH5A"
    }
  ]
}
```

Las imágenes van en la ruta:

https://images.igdb.com/igdb/image/upload/t_{tamaño}/{image_id}.jpg

Donde tamaño es uno de estos tamaños:

cover_small	90 x 128	Fit
screenshot_med	569 x 320	Lfill, Center gravity
cover_big	264 x 374	Fit
logo_med	284 x 160	Fit
screenshot_big	889 x 500	Lfill, Center gravity
screenshot_huge	1280 x 720	Lfill, Center gravity
thumb	90 x 90	Thumb, Center gravity
micro	35 x 35	Thumb, Center gravity
720p	1280 x 720	Fit, Center gravity
1080p	1920 x 1080	Fit, Center gravity

Se puede escalar al doble añadiendo “_2x” al nombre del tamaño.

Los vídeos van en la ruta:

https://www.youtube.com/watch?v={video_id}

4.4 ESTRUCTURA DE LA BASE DE DATOS.

4.4.1 DIAGRAMA DE LAS TABLAS.



4.4.2 DISPARADORES PARA AUDITORÍA.

Para llevar un control de la actividad de los usuarios se utiliza una serie de disparadores que se ejecutan cada vez que el usuario se registra, realiza un inserto, una modificación o una eliminación de un artículo de su colección.

- Disparador de registro para insertar nuevo usuario en la tabla de auditoría cuando un usuario se registra:

```
CREATE TRIGGER `add_new_audit_user` AFTER INSERT ON `users` FOR EACH
ROW BEGIN
    INSERT INTO auditing
    VALUES (null, NEW.id, 0, 0, 0);
END
```

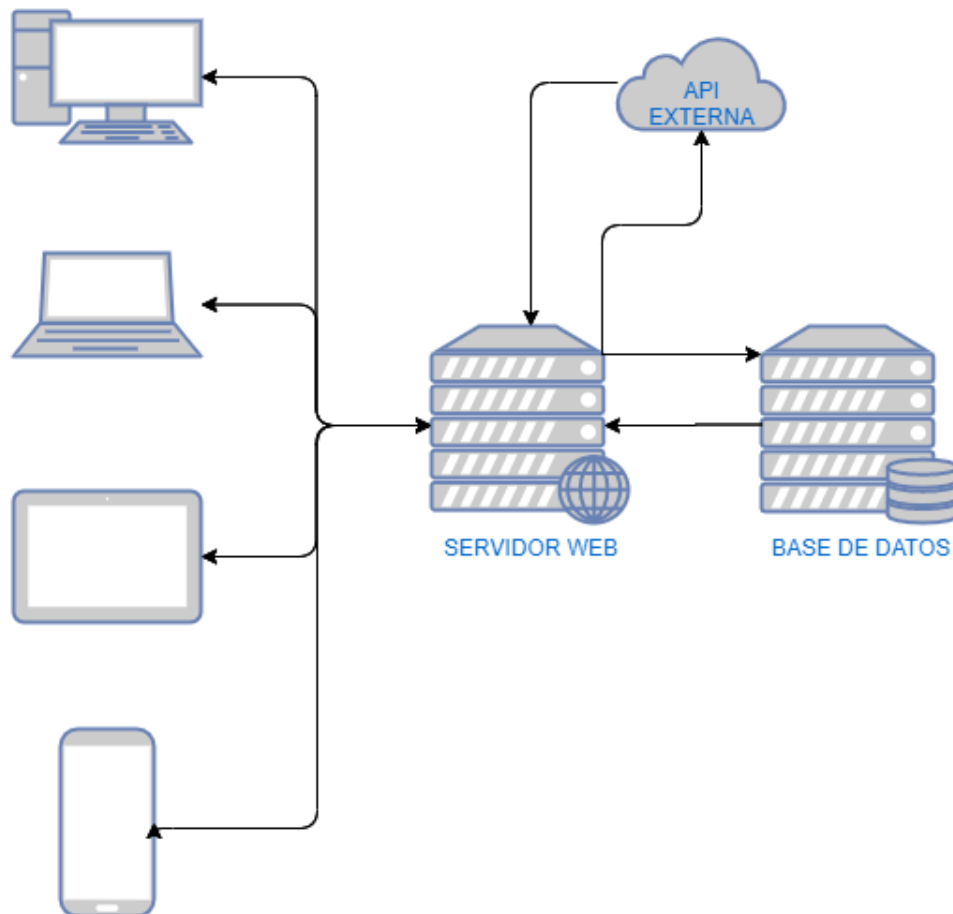
- Disparadores de inserto, modificación y eliminación para incrementar el total de operaciones de cada tipo cuando el usuario las realiza:

```
CREATE TRIGGER `increment_insert_ug` AFTER INSERT ON `user_games` FOR
EACH ROW BEGIN
    UPDATE auditing
    SET auditing.inserts = auditing.inserts+1
    WHERE auditing.user_id = NEW.user_id;
END
```

```
CREATE TRIGGER `increment_update_ug` AFTER UPDATE ON `user_games` FOR
EACH ROW BEGIN
    UPDATE auditing
    SET auditing.updates = auditing.updates+1
    WHERE auditing.user_id = NEW.user_id;
END
```

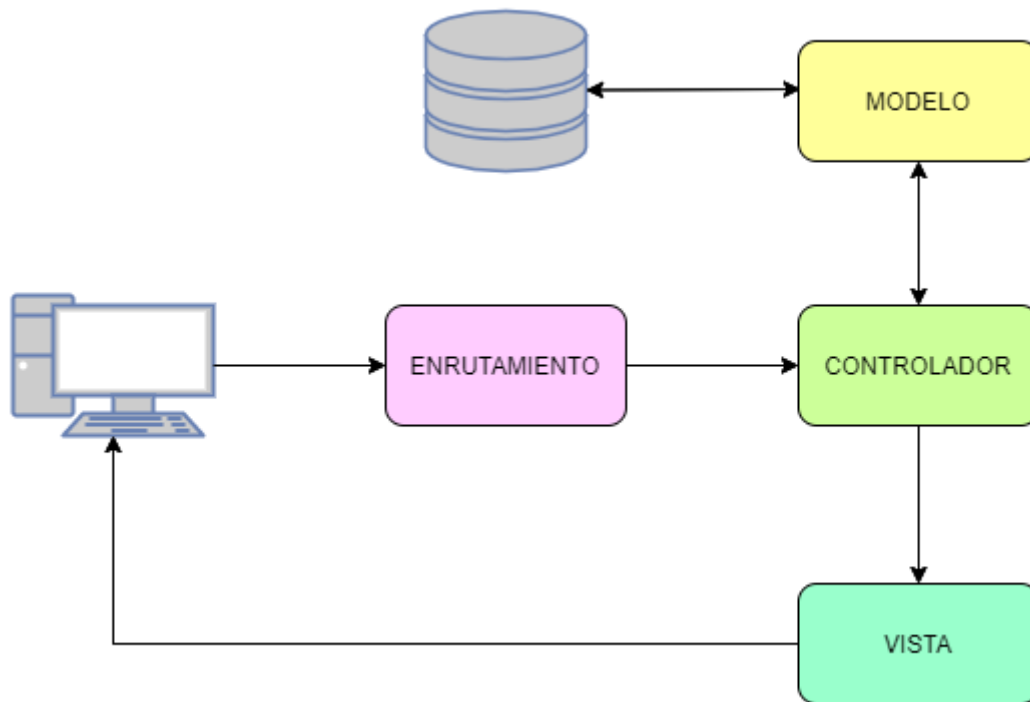
```
CREATE TRIGGER `increment_delete_ug` AFTER DELETE ON `user_games` FOR
EACH ROW BEGIN
    UPDATE auditing
    SET auditing.deletes = auditing.deletes+1
    WHERE auditing.user_id = OLD.user_id;
END
```

4.5 ARQUITECTURA FÍSICA.



4.6 ARQUITECTURA DE SOFTWARE.

La aplicación está desarrollada siguiendo los principios básicos del patrón MVC separando modelos, vistas y controladores e implementa un controlador frontal que gestiona el enrutamiento.



4.6.1 CAPA DE PERSISTENCIA.

En esta capa se representan y almacenan los datos de la aplicación y permite realizar las operaciones de creación, lectura, actualización y borrado (CRUD).

4.6.1.1 MODELO.

El modelo es el componente que representa la información en la base de datos mediante un conjunto de clases para ser gestionada por la aplicación.

Para la gestión de la información de la base de datos y la representación de esta mediante clases la aplicación utiliza el ORM (Mapeo Objeto-Relacional) Eloquent que integra el framework Laravel para hacer un mapeo de datos relacional.

Eloquent permite la creación de los modelos, definir sus relaciones y realizar operaciones CRUD sobre la base de datos mediante objetos.

4.6.2 CAPA DE PRESENTACIÓN.

Esta capa es la que permite al usuario interactuar con el sistema, se encarga de recoger la información del usuario y comunicarse con la capa de lógica.

Una vez la capa de lógica ha procesado la información esta capa se encarga de generar la presentación mediante vistas.

4.6.2.1 VISTA.

La vista se encarga de presentar toda la información del modelo que nos proporciona el controlador.

La aplicación hace uso del motor de plantillas Blade integrado en el framework Laravel para generar las diferentes vistas. Esto aporta a la aplicación una gran modularidad gracias a las secciones y a la herencia de plantillas.

Blade además permite el uso de estructuras de control, permitiendo que el código sea limpio y conciso.

4.6.3 CAPA DE LÓGICA.

Esta capa se encarga de procesar las interacciones del usuario con la aplicación, conectando la capa de presentación con la capa de persistencia.

4.6.3.1 CONTROLADORES.

El controlador es el que se encarga de responder a las acciones que se solicitan en la aplicación.

Es además el encargado de solicitar las operaciones de creación, lectura, actualización y borrado (CRUD) de los datos de la aplicación a la capa de persistencia.

4.6.3.2 ENRUTAMIENTO.

El enrutamiento implementado mediante Laravel permite determinar cuál es la acción que se ejecuta ante una petición HTTP del usuario de la aplicación y solicitarla al controlador adecuado.

Además ofrece herramientas para trabajar con funcionalidades como redireccionamiento, parámetros, restricciones, middleware o espacios de nombre.

5. IMPLEMENTACIÓN.

5.1 TECNOLOGÍAS.

En la actualidad existen una enorme cantidad de tecnologías para desarrollar aplicaciones web.

Para esta aplicación se ha optado por tecnologías de uso muy extendido y que a pesar de haber sido utilizadas durante años o incluso décadas se mantienen actualizadas y permiten desarrollar con un enfoque actual.

5.1.1 BACKEND.

5.1.1.1 PHP 7.4.11.

PHP (acrónimo recursivo de PHP: Hypertext Preprocessor) es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML.

El código PHP suele ser procesado en un servidor web por un intérprete PHP. El intérprete estándar de PHP es un software libre publicado bajo Licencia PHP.

PHP ha sido ampliamente portado y puede ser desplegado en la mayoría de los servidores web en casi todos los sistemas operativos y plataformas, de forma gratuita.

La mayoría de código backend de esta aplicación ha sido escrito en PHP.

5.1.1.2 LARAVEL 8.22.

Laravel es un framework para PHP de código abierto ampliamente utilizado gracias a su constante actualización para hacer posible trabajar con las últimas versiones de PHP así como implementar utilidades populares de otros frameworks.

Su filosofía es desarrollar código de forma elegante y simple basado en un modelo MVC.

Laravel pone énfasis en la calidad del código, la facilidad de mantenimiento y la escalabilidad. Además permite y facilita el trabajo en equipo y promueve buenas prácticas.

Para el desarrollo de esta aplicación se han utilizado numerosas características proporcionadas por este framework, como son el enrutamiento, la abstracción de la base de datos mediante ORM, el sistema de autenticación, la gestión de sesiones o el envío de email.

5.1.1.3 MariaDB 10.4.14.

MariaDB es un sistema de gestión de bases de datos de código abierto derivado de MySQL, un sistema de gestión de bases de datos relacional considerado como uno de los más populares para el desarrollo web.

Su popularidad se debe a que es de uso gratuito, fácil de usar, con alta compatibilidad, de excelente rendimiento y que requiere pocos recursos gracias a su gran eficiencia.

La aplicación hace uso de MariaDB para almacenar datos requeridos para el correcto uso de sus funcionalidades, tales como la información de los usuarios, de los objetos y de las colecciones de objetos de cada usuario.

5.1.1.4 BLADE.

Blade es un potente motor de plantillas integrado en el Framework Laravel.

Mediante Blade se han modularizado las vistas de la aplicación separando las distintas secciones de la aplicación en componentes independientes, proporcionando una gran flexibilidad a la hora de definir la estructura y permitiendo que el código sea reutilizable y limpio.

5.1.2 FRONTEND.

5.1.2.1 HTML 5.

HTML (HyperText Markup Language), hace referencia al lenguaje de marcado para la elaboración de páginas web.

Es un estándar a cargo del World Wide Web Consortium (W3C) o Consorcio WWW, organización dedicada a la estandarización de casi todas las tecnologías ligadas a la web, sobre todo en lo referente a su escritura e interpretación.

Mediante HTML se ha definido la estructura básica de la vista de la aplicación.

5.1.2.2 CSS 3.

CSS (Cascading Style Sheets) es un lenguaje para definir y crear la presentación de un documento estructurado escrito en un lenguaje de marcado.

La especificación CSS es mantenida por el World Wide Web Consortium (W3C).

CSS está diseñado principalmente para marcar la separación del contenido del documento y la forma de presentación de este. Esta separación busca mejorar la accesibilidad del documento, proveer más flexibilidad y reducir la complejidad y la repetición de código.

La separación del formato y el contenido ha hecho posible que la aplicación se haya desarrollado con una vista adaptable a distintos dispositivos (responsive).

5.1.2.3 BOOTSTRAP 4.

Bootstrap es un conjunto de herramientas de código abierto para desarrollos web responsive con HTML, CSS y JavaScript.

Permite utilizar estilos CSS predeterminados de forma sencilla, además de incluir muchos componentes listos para utilizar.

Mediante Bootstrap se ha creado para la aplicación una interfaz intuitiva y adaptable a distintos dispositivos (responsive).

5.1.2.4 JAVASCRIPT ES6.

JavaScript es un lenguaje de programación interpretado, dialecto del estándar ECMAScript.

Se utiliza principalmente del lado del cliente, permitiendo mejoras en la interfaz de usuario y páginas web dinámicas.

También es ampliamente utilizado para enviar y recibir información del servidor junto con ayuda de otras tecnologías como AJAX.

En esta aplicación mediante el uso de JavaScript se manipula el DOM y se manejan los eventos.

5.2 HERRAMIENTAS.

5.2.1 VISUAL STUDIO CODE.

Visual Studio Code es un editor de código fuente desarrollado por Microsoft para Windows, Linux y macOS.

Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código.

También es personalizable, por lo que los usuarios pueden cambiar el tema del editor, los atajos de teclado y las preferencias.

Es gratuito y de código abierto.

5.2.2 PHPMYADMIN.

Para gestionar la base de datos se usa la herramienta phpMyAdmin, que es una herramienta escrita en PHP con la intención de manejar la administración de MySQL y sus derivados a través de una interfaz web, utilizando un navegador.

Esta herramienta permite crear y eliminar Bases de Datos, crear, eliminar y alterar tablas, borrar, editar y añadir campos, ejecutar cualquier sentencia SQL, administrar claves en campos, administrar privilegios y exportar datos en varios formatos

5.2.3 COMPOSER.

Composer es un sistema de gestión de dependencias de paquetes y librerías para programar en PHP.

Composer trabaja e instala dependencias o librerías desde la línea de comandos. También permite al usuario instalar las aplicaciones PHP que estén disponibles en "Packagist", el repositorio principal que contiene todos los paquetes disponibles.

También dispone de capacidad de auto-descarga para las librerías necesarias que se especifiquen en la información de arranque para facilitar el uso de código de terceros.

Las dependencias del proyecto gestionadas mediante Composer son:

 [facade / ignition](#)

>= 2.5, < 3.0

 [FakerPHP / Faker](#)

>= 1.9.1, < 2.0.0

 [fideloper / TrustedProxy](#)

>= 4.4, < 5.0

 [fruitcake / laravel-cors](#)

>= 2.0, < 3.0

 [guzzle / guzzle](#)

>= 7.0.1, < 8.0.0

 [laravel / framework](#)

>= 8.12, < 9.0
 [laravel / sail](#)
 >= 1.0.1, < 2.0.0
 [laravel / socialite](#)
 >= 5.2, < 6.0
 [laravel / tinkler](#)
 >= 2.5, < 3.0
 [laravel / ui](#)
 >= 3.2, < 4.0
 [mockery / mockery](#)
 >= 1.4.2, < 2.0.0
 [nunomaduro / collision](#)
 >= 5.0, < 6.0
 [php / php-src](#)
 >= 7.3|^8.0, < 8.0.0
 [sebastianbergmann / phpunit](#)
 >= 9.3.3, < 10.0.0
 [symfony / dom-crawler](#)
 >= 5.2, < 6.0






5.2.4 NPM.







npm es el sistema de gestión de paquetes por defecto para Node.js, un entorno de ejecución para JavaScript.

npm trabaja e instala dependencias o librerías desde la línea de comandos.

También dispone de capacidad de auto-descarga para las librerías necesarias que se especifiquen en la información de arranque para facilitar el uso de código de terceros.

Las dependencias del proyecto gestionadas mediante npm son:

 [FontAwesome / Font-Awesome](#)
 ^ 5.15.3
 [axios / axios](#)
 ^ 0.21
 [twbs / bootstrap](#)
 ^ 4.0.0
 [jquery / jquery](#)
 ^ 3.2
 [JeffreyWay / laravel-mix](#)
 ^ 6.0.6

 [lodash / lodash](#)
^ 4.17.19
 [popperjs / popper-core](#)
^ 1.12
 [postcss / postcss](#)
^ 8.1.14
 [bholloway / resolve-url-loader](#)
^ 3.1.2
 [sass / dart-sass](#)
^ 1.15.2
 [webpack-contrib / sass-loader](#)
^ 8.0.0

5.2.5 GIT / GITHUB.

Se ha hecho uso en el proyecto del sistema de control de versiones Git junto con la plataforma de desarrollo colaborativo GitHub.

Git es un software de control de versiones diseñado pensando en la eficiencia, la confiabilidad y compatibilidad del mantenimiento de versiones de aplicaciones cuando estas tienen un gran número de archivos de código fuente. Su propósito es llevar el registro de los cambios en los archivos incluyendo coordinar el trabajo que varias personas realizan sobre archivos compartidos en un repositorio de código.

El uso de Git nos ayuda a mantener todo el histórico de cambios que se han producido en el proyecto, pudiendo volver a algún punto de estos, también nos permite crear ramas de desarrollo paralelas de distintas funcionalidades.

Por su parte GitHub nos permite tener nuestro código fuente almacenado en la nube, pudiéndolo gestionar haciendo uso de Git.

5.2.6 ARTISAN.

La herramienta Artisan es una de las herramientas que nos proporciona el framework Laravel para el desarrollo de aplicaciones.

Esta herramienta es una interfaz de línea de comandos donde el desarrollador puede dar instrucciones en forma de línea de comandos para realizar diferentes tareas durante el desarrollo e incluso cuando la aplicación se encuentra en producción.

Cada comando incluye una ayuda en pantalla, la cual muestra y describe los argumentos y opciones disponibles.

Además de los comandos proporcionados por Artisan, también puedes crear tus propios comandos personalizados que serán cargados mediante Composer.

5.2.7 PHPUNIT.

PHPUnit es un framework de código abierto para realizar pruebas unitarias en código PHP.

El objetivo de esta herramienta es crear pequeñas pruebas automatizadas para cada funcionalidad que implementemos y así poder comprobar de forma rápida y sencilla que la funcionalidad añadida o modificada funciona correctamente y que no ha afectado negativamente al resto de la aplicación.

5.2.8 LARAVEL DUSK.

Laravel Dusk es una herramienta para realizar pruebas automatizadas de navegador, las cuales simulan ser un usuario navegando por la aplicación y se utilizan para comprobar que determinadas acciones del usuario ofrecen el resultado esperado.

Laravel Dusk que proporciona una capa de abstracción sobre el PHP-Webdriver creado por Facebook para permitir controlar navegadores mediante PHP, al que se han agregado métodos para poder hacer pruebas sobre Laravel de forma más sencilla.

Por defecto utiliza Chromedriver, un driver para que las pruebas se ejecuten sobre Chrome, pero es posible usar otros navegadores soportados por Selenium, una herramienta de código abierto que se utiliza para automatizar las pruebas realizadas en los navegadores web.

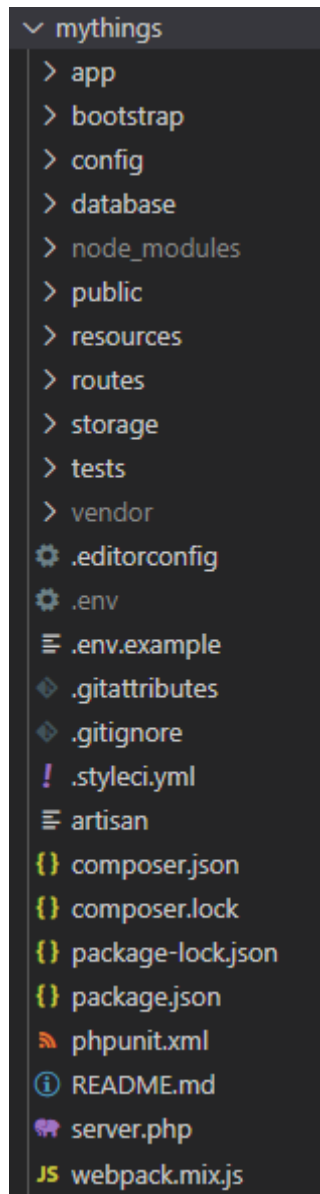
5.2.9 SONARQUBE

Sonarqube es una plataforma para evaluar código fuente.

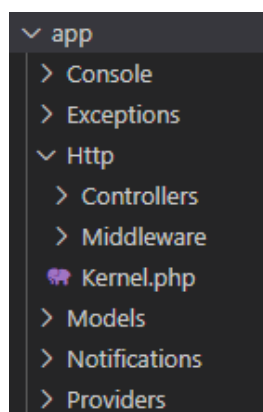
Es software libre y usa diversas herramientas de análisis estático de código fuente para informar sobre código duplicado, estándares de codificación, pruebas unitarias, cobertura de código, complejidad ciclomática, errores potenciales, comentarios y diseño del software.

5.3 ESTRUCTURA.

Se ha usado la estructura que por defecto establece Laravel.

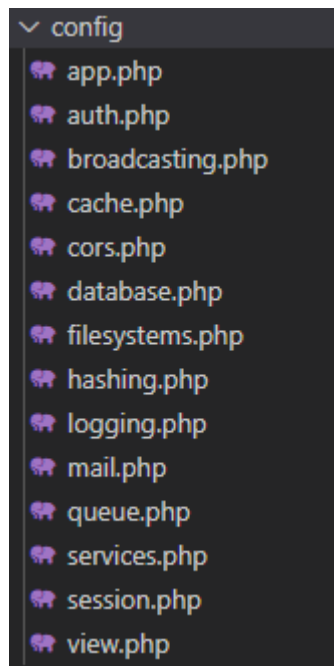


5.3.1 APP.



En este directorio se encuentran principalmente los modelos y controladores de la aplicación.

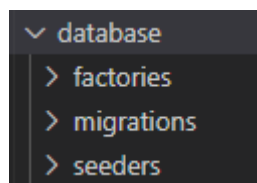
5.3.2 CONFIG.



En este directorio se encuentran los ficheros de configuración de aspectos como el idioma de la aplicación, la zona horaria, el tipo de base de datos.

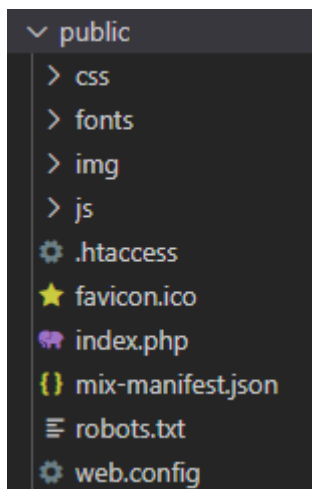
La mayoría de aspectos importan su valor desde el fichero de configuración `.env` situado en la raíz del proyecto para su fácil modificación.

5.3.3 DATABASE.



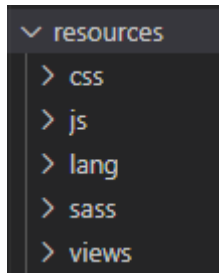
En este directorio se encuentran los ficheros necesarios para crear las tablas de la base de datos y poblarlas mediante migraciones.

5.3.4 PUBLIC.



En este directorio se encuentran todos los recursos estáticos de la aplicación, como imágenes, ficheros CSS, ficheros Javascript, fuentes, el favicon, el fichero de gestión de rastreadores, etc.

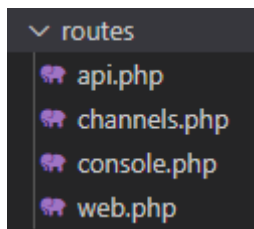
5.3.5 RESOURCES.



En este directorio se encuentran los archivos necesarios para generar las vistas de forma dinámica con ayuda de los controladores.

Estos archivos son por ejemplo las plantillas de Blade, los archivos de traducción y los archivos necesarios para las librerías de estilos.

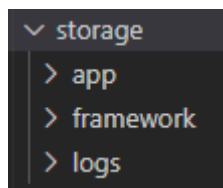
5.3.6 ROUTES.



En este directorio se establecen todas las rutas necesarias para el correcto enrutamiento de la aplicación.

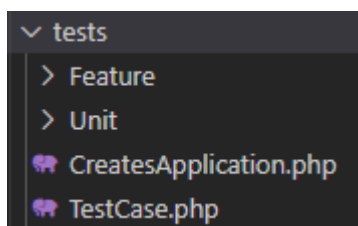
Además se pueden establecer las rutas para la API, los canales de difusión o la consola de comandos.

5.3.7 STORAGE.



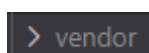
En este directorio se encuentran los logs, plantillas de Blade compiladas, sesiones, caché y demás archivos generados por Laravel.

5.3.8 TESTS.



En este directorio se encuentran los tests automatizados para comprobar el correcto funcionamiento de la aplicación.

5.3.9 VENDOR.



En este directorio se encuentran los archivos de terceros necesarios para las dependencias.

6. PRUEBAS.

Las pruebas de software son los procesos que permiten verificar la calidad de un producto de software. Son utilizadas para identificar posibles fallos de implementación, calidad o usabilidad de un programa.

6.1 PRUEBAS AUTOMATIZADAS.

Laravel incluye un entorno de pruebas pre-configurado, así como algunos tests de prueba.

Para las pruebas hace uso del framework PHPUnit, que permite escribir tests para luego ejecutarlos de forma automática y así comprobar tras añadir o modificar código que éste funciona correctamente y que no ha afectado al buen funcionamiento del resto de la aplicación.

6.1.1 PRUEBAS DE CARACTERÍSTICAS.

Mediante estos tests se prueba la funcionalidad de la aplicación, tanto de funcionalidades independientes como de las que necesitan interactuar entre ellas o con agentes externos. Estos tests tienen acceso a todo lo necesario para el funcionamiento de la aplicación, incluyendo la base de datos o peticiones a servicios externos.

Se han usado para comprobar funcionalidades como el enrutamiento:

```
public function testRouteLogin()
{
    $response = $this->get('login');
    $response->assertStatus(200);
}
```

O para comprobar la autenticación, como por ejemplo si un usuario tiene acceso a su propia lista de deseos marcada como privada:

```
public function testRoutePrivateWishlistWithAuthenticatedUserThatIsTheOwner()
{
    $userWithPrivateWishlist = User::where('wishlist', 0)->first();
    if ($userWithPrivateWishlist) {
        $response = $this->actingAs($userWithPrivateWishlist)->get('wishlist/' . $userWithPrivateWishlist->id);
        $response->assertStatus(200);
        $response->assertSee('Your wishlist is currently');
    } else {
        $this->assertNull($userWithPrivateWishlist);
    }
}
```

6.1.2 PRUEBAS UNITARIAS.

Mediante los tests unitarios se prueba la funcionalidad de un solo método. Estos tests son totalmente independiente unos de otros y no tienen acceso más que al método probado. No pueden acceder a la base de datos o servicios externos.

Se han usado estos tests para comprobar funciones concretas, como por ejemplo la de convertir una cadena de texto en minutos:

```
public function testConvertStringToMinutesFunction($expectedResult, $input)
{
    $gameController = new GameController();
    $results = $gameController->convertStringToMinutes($input);
    $this->assertEquals($expectedResult, $results);
}
```

Para comprobar los distintos parámetros que podría recibir la función, se ha hecho uso de los data provider, que nos permiten generar series de testeo con parámetros sobre un mismo método.

De este modo, se pueden utilizar distintas variaciones de los parámetros de entrada y sus correspondientes salidas, evitando repetir código.

```
public function provideStringsToMinutesData()
{
    return [
        'one hour' => [
            60,
            '1 Hour',
        ],
        'one hour and a half' => [
            90,
            '1½ Hour',
        ],
        'some hours' => [
            180,
            '3 Hours',
        ],
        'some hours and a half' => [
            210,
            '3½ Hours',
        ],
        'one minute' => [
            1,
            '1 Minute',
        ],
        'some minutes' => [
            45,
            '45 Minutes',
        ],
        'only seconds' => [
            0,
            '30 Seconds',
        ],
        'invalid string' => [
            0,
            'this is an invalid string to convert'
        ],
    ];
}
```


6.1.3 PRUEBAS DE NAVEGADOR.

Las pruebas automatizadas de navegador simulan ser un usuario navegando por la aplicación y se utilizan para comprobar que determinadas acciones del usuario ofrecen el resultado esperado.

Para realizar estas pruebas se ha hecho uso de Laravel Dusk, que proporciona una capa de abstracción sobre el PHP-Webdriver creado por Facebook para permitir controlar navegadores mediante PHP, al que se han agregado métodos para poder hacer pruebas sobre Laravel de forma más sencilla.

Por defecto utiliza Chromedriver, un driver para que las pruebas se ejecuten sobre Chrome, pero es posible usar otros navegadores soportados por Selenium, una herramienta de código abierto que se utiliza para automatizar las pruebas realizadas en los navegadores web.

Se han utilizado estas pruebas para comprobar la correcta usabilidad de la aplicación evaluando el resultado de las posibles interacciones del usuario.

Por ejemplo, para comprobar que se muestran errores de validación:

```
public function testRegisterWithWrongPasswordFormat()
{
    $this->browse(function (Browser $browser) {
        $browser->visit('/register')
            ->assertSee('Confirm Password')
            ->type('name', 'Dusk Test User')
            ->type('email', 'dusktestuser@email.com')
            ->type('password', 'password')
            ->type('password_confirmation', 'password')
            ->check('terms')
            ->press('Register')
            ->pause(1000)
            ->assertSee('The password must contain');
    });
}
```

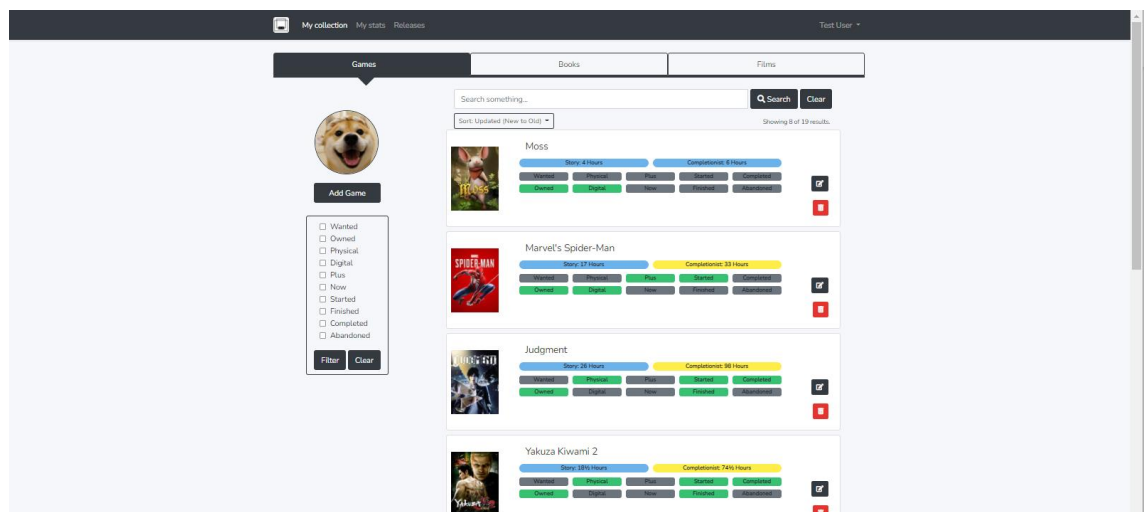
O que los enlaces que abren modales funcionan correctamente y muestran el contenido esperado:

```
public function testLandingLinkToTerms()  
{  
    $this->browse(function (Browser $browser) {  
        $browser->visit('/')  
        ->assertSee('Manage your collections')  
        ->press('Terms and conditions')  
        ->whenAvailable('#termsModal', function ($modal) {  
            $modal->waitForText('Terms and conditions of use')  
            ->assertSee('Terms and conditions of use');  
        });  
    });  
}
```

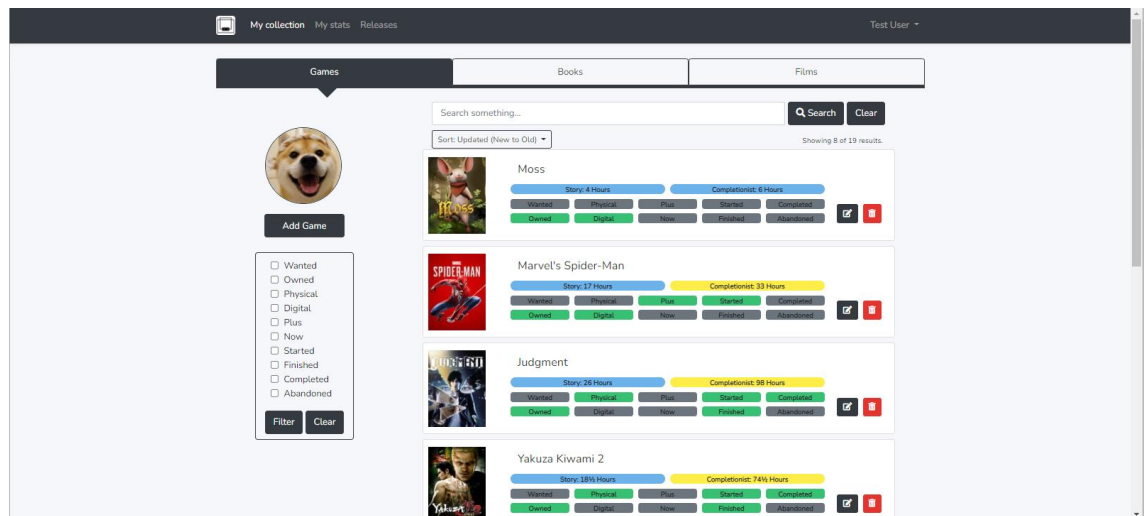
6.2 PRUEBAS MANUALES.

6.2.1 VISUALIZACIÓN EN DISTINTOS DISPOSITIVOS.

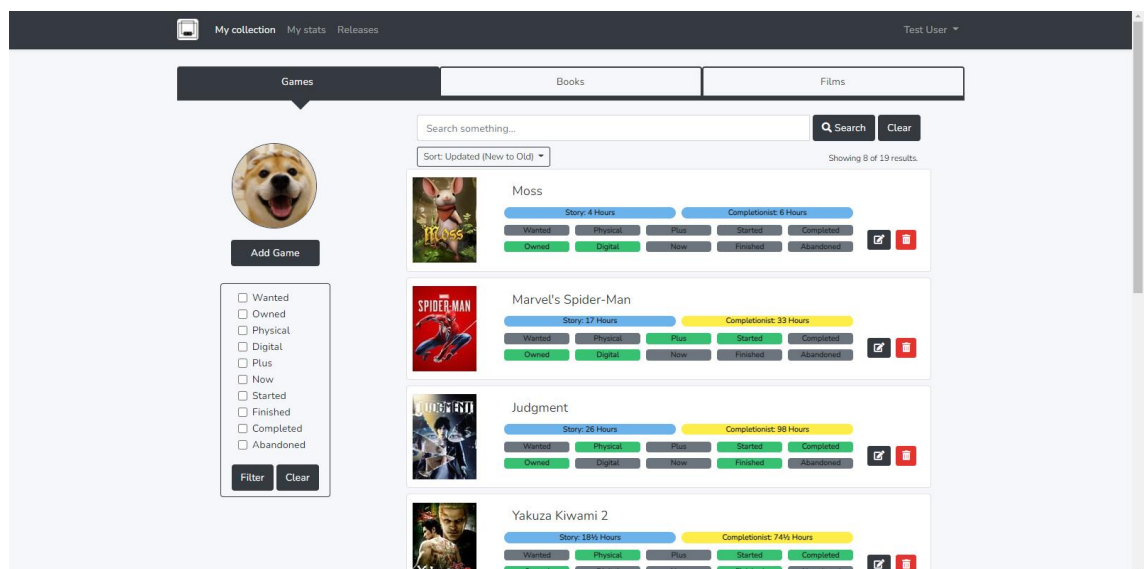
- Sobremesa Ultra-Wide (3440x1440)



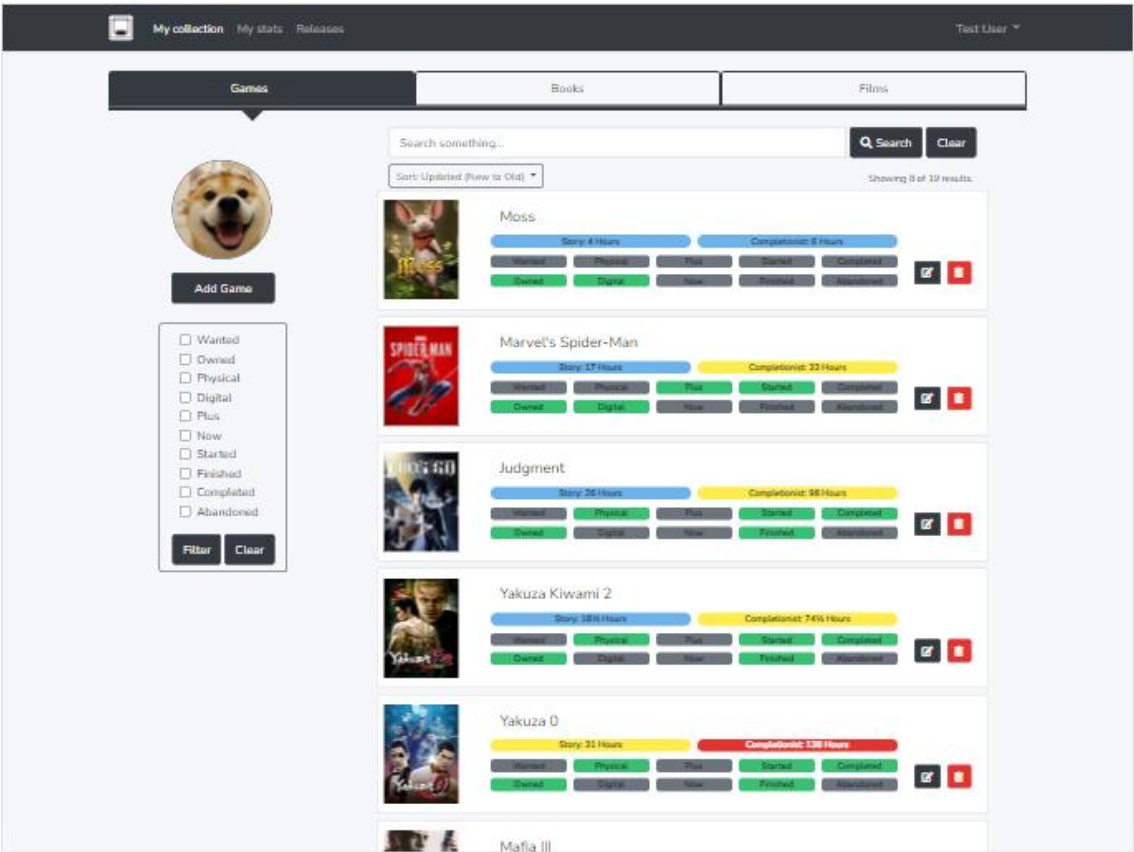
- Sobremesa Full-HD (1920x1080)



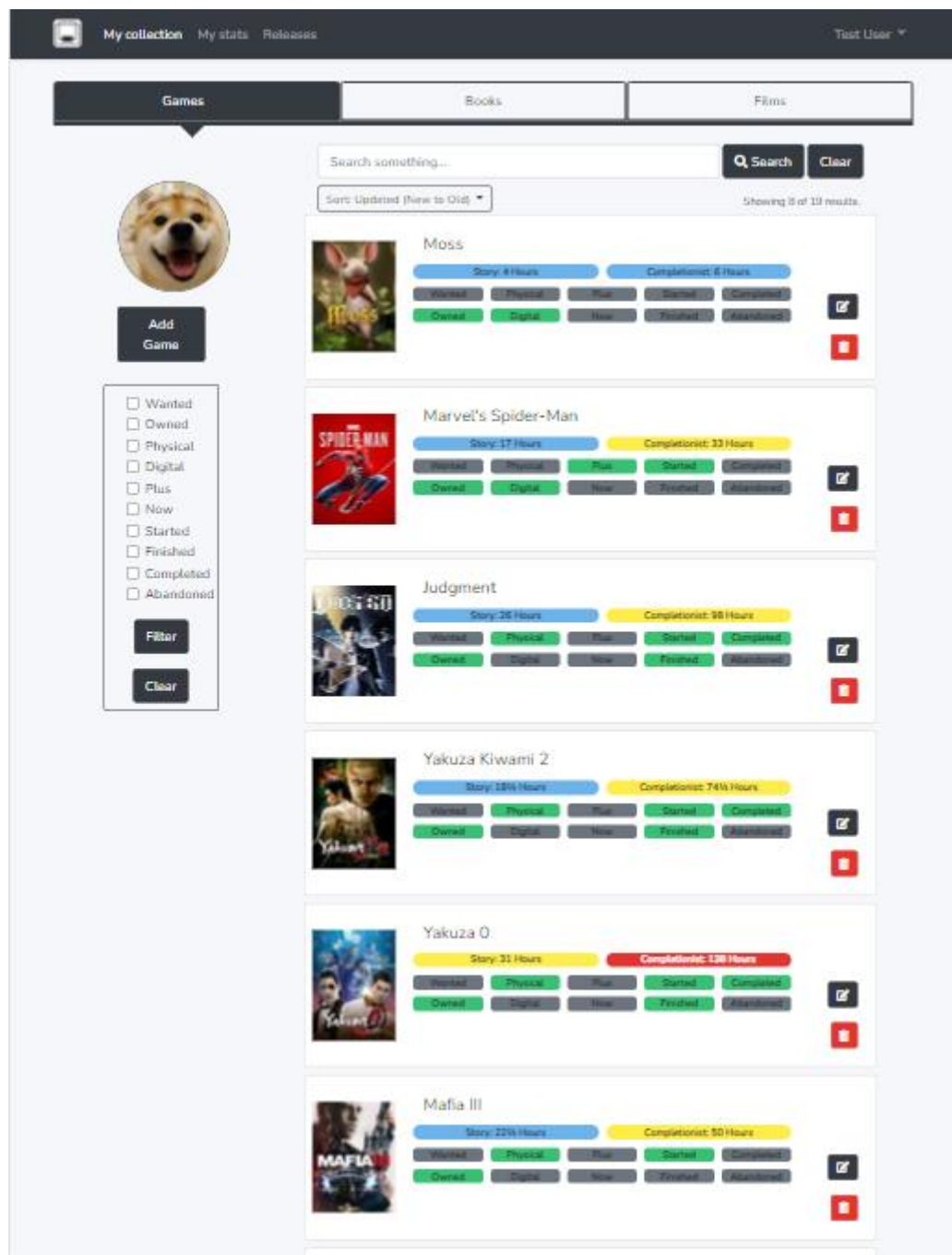
- Portátil (1600x900)



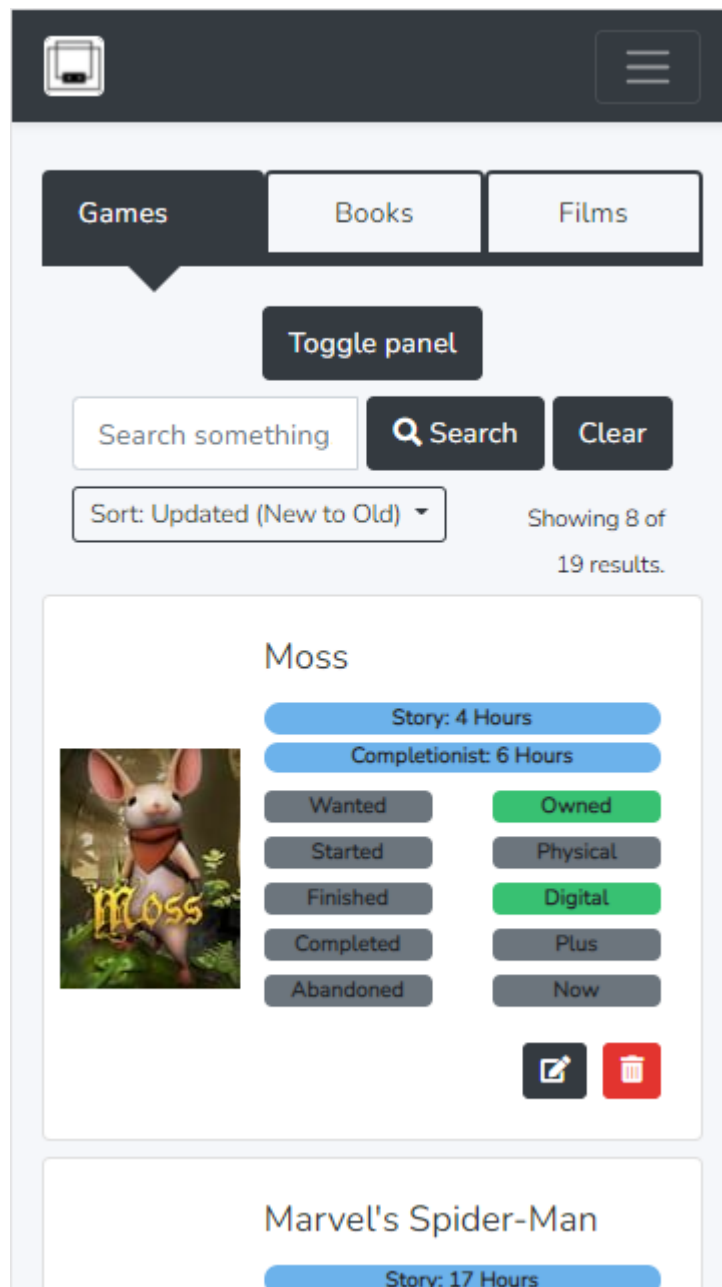
- Tablet iPad Pro (horizontal)



- Tablet iPad Pro (vertical)



- Móvil Android Samsung Galaxy S5



Games
Books
Films

Toggle panel

Sort: Updated (New to Old)
Showing 8 of 19 results

Story: 4 Hours
Completionist: 6 Hours

Wanted: ☒ Owned: ☒
Started: ☒ Physical: ☒
Finished: ☒ Digital: ☒
Completed: ☒ Plus: ☒
Abandoned: ☒ New: ☒

Story: 17 Hours
Completionist: 33 Hours

Wanted: ☒ Owned: ☒
Started: ☒ Physical: ☒
Finished: ☒ Digital: ☒
Completed: ☒ Plus: ☒
Abandoned: ☒ New: ☒

Story: 26 Hours
Completionist: 36 Hours

Wanted: ☒ Owned: ☒
Started: ☒ Physical: ☒
Finished: ☒ Digital: ☒
Completed: ☒ Plus: ☒
Abandoned: ☒ New: ☒

Story: 19½ Hours
Completionist: 74½ Hours

Wanted: ☒ Owned: ☒
Started: ☒ Physical: ☒
Finished: ☒ Digital: ☒
Completed: ☒ Plus: ☒
Abandoned: ☒ New: ☒

Story: 31 Hours
Completionist: 138 Hours

Wanted: ☒ Owned: ☒
Started: ☒ Physical: ☒
Finished: ☒ Digital: ☒
Completed: ☒ Plus: ☒
Abandoned: ☒ New: ☒

Story: 22½ Hours
Completionist: 50 Hours

Wanted: ☒ Owned: ☒
Started: ☒ Physical: ☒
Finished: ☒ Digital: ☒
Completed: ☒ Plus: ☒
Abandoned: ☒ New: ☒

Story: 19½ Hours
Completionist: 87 Hours

Wanted: ☒ Owned: ☒
Started: ☒ Physical: ☒
Finished: ☒ Digital: ☒
Completed: ☒ Plus: ☒
Abandoned: ☒ New: ☒

Story: 34 Hours
Completionist: 100 Hours

Wanted: ☒ Owned: ☒
Started: ☒ Physical: ☒
Finished: ☒ Digital: ☒
Completed: ☒ Plus: ☒
Abandoned: ☒ New: ☒

1 2 3

© 2021 My Things
About us

Terms and conditions
Privacy policy

My collection
My stats
Reviews
Add User

Games
Books
Films

Toggle panel

Add Game

☐ Wanted
☐ Owned
☐ Physical
☐ Digital
☐ Plus
☐ New
☐ Started
☐ Finished
☐ Completed
☐ Abandoned

Sort: Updated (New to Old)
Showing 8 of 19 results

Story: 4 Hours
Completionist: 6 Hours

Wanted: ☒ Owned: ☒
Started: ☒ Physical: ☒
Finished: ☒ Digital: ☒
Completed: ☒ Plus: ☒
Abandoned: ☒ New: ☒

Story: 17 Hours
Completionist: 33 Hours

Wanted: ☒ Owned: ☒
Started: ☒ Physical: ☒
Finished: ☒ Digital: ☒
Completed: ☒ Plus: ☒
Abandoned: ☒ New: ☒

Story: 26 Hours
Completionist: 36 Hours

Wanted: ☒ Owned: ☒
Started: ☒ Physical: ☒
Finished: ☒ Digital: ☒
Completed: ☒ Plus: ☒
Abandoned: ☒ New: ☒

Story: 19½ Hours
Completionist: 74½ Hours

Wanted: ☒ Owned: ☒
Started: ☒ Physical: ☒
Finished: ☒ Digital: ☒
Completed: ☒ Plus: ☒
Abandoned: ☒ New: ☒

Story: 31 Hours
Completionist: 138 Hours

Wanted: ☒ Owned: ☒
Started: ☒ Physical: ☒
Finished: ☒ Digital: ☒
Completed: ☒ Plus: ☒
Abandoned: ☒ New: ☒

Story: 22½ Hours
Completionist: 50 Hours

Wanted: ☒ Owned: ☒
Started: ☒ Physical: ☒
Finished: ☒ Digital: ☒
Completed: ☒ Plus: ☒
Abandoned: ☒ New: ☒

Story: 19½ Hours
Completionist: 87 Hours

Wanted: ☒ Owned: ☒
Started: ☒ Physical: ☒
Finished: ☒ Digital: ☒
Completed: ☒ Plus: ☒
Abandoned: ☒ New: ☒

Story: 34 Hours
Completionist: 100 Hours

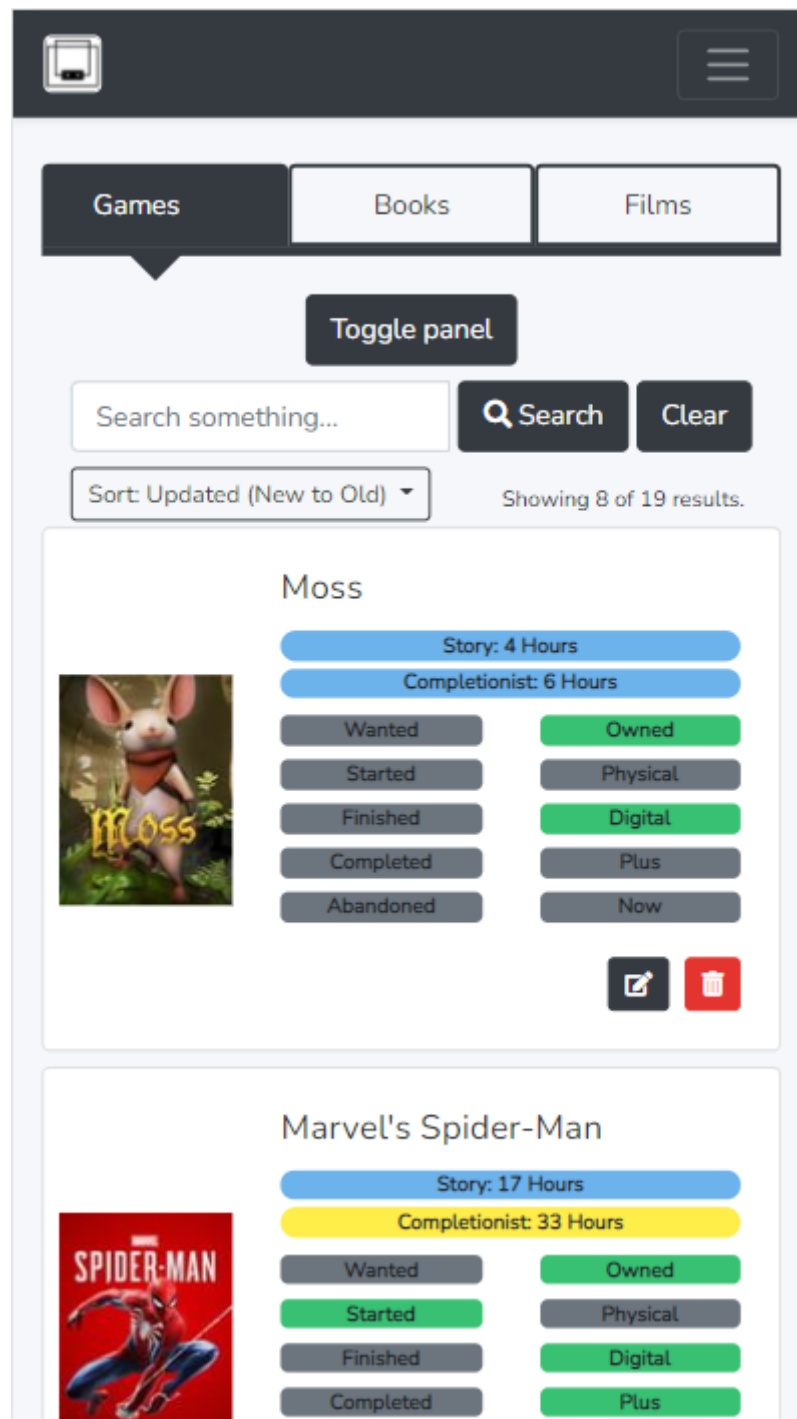
Wanted: ☒ Owned: ☒
Started: ☒ Physical: ☒
Finished: ☒ Digital: ☒
Completed: ☒ Plus: ☒
Abandoned: ☒ New: ☒

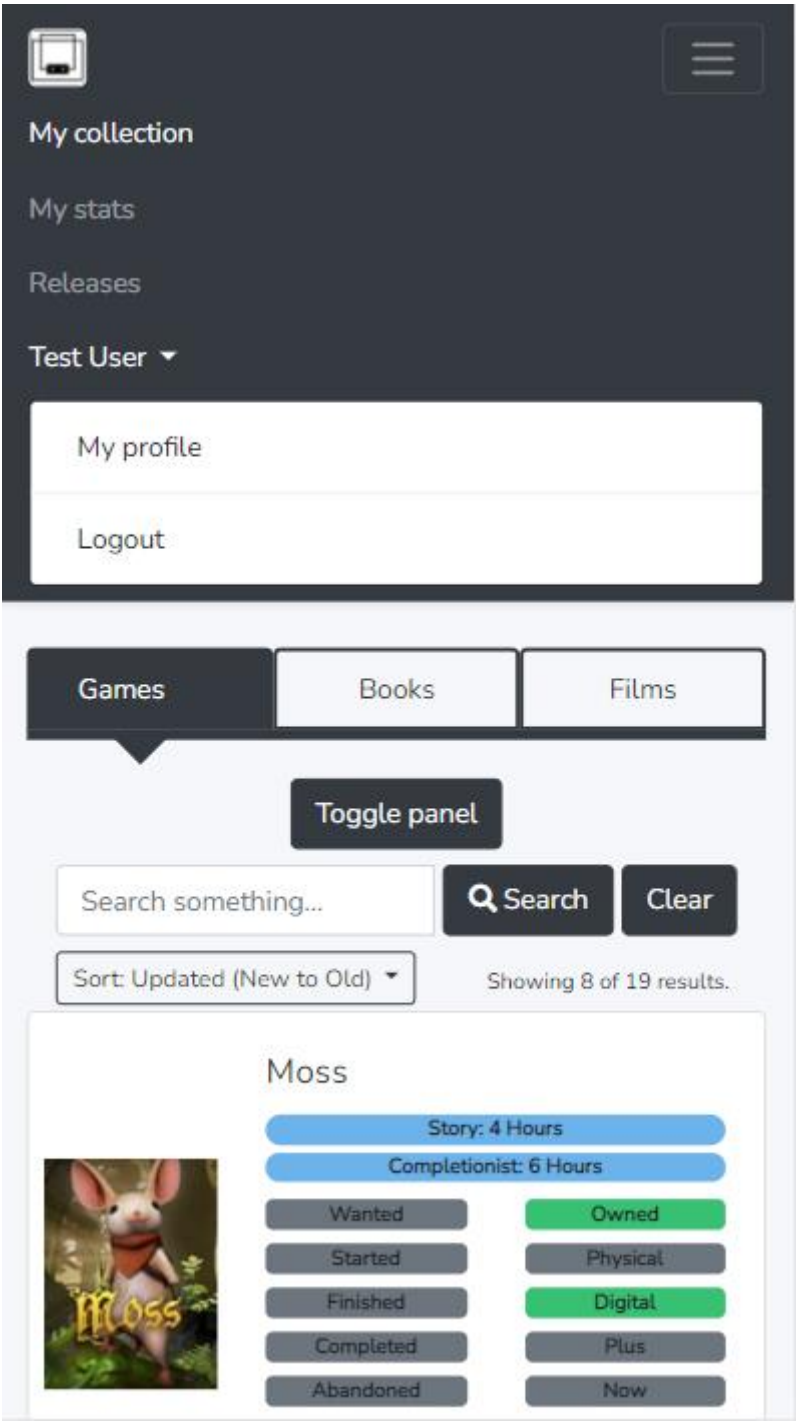
1 2 3

© 2021 My Things
About us

Terms and conditions
Privacy policy

- Móvil iPhone 8 Plus





Finished

Digital

Edit Game

X

Title

Yakuza Kiwami 2

Ownership

☐ Wanted

☒ Owned

☒ Physical

☐ Digital

☐ Plus

☐ Now

Play Info

☒ Started

☒ Finished

☒ Completed

☐ Abandoned

Cancel

Confirm changes



Story: 18½ Hours

Completionist: 74½ Hours

Wanted

Owned

Started

Physical

Finished

Digital

Completed

Plus



Abandoned

Now






Yakuza 0






Marvel's Spider-Man



Release date: 7 September 2018

Starring the world's most iconic Super Hero, Spider-Man features the acrobatic abilities, improvisation and web-slinging that the wall-crawler is famous for, while also introducing elements never-before-seen in a Spider-Man game. From traversing with parkour and utilizing the environment, to new combat and blockbuster set pieces, it's Spider-Man unlike any you've played before.



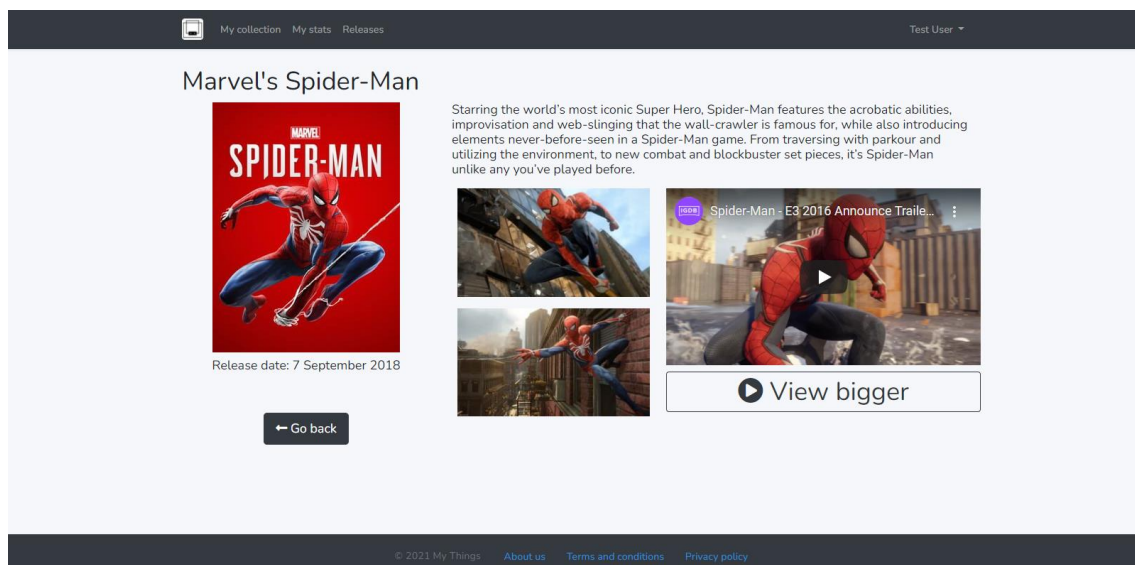
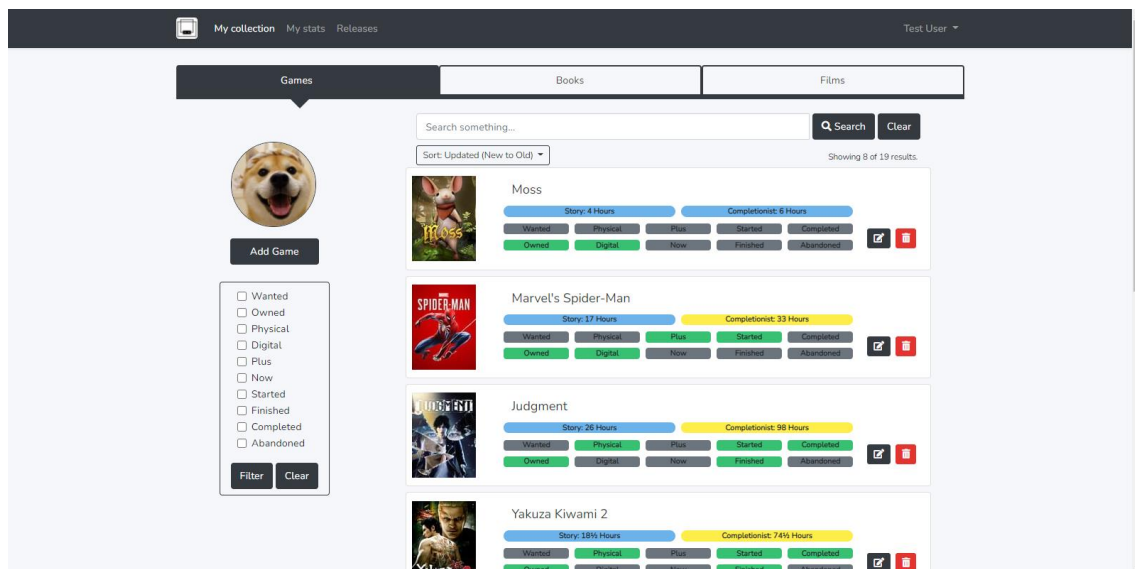
[← Go back](#)

© 2021 My Things [About us](#)

[Terms and conditions](#) [Privacy policy](#)

6.2.2 VISUALIZACIÓN EN DISTINTOS NAVEGADORES.


- Chrome



My collectionMy statsReleases

Test User

GamesBooksFilms



Your Game Collection Stats

Wanted:6Owned:14Started:10Finished:6Completed:5Abandoned:2

Total:19


Completion

This chart shows how many games have you either finished, fully completed or just abandoned.

Finished

Completed

Abandoned




Backlog

This chart shows how many owned games have you started playing and how many are waiting in your backlog.

Started

Not started



© 2021 My Things

About us

Terms and conditions

Privacy policy

My collectionMy statsReleases

Test User

May 2021

today<>

Sun	Mon	Tue	Wed	Thu	Fri	Sat
						1
2	3	4	5	6	7 Resident Evil Village	8
9	10	11	12	13	14 Mass Effect Legendary Edition	15
16	17	18	19	20	21	22
23	24	25 Semi-ant: Collector's Edition	26	27	28	29
30	31					

© 2021 My Things

About us

Terms and conditions

Privacy policy

Página 61 de 73

- Firefox


The screenshot shows the 'My Things' application interface. At the top, there's a navigation bar with 'My collection', 'My stats', and 'Releases'. Below this, there are tabs for 'Games', 'Books', and 'Films'. The 'Games' tab is active. On the left, there's a user profile section with a dog's head and an 'Add Game' button. Below that is a filter section with checkboxes for 'Wanted', 'Owned', 'Physical', 'Digital', 'Plus', 'Now', 'Started', 'Finished', 'Completed', and 'Abandoned'. The main area displays a list of games: 'Moss', 'Marvel's Spider-Man', 'Judgment', and 'Yakuza Kiwami 2'. Each game entry shows its cover art, title, story completion percentage, and a progress bar. For example, 'Moss' has a story completion of 41% and a completionist status of 61%. 'Marvel's Spider-Man' has a story completion of 17% and a completionist status of 33%. 'Judgment' has a story completion of 26% and a completionist status of 96%. 'Yakuza Kiwami 2' has a story completion of 18% and a completionist status of 74%.

The screenshot shows the detail page for the game 'Marvel's Spider-Man'. The title 'Marvel's Spider-Man' is at the top. Below it is the game's cover art. To the right of the cover art is a description: 'Starring the world's most iconic Super Hero, Spider-Man features the acrobatic abilities, improvisation and web-slinging that the wall-crawler is famous for, while also introducing elements never-before-seen in a Spider-Man game. From traversing with parkour and utilizing the environment, to new combat and blockbuster set pieces, it's Spider-Man unlike any you've played before.' Below the description are three images: two from the game and one from the E3 2016 Announce Trailer. A 'View bigger' button is located below the trailer image. At the bottom left, there's a 'Go back' button. The footer contains copyright information: '© 2021 My Things' and links to 'About us', 'Terms and conditions', and 'Privacy policy'.

My collectionMy statsReleases

Test User

GamesBooksFilms




Your Game Collection Stats

Wanted:6Owned:14Started:10Finished:6Completed:5Abandoned:2Total:19

Completion

This chart shows how many games have you either finished, fully completed or just abandoned.


FinishedCompletedAbandoned



Backlog

This chart shows how many owned games have you started playing and how many are waiting in your backlog.

StartedNot started



© 2021 My Things

About us

Terms and conditions

Privacy policy

My collectionMy statsReleases

Test User

May 2021

today<>

Sun	Mon	Tue	Wed	Thu	Fri	Sat
						1
2	3	4	5	6	7 Resident Evil Village	8
9	10	11	12	13 Mass Effect Legendary Edition	14	15
16	17	18	19	20	21	22
23	24 BioShock: Collector's Edition	25	26	27	28	29
30	31					

© 2021 My Things

About us

Terms and conditions

Privacy policy

Página 63 de 73

- Edge

My collection My stats Releases Test User

Games Books Films

Search something... Search Clear

Sort: Updated (New to Old) Showing 8 of 19 results

Add Game

☐ Wanted
☐ Owned
☐ Physical
☐ Digital
☐ Plus
☐ Now
☐ Started
☐ Finished
☐ Completed
☐ Abandoned

Filter Clear

Moss
 Story: 4 Hours Completionist: 6 Hours
 Wanted Physical Plus Started Completed
 Owned Digital Now Finished Abandoned

Marvel's Spider-Man
 Story: 17 Hours Completionist: 33 Hours
 Wanted Physical Plus Started Completed
 Owned Digital Now Finished Abandoned

Judgment
 Story: 26 Hours Completionist: 38 Hours
 Wanted Physical Plus Started Completed
 Owned Digital Now Finished Abandoned

Yakuza Kiwami 2
 Story: 18½ Hours Completionist: 74½ Hours
 Wanted Physical Plus Started Completed
 Owned Digital Now Finished Abandoned

My collection My stats Releases Test User

Marvel's Spider-Man

Release date: 7 September 2018

← Go back

Starring the world's most iconic Super Hero, Spider-Man features the acrobatic abilities, improvisation and web-slinging that the wall-crawler is famous for, while also introducing elements never-before-seen in a Spider-Man game. From traversing with parkour and utilizing the environment, to new combat and blockbuster set pieces, it's Spider-Man unlike any you've played before.


View bigger

© 2021 My Things [About us](#) [Terms and conditions](#) [Privacy policy](#)

My collectionMy statsReleases

Test User

GamesBooksFilms



Your Game Collection Stats


Wanted:6Owned:14Started:10Finished:6Completed:5Abandoned:2

Total:19

Completion

This chart shows how many games have you either finished, fully completed or just abandoned.


FinishedCompletedAbandoned



Backlog

This chart shows how many owned games have you started playing and how many are waiting in your backlog.

StartedNot started



© 2021 My Things

About us

Terms and conditions

Privacy policy

My collectionMy statsReleases

Test User

May 2021

today<>

Sun	Mon	Tue	Wed	Thu	Fri	Sat
						1
2	3	4	5	6	7 Resident Evil Village	8
9	10	11	12	13 Mass Effect Legendary Edition	14	15
16	17	18	19	20	21	22
23	24 Sonic Frontiers	25	26	27	28	29
30	31					

© 2021 My Things

About us

Terms and conditions

Privacy policy

Página 65 de 73

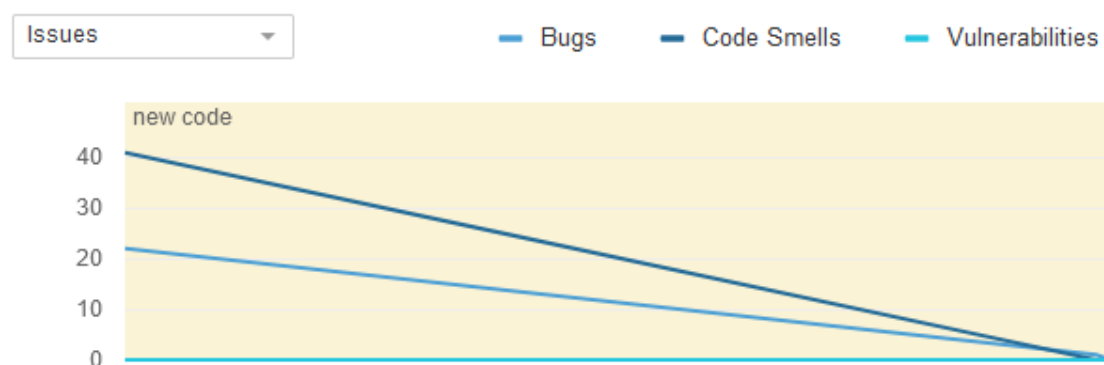
6.3 ANÁLISIS ESTÁTICO

Este análisis se realiza sobre el código fuente de la aplicación y proporciona distintas métricas para medir la calidad del código.

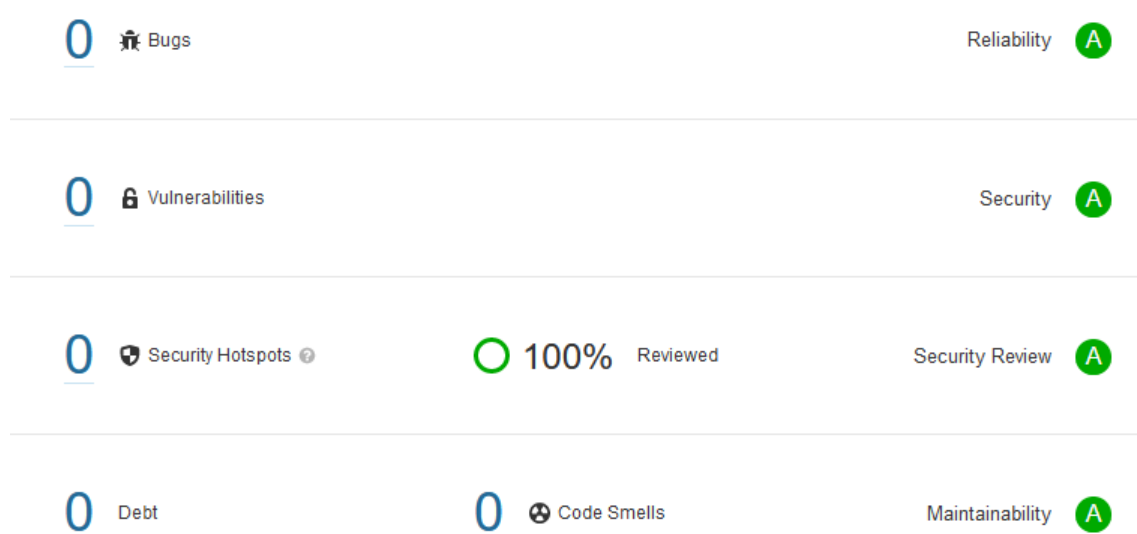
Para la realización de este análisis se ha utilizado la herramienta Sonarqube.

Sonarqube es una plataforma para evaluar código fuente. Es software libre y usa diversas herramientas de análisis estático de código fuente para informar sobre código duplicado, estándares de codificación, pruebas unitarias, cobertura de código, complejidad ciclomática, errores potenciales, comentarios y diseño del software.

Se ha utilizado Sonarqube para ir resolviendo los distintos problemas con el código:



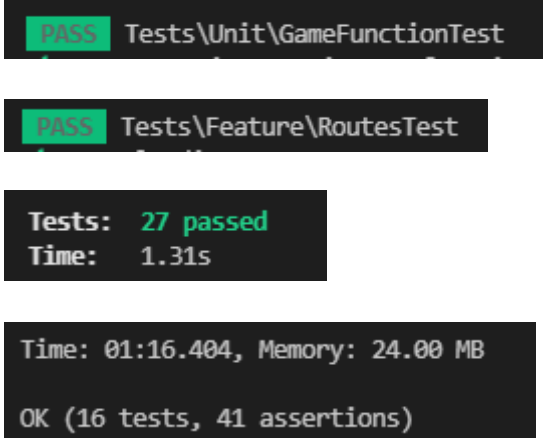
Hasta obtener un resultado final de calidad:



6.4 RESULTADOS OBTENIDOS.

Todos los tipos de tests realizados han devuelto en un principio varios errores de diversa gravedad, desde funciones devolviendo un resultado incorrecto hasta omisiones en la ayuda a la accesibilidad, pasando por prácticas poco recomendadas.

Estos errores se han ido solucionando y vuelto a comprobar hasta obtener resultados satisfactorios en todas las pruebas.











Tests\Unit\GameFunctionTest

Tests\Feature\RoutesTest

Tests: 27 passed
Time: 1.31s

Time: 01:16.404, Memory: 24.00 MB
OK (16 tests, 41 assertions)

 Bugs	 Vulnerabilities	 Hotspots Reviewed	 Code Smells
0 	0 	100% 	0 

7. CONCLUSIONES.

7.1 POSIBLES AMPLIACIONES Y MODIFICACIONES.

- Utilizar la fecha de añadido en la tabla de objetos del usuario para implementar estadísticas por años o periodos de tiempo personalizables.
- Permitir guardar un comentario personal del usuario en cada objeto.
- Versión demo con usuario de prueba para usuarios sin registrar.
- Elección del tema (otra paleta de colores, modo oscuro, etc.).
- Panel de administración.

7.2 LICENCIA

La licencia elegida para la aplicación es la GNU General Public License v3.0, la cual es compatible con las licencias de los paquetes utilizados en el proyecto.

Para obtener un listado de las licencias de las dependencias tanto de Composer como de npm se han utilizado las funcionalidades que incluyen estos gestores para proporcionar un listado de ellas:

```
PS C:\Users\Angela\Desktop\LARAVEL\mythings> composer licenses --format=summary
```

License	Number of dependencies
MIT	78
BSD-3-Clause	31
Apache-2.0	2

```
PS C:\Users\Angela\Desktop\LARAVEL\mythings> npx license-checker --summary
```

```
npx: instaló 57 en 6.339s
```

```

├─ MIT: 796
├─ ISC: 53
├─ BSD-2-Clause: 24
├─ BSD-3-Clause: 11
├─ Apache-2.0: 7
├─ Public Domain: 2
├─ CC0-1.0: 2
├─ 0BSD: 2
├─ (CC-BY-4.0 AND OFL-1.1 AND MIT): 1
├─ (MIT OR Apache-2.0): 1
├─ CC-BY-4.0: 1
├─ MIT OR GPL-2.0-or-later: 1
├─ (BSD-3-Clause OR GPL-2.0): 1
├─ (MIT AND Zlib): 1
├─ MIT*: 1
├─ (MIT AND BSD-3-Clause): 1
├─ (MIT OR CC0-1.0): 1
└─ BSD: 1
```

La GNU General Public License (GNU GPL) es una licencia de derecho de autor ampliamente usada en el mundo del software libre y código abierto que garantiza a los usuarios finales la libertad de usar, estudiar, compartir y modificar el software.

Su propósito es doble: declarar que el software cubierto por esta licencia es libre, y protegerlo de intentos de apropiación que restrinjan esas libertades a nuevos usuarios cada vez que la obra es distribuida, modificada o ampliada.

La GPL se distingue del dominio público o de otras licencias de software libre conocidas como permisivas por hacer hincapié en el copyleft, una práctica legal que consiste en propiciar el libre uso y distribución de una obra, exigiendo que las copias y derivados perpetúen la misma licencia.

Los derechos de distribución otorgados por la GPL para versiones modificadas de la obra no son incondicionales. Cuando alguien distribuye bajo GPL añadiendo a la obra sus propias modificaciones, los requisitos para la distribución de la totalidad de la obra no puede ser mayor que los requisitos que están en la GPL.

Copyleft por lo tanto utiliza la ley de copyright para lograr lo opuesto de su propósito usual: en lugar de imponer restricciones, otorga derechos, de tal manera que garantice que los derechos no puedan ser posteriormente quitados o restringidos. También asegura que si los derechos ilimitados de redistribución no se conceden o se produce cualquier falla legal se encuentra bajo la protección de la ley.

Copyleft solo se aplica cuando se trata de redistribuir el programa. Según sus bases, está permitido hacer privadas las modificaciones realizadas, sin obligación de divulgar las modificaciones siempre y cuando este software sea de uso propio.

Términos y condiciones de LA GNU GPL v3.0:

Da permiso a realizar modificaciones a una obra, realizar copias y distribuirla o distribuir cualquiera de sus versiones derivadas. Con esta licencia está permitido cobrar por la distribución de cada copia, o no cobrar nada. Este último punto distingue las licencias GPL de las licencias de software que prohíben la distribución comercial. La Free Software Foundation argumenta que en el software libre no debe haber cabida para las restricciones comerciales por lo que las obras bajo este tipo de licencias pueden ser vendidas a cualquier precio.

La GPL, además, establece que un distribuidor no puede imponer "restricciones sobre los derechos otorgados por la GPL". Esta prohíbe actividades como la distribución del software bajo un acuerdo de confidencialidad o contrato.

La Free Software Foundation no permite la aplicación de derechos de copyright a una obra licenciada bajo GPL, al menos que el autor los aplique explícitamente.

7.3 VALORACIÓN PERSONAL.

Con la realización de este proyecto he conseguido ampliamente los objetivos generales que me planteé al comienzo del proyecto:

- Aplicar y ampliar los conocimientos adquiridos durante el ciclo formativo.
- Aprender a consumir datos proporcionados por servicios de terceros.
 - Consumir datos de una API.
 - Extraer datos de un sitio web.

Al realizar un proyecto completo en el que he tenido que desempeñar el papel de varios roles he podido poner en práctica lo aprendido en todos los módulos profesionales del ciclo formativo.

Esto ha incluido entre otras cosas:

- Uso de entorno de desarrollo.
- Diseño de la interfaz.
- Diseño y gestión de la base de datos.
- Programación en entorno servidor.
- Programación en entorno cliente.
- Uso de lenguajes de marcas.
- Despliegue de la aplicación web.

Otro aspecto importante de lo aprendido en el proyecto es el haber tenido que emplear herramientas de planificación de proyecto tales como:

- Especificaciones del proyecto.
- Identificación de tareas.
- Planificación temporal (temporalización, diagrama de Gantt).
- Determinación de recursos necesarios.
- Seguimiento y control del estado de las tareas (tablero Kanban).

En conclusión, no solamente he puesto en práctica y afianzado los conocimientos adquiridos durante el curso sino que además he aprendido y utilizado nuevas técnicas, recursos y herramientas, por lo que ha sido una experiencia muy enriquecedora tanto a nivel profesional como personal.

8. BIBLIOGRAFÍA.

8.1 APIS

IGDB API docs. <https://api-docs.igdb.com/>

GIANT BOMB API documentation. <https://www.giantbomb.com/api/documentation/>

RAWG Video Games Database API. <https://api.rawg.io/docs/>

Apicalypse Syntax. <https://apicalypse.io/syntax/>

8.2 HERRAMIENTAS

PHP Documentation. <https://www.php.net/docs.php>

Laravel Documentation. <https://laravel.com/docs/8.x>

Bootstrap Documentation. <https://getbootstrap.com/docs/4.6>

Chart.js Documentation. <https://www.chartjs.org/docs/latest/>

FullCalendar Documentation. <https://fullcalendar.io/docs/getting-started>

8.3 LARAVEL

Laravel Controllers. <https://laravel.com/docs/8.x/controllers>

Laravel HTTP Client. <https://laravel.com/docs/8.x/http-client>

Laravel File Storage. <https://laravel.com/docs/8.x/filesystem>

Laravel Database: Pagination. <https://laravel.com/docs/8.x/pagination>

Laravel Validation. <https://laravel.com/docs/8.x/validation>

8.4 AUTENTICACIÓN

Laravel Authentication. <https://laravel.com/docs/7.x/authentication>

Laravel Daily - Laravel Too Many Login Attempts: Restrict and Customize.

<https://laraveldaily.com/laravel-too-many-login-attempts-restrict-and-customize/>

Laravel Passwords. <https://laravel.com/docs/7.x/passwords>

8.5 ORM

Laravel Eloquent. <https://laravel.com/docs/8.x/eloquent>

Laravel Database: Migrations. <https://laravel.com/docs/8.x/migrations>

8.6 EMAIL

Laravel Mail. <https://laravel.com/docs/8.x/mail>

Stopping Form SPAM in Laravel. <https://itnext.io/stopping-form-spam-in-laravel-76760bf84bd>

8.7 CRAWLER

Guzzle Documentation. <https://docs.guzzlephp.org/en/stable/>

Symfony - The Dom Crawler Component.

https://symfony.com/doc/current/components/dom_crawler.html

8.8 OAUTH

Laravel Socialite. <https://laravel.com/docs/8.x/socialite>

Using OAuth 2.0 to Access Google APIs.

<https://developers.google.com/identity/protocols/oauth2>

8.9 UX Y ACCESIBILIDAD

Primary & Secondary Action Buttons. <https://uxplanet.org/primary-secondary-action-buttons-c16df9b36150>

Web Content Accessibility Guidelines (WCAG) Overview.

<https://www.w3.org/WAI/standards-guidelines/wcag/>

Cómo volver tu página web accesible. <https://www.dreamhost.com/blog/es/volver-tu-pagina-web-accesible/>

Keyboard Accessibility. <https://webaim.org/techniques/keyboard/>

Why Autoplay is an accessibility issue. <https://www.abilitynet.org.uk/news-blogs/why-autoplay-accessibility-issue>

Sordoceguera, accesibilidad web y captchas.

<https://tothomweb.com/es/blog/sordoceguera-accesibilidad-web-y-captchas>

8.10 TESTS Y PRUEBAS

Laravel Dusk. <https://laravel.com/docs/8.x/dusk>

Laravel Dusk Tutorial Series. <https://5balloons.info/introduction-to-laravel-dusk/>

Documentation for PHPUnit. <https://phpunit.de/documentation.html>

SonarQube Documentation. <https://docs.sonarqube.org/latest/>

SonarScanner. <https://docs.sonarqube.org/latest/analysis/scan/sonarscanner/>

8.11 LICENCIAS

Free Software Foundation Licensing. <https://www.fsf.org/licensing/>

GNU General Public License.

https://es.wikipedia.org/wiki/GNU_General_Public_License

Copyleft. <https://es.wikipedia.org/wiki/Copyleft>

Adding a license to a repository. <https://docs.github.com/en/communities/setting-up-your-project-for-healthy-contributions/adding-a-license-to-a-repository>

8.12 DOCUMENTACIÓN

Doxygen - Installation. <https://www.doxygen.nl/manual/install.html>

Doxygen - Documenting the code. <https://www.doxygen.nl/manual/docblocks.html>

Doxygen - Usage. https://www.doxygen.nl/manual/doxygen_usage.html

8.13 GESTIÓN DE PROYECTO

Introducción a Trello. <https://trello.com/guide>

Priorización de Requisitos Software con MoSCoW. <https://adrianalonso.es/project-management/priorizacion-requisitos-software-con-moscow/>

8.14 UML

dbdiagram.io - Docs (Public). <https://www.notion.so/dbdiagram-io-Docs-Public-d405d7d9efd4419d9b6402eb6feb8cd1>

What is Entity Relationship Diagram (ERD)? <https://www.visual-paradigm.com/guide/data-modeling/what-is-entity-relationship-diagram/>

UML Class Diagram Tutorial. <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/>

What is Use Case Diagram? <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-use-case-diagram/>

What is Use Case Specification? <https://www.visual-paradigm.com/guide/use-case/what-is-use-case-specification/>

What is activity diagram? <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-activity-diagram/>

Wireframe vs Storyboard vs Wireflow vs Mockup vs Prototyping. <https://www.visual-paradigm.com/guide/ux-design/wireframe-vs-storyboard-vs-wireflow-vs-mockup-vs-prototyping/>