

BLOG DE VIDEOJUEGOS

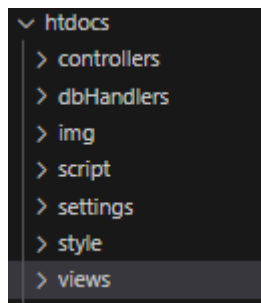


Raúl Caraballo y Serhiy Suvaryan

DOCUMENTACIÓN DEL EJERCICIO

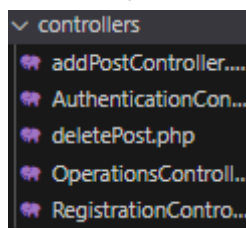
Este proyecto trata de un blog de videojuegos donde puedes consultar los posts que publican otros usuarios e incluso crear tú nuevos para enterarte de las últimas novedades y obtener información si es lo que buscas.

Este proyecto se divide en 7 carpetas las cuales iré explicando una por una:



CONTROLLERS

La primera carpeta llamada *controllers* contiene 5 ficheros .php que se encargan de manejar la lógica de la aplicación:

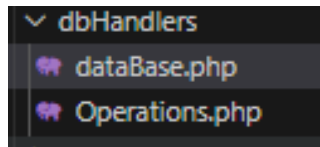


Si cogemos un fichero como ejemplo (*RegistrationController.php*) podemos comprobar que hace uso de un servidor de base de datos donde guardará información como tu correo electrónico o tu nombre de usuario por ejemplo:

```
1 <?php
2 try {
3     require_once("../dbHandlers/dataBase.php");
4 } catch (Exception $e) {
5     echo 'Error message';
6 }
7
8 try {
9     require_once("OperationsController.php");
10 } catch (Exception $e) {
11     echo 'Error operationController not found';
12 }
13
14 $settings = parse_ini_file('../settings/settings.ini');
15 $server = $settings['server'];
16 $username = $settings['username'];
17 $password = $settings['password'];
18 $dbName = $settings['database'];
19
```

DBHANDLERS

La siguiente carpeta llamada *dbHandlers* se compone de solo 2 ficheros también .php que se encargan únicamente de la lógica enfocada a base de datos:



Si abrimos el fichero *dataBase.php* podemos comprobar que se encarga de crear la base de datos, conectarse, crear las tablas para guardar la información del usuario...

```
1  <?php
2  class dataBase {
3      // variable declaration
4      private $server;
5      private $username;
6      private $password;
7      private $database;
8      private $connection;
9
10     public function __construct($server, $username, $password, $database) {
11         $this->server = $server;
12         $this->username = $username;
13         $this->password = $password;
14         $this->database = $database;
15
16         $this->connect($database); //execute connect function
17     }
18
19     private function createDatabase($dbName)
20     {
21         $sql = "CREATE DATABASE IF NOT EXISTS {$dbName}";
22
23         if ($this->connection->query($sql) === false) {
24             throw new RuntimeException("Error creating database");
25         }
26     }
27
28     private function connect() {
29         $this->connection = new mysqli($this->server, $this->username, $this->password, $this->database);
30
31         if ($this->connection->connect_error) {
32             throw new RuntimeException("Connection failed");
33         }
34     }
35 }
```

Debajo de esta carpeta se encuentra *img* la cual guarda en otras 3 carpetas las fotos que se utilizan en el resto de ficheros

SCRIPT

La carpeta *script* es donde se guardan dos ficheros llamados *script_Error_page.js* y *script.js*.

Uno maneja todos los errores que sucedan es decir cuando se produzca un error de cualquier tipo se encargará de enviar al usuario al index del error y que no se le quede la pantalla en blanco (mas adelante está explicado el .html del error)

Luego por otro lado el *script.js* se encarga de la animación de las fotos en la sección de galería de fotos que más adelante será explicada.

```
1  document.addEventListener("DOMContentLoaded", function() {
2      const galleryImages = document.querySelectorAll(".galleryImage");
3      let currentIndex = 0;
4
5      // Ocultar todas las imágenes excepto la primera
6      for (let i = 1; i < galleryImages.length; i++) {
7          galleryImages[i].style.display = "none";
8      }
9
10     // Función para mostrar la siguiente imagen
11     function showNextImage() {
12         if (currentIndex < galleryImages.length - 1) {
13             galleryImages[currentIndex].style.display = "none";
14             currentIndex++;
15             galleryImages[currentIndex].style.display = "block";
16         }
17     }
18
19     // Función para mostrar la imagen anterior
20     function showPreviousImage() {
21         if (currentIndex > 0) {
22             galleryImages[currentIndex].style.display = "none";
23             currentIndex--;
24             galleryImages[currentIndex].style.display = "block";
25         }
26     }
27
28     // Agregar listeners a los botones de navegación
29     document.getElementById("nextBtn").addEventListener("click", showNextImage);
30     document.getElementById("prevBtn").addEventListener("click", showPreviousImage);
31 });
```

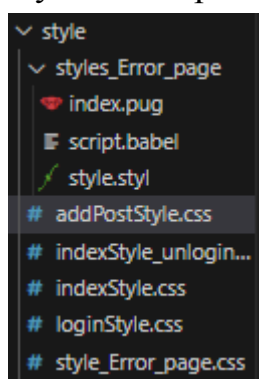
SETTINGS

Settings contiene un archivo .ini que ejecuta las siguientes sentencias:

```
3  server = 'localhost'
4  username = 'root'
5  password = ''
6  database = 'simpleBlog'
7  table = 'Users';
```

STYLE

Style se compone de 5 ficheros y una carpeta.



Estos ficheros son .css y se encargan de darle forma a cada uno de los .php que el usuario visualiza.

El primero es *addPostStyle.css* que es el estilo de *addPost.php*.

El segundo es *index_Style_unlogged.css* que es el estilo de *index_unlogged.php*.

El tercero es *indexStyle.css* que es el estilo de *index_logged.php*.

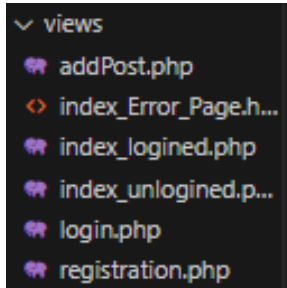
El cuarto es *loginStyle.css* que es el estilo de *login.php* y de *registration.php*

Y por último *style_Error_page.css* que es el estilo de *index_Error_page.html*.

Después encontramos una carpeta ya mencionada anteriormente que simplemente es parte de la lógica y estilo de *index_Error_page.html*.

VIEWS

Views es la carpeta que contiene todos los .php que se muestran al usuario de ahí su nombre. Está compuesta de 6 ficheros:



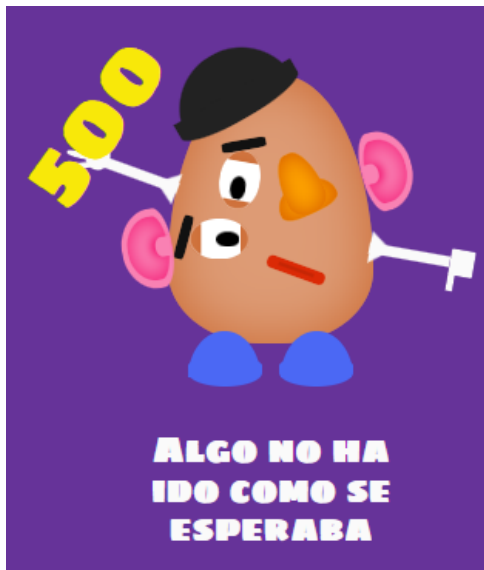
El primero *addPost.php*:

Este php muestra al usuario lo siguiente:

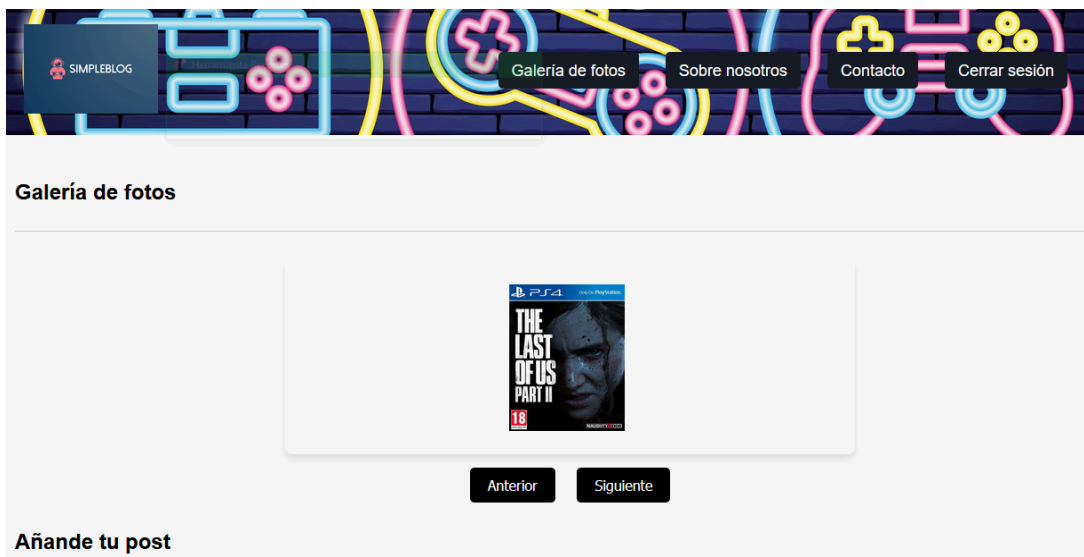
A screenshot of a web form titled 'Añade tu post' (Add your post). The form has a teal background and is set against a collage of video game characters. It includes fields for 'Título:' (Title) and 'Contenido:' (Content), a section for 'Sube una imagen:' (Upload an image) with an 'Examinar...' (Browse...) button and a message 'No se ha seleccionado ningún archivo.' (No file has been selected), and a blue 'Enviar' (Send) button at the bottom.

El usuario puede añadir un post para sumarlo al blog donde tiene que poner el título del videojuego, su contenido/descripción y luego subir una imagen de la carátula del juego por ejemplo.

El segundo fichero es *index_Error_page.html* que es la interfaz gráfica que se muestra al usuario cuando ocurre cualquier error:



En *index_logged.php* sirve para mostrar al usuario que se ha logueado la interfaz de la página. Así lo vería el usuario:



Añade tu post

Añadir Post

Últimos Juegos

Sobre Nosotros

Somos un equipo de entusiastas de los videojuegos, dedicados a explorar y compartir las maravillas del mundo del gaming.

► Ver más

Contacto

Contáctanos:

Email: simpleBlog@gmail.com

Teléfono: +34 724685432

Dirección: Calle Principal 123, Collado Villalba, Madrid

Síguenos en:



Después en el apartado de “Últimos juegos se mostraría lo siguiente”:

Últimos Juegos



The Last Of Us Parte II

Ambientado cinco años después de The Last of Us (2013), el juego se centra en dos p en un conflicto entre su milicia y un culto religioso ...



Spider-Man 2

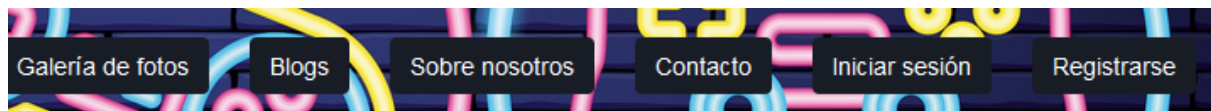
Marvel's Spider-Man 2 es un juego para un jugador. Sin embargo, podrás jugar tanto c únicos...



Hogwarts Legacy

Son los posts hechos por la comunidad.

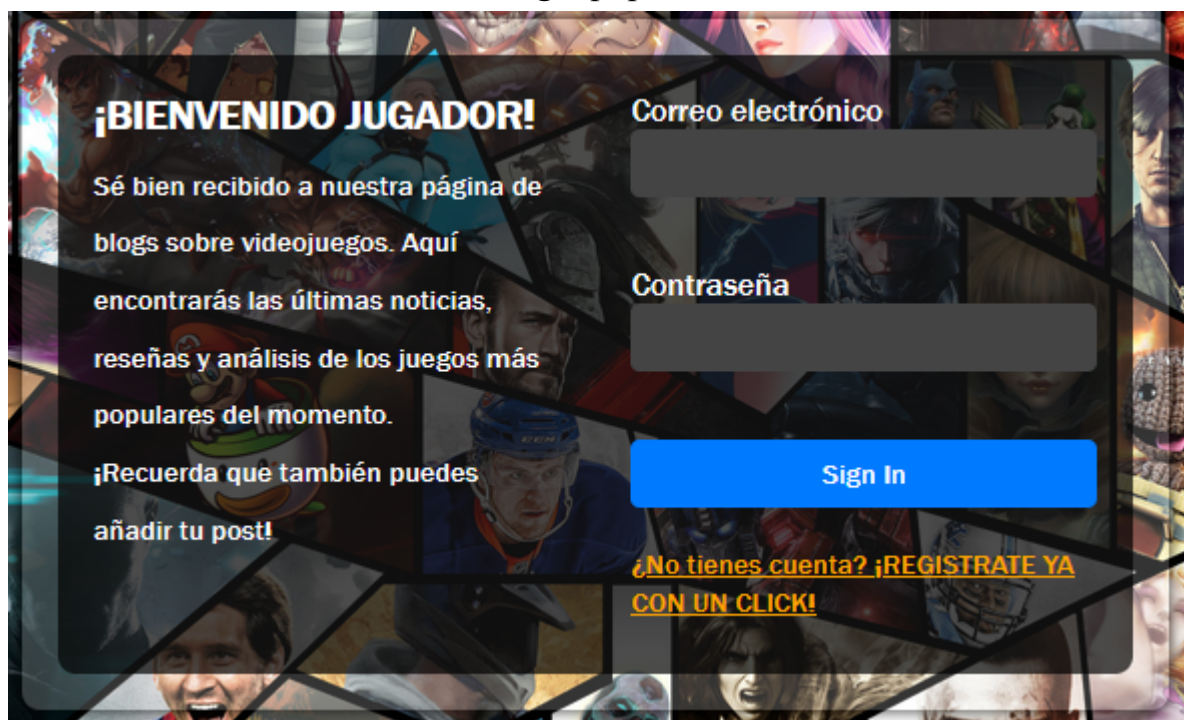
Luego tendríamos la interfaz de *index_unlogged.php* que sería muy parecida con ligeros cambios en la barra de navegación donde en vez de aparecer “cerrar sesión” aparece “iniciar sesión” y también “registrarse”:



El resto se mantiene parecido a la interfaz del usuario logueado exceptuando que si no estás registrado o no has iniciado sesión no tienes la opción de añadir un post como vimos en el .php de login pero podrás ver los posts que se publican.

Como últimos dos tendríamos *login.php* y *registration.php*:

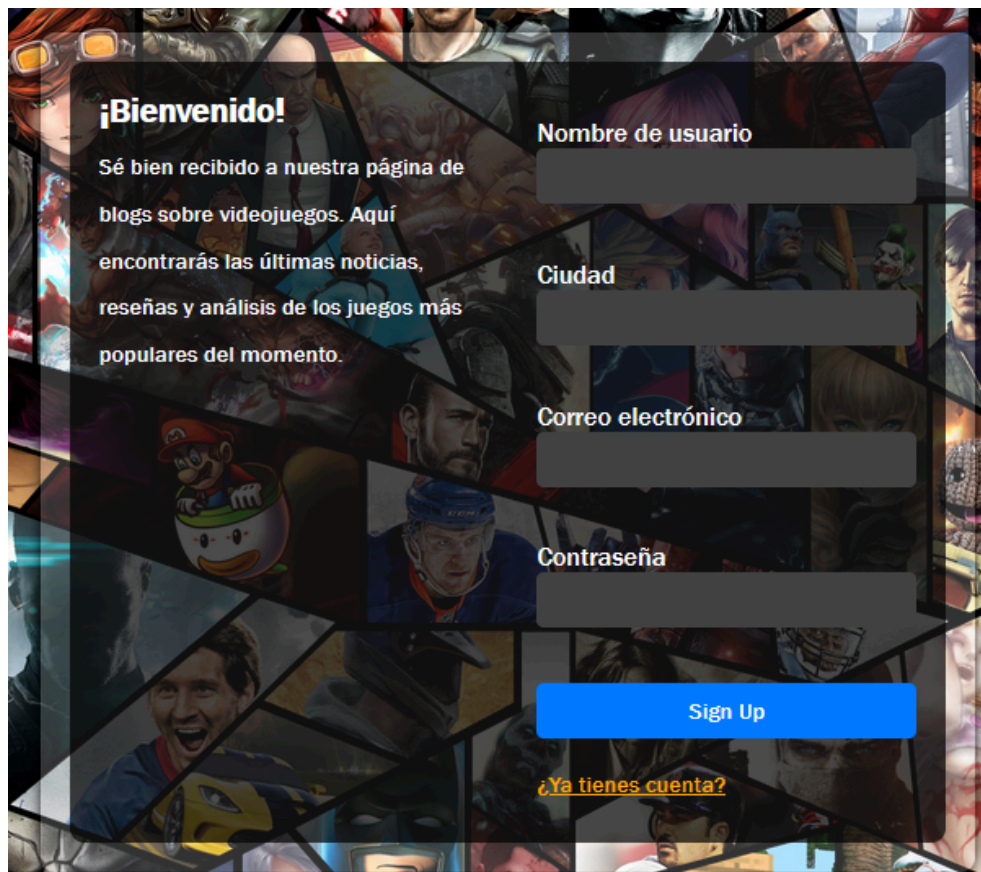
Interfaz de la vista del usuario de *login.php*:



Como podemos ver al usuario que ya está registrado anteriormente su información se guarda en una base de datos la cual utiliza luego esta misma para cuando introduzcas tu correo y contraseña te reconozca y te lleve directo a la interfaz de usuario logueado vista anteriormente.

En caso de que te hayas equivocado o no tengas cuentas registrada anteriormente debajo de “sign in” Pone: ¿No tienes cuenta? ¡REGISTRATE YA CON UN CLCIK! este enlace te llevará a la página de registro es decir

registration.php:



¡Bienvenido!

Sé bien recibido a nuestra página de blogs sobre videojuegos. Aquí encontrarás las últimas noticias, reseñas y análisis de los juegos más populares del momento.

Nombre de usuario

Ciudad

Correo electrónico

Contraseña

[Sign Up](#)

[¿Ya tienes cuenta?](#)

Esta interfaz es algo distinta a la anterior porque te pide que introduzcas “Nombre de usuario” y “Ciudad” (Datos que posteriormente se guardarán en la base de datos para cuando inicies sesión de nuevo) y este también cuenta con un enlace “¿Ya tienes cuenta?” que te redirigirá a *login.php*.