**IES Laguna de Joatzel Grado Superior de Desarrollo de Aplicaciones Web**

**Trabajo Fin de Grado**

**Plataforma Web**

**Iván De Las Heras Martín | José Galea Torrecilla**

**Tutor: Alain Fernández Fernández**

**abril de 2022**

**Índice**

Contenido

[PALABRAS CLAVE 4](#_Toc101891146)

[GLOSARIO 5](#_Toc101891155)

[1| INTRODUCCIÓN 5](#_Toc101891156)

[1.1 OBJETIVOS DEL PROYECTO 6](#_Toc101891157)

[1.2 JUSTIFICACIÓN 6](#_Toc101891158)

[2| ANÁLISIS 7](#_Toc101891159)

[2.1 CATÁLOGO DE REQUISITOS 8](#_Toc101891160)

[2.1.1 REQUISITOS FUNCIONALES 8](#_Toc101891161)

[2.1.2 REQUISITOS NO FUNCIONALES 9](#_Toc101891162)

[2.2 PLANIFICACIÓN PREVIA 10](#_Toc101891163)

[2.3 FASES DE REALIZACIÓN 11](#_Toc101891164)

[3| DISEÑO 12](#_Toc101891165)

[3.1 TECNOLOGÍAS UTILIZADAS 12](#_Toc101891166)

[3.1.1 HTML 12](#_Toc101891167)

[3.1.2 CSS 13](#_Toc101891168)

[3.1.3 PHP 14](#_Toc101891169)

[3.1.4 JavaScript 15](#_Toc101891170)

[3.1.5 PHPStorm 15](#_Toc101891171)

[3.1.6 XAMPP 16](#_Toc101891172)

[3.1.7 GitHub 17](#_Toc101891173)

[3.2 DISEÑO DEL PROYECTO 17](#_Toc101891174)

[3.2.1Diagramas de Casos de Uso 17](#_Toc101891175)

[3.2.1.1 Inicio de Sesión 18](#_Toc101891176)

[3.2.1.2Menú de Juegos 18](#_Toc101891177)

[3.2.1.3Mi Cuenta 18](#_Toc101891178)

[3.2.1.4Mis Juegos 19](#_Toc101891179)

[3.2.1.5Ficha de Juego 19](#_Toc101891180)

[3.2.2 Diagrama de Flujo del proyecto 21](#_Toc101891181)

[3.3 DISEÑO 22](#_Toc101891182)

[4| IMPLEMENTACIÓN 22](#_Toc101891183)

[- Inicio de sesión y creación de nuevas cuentas 22](#_Toc101891184)

[En caso de que el usuario ya esté registrado en el sistema procederá a iniciar sesión, para ello entrará en el apartado de “Iniciar sesión” que se encuentra en la misma página de inicio. 24](#_Toc101891185)

[- Cerrar sesión 26](#_Toc101891186)

[- Validación Usuario (PHP) 27](#_Toc101891187)

[- Validación Usuario (PHP) 28](#_Toc101891188)

[- Validación Usuario (PHP) 29](#_Toc101891189)

[- Validación Usuario (PHP) 31](#_Toc101891190)

[- Validación Usuario (JS) 37](#_Toc101891191)

[- Validación Usuario (JS) 38](#_Toc101891192)

[5| PRUEBAS 46](#_Toc101891193)

[6| TRABAJO FUTURO 48](#_Toc101891194)

[7| DESPLIEGUE 48](#_Toc101891195)

[7.1 CONFIGURACIÓN Y DESPLIEGUE 48](#_Toc101891196)

[8| CONCLUSIONES 48](#_Toc101891197)

[9| ANEXO 49](#_Toc101891198)

[9.1 MANUAL DE USUARIO 49](#_Toc101891199)

[10| BIBLIOGRAFÍA 49](#_Toc101891200)

# PALABRAS CLAVE

La siguiente lista contiene palabras descriptivas que podrán utilizarse para catalogar el proyecto.

* Php
* JavaScript
* HTML
* CSS
* Aplicación Web
* MySQL
* AJAX
* JQuery
* Cookies
* Sesiones

# GLOSARIO

# 1| INTRODUCCIÓN

## 1.1 OBJETIVOS DEL PROYECTO

Los objetivos de un proyectoson el conjunto de resultados que se espera obtener tras concluir con el desarrollo activo del mismo.

El objetivo principal de este proyecto es el de desarrollar una plataforma web en la que los usuarios puedan acceder a un conjunto de videojuegos. La plataforma debe presentar una interfaz gráfica sencilla e intuitiva que invite al uso por parte del usuario; así mismo, cada usuario podrá acceder a los videojuegos que haya adquirido siempre que quiera, y podrá adquirir nuevos y eliminar los ya presente siempre que estime oportuno. Se implementará una serie de opciones para facilitar la búsqueda de videojuegos por parte del usuario (búsqueda por categoría, por nombre, etc).

Además, la plataforma dispondrá de un menú de navegación cómodo y efectivo mediante el cual los usuarios podrán navegar por los distintos apartados de la misma sin desubicarse.

Para garantizar la seguridad de los datos de los usuarios y el correcto uso de los mismos, la plataforma presentará un sistema de inicio de sesión que solicitará unas credenciales (usuario y contraseña) a todo aquel que intente acceder al contenido de la plataforma. Así mismo, se facilitarán distintas herramientas para facilitar la gestión de las credenciales por parte de los usuarios: la interfaz del inicio de sesión presentará un apartado para la creación de nuevas cuentas así como la posibilidad de recuperar la contraseña en caso de que se haya extraviado.

En conclusión, la finalidad de este proyecto es desarrollar una completa plataforma web en la que los usuarios puedan interactuar con los distintos elementos de la misma permitiéndoles realizar acciones como iniciar sesión, ver los distintos videojuegos, adquirir y eliminar los que se consideren y, en última instancia, jugar a todos aquellos videojuegos que se hayan adquirido de forma cómoda y sin necesidad de realizar ningún tipo de instalación.

## 1.2 JUSTIFICACIÓN

La finalidad de este apartado consiste en explicar, de forma argumentada, las razones que motivan a la realización de este proyecto; intentando responder a las preguntas “¿por qué?” y “¿para qué?” se lleva a cabo este proyecto.

A continuación, se procede a detallar dichas razones:

El uso de plataformas web ha crecido exponencialmente en los últimos años, cada vez son más los usuarios que prescinden de realizar tediosas instalaciones para poder disfrutar de los servicios de una aplicación. En su lugar, el mercado se está decantando por el uso de espacios en Internet que permiten la ejecución de diversas aplicaciones o programas en un mismo lugar. De esta forma, se aspira a posicionar a la web como principal canal para adquirir contenidos.

De hecho, las plataformas web proporcionan una mayor libertad de uso a los usuarios ya que pueden acceder a su contenido desde múltiples dispositivos. Si bien es cierto que la experiencia del usuario puede estar algo ligada a la calidad de la conexión a la red que presente, el poder conectarse desde cualquier lugar del mundo sitúa a las plataformas web por encima de las aplicaciones de escritorio en lo que a usabilidad y accesibilidad se refiere.

Nuestra intención es la de implementar un software que permita a los usuarios disfrutar de un conjunto de videojuegos en un entorno cómodo y accesible; proporcionándoles un amplio abanico de opciones.

Por tanto, hemos decidido que la mejor manera de llevar a cabo dicho propósito es desarrollando una plataforma web de videojuegos como proyecto.

# 2| ANÁLISIS

En este capítulo se presenta la fase de análisis, parte inicial de todo proyecto software, se definen los requisitos y se muestra una vista global de la arquitectura pensada para el sistema. En este capítulo se tendrá en cuenta este catálogo de requisitos como base para el diseño de todos los aspectos de la plataforma web, así como la planificación y sus fases.

Es por ello por lo que la fase de análisis es de suma importancia para las siguientes fases de todo el producto software. Es en este capítulo donde se deben asentar las bases, a modo de cimientos, del proyecto, y, a partir de las cuales se construirá todo lo demás. El proceso a seguir se basa en primer lugar en una buena definición de requisitos y en elegir una metodología de desarrollo acorde con el proyecto. La arquitectura de la plataforma web estará basada en un conjunto de páginas web a las que el usuario podrá acceder a través de un menú de navegación presente en todas ellas.

A continuación, se desglosan la funcionalidad y las características a modo de catálogo de requisitos, teniendo en cuenta tanto requisitos funcionales como no funcionales.

## 2.1 CATÁLOGO DE REQUISITOS

El catálogo de requisitos es la especificación del comportamiento que se espera de cualquier proyecto software. Estudiando otras aplicaciones similares, se ha predefinido una serie de requisitos que se consideran indispensables para el proyecto. A continuación, se muestra una enumeración y breve descripción de los requisitos establecidos para el diseño y desarrollo de la plataforma web.

### 2.1.1 REQUISITOS FUNCIONALES

En este apartado se especifica cuáles son las tareas principales que debe realizar nuestro proyecto software.

En esta tabla se detallan los siguientes requisitos funcionales:

|  |  |  |
| --- | --- | --- |
| **ID** | **Requisito** | **Descripción** |
| **RF-01** | Inicio de sesión | Inicio de sesión del usuario al introducir el su identificador y su contraseña. |
| **RF-02** | Menú desplegable | Barra lateral izquierda con menú desplegable para la navegación del usuario por la plataforma web. |
| **RF-03** | Cerrar Sesión | La plataforma web presentará la opción de cerrar sesión por parte del usuario. |
| **RF-04** | Fichas de videojuegos | Cada videojuego de la plataforma web tendrá su respectiva ficha técnica a la que los usuarios autorizados podrán acceder. |
| **RF-05** | Adquirir videojuegos | La plataforma web permitirá a los usuarios autorizados adquirir videojuegos. |
| **RF-06** | Eliminar videojuegos | La plataforma web permitirá a los usuarios autorizados eliminar de su biblioteca los videojuegos adquiridos que consideren oportunos. |
| **RF-07** | Búsqueda de videojuegos | La plataforma web presentará diversos métodos de búsqueda de videojuegos (por categoría, nombre, etc). |
| **RF-08** | Envío correo electrónico | Se enviará un correo electrónico a cada usuario registrado dándole la bienvenida. |
| **RF-09** | Creación nuevas cuentas | La plataforma web gestionará la creación de nuevas cuentas realizando las validaciones pertinentes. |
| **RF-10** | Ajustes de usuario | Los usuarios podrán modificar sus datos personales (nombre, apellidos, email, foto de perfil, etc) una vez creada la cuenta en la plataforma. |
| **RF-11** | Recordar contraseña usuario | La plataforma web permitirá modificar la contraseña y recuperar la cuenta en caso de olvidarla llevando a cabo una serie de validaciones de seguridad. |
| **RF-12** | Modo Claro/Oscuro | Los usuarios podrán configurar la apariencia de la plataforma web pudiendo elegir entre un modo claro y otro oscuro. |

### 2.1.2 REQUISITOS NO FUNCIONALES

En este apartado se especifica cuáles son los requisitos complementarios o atributos de calidad de nuestro proyecto software.

Estos especifican criterios que juzgan operaciones del sistema en lugar de su comportamiento (requisitos funcionales).

En esta tabla se detallan los siguientes requisitos no funcionales:

|  |  |  |
| --- | --- | --- |
| **ID** | **Requisito** | **Descripción** |
| **RNF-01** | Interfaz y usabilidad | La plataforma web debe constar de interfaces intuitivas y sencillas, que resulten atractivas a los usuarios. |
| **RNF-02** | Accesibilidad | La plataforma web debe ser accesible para distintos usuarios. El usuario debe saber en todo momento en que apartado de la plataforma se encuentra. |
| **RNF-03** | Páginas web responsivas | La plataforma web debe presentar páginas web que se adapten a los diferentes tamaños de pantalla que los usuarios puedan presentar. |
| **RNF-04** | Rendimiento | La plataforma web debe presentar mínimos tiempos de carga entre páginas y un acceso a BBDD lo más optimizado posible. |
| **RNF-05** | Escalabilidad y desarrollo ágil | El proyecto software debe estar diseñado para poder aumentar exponencialmente el número de usuarios, videojuegos, apartados, etc sin comprometer el sistema. |
| **RNF-06** | Pruebas | Deben realizarse pruebas de todo tipo para garantizar la calidad del producto y eliminar bugs y errores del sistema. |
| **RNF-07** | Refactorización | Reestructuración del código para una mayor legibilidad de los métodos. |

## 2.2 PLANIFICACIÓN PREVIA

La planificación previa del proyecto se puede definir como la estructuración que el equipo ha hecho por periodos de tiempo de dicho proyecto hasta llegar al producto final.

Dicha planificación ha sido la siguiente:

**Semana 1:** Elaboración de la estructura del proyecto, software a utilizar y herramientas necesarias.

**Semana 2:**

**Semana 3:**

**Semana 4:**

**Semana 5:**

**Semana 6:**

**Semana 7:**

**Semana 8:**

**Semana 9:**

**Semana 10:**

**Semana 11:**

**Semana 12:**

## 2.3 FASES DE REALIZACIÓN

En este apartado se da una visión general de las fases seguidas a lo largo de la realización del proyecto, que son las correspondientes a la mayoría de un desarrollo software.

•**Estudio**: Investigación sobre las posibles tecnologías a utilizar, posible estudio de mercado y de viabilidad. Indagación y pruebas sobre aplicaciones relacionadas, con el objetivo de extraer lo que se suele utilizar en proyectos similares.

•**Análisis**: Definición de requisitos con el objetivo de dilucidar exactamente qué es lo que se desea construir.

•**Diseño**: Tras finalizar las fases anteriores, se procede a diseñar la arquitectura de la aplicación web. Esto incluye diversos diagramas que definirán el diseño las páginas, la interfaz de usuario, las herramientas a utilizar, etc.

•**Implementación**: Programación

•**Pruebas**: Una vez finalizadas todas las fases anteriores, se hace necesaria la realización de una fase de pruebas generales para comprobar la calidad y efectividad del software desarrollado.

•**Documentación**: La documentación es algo indispensable para el correcto mantenimiento de cualquier producto software. Tanto a nivel de código como de documento, con el fin de dejar constancia por escrito de las fases llevadas a cabo en el ciclo de vida del producto software.

•**Mantenimiento**: Una vez finalizadas las fases anteriores, quedaría realizar un mantenimiento periódico para evitar errores o si surgen solventarlos con actualizaciones en la plataforma web.

# 3| DISEÑO

## 3.1 TECNOLOGÍAS UTILIZADAS

En este apartado se enumeran las diferentes herramientas y tecnologías utilizadas durante el desarrollo del proyecto.

### 3.1.1 HTML

HTML(HyperText Markup Language), hace referencia al [lenguaje de marcado](https://es.wikipedia.org/wiki/Lenguaje_de_marcado) para la elaboración de [páginas web](https://es.wikipedia.org/wiki/P%C3%A1gina_web). Propone una estructura básica y un código (denominado código HTML) para la definición de contenido de una página web, como texto, imágenes, videos, entre otros. Es un estándar a cargo del [World Wide Web Consortium](https://es.wikipedia.org/wiki/World_Wide_Web_Consortium) ([W3C](https://es.wikipedia.org/wiki/W3C)), organización dedicada a la estandarización de casi todas las tecnologías ligadas a la web, sobre todo en lo referente a su escritura e interpretación.

HTML se considera el lenguaje web más importante ya que es el estándar que se ha impuesto en la visualización de páginas web y es el que todos los navegadores actuales han adoptado.



Logo de HTML5

### 3.1.2 CSS

CSS (Cascading Style Sheets) es un lenguaje de [diseño gráfico](https://es.wikipedia.org/wiki/Dise%C3%B1o_gr%C3%A1fico) para definir y crear la presentación de un documento estructurado escrito en un [lenguaje de marcado](https://es.wikipedia.org/wiki/Lenguaje_de_marcado). ​Es muy usado para establecer el diseño visual de los documentos web, e interfaces de usuario escritas en [HTML](https://es.wikipedia.org/wiki/HTML) o [XHTML](https://es.wikipedia.org/wiki/XHTML).

Junto con HTML y [JavaScript](https://es.wikipedia.org/wiki/JavaScript), CSS es una tecnología usada por muchos [sitios web](https://es.wikipedia.org/wiki/Sitio_web) para crear páginas visualmente atractivas, interfaces de usuario para [aplicaciones web](https://es.wikipedia.org/wiki/Aplicaci%C3%B3n_web) y [GUIs](https://es.wikipedia.org/wiki/GUI" \o "GUI) para muchas aplicaciones [móviles](https://es.wikipedia.org/wiki/Smartphone).​

CSS está diseñado principalmente para marcar la separación del contenido del documento y la forma de presentación de este, características tales como las capas o layouts, los colores y las fuentes.​ Esta separación busca mejorar la accesibilidad del documento, proveer más flexibilidad y control en la especificación de características presentacionales, permitir que varios documentos HTML compartan un mismo estilo usando una sola hoja de estilos separada en un archivo .css, y reducir la complejidad y la repetición de código en la estructura del documento.

La especificación CSS es mantenida por el [World Wide Web Consortium (W3C)](https://es.wikipedia.org/wiki/World_Wide_Web_Consortium); que proporciona una herramienta de validación de CSS gratuita para los documentos CSS.



Logo de CSS3

### 3.1.3 PHP

PHP es un [lenguaje de programación](https://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n) de uso general que se adapta especialmente al [desarrollo web](https://es.wikipedia.org/wiki/Desarrollo_web).

Puede ser desplegado en la mayoría de los [servidores web](https://es.wikipedia.org/wiki/Servidor_web) y en todos los sistemas operativos y plataformas. El lenguaje PHP se encuentra instalado en más de 20 millones de [sitios web](https://es.wikipedia.org/wiki/Sitio_web) y en un millón de servidores.

El código PHP suele ser procesado en un [servidor web](https://es.wikipedia.org/wiki/Servidor_web) por un [intérprete](https://es.wikipedia.org/wiki/Int%C3%A9rprete_(inform%C3%A1tica)) PHP implementado como un [módulo](https://es.wikipedia.org/wiki/Complemento_(inform%C3%A1tica)), un “[daemon](https://es.wikipedia.org/wiki/Daemon_(inform%C3%A1tica)" \o "Daemon (informática))” o como un ejecutable de [interfaz de entrada común](https://es.wikipedia.org/wiki/Interfaz_de_entrada_com%C3%BAn) (CGI). En un servidor web, el resultado del código PHP interpretado y ejecutado (que puede ser cualquier tipo de datos, como [HTML](https://es.wikipedia.org/wiki/HTML) generado o datos de imágenes binarias) formaría la totalidad o parte de una respuesta [HTTP](https://es.wikipedia.org/wiki/Protocolo_de_transferencia_de_hipertexto).



Logo de PHP

### 3.1.4 JavaScript

JavaScript (abreviado comúnmente JS) es un [lenguaje de programación](https://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n) [interpretado](https://es.wikipedia.org/wiki/Int%C3%A9rprete_(inform%C3%A1tica)) que se define como [orientado a objetos](https://es.wikipedia.org/wiki/Programaci%C3%B3n_orientada_a_objetos),​[basado en prototipos](https://es.wikipedia.org/wiki/Programaci%C3%B3n_basada_en_prototipos), imperativo, [débilmente tipado](https://es.wikipedia.org/wiki/Tipado_fuerte#Lenguajes_no_tipados) y [dinámico](https://es.wikipedia.org/wiki/Tipado_din%C3%A1mico).

Se utiliza principalmente del [lado del cliente](https://es.wikipedia.org/wiki/Cliente_(inform%C3%A1tica)), implementado como parte de un [navegador web](https://es.wikipedia.org/wiki/Navegador_web) permitiendo mejoras en la [interfaz de usuario](https://es.wikipedia.org/wiki/Interfaz_de_usuario) y [páginas web](https://es.wikipedia.org/wiki/P%C3%A1gina_web) dinámicas. No obstante, el uso de JavaScript en el [lado del servidor](https://es.wikipedia.org/wiki/Script_del_lado_del_servidor)en [aplicaciones](https://es.wikipedia.org/wiki/Aplicaci%C3%B3n_inform%C3%A1tica) externas a la [web](https://es.wikipedia.org/wiki/World_Wide_Web)(documentos [PDF](https://es.wikipedia.org/wiki/PDF), aplicaciones de escritorio, etc.) es también significativo.

JavaScript se diseñó con una sintaxis similar a [C](https://es.wikipedia.org/wiki/C_(lenguaje_de_programaci%C3%B3n)" \o "C (lenguaje de programación))aunque adopta nombres y convenciones del lenguaje de programación [Java](https://es.wikipedia.org/wiki/Java_(lenguaje_de_programaci%C3%B3n)). Sin embargo, Java y JavaScript tienen semánticas y propósitos diferentes.

Todos los navegadores modernos interpretan el código JavaScript integrado en las páginas web. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del [Document Object Model](https://es.wikipedia.org/wiki/Document_Object_Model) (DOM).

Javascript es el único lenguaje de programación que entienden de forma nativa los navegadores.



Logo de JavaScript

### 3.1.5 PHPStorm

PhpStorm es un [IDE](https://en.wikipedia.org/wiki/Integrated_development_environment) (entorno de desarrollo integrado) multiplataforma para [PHP](https://en.wikipedia.org/wiki/PHP),[desarrollado por la empresa JetBrains](https://en.wikipedia.org/wiki/JetBrains).

PhpStorm proporciona un editor para [PHP](https://en.wikipedia.org/wiki/PHP), [HTML](https://en.wikipedia.org/wiki/HTML) y [JavaScript con análisis de código en tiempo real, prevención de errores y](https://en.wikipedia.org/wiki/JavaScript)[refactorizaciones](https://en.wikipedia.org/wiki/Refactoring) automatizadas para código PHP y JavaScript.

[El autocompleta de código](https://en.wikipedia.org/wiki/Code_completion) de PhpStorm es compatible con PHP 5.3, 5.4, 5.5, 5.6, 7.0, 7.1, 7.2, 7.3, 7.4, 8.0 y 8.1. [Incluye un editor de SQL](https://en.wikipedia.org/wiki/SQL) completo con resultados de consultas editables.

PhpStorm está escrito en [Java](https://en.wikipedia.org/wiki/Java_(programming_language)). Los usuarios pueden ampliar el IDE instalando plugings creados para PhpStorm o escribiendo los suyos propios. El software también se comunica con fuentes externas como XDebug.



Logo de PHPStorm

### 3.1.6 XAMPP

XAMPP es un paquete de [software libre](https://es.wikipedia.org/wiki/Software_libre), que consiste principalmente en el [sistema de gestión de bases de datos](https://es.wikipedia.org/wiki/Sistema_de_gesti%C3%B3n_de_bases_de_datos) [MySQL](https://es.wikipedia.org/wiki/MySQL), el [servidor](https://es.wikipedia.org/wiki/Servidor) web [Apache](https://es.wikipedia.org/wiki/Servidor_HTTP_Apache) y los [intérpretes](https://es.wikipedia.org/wiki/Int%C3%A9rpretes) para [lenguajes de script](https://es.wikipedia.org/wiki/Lenguaje_de_script) [PHP](https://es.wikipedia.org/wiki/PHP) y [Perl](https://es.wikipedia.org/wiki/Perl).

El nombre es en realidad un [acrónimo](https://es.wikipedia.org/wiki/Acr%C3%B3nimo): X (para cualquiera de los diferentes [sistemas operativos](https://es.wikipedia.org/wiki/Sistema_operativo)), A (Apache), M ([MariaDB/MySQL](https://es.wikipedia.org/wiki/MariaDB" \o "MariaDB)), P (PHP), P (Perl).

A partir de la versión 5.6.15, XAMPP cambió la [base de datos](https://es.wikipedia.org/wiki/Base_de_datos) MySQL por MariaDB, una bifurcación del desarrollo de MySQL con [licencia GPL](https://es.wikipedia.org/wiki/Licencia_GPL).

El programa se distribuye con la licencia [GNU](https://es.wikipedia.org/wiki/GNU) y actúa como un servidor web libre, fácil de usar y capaz de interpretar páginas dinámicas. A día de hoy, XAMPP está disponible para [Microsoft Windows](https://es.wikipedia.org/wiki/Microsoft_Windows), [GNU/Linux](https://es.wikipedia.org/wiki/GNU/Linux), [Solaris](https://es.wikipedia.org/wiki/Solaris_(sistema_operativo)" \o "Solaris (sistema operativo)) y [Mac OS X](https://es.wikipedia.org/wiki/Mac_OS_X).



Logo de XAMPP

### 3.1.7 GitHub

GitHub es una plataforma de desarrollo colaborativo para alojar proyectos que utiliza el sistema de [control de versiones](https://es.wikipedia.org/wiki/Control_de_versiones) [Git](https://es.wikipedia.org/wiki/Git). Se utiliza principalmente para la creación de [código fuente](https://es.wikipedia.org/wiki/C%C3%B3digo_fuente) de [programas](https://es.wikipedia.org/wiki/Programa_inform%C3%A1tico) de [ordenador](https://es.wikipedia.org/wiki/Ordenador).

El código de los proyectos alojados en GitHub se almacena generalmente de forma [pública](https://es.wikipedia.org/wiki/C%C3%B3digo_abierto).

GitHub es famoso por el uso de los repositorios (lugares virtuales alojados en la [nube](https://es.wikipedia.org/wiki/Computaci%C3%B3n_en_la_nube) en donde los usuarios pueden almacenar cualquier tipo de archivo).Suelen usarse para guardar archivos de texto que representan código en distintos [lenguajes de programación](https://es.wikipedia.org/wiki/Lenguajes_de_programaci%C3%B3n), los cuales a su vez conforman un programa o aplicación. Este es el servicio principal que ofrece GitHub: repositorios en donde los usuarios pueden almacenar el código de sus aplicaciones, ya sea de forma pública, privada o a través de una organización.

Actualmente, GitHub continúa siendo la plataforma más importante de colaboración para proyectos de [código abierto](https://es.wikipedia.org/wiki/C%C3%B3digo_abierto).



Logo de GitHub

## 3.2 DISEÑO DEL PROYECTO

### 3.2.1Diagramas de Casos de Uso

Los casos de uso que se presentan en el proyecto son los siguientes:

### 3.2.1.1 Inicio de Sesión



### 3.2.1.2Menú de Juegos



### 3.2.1.3Mi Cuenta



### 3.2.1.4Mis Juegos



### 3.2.1.5Ficha de Juego



### 3.2.2 Diagrama de Flujo del proyecto

Imagen 8. Diagrama de Flujo del proyecto

## 

## 3.3 DISEÑO

# 4| IMPLEMENTACIÓN

En el siguiente punto se detallará la parte de programación e implementación de dicho código en el proyecto.

## Inicio de sesión y creación de nuevas cuentas

A continuación se muestra una captura de pantalla de la página que se encarga del logging por parte del usuario.



Registro de nuevos usuarios

Como se puede ver en el apartado “Registrarse” se solicitan una serie de datos al usuario para generar una nueva cuenta en el sistema. Una vez enviado, los datos del formulario son debidamente tratados y si cumplen los valores esperados se procederá a crear la nueva cuenta al usuario.



Método utilizado para la creación de la nueva cuenta y su inserción en el sistema

Si el proceso se ha realizado correctamente, el usuario podrá acceder a la plataforma, en caso contrario se le indicará el error en la misma página de registro ya que esta no necesita ser refrescada en ningún caso (pues el sistema actualiza el estado de la página mediante interacciones AJAX).



Método utilizado para la validación de los datos de registro



El sistema notifica al usuario que el nickname que ha introducido ya está en uso mediante llamadas AJAX sin necesidad de actualizar la página.

# En caso de que el usuario ya esté registrado en el sistema procederá a iniciar sesión, para ello entrará en el apartado de “Iniciar sesión” que se encuentra en la misma página de inicio.



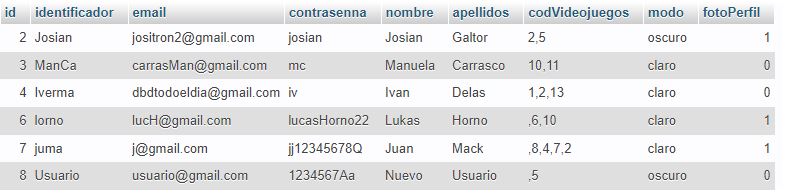
Inicio de sesión

Si el proceso se ha realizado correctamente, el usuario podrá acceder a la plataforma con su usuario. En caso de que los datos del logging (nickname y password) introducidos por el usuario sean incorrectos, el sistema le informará de ello.



Datos incorrectos

Tanto para el registro de nuevos usuarios como para el inicio de sesión de los que están ya registrados el sistema consulta la tabla “Usuarios” de la BBDD del proyecto.



Ejemplo de usuarios registrados en la tabla de “Usuarios”

## Cerrar sesión

Una vez que el usuario se encuentra en el sistema podrá salir del mismo sin riesgo de perder datos siempre que cierre sesión. Para ello, se le facilita en un menú desplegable una serie de opciones entre las que se encuentra el cierre de sesión.



Menú desplegable del usuario “Josian”

Cuando el usuario hace clic en esta opción se llama a la misma página que muestra las opciones para el logging del usuario sólo que esta vez el sistema le envía un parámetro indicando que se trata de un cierre de sesión por parte del usuario.



Sesión cerrada con éxito



Código necesario para mostrar el mensaje correspondiente al usuario

## Validación Usuario (PHP)

<?php  
require\_once "../../\_Varios.php";  
  
function validarUsuario($identificador, $email): int  
{  
 /\*SIGNIFICADOS RETURNS:  
 -1 -> El nickname NO EXISTE ERROR  
 -2 -> El email NO EXISTE ERROR  
 0 -> Los datos son correctos BIEN  
 \*/  
 $conexion = obtenerPdoConexionBD();  
  
//Primero compruebo que no exista el identificador ni el email en la BBDD  
 $sql = "SELECT \* FROM usuario WHERE BINARY identificador=?";  
 $select = $conexion->prepare($sql);  
 $select->execute([$identificador]);  
 $numRegistros = $select->rowCount();  
 if ($numRegistros == 0) {  
 //Devuelvo -1 que quiere decir que el nickname NO EXISTE  
 return -1;  
 } else {  
 $sql = "SELECT \* FROM usuario WHERE BINARY email=?";  
 $select = $conexion->prepare($sql);  
 $select->execute([$email]);  
 $numRegistros = $select->rowCount();  
 if ($numRegistros == 0) {  
 //Devuelvo -2 que quiere decir que el email NO EXISTE  
 return -2;  
 } else {  
 //Devuelvo 0 que quiere decir que Los datos son correctos  
 return 0;  
 }  
 }  
}  
echo json\_encode(validarUsuario($\_REQUEST["identificador"], $\_REQUEST["email"]));  
?>

Este script realiza una consulta en la base de datos para verificar si el nickname (identificador) y correo electrónico existen y es de un mismo usuario, en caso de que el nickname no exista, se devolverá un -1 (error), en caso de que el correo electrónico no exista se devolverá un -2 (error) y en caso de que los datos sean correctos y coincidan se devolverá un cero.

## Nueva Contraseña (PHP)

<?php  
require\_once '../../\_Sesion.php';  
require\_once '../../\_Varios.php';  
  
entrarSiSesionIniciada("../../Menu.php");  
?>  
  
<!doctype html>  
<html lang="en">  
 <head>  
 <meta charset="UTF-8">  
 <title>Reestablecer Contraseña</title>  
 <link href='https://fonts.googleapis.com/css?family=Titillium+Web:400,300,600' rel='stylesheet' type='text/css'><link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/normalize/5.0.0/normalize.min.css">  
 <link rel="stylesheet" href="../../CSS/style.css">  
 </head>  
 <body>  
 <header id="cabecera">  
 <h1 style="text-align: center; font-size: xxx-large;">Fandom Game Library</h1>  
 </header>  
 <section class="bloqueInfoSesion">  
 <?php if (isset($\_REQUEST["sesionCerrada"])) {?>  
 <p style="color: blue">Se ha cerrado correctamente la sesión.</p>  
 <?php } if (isset($\_REQUEST["error"])) {?>  
 <p style='color:red;'>Error al introducir los datos<p>;  
 <?php } if (isset($\_REQUEST["errorCreacionCuenta"])) {?>  
 <p style='color:red;'>Se ha producido un error inesperado. Por favor, vuelve a introducir los datos<p>;  
 <?php }?>  
 </section>  
  
 <div class="form">  
 <div id="signup">  
 <form action="EnviarCorreoContrasenna.php" method="post">  
 <div class="top-row"></div>  
  
 <div class="field-wrap">  
 <label>  
 Nickname<span class="req">\*</span>  
 </label>  
 <input type="text" name="identificador" id="identificador" required autocomplete="off"/>  
 <br>  
 <p style="color: red" id="pErrorNickName"></p>  
 </div>  
  
 <div class="field-wrap">  
 <label>  
 Email<span class="req">\*</span>  
 </label>  
 <input type="email" name="email" id="email" required autocomplete="off"/>  
 <br>  
 <p style="color: red" id="pErrorEmail"></p>  
 </div>  
  
 <button type="submit" class="button button-block" id="btnReestablecerPassword"/>Reestablecer</button>  
  
 </form>  
 </div>  
 </div>  
 <!-- partial -->  
 <script src='https://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js'></script><script src="../../JS/script.js"></script>  
 <script src="../../JS/ValidacionFormularioPassword.js"></script>  
 </body>  
</html>

## Generar Nueva Contraseña (PHP)

<?php  
require\_once '../../\_Sesion.php';  
require\_once '../../\_Varios.php';  
//*TODO hay que hacer algun tipo de control para asegurar que no hay dos usuarios iguales (con el mismo nombre, email, etc)*$filasNuevas = crearNuevaContrasenna($\_REQUEST["identificador"], $\_REQUEST["email"],$\_REQUEST["contrasenna"]);  
  
?>  
  
<!doctype html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>Crear cuenta</title>  
 <link href='https://fonts.googleapis.com/css?family=Titillium+Web:400,300,600' rel='stylesheet' type='text/css'><link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/normalize/5.0.0/normalize.min.css">  
 <link rel="stylesheet" href="../../CSS/style.css">  
</head>  
<body>  
<?php if ($filasNuevas == 1) {?>  
 <section class="form">  
 <h2 style="color:green; font-size: xx-large">Contraseña cambiada con éxito.</h2>  
 </section>  
 <button><a href="SesionFormulario.php"></a></button>  
<?php } else {  
 redireccionar("SesionFormulario.php?errorCreacionCuenta");  
}?>  
</body>  
</html>

Este script notifica al Usuario si el proceso de actualización de la contraseña ha sido realizado de manera exitosa o no.

Primero, recibe como parámetros el nickname, correo electrónico y la contraseña nueva, llamando a un método externo para actualizar en la base de datos la contraseña nueva al usuario con el nickname y el correo electrónico enviado, el cual, si uno de los dos datos no es válido, se cancelará la actualización.

Si el método devuelve el número uno, la operación ha resultado exitosa, notificando con un mensaje al usuario, en caso contrario, se le notificará un error al usuario y se le dirigirá al formulario de inicio de sesión.

## Enviar Correo Contraseña (PHP)

<?php  
require\_once '../../\_Sesion.php';  
require\_once '../../\_Varios.php';  
  
if (isset($\_REQUEST["email"]) && isset($\_REQUEST["identificador"])){  
 $email = $\_REQUEST["email"];  
 $identificador = $\_REQUEST["identificador"];  
} else {  
 $email = "";  
 $identificador = "";  
}  
  
if (isset($\_REQUEST['cod'])){  
 redireccionar("CorreoContrasenna.php");  
} else {  
 $correoEnviado = enviarMensajeCorreoPassword($email,$identificador);  
}  
?>  
  
<!doctype html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>Reestablecer Contraseña</title>  
 <link href='https://fonts.googleapis.com/css?family=Titillium+Web:400,300,600' rel='stylesheet' type='text/css'><link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/normalize/5.0.0/normalize.min.css">  
 <link rel="stylesheet" href="../../CSS/style.css">  
</head>  
<body>  
 <section class="form">  
 <h2 style="color:green; font-size: xx-large">Un paso más cerca.</h2>  
 <br>  
 <?php if ($correoEnviado) {?>  
 <h4 style="color:green; font-size: medium">Se ha enviado un correo a la cuenta especificada</h4>  
  
 <input type="hidden" id="email" name="email" value="<?= $email ?>">  
 <input type="hidden" id="identificador" name="identificador" value="<?= $identificador ?>">  
 <form method="post">  
 <div class="field-wrap">  
 <label>  
 Código<span class="req">\*</span>  
 </label>  
 <input type="text" name="cod" id="cod" required autocomplete="off"/>  
 <br>  
 <p style="color: red" id="pErrorCodigo"></p>  
 </div>  
 <input type="submit" value="Enviar" id="btnEnviarCodigo" name="btnEnviarCodigo">  
 </form>  
 <?php } else {?>  
 <h4 style="color:red; font-size: medium">No se ha podido enviar un correo a la cuenta especificada</h4>  
 <br>  
 <button class="btnVolver">  
 <a href="SesionFormulario.php">Volver al Inicio de Sesión</a>  
 </button>  
 <?php } ?>  
 </section>  
<br><br>  
  
</body>  
<script src="../../JS/ValidacionCodigoRecuperacion.js"></script>  
<script src='https://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js'></script><script src="../../JS/script.js"></script>  
</html>

Este script recibe el nickname y correo electrónico del usuario para después enviar un código de verificación a dicho correo, este código será utilizado para validar que el usuario que ha solicitado el cambio de contraseña es el propietario de dicha cuenta.

Todos los usuarios, al crear una cuenta se les asigna un código de verificación aleatorio que se guarda en una tabla diferente en la base de datos.

Al recibir el correo de solicitud de cambio de contraseña, el nuevo código generado renueva al código antiguo de la base de datos, impidiendo así que el código de recuperación sea el mismo una vez cambiada la contraseña.

En todo caso, para que el correo sea enviado se comprobará que dicho correo existe, en el caso de que no exista, se le notificará al usuario de un error para el envío del correo.

Si dicho correo es enviado correctamente, se le notificará al usuario de que el correo se envió sin problemas y aparecerá automáticamente un formulario para la validación del código de verificación.

Mediante una llamada a la base de datos, se verificará que el código introducido en el formulario es correcto, en dicho caso, el usuario será dirigido a la página de actualización de la contraseña, en cambio, si el código es incorrecto, aparecerá un mensaje de error.

## Correo Contraseña (PHP)

<!doctype html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>Reestablecer Contraseña</title>  
 <link href='https://fonts.googleapis.com/css?family=Titillium+Web:400,300,600' rel='stylesheet' type='text/css'><link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/normalize/5.0.0/normalize.min.css">  
 <link rel="stylesheet" href="../../CSS/style.css">  
</head>  
<body>  
 <div id="msg"></div>  
  
 <!-- Mensajes de Verificación -->  
 <div id="error" class="alert alert-danger ocultar" role="alert" style="color: red">  
 Las Contraseñas no coinciden, vuelve a intentar !  
 </div>  
 <div id="ok" class="alert alert-success ocultar" role="alert" style="color: green">  
 Las Contraseñas coinciden! (Procesando formulario ...)  
 </div>  
 <div class="form">  
 <form action="GenerarNuevaContrasenna.php" method="post" onsubmit="verificarPasswords(); return false">  
 <div class="top-row"></div>  
  
 <div class="field-wrap">  
 <label>  
 Nickname<span class="req">\*</span>  
 </label>  
 <input type="text" name="identificador" id="identificador" required autocomplete="off"/>  
 <br>  
 <p style="color: red" id="pErrorNickName"></p>  
 </div>  
  
 <div class="field-wrap">  
 <label>  
 Email<span class="req">\*</span>  
 </label>  
 <input type="email" name="email" id="email" required autocomplete="off"/>  
 <br>  
 <p style="color: red" id="pErrorEmail"></p>  
 </div>  
 <div class="field-wrap">  
 <label>  
 Contraseña Nueva<span class="req">\*</span>  
 </label>  
 <input type="password" name="contrasenna" id="contrasenna"  
 pattern="(?=.\*\d)(?=.\*[a-z])(?=.\*[A-Z]).{8,}"  
 title="Introduce una contraseña con al menos un numero, una letra minúscula,  
 otra mayúscula y con una longitud mínima de 8 carácteres"  
 required autocomplete="off"/>  
 </div>  
 <div class="field-wrap">  
  
 <label>  
 Confirmar Contraseña<span class="req">\*</span>  
 </label>  
 <input type="password" name="contrasennaConfirmar" id="contrasennaConfirmar"  
 pattern="(?=.\*\d)(?=.\*[a-z])(?=.\*[A-Z]).{8,}"  
 title="Introduce una contraseña con al menos un numero, una letra minúscula,  
 otra mayúscula y con una longitud mínima de 8 carácteres"  
 required autocomplete="off"/>  
 </div>  
  
 <button type="submit" class="button button-block" id="btnReestablecerPassword"/>Reestablecer</button>  
  
 </form>  
 </div>  
 <script src='https://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js'></script><script src="../../JS/script.js"></script>  
 <script src="../../JS/ValidacionFormularioPassword.js"></script>  
 <script src="../../JS/ConfirmacionPassword.js"></script>  
</body>  
</html>

Este script sirve para la actualización de la contraseña del usuario, una vez introducido el código recibido correctamente, el usuario será enviado a esta página donde deberá introducir como doble verificación su nickname y su correo electrónico, a continuación, el usuario deberá introducir su contraseña nueva la cual deberá contener una letra mayúscula, una letra minúscula, un número y con una longitud mínima de ocho caracteres.

Para comprobar la existencia del nickname y correo electrónico del usuario se realiza una llamada AJAX la cual llama a un script PHP el cual obtiene de la base de datos el nickname y correo electrónico del usuario, si no existen, los campos se mostrarán en rojo con un texto para el usuario el cual contiene: “El nickname/correo electrónico no existe”.

En cambio, si los datos introducidos existen, los campos se mostrarán en verde con un texto el cual contiene: “Datos correctos”.

Para dicha comprobación de los datos, también se necesita otro script JS el cual muestra los mensajes de error.

También, se realiza una llamada independiente a otro script JS para la confirmación de la contraseña en los dos campos (contraseña y confirmar contraseña).

## Validación Formulario Contraseña (JS)

addEventListener("load", inicializar, false);  
  
function inicializar() {  
 inputNickName = document.getElementById("identificador");  
 inputNickName.addEventListener("focus", prepararEventoNickName, false);  
 inputEmail = document.getElementById("email");  
 inputEmail.addEventListener("focus", prepararEventoEmail, false);  
}  
function prepararEventoNickName() {  
 inputNickName.addEventListener("focusout", validarCampos,false);  
}  
function prepararEventoEmail() {  
 inputEmail.addEventListener("focusout", validarCampos,false);  
}  
function validarCampos() {  
 document.getElementById("pErrorNickName").innerHTML = "";  
 document.getElementById("pErrorEmail").innerHTML = "";  
 document.getElementById("btnReestablecerPassword").removeAttribute("disabled");  
 document.getElementById("btnReestablecerPassword").style.color = "white";  
 //console.log("Se salio del nickName o del email");  
 nickname = document.getElementById("identificador").value;  
 email = document.getElementById("email").value;  
 //console.log("Los parametros dados son --> nickname: "+ nickname + " y email: " + email);  
 llamadaAjax("ValidacionPassword.php",nickname, email,"",  
 function(texto) {  
 // debugger  
 var numOperacion = JSON.parse(texto);  
 // console.log("numOperacion es: " + numOperacion);  
 if(numOperacion == -1) {  
 document.getElementById("pErrorNickName").innerHTML = "El nickname introducido NO EXISTE, por favor, introduce otro";  
 document.getElementById("btnReestablecerPassword").setAttribute("disabled", "true");  
 document.getElementById("btnReestablecerPassword").style.color = "grey";  
 }else if (numOperacion == -2) {  
 document.getElementById("pErrorEmail").innerHTML = "El email introducido NO EXISTE, por favor, introduce otro";  
 document.getElementById("btnReestablecerPassword").setAttribute("disabled", "true");  
 document.getElementById("btnReestablecerPassword").style.color = "grey";  
 }},  
 function(texto) {  
 notificarUsuario("Error Ajax al cargar personas al inicializar: " + texto);  
 }  
 );  
}

function llamadaAjax(url, nickname, email, parametros, manejadorOK, manejadorError) {  
 var request = new XMLHttpRequest();  
  
 request.open("POST", url+"?identificador="+nickname + "&email="+email);  
 request.setRequestHeader("Content-type", "application/x-www-form-urlencoded");  
  
 request.onreadystatechange = function() {  
 if (this.readyState == 4) { // 4 equivale a "hemos terminado".  
 if (request.status == 200) { // 200 significa "todo bien".  
 manejadorOK(request.responseText);  
 } else {  
 if (manejadorError != null) manejadorError(request.responseText);  
 }  
 }  
 };  
 request.send(parametros);  
}

La funcionalidad de este script permite la validación del nickname y correo electrónico introducido por el usuario para su recuperación de la contraseña.

Para ello, al cargar todos los elementos de la página, se le añade a los campos un evento focus (permite realizar una opción cuando el usuario tiene el foco en dicho campo) el cual llamará a una función la cual añadirá un evento focusout (permite realizar una opción cuando el foco del usuario cambia de campo) el cual validará si lo datos introducidos en ese campo son correctos o no.

La función validarCampos desactivará (disabled = true) el botón para continuar si uno o todos los campos son incorrectos, en caso de que todos los datos sean correctos se le notificará mediante un mensaje al usuario de que puede continuar.

Para validar los datos, esta función realiza una llamada AJAX a un script PHP el cual recibe el nickname y correo electrónico y a sobre la base de datos ejecuta una consulta, comprobando si el nickname y correo electrónico existen.

La llamada AJAX devolverá un número según el resultado de la consulta, si el número devuelto es -1 quiere decir que el nickname no existe, si es un -2, el correo electrónico no existe y en caso de cualquier otro número, la consulta ha encontrado el nickname y correo electrónico de dicho usuario.

## Validación Usuario (JS)

addEventListener("load", inicializar, false);  
  
function inicializar() {  
 inputCodigo = ***document***.getElementById("cod");  
 inputCodigo.addEventListener("focus", prepararEventoCodigo, false);  
}  
  
function prepararEventoCodigo() {  
 inputCodigo.addEventListener("focusout", validarCampos, false);  
}  
  
function validarCampos() {  
 ***document***.getElementById("pErrorCodigo").innerHTML = "";  
 ***document***.getElementById("btnEnviarCodigo").removeAttribute("disabled");  
 ***document***.getElementById("btnEnviarCodigo").style.color = "white";  
 //console.log("Se salio del nickName o del email");  
 nickname = ***document***.getElementById("identificador").value;  
 ***email*** = ***document***.getElementById("email").value;  
 //console.log("Los parametros dados son --> nickname: "+ nickname + " y email: " + email);  
 llamadaAjax("ValidacionCodigo.php",nickname, email,"",  
 function(texto) {  
 // debugger  
 var txt = ***JSON***.parse(texto);  
 ***console***.log("texto es: " + txt);  
 if(txt !== ***document***.getElementById("cod").value) {  
 ***document***.getElementById("pErrorCodigo").innerHTML = "El Código introducido NO COINCIDE, por favor, introduce otro";  
 ***document***.getElementById("btnEnviarCodigo").setAttribute("disabled", "true");  
 ***document***.getElementById("btnEnviarCodigo").style.color = "grey";  
 }  
 },  
 function(texto) {  
 notificarUsuario("Error Ajax al cargar codigos al inicializar: " + texto);  
 }  
 );  
}

## Confirmación Contraseña (JS)

function llamadaAjax(url, nickname, email, parametros, manejadorOK, manejadorError) {  
 var request = new XMLHttpRequest();  
  
 request.open("POST", url+"?identificador="+nickname + "&email="+email);  
 request.setRequestHeader("Content-type", "application/x-www-form-urlencoded");  
  
 request.onreadystatechange = function() {  
 if (this.readyState == 4) { // 4 equivale a "hemos terminado".  
 if (request.status == 200) { // 200 significa "*todo bien".* manejadorOK(request.responseText);  
 } else {  
 if (manejadorError != null) manejadorError(request.responseText);  
 }  
 }  
 };  
 request.send(parametros);  
}

function verificarPasswords() {  
  
 // Obtenemos los valores de los campos de contraseñas  
 pass1 = ***document***.getElementById('contrasenna');  
 pass2 = ***document***.getElementById('contrasennaConfirmar');  
  
 // Verificamos si las constraseñas no coinciden  
 if (pass1.value != pass2.value) {  
  
 // Si las constraseñas no coinciden mostramos un mensaje  
 ***document***.getElementById("error").classList.add("mostrar");  
  
 return false;  
 } else {  
  
 // Si las contraseñas coinciden ocultamos el mensaje de error  
 ***document***.getElementById("error").classList.remove("mostrar");  
  
 // Mostramos un mensaje mencionando que las Contraseñas coinciden  
 ***document***.getElementById("ok").classList.remove("ocultar");  
  
 // Desabilitamos el botón de login  
 ***document***.getElementById("login").disabled = true;  
  
 // Refrescamos la página (Simulación de envío del formulario)  
 setTimeout(function() {  
 ***location***.reload();  
 }, 3000);  
  
 return true;  
 }  
  
}

La funcionalidad de este script comprueba que la nueva contraseña sea verificada e insertada correctamente.

Para ello se recogen los valores de los campos contrasenna y contrasenna Confirmar, si el valor es distinto entre los dos campos, aparecerá un mensaje al usuario de que las contraseñas no coinciden y deshabilitando el botón de cambio de contraseña, en caso de que coincidan, aparecerá un mensaje al usuario y el botón de cambio de contraseña se activará.

## Gestión Menú Principal (JS)

addEventListener("load", inicializar, false);  
  
var ***todosLosDatosCargados*** = false;  
  
var ***categorias*** = [];  
  
  
function llamadaAjax(url, parametros, manejadorOK, manejadorError) {  
 var request = new XMLHttpRequest();  
  
 request.open("POST", url);  
 request.setRequestHeader("Content-type", "application/x-www-form-urlencoded");  
  
 request.onreadystatechange = function() {  
 if (this.readyState == 4) { // 4 equivale a "hemos terminado".  
 if (request.status == 200) { // 200 significa "*todo bien".* manejadorOK(request.responseText);  
 } else {  
 if (manejadorError != null) manejadorError(request.responseText);  
 }  
 }  
 };  
 request.send(parametros);  
}  
  
function inicializar() {  
  
 //Obtengo todos los videojuegos de la base de datos  
 llamadaAjax("ObtenerJuegosBBDD.php", "",  
 function(texto) {  
 // debugger  
 var videojuegos = ***JSON***.parse(texto);  
 for (let i = 0; i < videojuegos.length; i++) {  
 // console.log(videojuegos[i]);  
 insertarVideojuego(videojuegos[i]);  
 addVideojuegoSelectFiltro("selectTipos", videojuegos[i]);  
 ***todosLosDatosCargados*** = true;  
 ***document***.getElementById("selectTipos").addEventListener("click", realizarFiltro, false);  
 }  
 },  
 function(texto) {  
 notificarUsuario("Error Ajax al cargar las tarjetas de los juegos: " + texto);  
 });  
  
}

PARTE 1: Este script permite crear las tarjetas personalizadas de los videojuegos, asignando automáticamente un estilo CSS de un script CSS ya creado.

Primero, al cargas todos los elementos de la página, se llamará a la función inicializar la cual realiza una llamada AJAX a un script PHP (ObtenerJuegosBBDD), la cual obtiene todos los datos sobre cada videojuego, cada vez que los datos de un videojuego es recogido, mediante un bucle se envían los datos de cada videojuego a una función llamada insertarVideojuego donde se crearán mediante JavaScript todos los elementos de la tarjeta y a la función addVideojuegoSelectFiltro la cual llenará un select con las categorias de los videojuegos para obtener un buscador de videojuegos por categorias.

Cuando todos los datos estén cargados, el select creado tendrá un evento click el cual permitirá mostrar los videojuegos de cada categoria al ser pulsada una categoria específica.

function obtenerTodasLasCategoriasAJAX(select, videojuegoActual) {  
 //obtengo todas las categorias de la base de datos  
 //y creo los options del select (uno por cada tipo DISTINTO de categoria que aparezca)  
 //*TODO esto creo que se puede hacer mejor porque al final estoy llamando mil veces a un metodo que necesito una vez pero bueno.* llamadaAjax("ObtenerCategoriasBBDD.php", "",  
 function(texto) {  
 // debugger  
 ***categorias*** = ***JSON***.parse(texto);  
 // console.log("categorias al inicializar tiene: " + categorias.length);  
 var optionsExistentes = select.options;  
 var existe = false;  
 for (let i = 0; i < optionsExistentes.length; i++) {  
 if (optionsExistentes[i].value == videojuegoActual.categoriaId) {  
 existe = true;  
 }  
 }  
 if (!existe) {  
 for (let i = 0; i < ***categorias***.length; i++) {  
 if (***categorias***[i].id == videojuegoActual.categoriaId) {  
 var opcion = new Option(***categorias***[i].categoria, videojuegoActual.categoriaId);  
 select.appendChild(opcion);  
 }  
 }  
 }  
 },  
 function(texto) {  
 notificarUsuario("Error Ajax al cargar las tarjetas de los juegos: " + texto);  
 });  
}

PARTE 2: La función obtenerTodasLasCategoriasAJAX realiza una llamada AJAX al script PHP ObtenerCategoriasBBDD el cual realiza una consulta en la base de datos y devuelve todas las categorías existentes, al recoger todos los datos se añade cada categoria en cada option del select creado mediante un bucle, creando un option por cada categoria y añadiéndolo al select con la función appendChild().

function realizarFiltro(e) {  
 eliminarTodosLosHijosDivDatos();  
 //Aqui obtengo el valor que elije el usuario para hacer el filtrado  
 var filtrarPor = e.target.value;  
 ***console***.log("filtrar por vale: " + filtrarPor);  
 if (filtrarPor == "Todos" && !***todosLosDatosCargados***) {  
 //si el usuario ha filtrado por "Todos" y no estan ya cargados todos los datos entonces muestro todos los datos  
 llamadaAjax("ObtenerJuegosBBDD.php", "",  
 function(texto) {  
 productosInicio = ***JSON***.parse(texto);  
 // debugger  
 for (var i=0; i<productosInicio.length; i++) {  
 insertarVideojuego(productosInicio[i]);  
 }  
 ***todosLosDatosCargados*** = true;  
 ***document***.getElementById("selectTipos").addEventListener("click", realizarFiltro, false);  
 },  
 function(texto) {  
 // alert(productosInicio);  
 notificarUsuario("Error Ajax al cargar al inicializar: " + texto);  
 }  
 );  
 } else {  
 ***console***.log("filtro de: " + filtrarPor)  
 //si el usuario a seleccionado un filtro determinado aplico la busqueda segun dicho filtro  
 llamadaAjax("ObtenerJuegosFiltrados.php?filtro="+filtrarPor, "",  
 function(texto) {  
 var videojuego = ***JSON***.parse(texto);  
  
 for (var i=0; i<videojuego.length; i++) {  
 insertarVideojuego(videojuego[i]);  
 addVideojuegoSelectFiltro("selectTipos",videojuego[i]); //-------------------  
 }  
 },  
 function(texto) {  
 // alert(productosInicio);  
 notificarUsuario("Error Ajax al cargar al inicializar: " + texto);  
 }  
 );  
 }  
}

PARTE 3: La función realizarFiltro realiza una llamada AJAX al script PHP ObtenerJuegosBBDD y otra llamada AJAX al script PHP ObtenerJuegosFiltradosBBDD con el valor de un option seleccionado para mostrar los videojuegos de una categoría en concreto.

Si el filtro de categorías tiene como valor por defecto realizará la primera llamada AJAX, en caso contrario realizará la llamada AJAX de filtrado.

function addVideojuegoSelectFiltro(nombreSelectHTMl, videojuegoActual) {  
  
 var select = ***document***.getElementById(nombreSelectHTMl);  
 obtenerTodasLasCategoriasAJAX(select, videojuegoActual);  
  
}  
  
/\* GESTION DEL DOM \*/  
  
function insertarVideojuego(videojuegoActual) {  
 var juego = tarjetaJuego();  
 var poster = tarjetaPoster();  
 var imagenEnlace = tarjetaImagenEnlace();  
 var imagen = tarjetaImagen(videojuegoActual);  
 var ticket = tarjetaTicket();  
 var ticketContenido = tarjetaTicketContenido();  
 var titulo = tarjetaTitulo(videojuegoActual);  
 var descripcion = tarjetaDescripccion(videojuegoActual);  
 var precioActual = tarjetaPrecioActual(videojuegoActual);  
 var precioViejo = tarjetaPrecioViejo(videojuegoActual);  
 var btnJuego = tarjetaBtnJuego();  
 var enlaceFicha = tarjetaEnlaceFicha(videojuegoActual);  
  
 insertarTarjeta(juego,poster,imagenEnlace,imagen,ticket,  
 ticketContenido,titulo,descripcion,precioActual,precioViejo,btnJuego,enlaceFicha);  
}  
  
function insertarTarjeta(juego,poster,imagenEnlace,imagen,ticket,  
 ticketContenido,titulo,descripcion,precioActual,precioViejo,btnJuego,enlaceFicha){  
 var contenedorVideojuegos = ***document***.getElementById("games-container");  
  
 contenedorVideojuegos.appendChild(juego);  
 juego.appendChild(poster);  
 juego.appendChild(ticket);  
 imagenEnlace.appendChild(imagen);  
 poster.appendChild(imagenEnlace);  
 ticket.appendChild(ticketContenido);  
 ticketContenido.appendChild(titulo);  
 ticketContenido.appendChild(descripcion);  
 ticketContenido.appendChild(precioActual);  
 ticketContenido.appendChild(precioViejo);  
 ticketContenido.appendChild(btnJuego);  
 btnJuego.appendChild(enlaceFicha);  
}

PARTE 4: La función addVideojuegoSelectFiltro recoge el select creado y llama a la función obtenerTodasLasCategoriasAJAX y le pasa como parámetro el videojuego obtenido y el select para crear en dicho select un option de esa categoría.

La siguiente función (insertarVideojuego) permite crear cada tarjeta por cada videojuego. Cada parte de la tarjeta es creada de forma independiente llamando a distintas funciones por cada parte, después es llamada una función (insertarTarjeta) la cual añade cada tarjeta a la página web.

Dicha función recibe cada parte de la tarjeta y esta es creada para ser insertada en un div llamado games-container de la página.

function tarjetaJuego(){  
 var juego = ***document***.createElement("div");  
 juego.setAttribute("class","main-container");  
 return juego;  
}  
function tarjetaPoster(){  
 var poster = ***document***.createElement("div");  
 poster.setAttribute("class","poster-container");  
 return poster;  
}  
  
function tarjetaImagenEnlace(){  
 var imagenEnlace = ***document***.createElement("a");  
 imagenEnlace.setAttribute("href","#");  
 return imagenEnlace;  
}  
  
function tarjetaImagen(videojuegoActual){  
 var imagen = ***document***.createElement("img");  
 imagen.setAttribute("id","img-" + videojuegoActual.id);  
 imagen.setAttribute("class","poster");  
 imagen.setAttribute("height","150px");  
 imagen.setAttribute("width","100px");  
 imagen.setAttribute("src","Imagenes/ImagenesCardsMenu/" +  
 videojuegoActual.id + ".png");  
 return imagen;  
}  
  
function tarjetaTicket(){  
 var ticket = ***document***.createElement("div");  
 ticket.setAttribute("class","ticket-container");  
 return ticket;  
}  
  
function tarjetaTicketContenido(){  
 var ticketContenido = ***document***.createElement("div");  
 ticketContenido.setAttribute("class","ticket-contenido");  
 return ticketContenido;  
}

function tarjetaTitulo(videojuegoActual){  
 var titulo = ***document***.createElement("h4");  
 titulo.setAttribute("class","ticket-game-titulo");  
 titulo.textContent = videojuegoActual.nombre;  
 titulo.style.color = "black";  
 return titulo;  
}

PARTE 5:

La función tarjetaJuego() crea un div principal en el que se recopilan todos los datos del videojuego.

La función tarjetaPoster() crea un div dentro del div tarjetaJuego donde se colocará el logo del videojuego.

La función tarjetaImagenEnlace() crea un enlace dentro del div tarjetaPoster para acceder a la ficha del videojuego (no se utiliza pero está implementando para un futuro).

La función tarjetaImagen() obtiene una imagen dentro de una carpeta específica del proyecto donde se encuentran los logos de los videojuegos, después, se da formato CSS y es colocada en el div tarjetaPoster.

La función tarjetaTicket() crea un div dentro del div tarjetaJuego donde se colocará toda la información sobre el videojuego.

La función tarjetaTicket Contenido() crea un div dentro del div ticket donde se colocará toda la información del videojuego (precios, título y descripción).

La función tarjetaTitulo() recoge el título del videojuego, crea una cabecera (h4) y es devuelta con un estilo CSS.

function tarjetaDescripccion(videojuegoActual){  
 var descripcion = ***document***.createElement("p");  
 descripcion.setAttribute("class","ticket-game-desc");  
 descripcion.textContent = videojuegoActual.descripcion;  
 return descripcion;  
}

function tarjetaPrecioActual(videojuegoActual){  
 var precioActual = ***document***.createElement("p");  
 precioActual.setAttribute("class","ticket-precio-actual");  
 if (videojuegoActual.precioActual == 0){  
 precioActual.textContent = "GRATIS";  
 } else {  
 precioActual.textContent = videojuegoActual.precioActual + "€";  
 }  
  
 return precioActual;  
}

function tarjetaPrecioViejo(videojuegoActual){  
 var precioViejo = ***document***.createElement("p");  
 precioViejo.setAttribute("class","ticket-precio-viejo");  
 precioViejo.textContent = videojuegoActual.precioViejo + "€";  
  
 return precioViejo;  
}  
  
function tarjetaBtnJuego(){  
 var btnJuego = ***document***.createElement("button");  
 btnJuego.setAttribute("class","ticket-btn-jugar");  
 return btnJuego;  
}  
  
function tarjetaEnlaceFicha(videojuegoActual){  
  
 var enlaceFicha = ***document***.createElement("a");  
 enlaceFicha.setAttribute("href", "../Proyecto/z\_FichasVideojuegos/"+videojuegoActual.id+".html");  
 enlaceFicha.textContent = "Ver ficha";  
 enlaceFicha.setAttribute("class","enlacePagina");  
 return enlaceFicha;  
}  
  
//Este metodo elimina todos los div que el metodo de obtener todos los productos crea  
function eliminarTodosLosHijosDivDatos() {  
 var divHijos = ***document***.getElementById("gamescontainer").children;  
 var numDivs = divHijos.length;  
 var cont = numDivs - 1;  
 while (divHijos.length > 0) {  
 divHijos[cont].remove();  
 cont--;  
 }  
 ***todosLosDatosCargados*** = false;  
}

PARTE 6:

La función tarjetaDescripcion() crea un parrafo dentro del div tarjetaTicketContenido con una pequeña descripcion sobre el videojuego.

La función tarjetaPrecioActual() crea un parrafo dentro del div tarjetaTicketContenido con el precio actual del videojuego.

La función tarjetaPrecioViejo()crea un parrafo dentro del div tarjetaTicketContenido con el precio viejo del videojuego (solo aparece si el juego tiene un nuevo precio por ofertas o descuentos).

La función tarjetaBtnJuego() crea un botón para el acceso a la ficha del videojuego en el div tarjetaContenido el cual recibe un estilo CSS.

La función tarjetaEnlaceFicha() crea un enlace dentro del botón el cual permite el acceso a la ficha del videojuego.

La función tarjetaEliminarTodosLosHijosDivDatos() elimina divs creados redundantemente dentro del div principal de la ficha del videojuego.

# 5| PRUEBAS

Tabla 3. Recopilación de pruebas llevadas a cabo

|  |  |  |  |
| --- | --- | --- | --- |
| **RF (Requisito Funcional)** | **Prueba realizada** | **Resultado Obtenido** | **Resultado aceptado** |
| **RF-01**  **(Inicio de sesión)** | Se inicia sesión en la plataforma web con un usuario de prueba. | Se accede al menú principal de la plataforma con el usuario de prueba con éxito. | **TRUE** |
| **RF-02**  **(Menú desplegable)** | Una vez en el menú principal de la plataforma web, se procede a desplegar el menú y a comprobar que los enlaces funcionan correctamente. | El menú se despliega correctamente y los enlaces funcionan como se espera. | **TRUE** |
| **RF-03**  **(Cerrar sesión)** | Se cierra la sesión del usuario de prueba. | La sesión se cierra con éxito y se redirige al usuario al inicio de sesión | **TRUE** |
| **RF-04**  **(Fichas de videojuegos)** | Se accede a la ficha de cada videojuego para comprobar que la disposición de los elementos y su contenido es el adecuado. | Las fichas presentan la disposición deseada y el contenido es el correcto. | **TRUE** |
| **RF-05**  **(Adquirir videojuegos)** | Se comprueba si los distintos usuarios pueden adquirir videojuegos y si el resultado de esta operación se almacena correctamente. | Cada usuario puede adquirir nuevos videojuegos. Los datos pertinentes son almacenados en la BBDD de forma correcta. Los usuarios pueden acceder a los videojuegos adquiridos mediante el apartado correspondiente. | **TRUE** |
| **RF-06**  **(Eliminar videojuegos)** | Se comprueba si los distintos usuarios pueden eliminar los videojuegos previamente adquiridos y si el resultado de esta operación se almacena correctamente. | Cada usuario puede eliminar los videojuegos adquiridos que considere oportunos. Los datos pertinentes son almacenados en la BBDD de forma correcta. Los usuarios pueden volver a adquirir el videojuego eliminado en el apartado correspondiente. | **TRUE** |
| **RF-07**  **(Búsqueda de videojuegos)** | Se comprueba si los métodos de búsqueda de videojuegos funcionan correctamente. | El filtrado por categoría funciona correctamente.  La búsqueda por nombre funciona correctamente. | **TRUE** |
| **RF-08**  **(Envío correo electrónico)** | Se comprueba que el envío de emails funciona correctamente creando un usuario de prueba. | El email introducido para el usuario de prueba recibe el correo notificando el acceso a la plataforma correctamente. | **TRUE** |
| **RF-09**  **(Creación nuevas cuentas)** | Se comprueba que se pueden crear nuevas cuentas respetando las condiciones de validación establecidas y que estas se integren de forma eficiente en el sistema creando una cuenta de usuario de prueba. | La cuenta se crea correctamente. La validación de datos reacciona de la manera esperada. La información se almacena en la BBDD como se estableció. | **TRUE** |
| **RF-10**  **(Ajustes de usuario)** | Se comprueba que los usuarios puedan modificar los datos de su cuenta. | Los datos modificados por el usuario se guardan y se actualizan correctamente | **TRUE** |
| **RF-11**  **(Recordar contraseña usuario)** | Se comprueba que la plataforma web permita modificar la contraseña y recuperar la cuenta a los usuarios que así lo quieran. | La contraseña se resetea y se actualiza con los nuevos datos de forma correcta. | **TRUE** |
| **RF-12**  **(Modo Claro/Oscuro)** | Se comprueba que los usuarios puedan cambiar la apariencia de la plataforma web de claro a oscuro y viceversa y que sea una operación individual para cada usuario. | El usuario puede cambiar el modo de claro a oscuro correctamente. La operación se realiza de manera individual para cada usuario. | **TRUE** |

# 6| TRABAJO FUTURO

Dentro del trabajo futuro en este proyecto pueden destacarse varias vertientes.

# 7| DESPLIEGUE

## 7.1 CONFIGURACIÓN Y DESPLIEGUE

# 8| CONCLUSIONES

Este capítulo se dedica a conclusiones finales y personales del proyecto, extraídas a lo largo de todo el proceso de desarrollo del mismo.

# 9| ANEXO

## 9.1 MANUAL DE USUARIO

# 10| BIBLIOGRAFÍA