

CFGX CICLO
Proyecto Final de Ciclo

SENET



Autor: mathieu chopplet

Tutor: Sergio Malo Molina

Fecha de entrega: 30/05/2020

Convocatoria: Semestre – 2020

Enlace hosting: <https://senet.es>

Tabla de contenido

1.- INTRODUCCIÓN	2
1.1- MOTIVACIÓN	2
1.3.- OBJETIVOS PROPUESTOS (GENERALES Y ESPECÍFICOS).	3
2.- METODOLOGÍA UTILIZADA.....	4
3.- TECNOLOGÍAS Y HERRAMIENTAS UTILIZADAS EN EL PROYECTO	5
3.1 Lenguajes front-end	5
3.1.1 HTML.....	5
3.1.2 CSS	5
3.1.3 JavaScript	6
3.1.4 Ajax	7
3.2 Lenguajes back-end	7
3.2.1 PHP	7
3.3 PhpMyadmin-SQL	8
3.4 Editor de código	8
3.4.1 Sublime Text 3	8
3.5 Framework.....	8
3.5.1 jQuery	8
3.5.2 Bootstrap	9
4.- ESTIMACIÓN DE RECURSOS Y PLANIFICACIÓN	10
5.- DESARROLLO DEL PROYECTO.	12
5.1.- INICIO	12
5.2.- ANÁLISIS	12
5.2.1- Modelo Entidad-Relación	12
5.2.2- Diagramas de Clases	13
5.2.3- Requisitos funcionales y no funcionales	14
5.2.3.1- Análisis de los requisitos funcionales:	15
1.1.- Diagramas de casos de uso - gestión de los usuarios:	15
1.2.- Diagramas de casos de uso - gestión de los productos:	15
5.2.3.2- Análisis de los requisitos no funcionales:	16
5.2.4 Concepción de la base de datos	17
5.3.- DISEÑO	18
5.3.1 MVC:	18
5.3.2 Explicación del código con sus funcionalidades	19
5.3.2.1 El index	19

5.3.2.2 El header	19
5.3.2.3 El footer	20
5.3.2.4 Crear una cuenta	21
5.3.2.5 Login del usuario.....	23
5.3.2.6 Modificar cuenta usuario	23
5.3.2.7 Vuestro juego de mesa	26
5.3.2.8 Proponer mis juegos	27
5.3.2.9 Busco juegos de mesa	27
5.3.2.10 Ver juego de mesa	36
5.3.2.11 Alquilar juego de mesa.....	37
5.3.2.12 Comprar juego de mesa	38
6.- DESPLIEGUE Y PRUEBAS	40
6.1.- PLAN DE PRUEBA.....	40
6.1.1 creación de un usuario.....	40
6.1.2 Login de un usuario	45
6.1.3 Registrar un juego de mesa.....	46
7.-CONCLUSIONES.....	49
7.1.- OBJETIVOS ALCANZADOS	49
7.2.- CONCLUSIONES DEL TRABAJO	49
7.3.- VÍAS FUTURAS.....	49
9.- BIBLIOGRAFÍA	51
10.- ANEXOS	51
10.1.- MANUAL DE INSTALACIÓN	51
10.2.- MANUAL DE USUARIO.....	51

1.- INTRODUCCIÓN.

Mi proyecto consiste en página web para la venta y el alquiler de juegos de mesa entre particulares. Se he desarrollado con Sublime Text 3, utilizando JavaScript y Php, una base de datos MySQL, y CSS.

1.1- MOTIVACIÓN

Lo que pretendía hacer en el proyecto, era una página web responsive que se pueda utilizar tanto en un móvil como en un portátil. Mi proyecto claramente no tiene todas las herramientas, restricciones o especificaciones de una página profesional pero mi objetivo era hacer algo utilizable y con funcionalidades.

Me he decidido por este tema porque creo que es algo que existe poco hoy en día en España, y podría ser más valorado. Más adelante presentaré datos recientes que muestran que el mercado de los juegos de mesa está creciendo años tras años. De hecho, encontré varias páginas webs similares a la que presento, pero en otros países.

Por otra parte, me interesaba desarrollar un proyecto que permita la interacción entre ciudadanos. Este tipo de herramienta, capaz de crear relaciones sociales entre vecinos me interesa particularmente. Al utilizar la página web, el usuario entra en un proceso que le permitirá no solamente pasarlo bien disfrutando de nuevos juegos, pero también si alquila o vende o alquila juegos ganar dinero. Le permitirá también crearse una red de jugadores dispuestos a compartir su afición.

1.2.- ABSTRACT.

SENET is a Web Page designed to rent or buy board games between private parties. I tried to create a responsive web page.

This Project is divided into seven parts. At first will see the main idea of the program, why I decided to create it and my motivations. Then, we will have a look at the methodology, technology and tools planned to be involved in the project.

Once the project is correctly described and planned, we will see his organization, and how the coding part has been performed.

At last, a series of test we allow us to see if the program has been correctly managed and if it reacts to users requests as expected. As a conclusion, we will analyses what goes wrong and what could be done to improve the program.

Here is a small description of the program himself:

First, the user access to the web Page SENET. He has the possibility to login/logout, see a description of the main object of the web page and look for board games close to him.

Once registered, all the activities of creation-modification or elimination of any data will be registered in the Database with a reference to the user Id.

The user can access to register his board games. In this layout, the user has the possibility to register new board games that he wants to sell or rent. If a user wants to rent o buy a board game, he has to be login. Then he will have access to a list of games, choose the ones he wants and confirm his order.

The tools involved to create the web Page are: Html5, CSS, Javascript, PHP, SQL, and the framework Bootstrap.

I choose this project among others because I like board games and it seems interesting to me to create a web page that allow people to interact with others. Though my project is very simple, I found it very interesting to understand how a simple web page could be done.

1.3.- OBJETIVOS PROPUESTOS (GENERALES Y ESPECÍFICOS).

De una forma general, pretendo crear una página web sencilla, que permita gestionar la compraventa -alquiler de juegos de mesa entre particulares. Con la posibilidad de subir fotos, nombres, descripción de los productos.

De una forma más específica, la aplicación debería localizar el usuario, y proponerle la visualización de otros usuarios en un rango determinado. Una vez localizado el usuario, podrá acceder a los juegos del usuario y visualizar los precios que propone.

Una vez que el usuario haya definido lo que desea, tendrá que hacer su pedido enviando un email automático al otro usuario.

2.- METODOLOGÍA UTILIZADA

En este apartado, definiremos la metodología utilizada para la definición de nuestro proyecto. La metodología se divide en dos partes. Por una parte, la preparación del proyecto, y por otra parte su evaluación. Primero se debe definir la información requerida, definimos lo que se llama el diseño lógico y físico. La parte de la evaluación consiste en definir atributos, asignarles un porcentaje, y calcular un índice de eficiencia.

- Preparación del proyecto:

El diseño lógico de nuestro proyecto nos permite definir los sistemas operativos, que flujos de información son relevantes, que procesamiento se requiere, el tipo de datos que se maneja.

El diseño físico, permite definir qué tipo de archivos se necesitan, los tipos de acceso a los archivos, los programas, los lenguajes o aplicaciones, la configuración hardware/software...

En nuestro caso, el sistema operativo utilizado para desarrollar la aplicación es Windows.

Podemos definir el flujo de información de un sistema distribuido como el conjunto de todas sus transferencias de información de acuerdo con un cierto análisis y en referencia a un cierto período de tiempo. El tipo de dato, son variables con sus atributos y datos almacenados en un base de datos.

Para el programa, se necesita la utilización del lenguaje de programación Javascript y Php, el editor de código multiplataforma Sublime, el manejo de datos con json.

- El ciclo de vida

Podemos definir el ciclo de vida del proyecto como el proceso de transformación de las ideas, a la parte de diseño, hasta llegar a su puesta en marcha.

La primera etapa a veces llamada de preinversion, consiste en definir la factibilidad técnica y económica del proyecto. Aquella etapa tiene al menos dos objetivos claramente definidos que son la asignación de los recursos, y la intención de medir, valorizar, identificar, las ratios costos-beneficios del proyecto. En nuestro caso, teniendo en cuenta el carácter particular del proyecto (no hay otros proyectos en competencia), la definición económica y de rentabilidad no fue decisiva a la hora de concretar una idea.

3.- TECNOLOGÍAS Y HERRAMIENTAS UTILIZADAS EN EL PROYECTO.

3.1 Lenguajes front-end

Los lenguajes del lado del cliente son aquellos asimilados directamente por el navegador y no necesitan pretratamiento. Los tres lenguajes principales del lado del cliente son: HTML, CSS y JavaScript.

Sirven para visualizar las páginas y presentar el contenido al usuario. Es importante que los navegadores sean capaces de decodificar un lenguaje para que la página se muestre en el formato adecuado, tal cual fue diseñada.

3.1.1 HTML

HTML (HyperText Markup Language o lenguaje de marcas de hipertexto) es un lenguaje que marca el texto de modo que el ordenador pueda manipularlo. Permite poner marcas como títulos, marcas de texto en negrita, subrayar, crear enlaces, etc.

El lenguaje HTML es el encargado de definir los contenidos de un sitio web de forma textual y estructurado. Sirve para estructurar textos y establecer enlaces con otros archivos. Es un lenguaje para la elaboración de páginas web.

El Consorcio World Wide Web (W3C), regula las recomendaciones y versiones normalizadas del lenguaje HTML, aceptado como norma ISO, la ISO 15445:2000.

HTML5 es la quinta y última versión de HTML. Sólo con HTML no se pueden crear estructuras complejas, por eso son necesarias las hojas de estilo, CSS. El lenguaje utilizado por la World Wide Web es el HTML. Los archivos pueden tener las extensiones htm y html. HTML organiza un documento en elementos lógicos, tales como: encabezado, párrafo, etc. Define las operaciones tipográficas y las funciones que debe ejecutar un programa visualizador sobre dichos elementos.

HTML permite la publicación de documentos en línea con títulos, textos, tablas, listas, fotos, etc. La recuperación de la información en línea mediante enlaces de hipertexto. Diseñar formularios para la realización de transacciones con servicios remotos, para su uso en la búsqueda de información. Insertar hojas de cálculo, videoclips y otras aplicaciones directamente en sus documentos.

3.1.2 CSS

CSS (Cascading Style Sheets u hojas de estilo en cascada) es el código que se aplica para dar un aspecto visual de una página web.

Fueron diseñadas por el W3C y puede decirse que son las instrucciones utilizadas para definir los tipos de letras, tamaños, colores, márgenes, ...

Este código está separado de la estructura, que es aportada por el lenguaje HTML y otro lenguaje. CSS es el complemento de HTML y gracias a este lenguaje se puede seleccionar un elemento de una página y definir un color, tamaño, tipo de letra, dimensiones, etc.

Al igual que HTML, CSS es un lenguaje de marcado no un lenguaje de programación.

Es el tipo de documento que utiliza un navegador web para redefinir las propiedades de los distintos elementos y las etiquetas en el código HTML, es decir, permite dar formato a los documentos de forma global. Además, proporciona al diseñador de páginas web definir un conjunto de ampliaciones HTML especiales y aplicarlas.

3.1.3 JavaScript

Fue creado por Brendan Eich en la empresa Netscape Communications. Primeramente, se creó con el nombre de LiveScript y con el objetivo de extender HTML y CSS para que los programadores pudieran evaluar las interacciones de los usuarios y presentar el contenido de forma dinámica. El nombre de JavaScript se eligió en base a la popularidad de Java, lenguaje al que se quería complementar.

Se utiliza especialmente para sitios webs que tienen animaciones, páginas dinámicas y es un lenguaje de programación que aporta dinamismo a una página HTML. También se podría utilizar como lenguaje del lado del servidor.

Es un lenguaje interpretado, no requiere compilación. JavaScript no está orientado a objetos. Actualmente, JavaScript no se utiliza sólo en navegadores, sino también en microcontroladores y en servidores. Gracias a tecnologías como AJAX es utilizado para enviar y recibir información del servidor. Presenta una escritura dinámica y no tiene clases. Las variables no necesitan ser introducidas antes de su uso y los tipos de variables se resuelven dinámicamente durante su ejecución.

Permite que las definiciones de funciones y otro tipo de código sean modificados mientras el programa se esté ejecutando. El modelo de ejecución de Java Script se basa en la interpretación del código fuente.

3.1.4 Ajax

Asynchronous JavaScript and XML (JavaScript y XML asíncronos) usa varias tecnologías web para que el comportamiento de las aplicaciones basadas en lotes sea más rápido.

Ajax evita las recargas constantes de la página, ya que se crea una capa intermedia entre el usuario y el servidor. Esta capa ayuda a que el usuario nunca se encuentre con una ventana del navegador vacía esperando la respuesta del servidor. Las peticiones HTTP al servidor se sustituyen por peticiones JavaScript, son intervenciones simples, realizadas de manera asíncrona que no requieren una respuesta del servidor.

Las tecnologías que forman AJAX son:

- XHTML y CSS, para crear una presentación basada en estándares.
- DOM, para la interacción y manipulación dinámica de la presentación.
- XML y JSON, para el intercambio y manipulación de la información.
- XMLHttpRequest, para el intercambio síncrono de información.
- Un lenguaje de servidor como PHP o ASP.NET, para procesar las peticiones en el servidor y enviar las respuestas.
- JavaScript, para unir todas las anteriores tecnologías.

3.2 Lenguajes back-end

3.2.1 PHP

PHP Hypertext Pre-processor, inicialmente se llamó Personal Home Page. Surgió en 1995, desarrollado por PHP Group. Los scripts de PHP son interpretados por el servidor.

Es un lenguaje que está enfocado a la creación de webs dinámicas, embebidas en páginas HTML y ejecutadas en el servidor. No necesita ser compilado para ejecutarse, pero requiere librerías de PHP, Apache o IIS y hereda la sintaxis de C, Java y Perl.

PHP es capaz de acceder a archivos, ejecutar comandos y abrir conexiones de red en el servidor. PHP procesa el código del lado del servidor evitando así, la interpretación por parte del navegador.

PHP5 se lanzó oficialmente en 2004 con el fin de perfeccionar los mecanismos de la programación orientada a objetos para solucionar las carencias de versiones anteriores. Incluye modificadores de control de acceso para implementar el encapsulamiento y el manejo de excepciones.

En PHP 5, no es necesario pasar objetos por referencia. Incluye funcionalidades parecidas a Java como: Constructores y destructores. Objeto de clonación. La clase de abstracción. Herencia. Esta última versión hace que PHP sea un lenguaje orientado a objetos muy potente y con un rendimiento mejorado

3.3 PhpMyadmin-SQL

Para la gestión y administración de las bases de datos, se utilizó el sistema de gestión de base de datos relacional PhpMyAdmin. Gracias a una interfaz gráfica intuitiva, permite crear, modificar o suprimir tablas, cuentas de usuarios entre otros. Para ello debe estar conectado a un servidor MySQL. En el caso del proyecto hemos utilizado Xampp. Integra el desarrollo software, Administración de base de datos, su diseño, creación y mantenimiento de base de datos.

Las principales características de PhpMyAdmin son :

Aporta seguridad y libertad a los usuarios. Puede ser utilizado, copiado, estudiado, y redistribuido. Es muy eficiente, y soporta una gran carga de operaciones. Consume poco recurso del equipo.

3.4 Editor de código

3.4.1 Sublime Text 3

Sublime text es un editor de código. Esta escrito en C++ y Python para los plugins. Permite ejecutar varias tareas a la vez, de forma muy rápida y sencilla. Se debe tener una licencia para su su uso continuo, aunque la versión de evaluación es plenamente funcional y no tiene fecha de caducidad. Dispone de un sistema de instalación de paquetes adicionales que amplían sus características de forma ilimitada.

3.5 Framework

3.5.1 jQuery

jQuery es un framework JavaScript libre y Open source, del lado del cliente. Se centra en la interacción entre el DOM, JavaScript, AJAX y HTML. Su objetivo es simplificar los comandos comunes de Java Script. JQuery fue diseñado por John Resig en el año 2006. Se centra en reunir un conjunto de elementos que gestiona el DOM.

3.5.2 Bootstrap

Bootstrap es un framework originalmente llamado Blueprint de Twitter, fue desarrollado por Mark Otto y Jacob Thornton de Twitter, como un marco de trabajo (*framework*) para fomentar la consistencia entre las herramientas internas. Bootstrap es modular y consiste esencialmente en una serie de hojas de estilo LESS que implementan la variedad de componentes de la herramienta. Una hoja de estilo llamada bootstrap.less incluye los componentes de las hojas de estilo. Los desarrolladores pueden adaptar el mismo archivo de Bootstrap, seleccionando los componentes que deseen usar en su proyecto. Es una biblioteca multiplataforma o conjunto de herramientas de código abierto para diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, así como extensiones de JavaScript adicionales. A diferencia de muchos frameworks web, solo se ocupa del desarrollo front-end.

4.- ESTIMACIÓN DE RECURSOS Y PLANIFICACIÓN

Ahora que hemos definido el propósito de nuestro proyecto, con el método y protocolo que seguiremos, podemos establecer una estimación de los recursos que serán necesarios para su realización. Se establece para la realización entera del proyecto un periodo de 2 meses de febrero a abril 2020. Este periodo engloba tanto la memoria como el programa en sí. Para la planificación, utilizamos un fichero Excel con las distintas tareas que supone la creación del proyecto y sus planificaciones temporales. La representación gráfica se hace a través de un diagrama de Gantt.

Proyecto SENET				
Nº	Actividad	Duración días	Fecha inicio	Fecha Fin
Definición del proyecto				
1	Definición del concepto del proyecto	4	25/02/2020	29/02/2020
2	Definición del nombre del proyecto	4	25/02/2020	29/02/2020
3	Análisis de las motivaciones para la realización del proyecto	4	25/02/2020	29/02/2020
4	Análisis de los conocimientos necesarios	3	01/03/2020	04/03/2020
5	Análisis de la factibilidad	3	01/03/2020	04/03/2020
6	Análisis de las herramientas necesarias	3	01/03/2020	04/03/2020
7	Análisis coste-tiempo realización	3	05/03/2020	08/03/2020
8	Planning realización proyecto	3	05/03/2020	08/03/2020
Desarrollo del proyecto				
9	búsqueda información de las "best-practice"	2	09/03/2020	11/03/2020
10	Formación-búsqueda de información	14	11/03/2020	25/03/2020
11	Creación Base de Datos	2	25/03/2020	27/03/2020
12	Desarrollo proyecto	45	01/03/2020	15/04/2020
13	Modificaciones código + BDD	19	27/03/2020	15/04/2020
Memoria del proyecto				
14	Introducción + abstract	2	16/03/2020	18/03/2020
15	redacción de la memoria	40	19/03/2020	28/04/2020
16	Modificaciones y correcciones	40	19/03/2020	28/04/2020
17	realización del video	4	24/04/2020	28/04/2020
18	Entrega final del proyecto	1	28/04/2020	28/04/2020

Fig 4.1 Planificación proyecto SENET

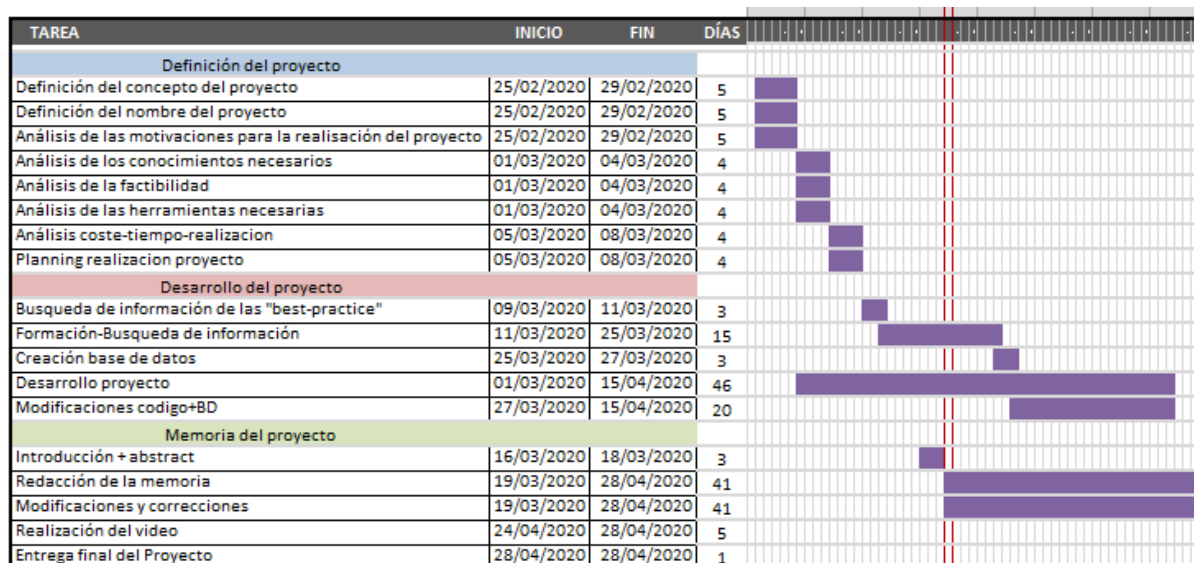


Fig 4.2 Diagrama de Gantt proyecto SENET

5.- DESARROLLO DEL PROYECTO.

En este apartado, vamos a introducir, analizar y detallar la implementación de nuestro proyecto. Este desarrollo lo repartiremos en 3 puntos a destacar.

5.1.- INICIO

Primero, hacemos una pequeña representación gráfica del funcionamiento del programa SENET:

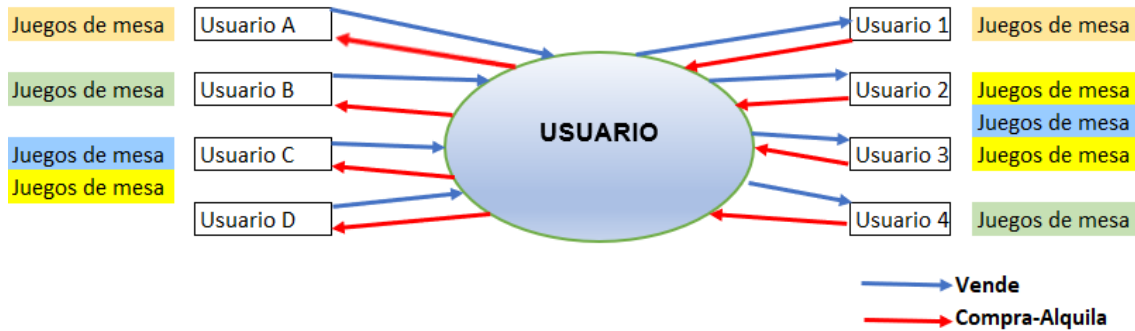


Fig 5.1.1 Funcionamiento del sistema

Como lo podemos ver, un usuario que utilizaría la página web SENET podría interactuar con otros usuarios. Podría, comprar o alquilar artículos al usuario A o B o C, y después vender o alquilar otros juegos de mesa a sus los usuarios 1, 2 o 3. El funcionamiento es por lo tanto bastante sencillo y fácil de entender.

5.2.- ANÁLISIS

El análisis del funcionamiento del programa se hace mediante diagramas de la estructura interna del proyecto. Para simplificar el análisis de nuestro programa, lo dividimos en distintas partes:

5.2.1- Modelo Entidad-Relación

La primera es el modelo Entidad-Relación de nuestro proyecto. Este diagrama es muy importante porque nos permite tener una representación gráfica de la futura Base de datos. Para ello, separamos las distintas entidades de nuestro programa y anotamos la relación que pueden tener entre ellos. Dicha relación, puede ser de varios tipos, como lo veremos más adelante.

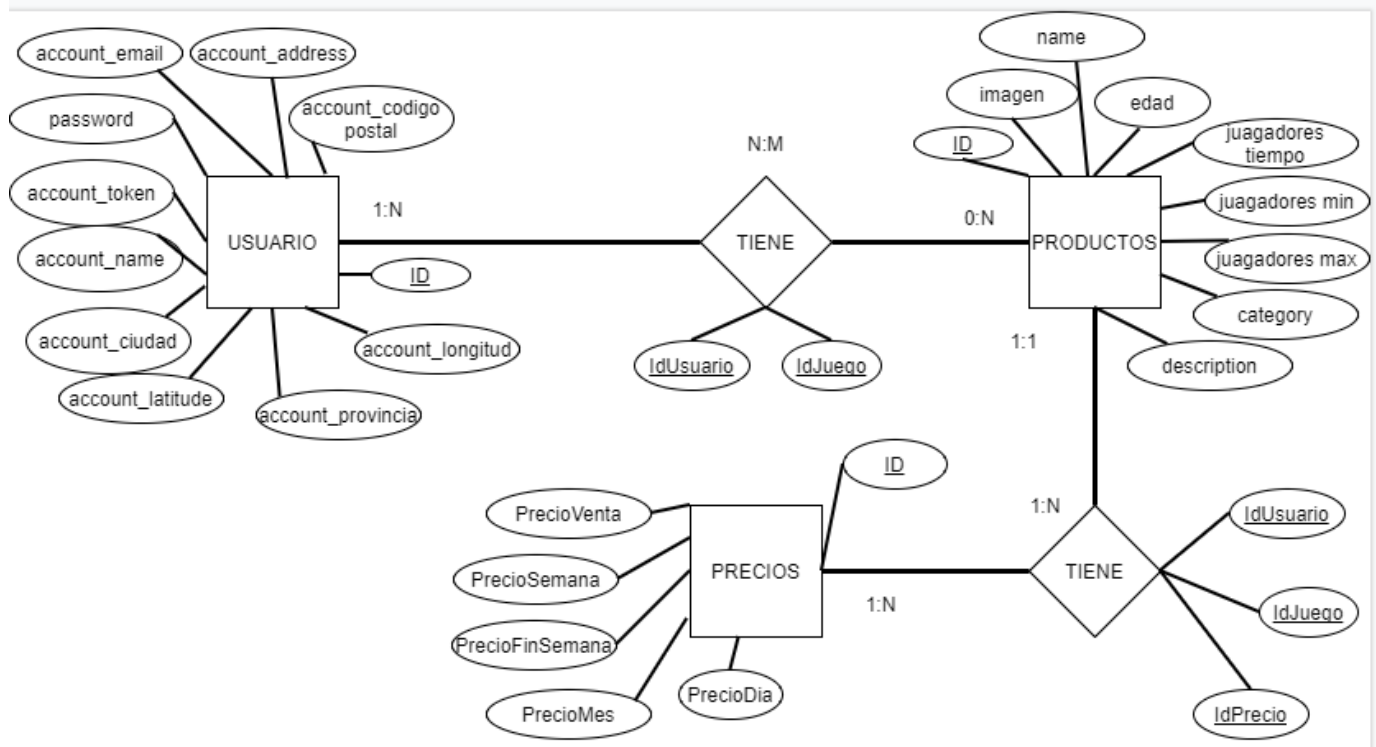


Fig 5.2.1 Modelo Entidad-Relación

5.2.2- Diagramas de Clases

El diagrama de clases explica la estructura estática de nuestro sistema en términos de clases y de la relación entre ellas. El interés de los diagramas de clases es de modelizar las entidades del sistema de información. Permite representar el conjunto de las informaciones de forma estructurada y visualmente fácil de entender.

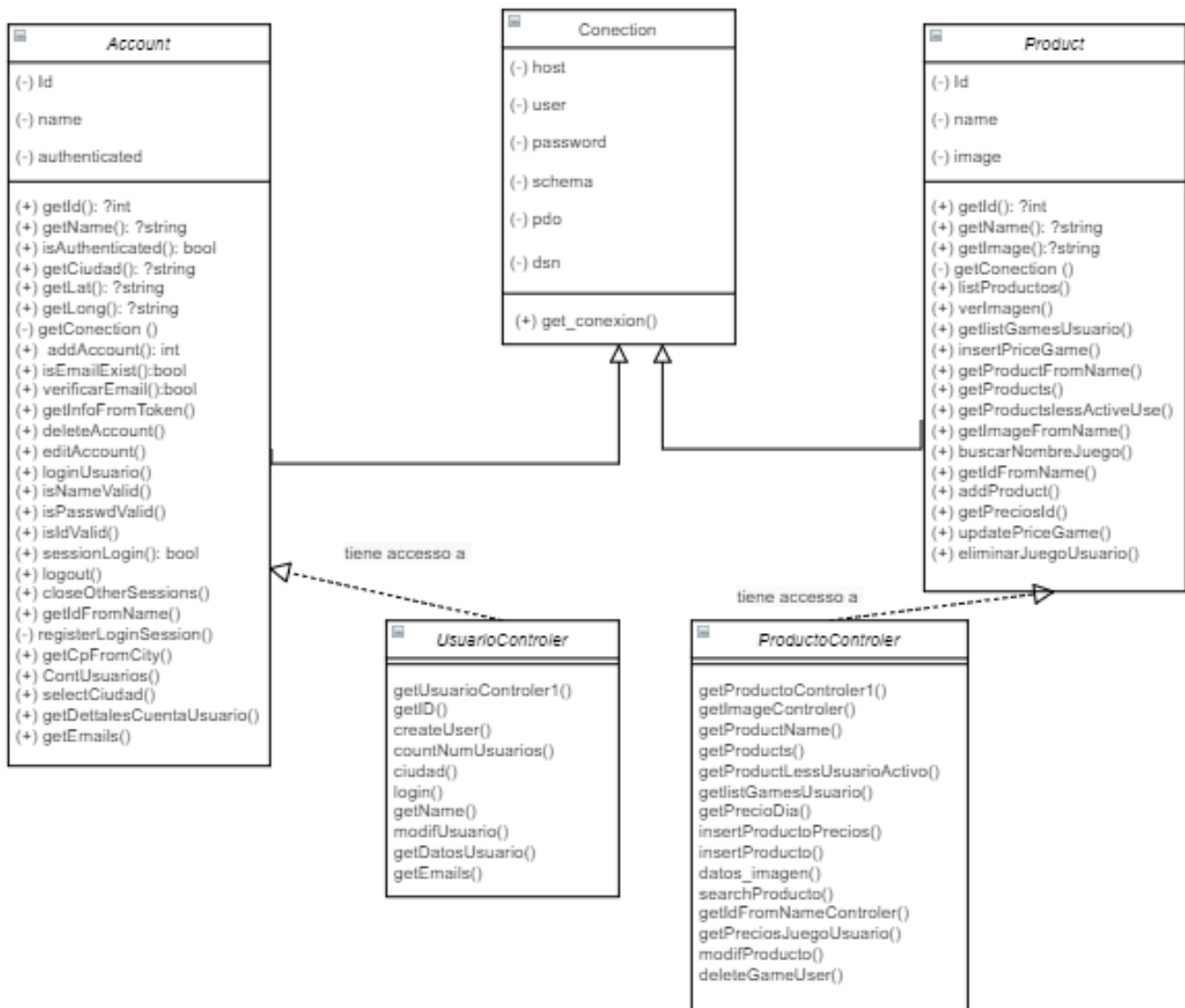


Fig 5.2.2 Diagrama de clases.

Este diagrama de clase nos permite ver que se instancia las clases “UsuarioControler” o “ProductoControler”. Las clases productos y usuarios utilizan métodos para visualizar, crear, modificar o eliminar datos de la BDD están relacionados con la clase Connection que crea la conexión con la base de datos.

5.2.3- Requisitos funcionales y no funcionales

Existen dos tipos de requisitos.

- Funcionales: Los requerimientos funcionales son las descripciones explícitas del comportamiento que debe tener una solución de software y que información debe manejar.

- No Funcionales: Son requisitos que imponen restricciones en el diseño o la implementación como restricciones en el diseño o estándares de calidad. Son propiedades o cualidades que el producto debe tener.

5.2.3.1- Análisis de los requisitos funcionales:

Nuestro programa está dividido en módulos. Para el análisis de los requisitos funcionales, utilizaremos diagramas de casos de uso para cada uno.

- Gestión de los usuarios
- Gestión de los productos

1.1.- Diagramas de casos de uso - gestión de los usuarios:

El programa debe permitir gestionar quien tiene acceso al programa. Un usuario, una persona física, puede registrarse, modificar su usuario y su contraseña. Además, el sistema debe ser capaz de controlar que dos usuarios no tienen las mismas claves de acceso. La contraseña debe ser encriptada, y la información de los usuarios correctamente guardada en la base de datos.

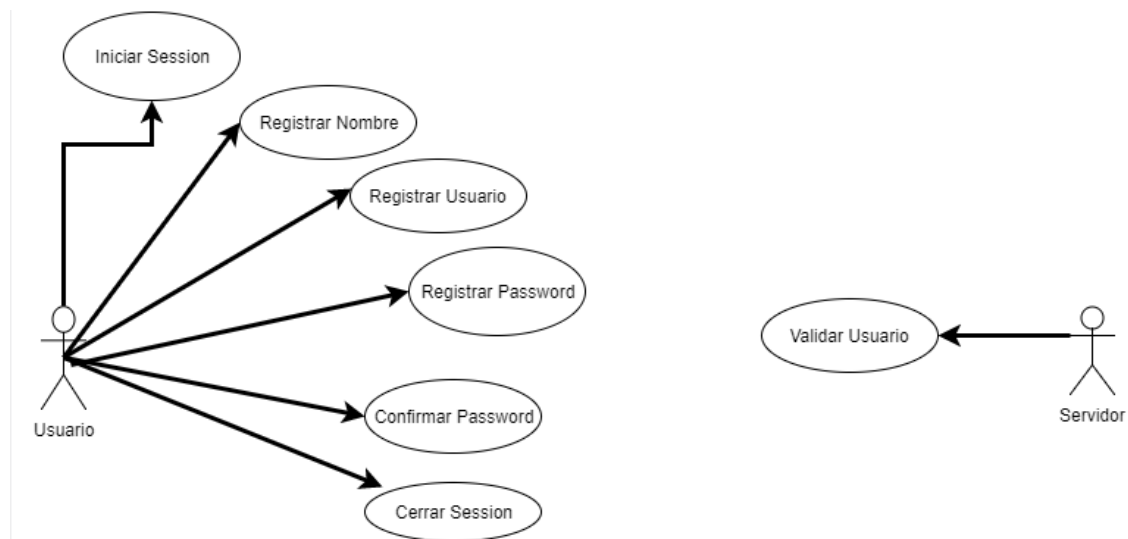


Fig 5.2.3 Diagrama de casos de uso Usuario

1.2.- Diagramas de casos de uso - gestión de los productos:

El programa permite la gestión de los productos. Para ello, nos permitirá visualizar los juegos de mesa o de todos los usuarios o de todos los usuarios menos el usuario activo. Se visualiza su categoría, sus precios de ventas y de alquiler. Por otra parte, nos permitirá modificar los precios de un juego, así como registrar uno nuevo. No puede eliminar, ni modificar la descripción de un juego existente en la Base de datos.

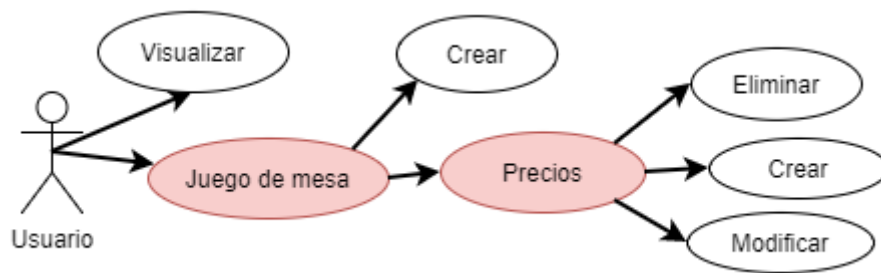


Fig 5.2.4 Diagrama de casos de uso productos

5.2.3.2- Análisis de los requisitos no funcionales:

Los requisitos no funcionales incluyen restricciones, y atributos de calidad.

Las restricciones son las características que impone el cliente. Son imprescindibles al programa, y no son negociables. Los atributos de calidad se dividen en dos, internos y externos.

Externamente, lo que prevalece es el desempeño, la usabilidad, la disponibilidad, la seguridad, y la confiabilidad.

Internamente (para el desarrollador), la calidad es sinónimo de modificabilidad, facilidad de prueba, reusabilidad, portabilidad, y facilidad de instalación.

El programa debe por lo tanto satisfacer a estos criterios de calidad.

Por ejemplo, si contemplamos el criterio de seguridad y de confiabilidad conviene destacar que el programa debe asegurarse que el usuario que manipula el programa está correctamente registrado, que no haya dos o varios usuarios registrados con el mismo usuario. Que la contraseña sea correctamente cifrada.

En términos de desempeño y de usabilidad, los layout del programa deben ser “user-friendly” intuitivos y atractivos. Una de las reglas fundamentales por el diseño del programa es que sea lo más sencillo posible, sin muchos colores (tonos pasteles) y que la información sea accesible rápidamente.

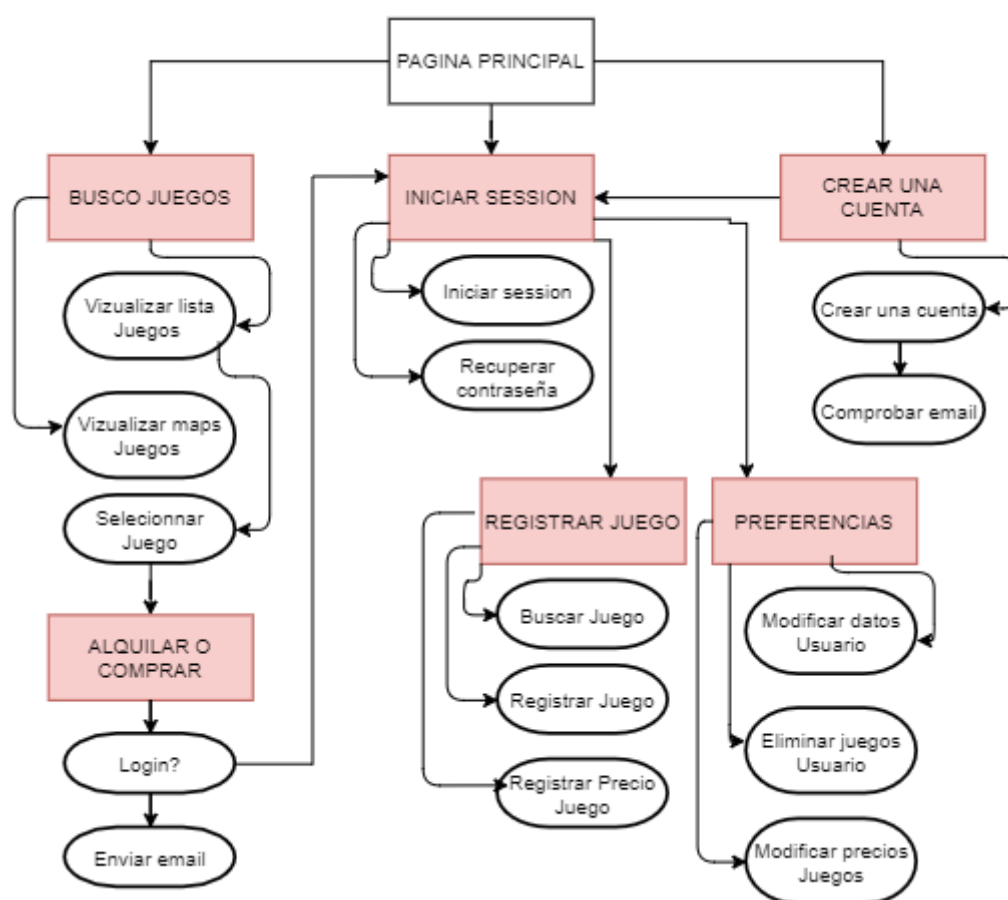


Fig 5.2.5 Diseño del Programa

5.2.4 Concepción de la base de datos

La base de datos será hecha con PhpMyAdmin. Debemos respetar unas reglas:

Una clase está representada por una tabla. Una asociación “uno a varios” implica la integración de la llave de la tabla relacionada a la clase que tiene la cardinalidad uno en la tabla que tiene la cardinalidad varios.

Una asociación varios a varios implica la creación de una nueva tabla teniendo como llave la concatenación de las dos tablas asociadas a las clases.

Siguiendo estas normas, la base de datos relacional es la siguiente:

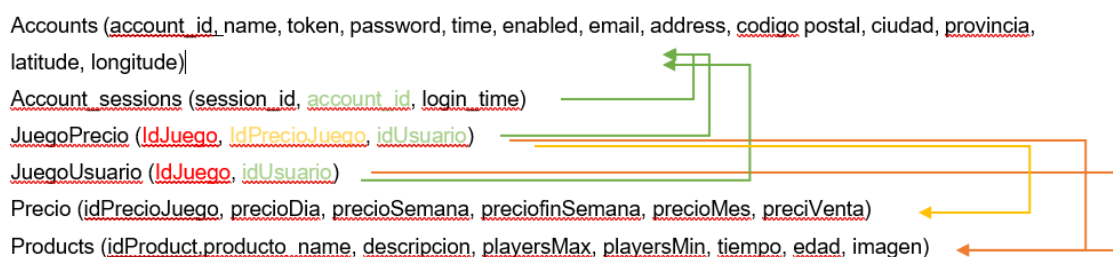


Fig 5.2.6 Modelo Relacional

5.3.- DISEÑO

Para describir el diseño y la arquitectura del programa, analizaremos las tres capas del modelo vista controlador. Para cada una de las capas, detallaremos como se estructura y su relación con las otras capas. Primero describiremos en que consiste, y luego como lo hemos utilizado en nuestro programa.

5.3.1 MVC:

El modelo vista-controlador (MVC) consiste en la separación en tres capas del programa.

La capa denominada modelo sirve para trabajar con los datos. Por tanto, concentra la parte del código necesaria para acceder a la información. Los datos por supuesto están almacenados en una base de datos. Por lo tanto, en esta capa es donde almacenaremos la parte del código necesaria a la conexión con la o las bases de datos, y donde haremos las operaciones que sea necesarias (como pueden ser select, update o insert) por ejemplo. En esta capa, se puede utilizar sentencias SQL directamente o un dialecto de acceso a datos basado en claves y objetos.

La capa Vista sirve para trabajar con el diseño del programa. Es en esta capa donde el desarrollador pondrá el código referente a las interfases con el usuario. La capa Vista no puede por motivos de seguridad interactuar con la capa Modelo gestionando los datos.

La capa Controlador como su nombre lo indica es la capa que nos permite relacionar la Vista con el Modelo. Contiene el código necesario para responder a las acciones del o de los usuarios.

MVC tiene por ventaja ser muy intuitiva y permitir diferenciar bien los componentes y actuar sobre ellos de una manera más precisa.

Un ejemplo utilizando MVC podría ser el siguiente:

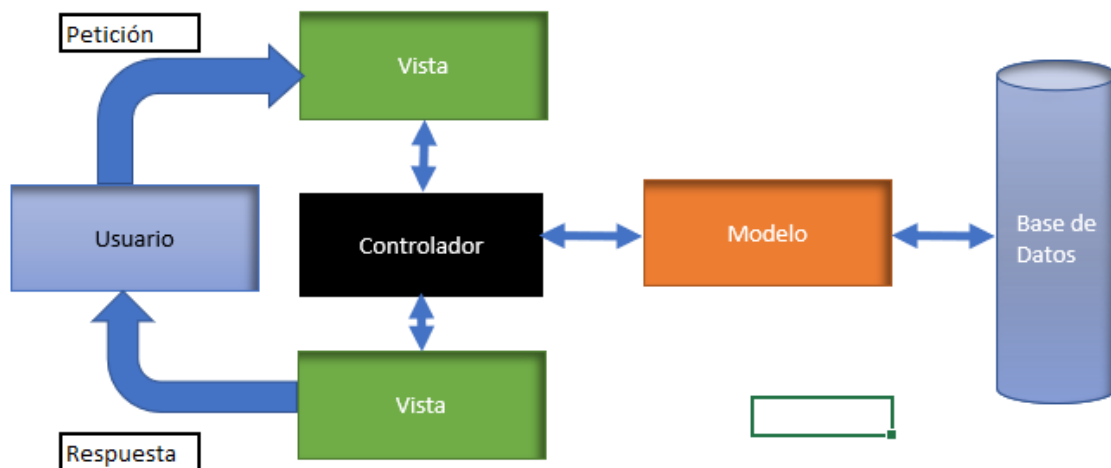


Fig 5.3.1 MVC

5.3.2 Explicación del código con sus funcionalidades

5.3.2.1 El index

Ahora podemos ver como el programa está estructurado. Primero el index.php es nuestra landing page. Esta compuesta por el header, el footer, y la página inicio.php.

```
1  <?php
2  session_start();
3      include_once 'View/includes/header.php';
4  ?>
5      <section class="home"> </section>
6  <?php
7      include_once 'inicio.php';
8      include_once 'View/includes/footer.html';
9  ?>
```

Fig 5.3.2.1 index.php

Según php.net, session_start() “crea una sesión o reanuda la actual basada en un identificador de sesión pasado mediante una petición GET o POST, o pasado mediante una cookie”.

5.3.2.2 El header

El header se va a repetir en todas las paginas de nuestra web. En la etiqueta <head>, incluimos las referencias a las hojas de estilo propias, las de Bootstrap, una hoja normalize.css descargada de github que nos permite ajustar el css en función de los navegadores, y una hoja font-awesome descargada de fontawesome.io que nos permite añadir objetos visuales a nuestras páginas. Por fin definimos un estilo propio a nuestra web, definido por una Api de Google.

Nuestro header es distinto dependiendo del inicio de una sesión o no. En caso afirmativo, se presenta la opción de registrar sus juegos, y el nombre del usuario activo.



En caso contrario se presenta las opciones de iniciar sesión o de crear una cuenta.



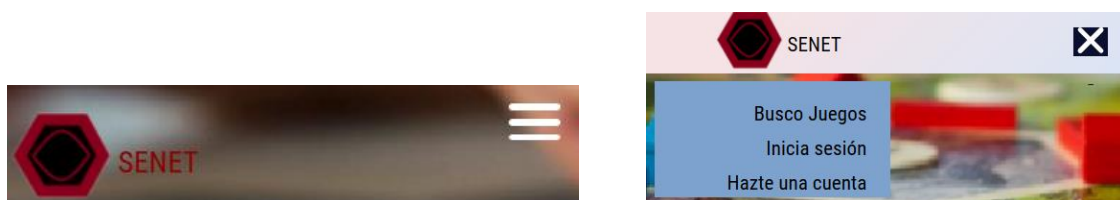
También el header incluye referencias a Javascript, paginas propias, a Bootstrap, y a jquery. Así el header, cambia su apariencia cuando el usuario hace scroll en la página.



Se ha añadido un dropdown al elemento usuario con opciones.



Por fin hay que mencionar que, si la pantalla del usuario es inferior a 768px, los elementos del header se contraen en barras.



5.3.2.3 El footer

Como el header se va a repetir en todas nuestras páginas.

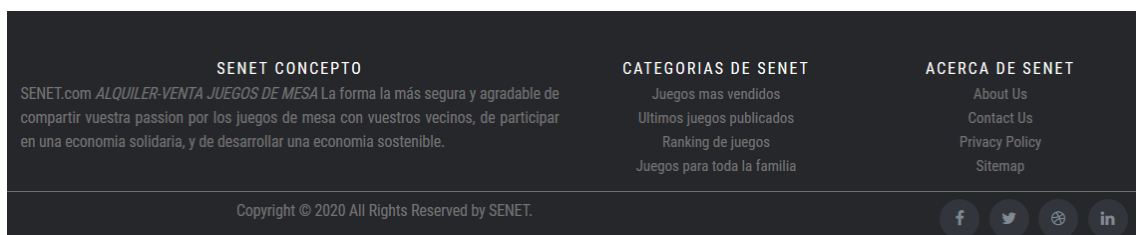


Fig 5.3.2.3 footer.php

Este compuesto por una breve descripción de la web, y links a más informaciones. Estos links no funcionan del momento.

5.3.2.4 Crear una cuenta

Fig 5.3.2.4 crearCuentaUsuario.php

Para la dirección utilizamos la Api de Google, `google.maps.places.Autocomplete()`. Para utilizarla, debemos declararla con nuestra clave:

```
<script
src="https://maps.googleapis.com/maps/api/js?v=3.exp&libraries=places&key=AlzaSy
AhS3viROB_tPYZ-MFSW1CHLpV9Mxxnyus"></script>
```

Desde html llamamos a la función javascript: `onKeyPress="javascript:validarAdresse()"`

```
2 function validarAdresse()
3 {
4     var searchInput = 'direccion';
5     $(document).ready(function () {
6         var autocomplete;
7         autocomplete = new google.maps.places.Autocomplete((document.getElementById(searchInput)), {
8             types: ['geocode'],
9         });
10    });
```

Lo que nos interesa es obtener la dirección, la ciudad, el código postal y las coordenadas del usuario para poder utilizarlas en el mapa después. Utilizamos el

proceso Geocoding de Google. Gracias a ello podemos convertir cualquier dirección en latitudes y longitudes.

La clase `google.maps.places.Autocomplete` toma como parámetro un elemento `html: HTMLInputElementAutocomplete`. `Types:['geocode']` modifica el tipo de respuesta obtenido de `Autocomplete`.

https://developers.google.com/places/web-service/autocomplete?csw=1#place_types

“`geocode` instructs the Place Autocomplete service to return only geocoding results, rather than business results. Generally, you use this request to disambiguate results where the location specified may be indeterminate.”

```
11 google.maps.event.addListener(autocomplete, 'place_changed', function ()
12 {
13     var place = autocomplete.getPlace();
14     console.log(place)
15
16     document.getElementById('latitude').value = place.geometry.location.lat();
17     document.getElementById('longitude').value = place.geometry.location.lng();
18
19     for(var count=0; count<place.address_components.length; count++){
20         switch(place.address_components[count].types[0]){
21             case 'route':
22                 document.getElementById('address').value = place.address_components[count].long_name;
23                 break;
24             case 'locality':
25                 document.getElementById('ciudad').value = place.address_components[count].long_name;
26                 break;
27             case 'postal_code':
28                 document.getElementById('cp').value = place.address_components[count].long_name;
29                 break;
30             case 'administrative_area_level_2':
31                 document.getElementById('provincia').value = place.address_components[count].long_name;
32                 break;
33         }
34     }
35 });
36 });
37 }
```

Luego añadimos un evento `addListener`. Declaramos una variable `place` utilizando la función `getPlace()` de Google, lo que nos permite tener un resultado con el id, el tipo y el nombre.

<https://developers.google.com/maps/documentation/javascript/places-autocomplete?hl=es>

Luego definimos los elementos `html` con los resultados de Google. Estos resultados se presentan como un `json` donde cada elemento está claramente definido.

<https://developers.google.com/maps/documentation/geocoding/intro?hl=es-419#geocoding>

Los elementos así definidos son transferidos a nuestro formulario:

```
<input type="hidden" name="longitude" id="longitude">
<input type="hidden" name="latitude" id="latitude">
<input type="hidden" name="addrese" id="addrese">
<input type="hidden" name="ciudad" id="ciudad">
<input type="hidden" name="cp" id="cp">
<input type="hidden" name="provincia" id="provincia">
```

5.3.2.5 Login del usuario



SENET

Inicia sesión

Nombre usuario

Contraseña

INICIAR SESSION

No me acuerdo de mi contraseña

[Darse de Alta](#)

[Volver a la pagina principal](#)

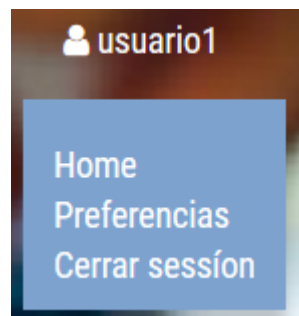
Fig 5.3.2.5 crearCuentaUsuario.php

La página para iniciar sesión es un formulario donde podemos introducir el nombre del usuario y su contraseña. El formulario esta en la vista, hace referencia a la página `comprueba_login_controlador.php` que instancia la clase `Account` del modelo y llama a la función `loginUsuario()` del modelo. Esta función toma por parámetros el nombre y la contraseña, y busca los resultados por el nombre introducido. En el que caso que haya un resultado, comprueba que la contraseña sea correcta con la función `password_verify` de php.

Si es correcto, iniciamos la sesión en la Base de datos, con la función `registerLoginSession()`. Esta función coge el id del usuario, con el tiempo (`now`), y asociamos `session_id()`. `Sesión_id()` se usa para obtener o establecer el id de sesión para la sesión actual. De esta forma, la próxima vez que el usuario se conecta, no tendrá que indicar sus credenciales.

5.3.2.6 Modificar cuenta usuario

En preferencias el usuario tiene la posibilidad de modificar sus atributos:



SENET

Hola, usuario1
Modificar su datos personales

Su nombre

Su Email

Dirección

Contraseña

Confirmar contraseña

[Cerrar sesión](#)

[Modificar usuario](#)

Para obtener los datos, hacemos una petición al Controlador, a la clase UsuarioControler.

```
<?php
$id = $_SESSION['id_user'];
$account = new UsuarioControler();
$res = $account->getDatosUsuario($id);

$name = $res['account_name'];
$passwd = $res['account_passwd'];
$email = $res['account_email'];
$address = $res['account_address'];
?>
```

Los modificamos en un formulario, y los obtenemos en modificar.php.

```
<form id="formCrearCuenta" class="form-horizontal" method="POST" action="Controlador/usuarios/modificar.php" >
submit="return validation(this)" >
```

```
// Recuperamos los datos del formulario modificar Usuarios
$cp = 0;
$nombre = $email = $direccion = $ciudad = $provincia = $password = "";
$id = $_SESSION['id_user'];
$nombre = $_POST['nameModif'];
$password = $_POST['passwordModif'];
$email = $_POST['emailModif'];
$direccion = $_POST['direccion'];
$ciudad = $_POST['ciudad'];
$cp = $_POST['cp'];
$provincia = $_POST['provincia'];
$lat = floatval($_POST['latitude']);
$long = floatval($_POST['longitude']);
$intEnabled=1;
if($cp=='cp' || $cp==''){
    $cp=00000;
}
```

Y hacemos la petición a la base de datos.

Al modificar el usuario controlamos que otro usuario ya no tiene el mismo nombre.

```
/* Check if an account having the same name already exists (except for this one). */
$idFromName = $this->getIdFromName($name);

if (!is_null($idFromName) && ($idFromName != $id))
{
    throw new Exception('User name already used');
}
```

El usuario tiene también la posibilidad de modificar o eliminar sus juegos. Solo modificar los precios que tiene registrado. No puede modificar el contenido del juego.

Modificar sus juegos

Juego	Precio día	Precio fin de semana	Precio semana	Precio mes	Precio venta		
CATAN	<input type="text" value="3 €"/>	<input type="text" value="6 €"/>	<input type="text" value="21 €"/>	<input type="text" value="90 €"/>	<input type="text" value="25.48 €"/>	<button>Modificar</button>	<button>Eliminar</button>
7_WONDERS_DUEL	<input type="text" value="0.99 €"/>	<input type="text" value="2 €"/>	<input type="text" value="6.93 €"/>	<input type="text" value="29.7 €"/>	<input type="text" value="30 €"/>	<button>Modificar</button>	<button>Eliminar</button>
TRIVIAL_PURSUIT	<input type="text" value="1 €"/>	<input type="text" value="2 €"/>	<input type="text" value="7 €"/>	<input type="text" value="30 €"/>	<input type="text" value="15 €"/>	<button>Modificar</button>	<button>Eliminar</button>

Contamos los juegos del usuario,

```
135 $resultados = $preciosProductos->getPreciosJuegoUsuario($id);
136 >
```

Y pintamos el formulario sobre estos resultados.

```
148 <table class="table table-striped table-sm ">
149     <thead >
150         <tr> <?php for($i=0; $i<$nbreLabels; $i++): ?>
151             <th> <?php echo $labels[$i] ?></th>
152         <?php endfor; ?>
153     </tr>
154 </thead>
155 <tbody>
156     <?php for($j=0; $j<count($resutados) ; $j++): ?>
157         <form action="" method="POST" >
158             <tr>
159                 <td><strong><?php echo $resutados[$j]['product_name'];?></strong></td>
160                 <td>
```

5.3.2.7 Vuestro juego de mesa

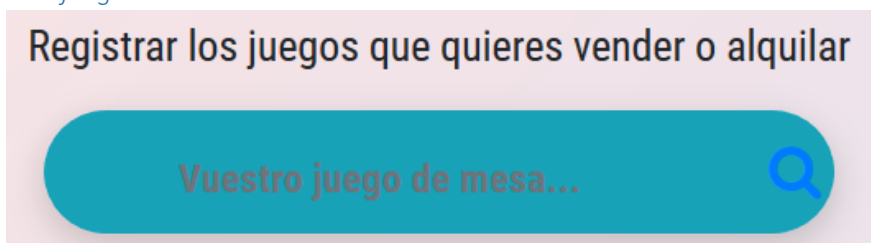


Fig 5.3.2.7 buscador de Juegos .php

En la parte para proponer los juegos de mesa, tenemos un botón que nos permite comprobar si el juego de mesa que vamos registrar ya existe en la Base de datos. Se hace mediante javascript y php. Es un formulario. Lo que introduce el usuario se guarda en una variable. En Javascript, recuperamos esta información en la función autocomplete, y con segundo parámetro el json de la lista de los juegos.

```
var js_array = <?php echo json_encode($lists);?>;
```

```
autocomplete('myInput', js_array);
```

Para obtener la lista de los juegos, instanciamos la clase ProductoControler, y llamamos a la función searchProducto(). La clase ProductoControler, instancia la clase Product, y llama a la función buscaNombreJuego() y devuelve un array.

```
public function buscarNombreJuego(string $term)
{
    $pdo = $this->getConnection();

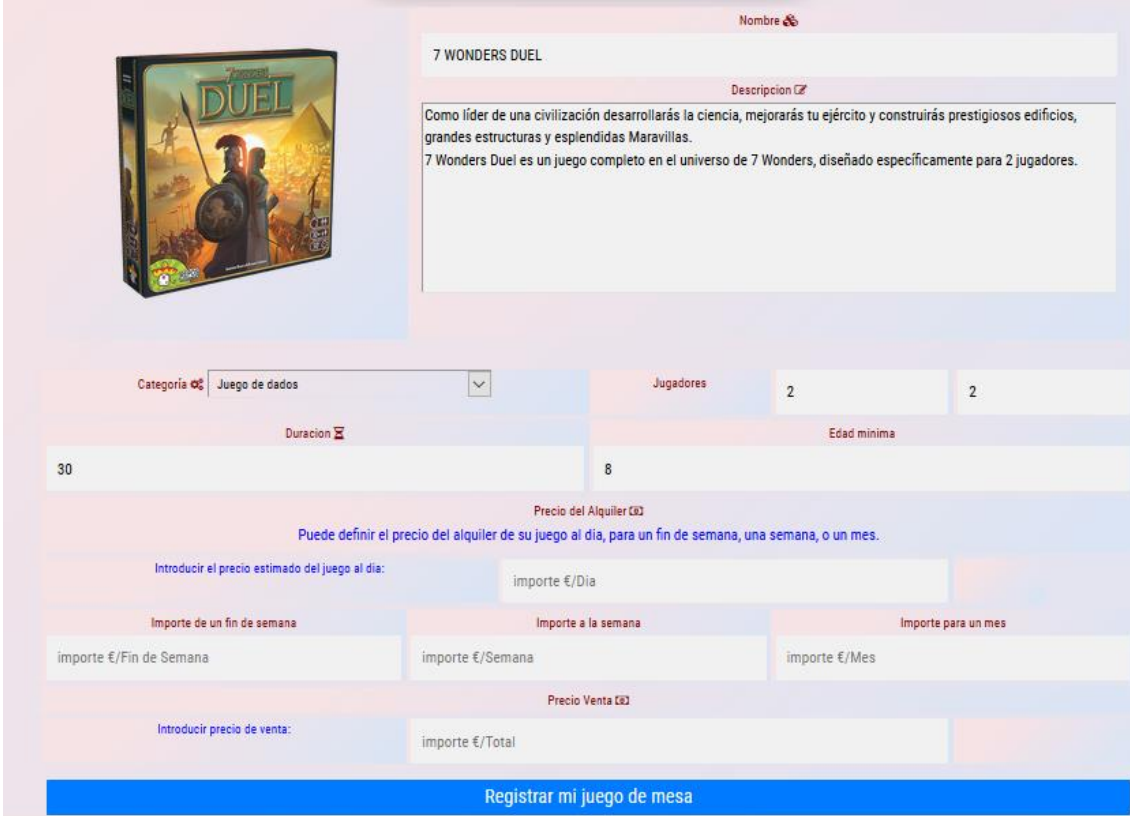
    $stmt = $pdo->prepare("SELECT * FROM products p WHERE p.product_name LIKE :search GROUP BY product_name");
    $stmt->execute([':search' => $term.'%']);
    $lists = array();
    while (false !== ($row = $stmt->fetch())) {
        $var = $row["product_name"];
        array_push($lists, $var);
    }
    return ($lists);
}
```


En Javascript, utilizamos la función del W3C autocomplete:

https://www.w3schools.com/howto/howto_js_autocomplete.asp


Comprobamos si el campo “myInPut” no es vacío, recuperamos los datos del juego que nos interesan, y rellenamos el formulario.

5.3.2.8 Proponer mis juegos




Nombre 


7 WONDERS DUEL

Descripcion 


Como líder de una civilización desarrollarás la ciencia, mejorarás tu ejército y construirás prestigiosos edificios, grandes estructuras y espléndidas Maravillas.
7 Wonders Duel es un juego completo en el universo de 7 Wonders, diseñado específicamente para 2 jugadores.

Categoría  Juego de dados

Jugadores 2 2

Duración  30

Edad mínima 8

Precio del Alquiler 

Puede definir el precio del alquiler de su juego al día, para un fin de semana, una semana, o un mes.


Introducir el precio estimado del juego al día:

importe €/Dia

Importe de un fin de semana importe €/Fin de Semana

Importe a la semana importe €/Semana

Importe para un mes importe €/Mes

Precio Venta 

Introducir precio de venta:

importe €/Total

Registrar mi juego de mesa

Fig 5.3.2.8 formulario de Juegos .php

El usuario puede registrar un juego de mesa. Varios parámetros se deben tener en cuenta. En la base de datos solo puede haber un registro por un nombre de juego de mesa, con sus características (nombre, descripción, categoría, numero de jugadores, duración de la partida, edad mínima del jugador). Si el juego existe, el usuario solo puede registrar los precios de alquiler y de venta. Si el juego no existe en la base de datos, el usuario podrá subir tanto la imagen, las características del juego y los precios. La explicación de las restricciones se hace en el apartado pruebas.

5.3.2.9 Busco juegos de mesa

Que el usuario este registrado o no, tiene la posibilidad de buscar juegos de mesa a la venta o en alquiler. La lista de los juegos esta dividida en dos partes. Por una parte, aparece la lista de los juegos. Si el usuario ha iniciado sesión, tendrá acceso a todos los juegos menos los que el ha registrado. Por otra parte la lista de los juegos se visualiza en un mapa de Google map según la dirección registrada cuando el usuario se da de alta.

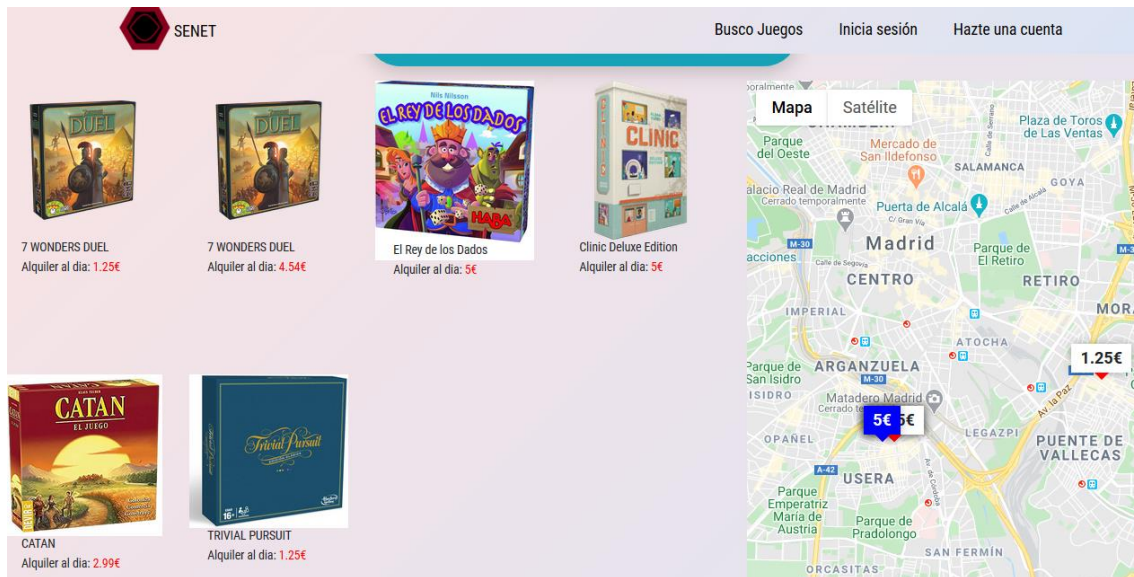


Fig 5.3.2.9 buscarJuegos.php

Vemos 2 veces el juego, porque varios usuarios pueden haber registrado el mismo juego con distintos precios. Si por el contrario, un usuario inicia sesión, sus juegos no aparecen en la lista:

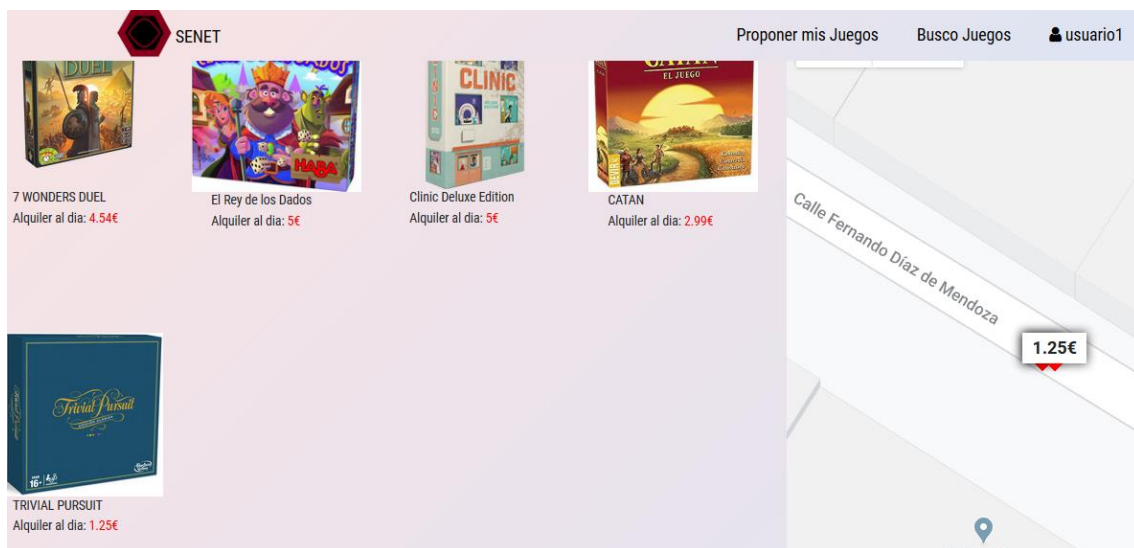


Fig 5.3.2.9.2 buscarJuegos.php

Para mostrar la lista de los juegos a la izquierda, contamos el numero de juegos total e iteramos en un div los elementos que queremos que aparezcan en la clase "item"

```
92 <div class="item js-marker" id="items"  
93 onclick="location.href='<?=$url onclick ?>'"  
94 data-lat="<?=$juegos[$i]['account_lat'] ?>"  
95 data-lng="<?=$juegos[$i]['account_lon'] ?>"  
96 data-price="<?php echo $juegos[$i]['product_precioDia'] ?>"  
97   
98 <h3><?php echo $juegos[$i]['product_name'] ?> </h3>  
99 <?php  
100 if($juegos[$i]['product_precioDia']==0 || $juegos[$i]['product_precioDia']=='')  
101 {  
102     $juegos[$i]['product_precioDia']=0;  
103 }  
104 >>  
105 <?php echo "Alquiler al dia: ".<span class='precioSpan'> . $juegos[$i]['product_precioDia']. '€'. "</span>";?></p>  
106 </div>
```

Para el mapa de Google, se crea un div:

```
<div class="map" id="map"></div>
```

y dos ficheros javascript: vendor.js y map .js.

Vendor.js nos permite trabajar con el mapa de google de forma asincrona y no bloquear otros recursos como el css o las imagines. Permite cargar un javascript y lanzar una funcion una vez que esta cargado.

En map.js, definimos la variable \$map, si es distinta de null, llamamos a la funcion initMap().

```
1 //selecciona id map  
2 let $map=document.querySelector('#map')
```

La funcion initMap(), instancia la clase GoogleMap.

La clase GoogleMap tiene un metodo load que nos permite cargar el mapa. Tiene por parametro (element) que nos permite definir en que elemento cargar este mapa. Nuestro elemento es \$map. El script que cargamos viene de google map:

<https://developers.google.com/maps/documentation/javascript/tutorial?hl=es>

No utilizamos el callBack de google &callback=initMap, porque utilizamos script.js.

```
$script("https://maps.googleapis.com/maps/api/js?key=AIzaSyAhS3viROB_tPYZ-MFSW1CHLpV9Mxxnyus",() => {
```

La funcion load es asincrona. Utilizamos el objeto Promise.

```
async load(element){  
    //El objeto Promise (Promesa) es usado para computaciones asincronas.  
    //Una promesa representa un valor que puede estar disponible ahora, en el futuro, o nunca.  
    return new Promise((resolve,reject)=>{  
        // para que haga referencia al constructor
```

Si todo se ha cargado correctamente, utilizamos la funcion resolve().

Luego debemos seleccionar los elementos que queremos mostrar en el mapa.

```
//recojemos los elementos con la class item  
Array.from(document.querySelectorAll('.item')).forEach(function(item)
```


Para utilizar Array.from en todos los navegadores utilizamos polyfill, copiando el código de la página:

https://developer.mozilla.org/es/docs/Web/JavaScript/Referencia/Objetos_globales/Array/from

Para cada elemento debemos añadir un marker. Creamos un método addMarker que tiene como parámetros la latitud, la longitud, y el texto.

```
addMarker(lat, lng, text){
  //parametros de longitud y latitud
  //A LatLng is a point in geographical coordinates: latitude and longitude.
  let point = new google.maps.LatLng(lat, lng)
  //parametros : la posicion/el mapa/el texto
  let marker = new this.textMarker(point, this.map, text)
  //añado en los callback un nuevo callback
  marker.onActivation.push(()=>{
    this.map.setCenter(marker.pos)
  })
  //cada vez que se añade un marker, hacemos un extend del punto
  //para crear una clase hija de otra.
  this.bounds.extend(point)
  //tenemos que devolver marker para utilizarlo en la class activ
  return marker;
}
```

```
let marker= map.addMarker(item.dataset.lat,item.dataset.lng, item.dataset.price+'€')
```

La propiedad dataset en HTMLElement proporciona una interfaz lectura/escritura para obtener todos los atributos de datos personalizados (data-*) de cada uno de los elementos.

Definimos un método para centrar el mapa:

```
//metodo para centrar el mapa
centerMap(){
  this.map.panToBounds(this.bounds)
  this.map.fitBounds(this.bounds)
}
```

Para modificar los markers de Google.

Para ello utilizamos la opción de Google de crear un Div y en esta div de crear una imagen:

<https://developers.google.com/maps/documentation/javascript/examples/overlay-hideshow>


```
/**
 * onAdd is called when the map's panes are ready and the overlay has been
 * added to the map.
 */
USGSOverlay.prototype.onAdd = function() {

    var div = document.createElement('div');
    div.style.border = 'none';
    div.style.borderWidth = '0px';
    div.style.position = 'absolute';

    // Create the img element and attach it to the div.
    var img = document.createElement('img');
    img.src = this.image_;
    img.style.width = '100%';
    img.style.height = '100%';
    div.appendChild(img);

    this.div_ = div;
```

Lo que debemos utilizar aquí es `OverlayView()` .
`USGSOverlay.prototype = new google.maps.OverlayView();`

“The Maps JavaScript API provides an [OverlayView](#) class for creating your own custom overlays. The `OverlayView` is a base class that provides several methods you must implement when creating your overlays. The class also provides a few methods that make it possible to translate between screen coordinates and locations on the map.”

```
/** @constructor */
function USGSOverlay(bounds, image, map) {

    // Initialize all properties.
    this.bounds_ = bounds;
    this.image_ = image;
    this.map_ = map;

    // Define a property to hold the image's div. We'll
    // actually create this div upon receipt of the onAdd()
    // method so we'll leave it null for now.
    this.div_ = null;

    // Explicitly call setMap on this overlay.
    this.setMap(map);
}
```

Declaramos la clase `TextMarker`. Esta clase tiene un constructor con tres parámetros: la posición, el mapa, y el texto.

```
//declaramos la class para modificar los marker de google:  
//objeto hereda de google.maps.OverlayView  
//The Maps JavaScript API provides an OverlayView class for creating your own custom overlays  
//https://developers.google.com/maps/documentation/javascript/customoverlays  
this.textMarker=class TextMarker extends google.maps.OverlayView {
```

El primer elemento por definir, el setMap:

```
this.setMap(map)
```

Después debemos crear el método onAdd():

```
/**  
 * onAdd is called when the map's panes are ready and the overlay has been  
 * added to the map.  
 */  
USGSOverlay.prototype.onAdd = function() {  
  
    var div = document.createElement('div');  
    div.style.borderStyle = 'none';  
    div.style.borderWidth = '0px';  
    div.style.position = 'absolute';  
  
    // Create the img element and attach it to the div.  
    var img = document.createElement('img');  
    img.src = this.image_;  
    img.style.width = '100%';  
    img.style.height = '100%';  
    img.style.position = 'absolute';  
    div.appendChild(img);  
  
    this.div_ = div;  
  
    // Add the element to the "overlayLayer" pane.  
    var panes = this.getPanes();  
    panes.overlayLayer.appendChild(div);  
};
```

Definimos la div, la posición absolute, y el texto que mostramos:

```
this.div = document.createElement('div')
```

```
this.div.style.position = 'absolute'
```

```
this.div.innerHTML = this.text
```

damos un class al marker para el css

```
this.div.classList.add('marker')
```

Despues debemos utilizar los panes.

```
// Add the element to the "overlayLayer" pane.  
var panes = this.getPanes();  
panes.overlayLayer.appendChild(div);  
};
```

this.getPanes().overlayImage.appendChild(this.div)

El marker en CSS es :

```
362  /***** estilo markers google map *****/  
363  .marker{  
364      background-color: #fff; /*white*/  
365      transform: translate(-50%,calc(-100%-10px));  
366      white-space: nowrap;  
367      font-size: 18px;  
368      padding: 5px 10px;  
369      font-weight: bold;  
370      text-align: center;  
371      box-shadow: 0 0 10px rgba(0,0,0, 0.3);  
372  }  
373  
374
```

Transform nos permite posicionarlo en el mapa: -50% a la izquierda y -100 % más alto.

White-space nowrap para que el texto no cambie de línea.

El método draw() nos permite que el elemento marker no se mueva en el mapa cuando el usuario lo desplaza. Para ello utilizamos lo que se llama projection.

```
// Retrieve the south-west and north-east coordinates of this overlay  
// in LatLngs and convert them to pixel coordinates.  
// We'll use these coordinates to resize the div.  
var sw = overlayProjection.fromLatLngToDivPixel(this.bounds_.getSouthWest());  
var ne = overlayProjection.fromLatLngToDivPixel(this.bounds_.getNorthEast());  
  
// Resize the image's div to fit the indicated dimensions.  
var div = this.div_;  
div.style.left = sw.x + 'px';  
div.style.top = ne.y + 'px';  
div.style.width = (ne.x - sw.x) + 'px';  
div.style.height = (sw.y - ne.y) + 'px';  
.
```

```
draw () {  
  //el elemnto marker se queda en su posicion en el mapa  
  //Computes the pixel coordinates of the given geographical location in the DOM element that holds the draggable map.  
  //https://developers.google.com/maps/documentation/javascript/reference/overlay-view  
  let position = this.getProjection().fromLatLngToDivPixel(this.pos)  
  this.div.style.left = position.x + "px"  
  this.div.style.top = position.y + "px"  
}
```

A mi elemento div se le aplica un estilo a la izquierda x y de altura y.

El método onRemove():

```
// The onRemove() method will be called automatically from the API if
// we ever set the overlay's map property to 'null'.
USGSOverlay.prototype.onRemove = function() {
  this.div_.parentNode.removeChild(this.div_);
  this.div_ = null;
};
```

Hacemos lo mismo.

```
onRemove () {
  this.div.parentNode.removeChild(this.div)
}
```

Para el infoWindow en el mapa:

En el constructor:

```
//necesitamos saber cuando el elemento html(al hacer click) es activado para ello añadimos un callBack
this.onActivation=[]
```

```
//les llamamos uno tras otro
this.onActivation.forEach ( function(cb)
{
  cb()
});
}
```

En el método On Add():

```
//al hacer click en mi elemento se añade el contenido de mi html
this.div.addEventListener('click',(e)=>{
//para impedir los datos google
  e.preventDefault()
  e.stopPropagation()
  //cambiamos el contenido para mostrar el contenido del metodo setContent
  //lo mismo que this.div.innerHTML=this.html
  this.activate()
})
```

Así definido, tenemos el mapa de Google y los markers en sus posiciones geográficas.

Para que saber que elemento está activado cuando pasamos el ratón, creamos dos

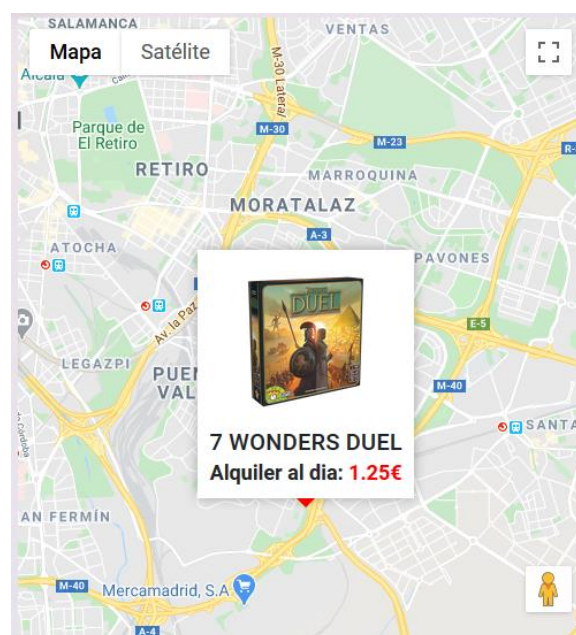
métodos:

```
//se crea 2 metodos más: active y non-active segun el focus del elemnto
hover(){
  if(this.div!==null){
    //class is-active ->style.css (372)
    this.div.classList.add('is-active')
  }
}
hoverOut(){
  if(this.div!==null){
    this.div.classList.remove('is-active')
  }
}
```

Y para cada item, creamos un addEventListener.

```
//Para mostrar en el mapa el element item selecionado
item.addEventListener('mouseover', function(){
  marker.hover()
  //desactivar marker
  if(activeMarker!==null){
    activeMarker.hoverOut()
  }
  activeMarker=marker
})
item.addEventListener('mouseleave', function(){
  if(activeMarker===marker)
    marker.hoverOut();
  //para hacer como un reset de la class active y reiniciarla.
  activeMarker=null;
})
```

La posibilidad de hacer click en el elemento en el mapa se define así:



Creamos un método setContent:

```
//parametro una cadena de texto que sera el contenido
setContent(html){
    this.html=html;
}
```

Que definiremos por cada marker: marker.setContent().

El elemento html se define en la clase TextMarker: this.html=null. Definimos un método onAdd(), con un addEventListener.

```
//al añadir el elemnto
onAdd () {
    this.div = document.createElement('div')
    //damos un class al marker para el css
    this.div.classList.add('marker')
    //overlay sigue el movimiento del mapa
    this.div.style.position = 'absolute'
    this.div.innerHTML = this.text
    this.getPanes().overlayImage.appendChild(this.div)
    //al hacer click en mi elemento se añade el contenido de mi html
    this.div.addEventListener('click',(e)=>{
        //para impedir los datos google
        e.preventDefault()
        e.stopPropagation()
        //cambiamos el contenido para mostrar el contenido del metodo setContent
        //lo mismo que this.div.innerHTML=this.html
        this.activate()
    })
}
```

```
activate(){
    if(this.div!=null){
        this.div.innerHTML=this.html
        if(this.div.innerHTML!=null){
            this.div.classList.add('cuadroMapa')
        }
    }
}
```

Y definimos el comportamiento según es activo o no.

```
//para activar y desactivar el elemento html
marker.onActivation.push(function(){
    //desactivar marker
    if(enableMarker!=null){
        enableMarker.desactivate()
    }
    enableMarker=marker
})
```

5.3.2.10 Ver juego de mesa



El usuario *usuario2* te propone **CATAN**

Alquila el juego:

➤ 2.99€ al día ➤ 6€ fin de semana
➤ 20.93€ a la semana ➤ 89.7€ al mes

Compra el juego:

➤ Comprar por: 35€

CATAN:

Catan es un juego de mesa para toda la familia que se ha convertido en un fenómeno mundial. Desde su aparición en Alemania ha vendido más que muchos

Cuando seleccionamos un juego de mesa, podemos o mandar un email para alquilarlo o comprarlo. Los datos del juego se transmiten por la URL:

Al alquilar el juego debemos definir las fechas de inicio y de fin. Se hace mediante datePicker.

Alquila el juego:

➤ 1.25€ al día ➤ 3€ fin de semana
➤ 8.75€ a la semana ➤ 37.5€ al mes

Elegir la fecha:

Inicio:

2020-05-09

Hasta:

☒ Manda un email para

Mandar

Mayo 2020

Lu	Ma	Mi	Ju	Vi	Sá	Do
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

Compra el

➤ Comprar

5.3.2.11 Alquilar juego de mesa

Para alquilar el juego de mesa, se manda un email al propietario del juego de mesa.

[Volver a la pagina Principal](#)

Muchas gracias por utilizar nuestro servicio

Su email se envió correctamente. Recibirá un email de confirmación cuando se apruebe su petición.

El propietario recibe un email:



Alquiler de juego de mesa !

El usuario, usuario1 esta interesado en alquilar su juego de mesa CATAN. Por favor compruebe que lo tiene disponible en estas fechas. Recuerda que puede ponerse en contacto con nosotros por cualquier duda que puedas tener.

Validar el alquiler

Validar la transaccion.

Se hace mediante PHPMailer.

Al validar la transacción aparece un mensaje de confirmación.

Muchas gracias por utilizar nuestro servicio

Su email se envió correctamente. El usuario recibirá un email confirmandole que puede alquilar el juego de mesa.

Y por fin el usuario que deseaba alquilar el juego recibe un email:



Alquiler juego de mesa !

El usuario usuario2 ha aceptado alquilarte CATAN.
Puedes pasar a buscarlo...

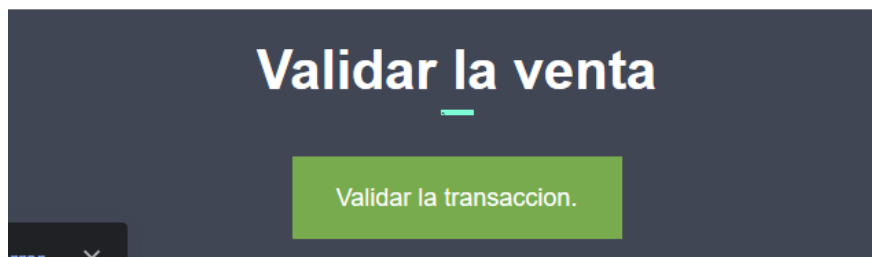
5.3.2.12 Comprar juego de mesa

La compra del juego de mesa sigue los mismos pasos que el alquiler. Se manda una solicitud al usuario, que puede validar la transacción o no. En caso afirmativo, se manda un email al comprador.



Venta de juego de mesa !

El usuario, usuario1 esta interesado en comprar su juego de mesa: CATAN. Por favor compruebe que lo tiene disponible. Recuerda que puede ponerse en contacto con nosotros por cualquier duda que puedas tener.



Email recibido si se confirma la transaccion:

S SENET <informationsenet@gmail.com>
Jue 28/05/2020 18:24
Para: Usted



Compra juego de mesa !

El usuario usuario2 ha aceptado venderte CATAN.
Puedes pasar a buscarlo...

6.- DESPLIEGUE Y PRUEBAS

A medida que se desarrollaba el programa, se ha hecho una serie de pruebas para definir el funcionamiento correcto del sistema. Primero se ha hecho una serie de pruebas unitarias, destinadas a comprobar que cada método devolvía el resultado esperado. También, una serie de pruebas han permitido definir lo que el usuario puede introducir en el programa y lo que no. Por fin para comprobar el correcto funcionamiento general del programa, se realizó una serie de pruebas globales.

6.1.- PLAN DE PRUEBA

Una Prueba Unitaria, es una forma de comprobar que nuestro código, hace lo que debe hacer. Nos asegura que no hay fallos, errores o cálculos inesperados.

6.1.1 creación de un usuario

Para crear una cuenta tenemos que comprobar que los elementos introducidos por el usuario respectan una serie de criterios. Esta revisión se puede hacer tanto en la parte front-end como en la parte back-end. Vamos a repasar estos criterios y como se han implementado en la página.

PRUEBAS UNITARIAS		
CLASE	ACCIONES	PRUEBAS
USUARIO	CREAR	Ningún campo vacío
		Formato del email sea correcto
		Comprobación password
		Comprobar si el usuario existe
		Comprobar si el email existe
		Validacion del email
		Encriptar password
		Datos introducidos correctos

Comprobaciones front-end:

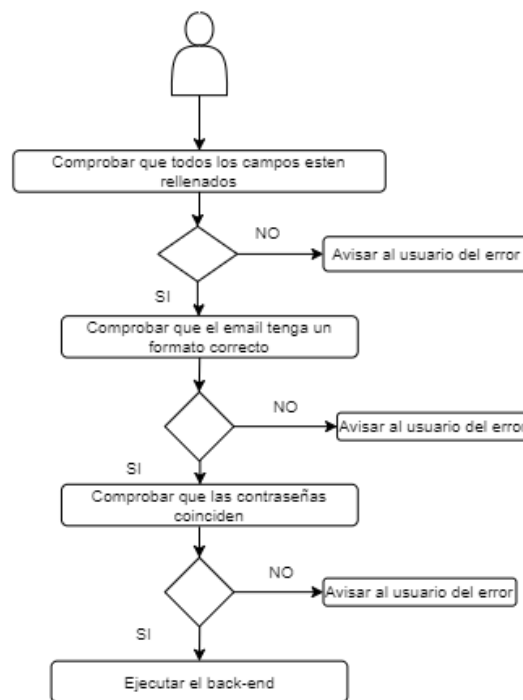


Fig 6.1.1 Diagrama de flujo de comprobación front-end

Con javascript comprobamos que todos los campos están rellenos en caso contrario avisamos al usuario modificando el color de la casilla. Validamos que el email sea un email cambiado el color del texto a rojo en caso contrario. No es una comprobación perfecta y por eso mandamos un email de confirmación al usuario como lo veremos más adelante.

```
42 function validateMail(idMail)
43 {
44     //Creamos un objeto
45     object = document.getElementById(idMail);
46     valueForm = object.value;
47     // Patron para el correo
48     var patron = /^[^<>()[\].,;:\s@"]+(\.[^<>()[\].,;:\s@"]+)*"["\n"]+">@(?![^<>()[\].,;:\s@"]{2,63})$/i;
49     if (valueForm.search(patron) == 0) {
50         //Mail correcto
51         object.style.color = "#000";
52         return;
53     }
54     //Mail incorrecto
55     object.style.color = "#f00";
56 }
57
```

En javascript comprobamos que todos los campos sean rellenos. En el caso de las contraseñas comprobamos que sean iguales.

```
else if (f.password.value != f.confirm.value) {
    f.password.style.backgroundColor = ' #ffdddd' ;
    f.confirm.style.backgroundColor = ' #ffdddd' ;
    f.password.focus();
    e.preventDefault();
    return;
}
```

Comprobaciones back-end:

Si todos los campos han sido validados por la parte front-end, mandamos los datos al back-end.

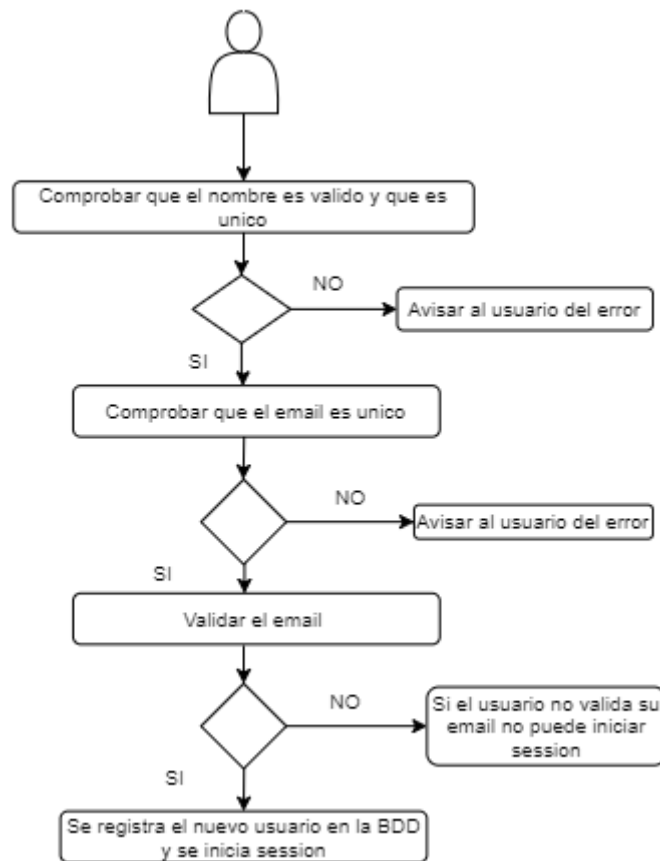


Fig 6.1.1.2 Diagrama de flujo de comprobación back-end

La página crear.php del controlador se encarga de recopilar la información y llama a la función addAccount() de la clase Account. Esta función comprueba que:

- El nombre es válido (que tenga una longitud de entre 2 y 16 caracteres)
- La contraseña sea correcta (que tenga una longitud de entre 2 y 16 caracteres)
- Si el nombre del usuario ya existe en la base de datos, para no duplicarlos
- Si el email introducido no existe en la base de datos.

Utilizamos la función hash para encriptar la contraseña. Según,

<https://www.php.net/manual/es/function.password-hash.php>

“**password_hash()** crea un nuevo hash de contraseña usando un algoritmo de hash fuerte de único sentido. **password_hash()** es compatible con [crypt\(\)](#). Por lo tanto, los hash de contraseñas creados con [crypt\(\)](#) se pueden usar con **password_hash()**.”

También generamos un token único con la función `bin2hex(random_bytes(16))`. Según, <https://www.php.net/manual/es/function.bin2hex.php>

“Devuelve una cadena ASCII que contiene la representación hexadecimal de `str`. La conversión se realiza byte a byte, con los 4 bits superiores primero.”

Definimos el campo `account_enabled` a 0. Una vez comprobado el email se modifica a 1.

Comprobación del email:

```
25     $account = new Account();
26     try
27     {
28         $newId = $account->addAccount($nombre, $email,$direccion,$cp,$ciudad,$provincia,$lat,$long,$password);
29     }
30     catch (Exception $e)
31     {
32         echo $e->getMessage();
33         die();
34     }
35     if($newId!='')
36     {
37         try
38         {
39             $row= $account->getDetallesCuentaUsuario($newId);
40         }
41         catch (Exception $e)
42         {
43             echo $e->getMessage();
44             die();
45         }
46         $token=$row['account_token'];
47         /* Mandar email de comprobacion al usuario */
48         $_SESSION['message'] = 'You are logged in!';
49         $_SESSION['type'] = 'alert-success';
50         $_SESSION['username'] = $nombre;
51         $_SESSION['verified'] = 0;
52         $_SESSION['email'] = $email;
53         header('location:../../infoUsuarioValidarEmail.php' );
54         sendVerificationEmail($email, $token);
55     }
56     ?>
```

Si el proceso de creación del nuevo usuario ha sido correcto, la función devuelve un nuevo id. Si no es vacío, la función `getDetallesCuentaUsuario` nos permite recuperar el nuevo token generado. Mandamos un mensaje al usuario:

Bienvenido! usuario10

Debe validar su direccion email.

Por favor haga click en el email de comprobación que le hemos enviado a: **matoo_4@hotmail.com**

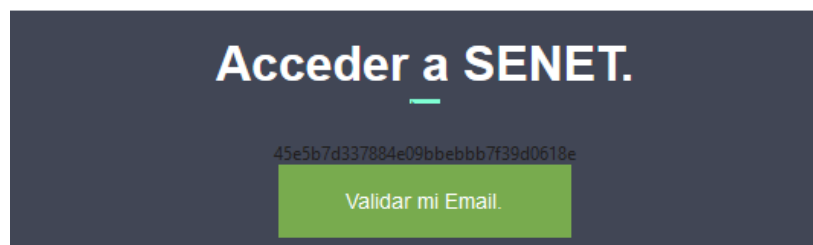
La función `sendVerificacionEmail` toma por parámetros el email del usuario y su token, e instancia la clase `PHPMailer()`. Para el mensaje utilizamos la función `file_get_contents`, la cual devuelve el contenido de un archivo en una variable.

```
17 $body.=file_get_contents('http://localhost/Proyecto/validarEmail.php?a='.$token);
18
19 $mail->SMTPOptions = array(
20     'ssl' => array(
21         'verify_peer' => false,
22         'verify_peer_name' => false,
23         'allow_self_signed' => true
24     )
25 );
26 $mail->IsSMTP();
27 $mail->Host = 'smtp.gmail.com';
28 $mail->SMTPSecure = 'tls';
29 $mail->Port = 587;
30 $mail->SMTPDebug = 1;
31 $mail->SMTPAuth = true;
32 $mail->Username = 'informationSenet@gmail.com';
33 $mail->Password = 'Info@000';
34 $mail->SetFrom('informationSenet@gmail.com', "SENET");
35 $mail->AddReplyTo('no-reply@mycomp.com', 'no-reply');
36 $mail->Subject = 'Petición de reservacion de un juego de mesa';
37 $mail->MsgHTML($body);
38 $mail->SMTPDebug = SMTP::DEBUG_SERVER;
39 $mail->AddAddress($userEmail, 'SENET');
40 $mail->send();
41 }
```

El email recibido por el usuario se visualiza así:

Bienvenido a SENET !

¡Bienvenido! Para registrarte en SENET, primero debes validar tu email en tu cuenta SENET. Una vez que hayas validado tu email, podrás usar tu cuenta de SENET para disfrutar de todas las ventajas de SENET. Con esta cuenta, también puedes acceder a un mundo lleno de juegos de mesa, más divertidos unos que otros. Si crees que recibiste este mensaje por error, visita nuestro Centro de Ayuda para más información sobre cómo eliminar esta cuenta. Es un gusto poder ayudarte.



La validación del email se hace mediante la función `verificarEmail()`, comprobando que el token existe en la base de datos, y modificando el parámetro `account_enabled` a 1 del usuario.

```

1  <?php
2  session_start();
3  require_once '../Model/account_class.php';
4  $account = new Account();
5  if (isset($_GET['token']))
6  {
7      $token = $_GET['token'];
8      try
9      {
10         $verif= $account->verificarEmail($token);
11     }
12     catch (Exception $e)
13     {
14         echo $e->getMessage();
15         die();
16     }
17     if($verif)
18     {
19         $infoUser=    $account->getInfoFromToken($token);
20         $username=    $infoUser['account_name'];
21         $iduser =    $infoUser['account_id'];
22         $_SESSION['login_user']= $username;
23         $_SESSION['id_user'] = $iduser;
24         header('location:../index.php' );
25     }
26     else
27     {
28         header('location:../login.html' );
29     }
30 }

```

Al validar su email el usuario esta automáticamente redirigido a la pagina web con su sesión activada.



6.1.2 Login de un usuario

Cuando un usuario inicia sesión, debemos comprobar que su nombre y su contraseña existen en la base de datos.

PRUEBAS UNITARIAS		
CLASE	ACCIONES	PRUEBAS
USUARIO	LOGIN	Ningún campo vacío
		Comprobación password

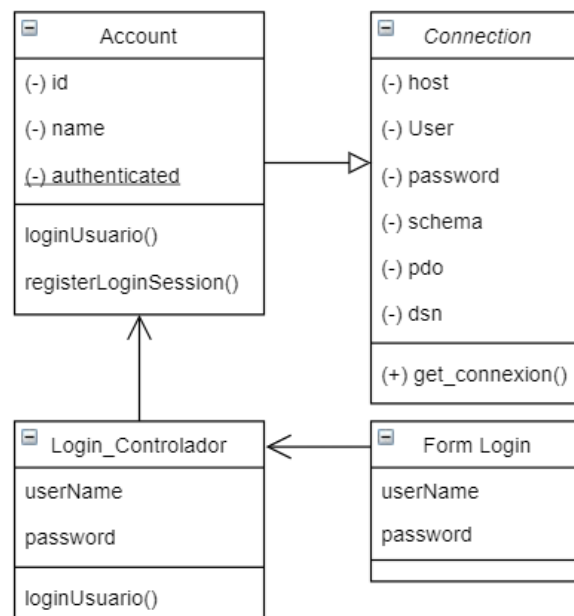


Fig 6.1.2 Diagrama de clases Login

6.1.3 Registrar un juego de mesa

Vamos a ver las comprobaciones a la hora de registrar un juego de mesa.

PRUEBAS UNITARIAS		
CLASE	ACCIONES	PRUEBAS
PRODUCTO	CREAR	Ningún campo vacío
		campo jugadores debe ser un numero entero
		campo tiempo debe ser un numero entero
		campo edad debe ser un numero entero
		Los campos importes deben ser numericos y pueden ser doubles.
		Si el juego existe solo se puede rellenar los campos de precio
		si el juego no existe se debe rellenar todos los campos
		Si el juego existe y el el usuario lo tiene registrado, lo debe modificar en sus preferencias.

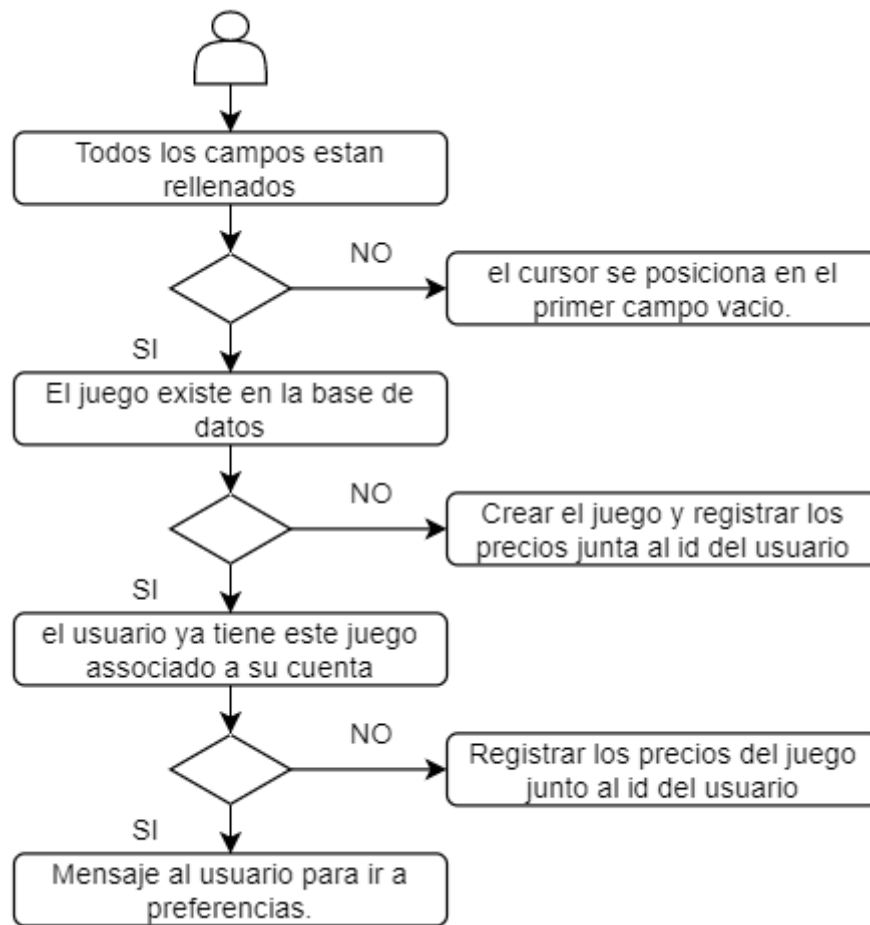


Fig 6.1.3 Diagrama de flujo de comprobación registrar un juego.

Las comprobaciones de introducción de los datos con el teclado se hace mediante javascript:

- Controlar si el importe es numérico:

```

//controla si el importe es numerico y tiene que ser un Integer
function isNumber(evt) {
    evt = (evt) ? evt : window.event;
    var charCode = (evt.which) ? evt.which : evt.keyCode;
    if (charCode > 31 && (charCode < 48 || charCode > 57)) {
        return false;
    }
    return true;
}
  
```

- Controlar si el valor introducido es numérico y puede ser double:

```
$('.filterme').keypress(function(eve) {  
  if ((eve.which != 46 || $(this).val().indexOf('.') != -1) && (eve.which < 48 || eve.which > 57) ||  
      (eve.which == 46 && $(this).caret().start == 0)) {  
    eve.preventDefault();  
  }  
})
```

7.-CONCLUSIONES

7.1.- OBJETIVOS ALCANZADOS

El proyecto alcanzo una serie de objetivos establecidos al principio. Es capaz de gestionar la base de datos, y es capaz de crear y gestionar el master de data de los juegos de mesa. La parte del diseño me parece bastante adecuada en términos de calidad y de usabilidad. La base de datos corresponde a lo que se estableció al principio. Al principio del proyecto pensaba poder realizar una aplicación mucha más amplia que el resultado conseguido. Unas de las tareas que pensaba incluir en el programa eran:

- Localización del usuario en el mapa
- Poner filtros de búsqueda de los juegos de mesa
- Incluir más detalles, enriquecer la página.
- Tener una base datos más amplia y profunda.

No he logrado todos los objetivos que tenía establecidos. Si tuviera que rehacer el proyecto ahora no lo haría de la misma manera.

7.2.- CONCLUSIONES DEL TRABAJO

En mi opinión, aprendí muchísimo gracias al proyecto. El hecho de tener que crearlo desde cero me obligo a revisar todo lo aprendido con Ilerna. Aunque es cierto que podría ser más complejo técnicamente, me parece un proyecto serio, correctamente desarrollado, y concorde a lo que se puede esperar de un primer proyecto después de dos años estudiando DAW. Encontré más dificultades en el desarrollo del código que en cualquier otra parte. Por ejemplo, renuncié a utilizar AJAX por no tener tiempo de investigar su desarrollo. Hubiera debido reflexionar más sobre la creación de la base de datos. Con el proyecto me di cuenta de que se debe dedicar mucho más tiempo de lo que hice al principio a los objetivos y a la estructura tanto de la base de datos como del código. Eso me hubiera permitido ahorrar mucho tiempo en lugar de hacer, deshacer y rehacer.

7.3.- VÍAS FUTURAS

Tal como se indicó previamente, el proyecto no está acabado. Se podría añadir más funcionalidades como la posibilidad de tener filtros en los juegos de mesa, añadir un sistema de notación de los juegos con comentarios. Añadir la posibilidad de tener una ficha del usuario con comentarios de otros usuarios. Añadir más seguridad al código. Añadir más mensajes de interacción con el usuario. Técnicamente, el programa debería

asegurarse de posibles fallos, o ataques a la base de datos. Debería tener un modo de pago online..

9.- BIBLIOGRAFÍA

Aquí dejo unas paginas webs utilizadas para el proyecto. Visualicé y utilicé muchas más.

<http://codewithawa.com/posts/user-registration-and-email-verification-php-and-mysql>

<http://www.bitrepository.com/a-collection-of-free-javascript-date-pickers.html>

<https://stackoverflow.com/questions/28951283/how-to-insert-image-to-database-using-phpmyadmin-insert-tab/28952131>

https://www.w3schools.com/howto/howto_js_autocomplete.asp

https://www.youtube.com/watch?v=AzSzkCUmH_g&list=PLty0cFLf07jXQA5_P9rDMWjpEet2wTXN1&index=12

<https://www.jose-aguilar.com/blog/autocompletar-campo-con-jquery-ajax-y-php/>

<https://jsfiddle.net/bootstrapious/za9dtgme>

<https://www.youtube.com/watch?v=duYbFLefqRw>

https://www.w3schools.com/howto/howto_js_form_steps.asp

<https://alexwebdevelop.com/user-authentication/>

<https://bootsnipp.com/snippets/emRPM>

<https://openclassrooms.com/fr/courses/1885491-prenez-en-main-bootstrap/1886111-une-grille>

<https://programadorwebvalencia.com/como-estructurar-una-pagina-web/>

10.- ANEXOS

10.1.- MANUAL DE INSTALACIÓN

La instalación del programa no necesita información particular, ni herramientas particulares.

10.2.- MANUAL DE USUARIO

Pagina inicial:

La idea de la web es proporcionar a los usuarios la posibilidad de alquilar o comprar juegos de mesa.

¿ Como alquilar o vender juegos en Tres pasos?

Nuestra aplicación es sencilla y eficaz

<p>1. El anuncio</p> <p>Registra tus juegos en nuestra plataforma indicando su nombre y sus características.</p> <p>Crea tu baúl de juegos de forma sencilla</p> <p>Proponer Juegos</p>	<p>2. El acuerdo</p> <p>Consulta las ofertas de venta o de alquiler de otros jugadores. Puedes aceptar o no, según tu disponibilidad, la fiabilidad del jugador, etc.</p> <p>Confianza en la transacción</p> <p>Buscar Juegos</p>	<p>3. El encuentro</p> <p>Encuentra jugadores cerca de tu casa para compartir los juegos. Nuestra aplicación está basada en la confianza y en la proximidad.</p> <p>Cercanía y simplicidad</p> <p>Ir somewhere</p>
--	--	---

Se puede registrar el juego, o si ya existe en la base datos, registrar sus precios.

Registrar los juegos que quieres vender o alquilar

Vuestro juego de mesa...

Nombre 🚗

TRIVIAL_PURSUIT

Descripcion 📄

La Edición Clásica de este juego de Trivial Pursuit es el mismo juego que conoces y que te encanta, pero con un diseño retro de los años 80.

Este juego de Trivial Pursuit incluye el juego y el tablero clásicos y contiene 2.400 preguntas en seis categorías: geografía, entretenimiento, historia, arte y literatura, ciencias y naturaleza, y deportes y pasatiempos.



Se puede buscar los juegos en el mapa:



Y se puede alquilar o comprar:



El usuario *usuario2* te propone **CATAN**

Alquila el juego:

► 1.25€ al día ► 3€ fin de semana
► 8.75€ a la semana ► 37.5€ al mes

Elegir la fecha:

Inicio:

Hasta:

☒ Manda un email para reservar tu juego:

[Mandar Email](#)

Compra el juego:

► Comprar por: **19€**

CATAN:

Catan es un juego de mesa para toda la familia que se ha convertido en un fenómeno mundial. Desde su aparición en Alemania ha vendido más que muchos de los juegos más tradicionales. Se trata de un juego que aúna la estrategia, la astucia y la capacidad para negociar y en el que los jugadores tratan de colonizar una isla, Catán, rica en recursos naturales. Construyendo pueblos, estableciendo rutas comerciales, etc... Catan ha vendido más de 2 millones de ejemplares en Europa y América. Por si esto no fuera bastante, ha sido galardonado en Alemania y Estados Unidos como Juego del Año. El juego básico de Catan, a la venta desde hace más de 10 años en España, ha marcado un hito en toda Europa en cuanto a juego de planificación, colaboración y, por supuesto, diversión. No hace falta comentar que es la única pieza indispensable de Catan en tu ludoteca. A partir del básico se abre todo el abanico de expansiones que la familia de Catan ofrece.