

BookHub

DOCUMENTACIÓN DEL PROYECTO



Elaborada por: Miguel Ángel de la Calle Cuadra

Grado: 2º Desarrollo de Aplicaciones Web

Módulo: Desarrollo web en entorno servidor

INDICE DE CONTENIDOS

1. Introducción al proyecto.....	Pag. 2
2. Estructura general	Pag. 3
3. Funcionalidad de la aplicación	Pag. 5
3.1. Login	Pag. 5
3.2. Registro	Pag. 8
3.3. Opciones de usuario estándar	Pag. 10
3.3.1. Mi perfil	Pag. 11
3.3.2. Catálogo	Pag. 13
3.3.3. Mis libros alquilados	Pag. 19
3.4. Opciones de administrador	Pag. 21
3.4.1. Gestión de usuarios	Pag. 21
3.4.2. Agregar libro	Pag. 25
3.4.3. Gestión de préstamos	Pag. 27
3.5. Seguridad y tratamiento de errores	Pag. 29
3.5.1. Logout	Pag. 29
3.5.2. Tratamiento de inserción de rutas indebidas	Pag. 30
3.5.3. Filtrado de email y contraseña	Pag. 34
3.5.4. Cifrado de contraseñas	Pag. 35
4. Conclusión del proyecto	Pag. 37

1. INTRODUCCIÓN

BookHub es una plataforma versátil que atiende tanto a los amantes de la lectura como a los administradores de bibliotecas con herramientas intuitivas y eficientes. Para los lectores, BookHub brinda una experiencia de usuario excepcional. Pueden explorar el catálogo de libros, encontrar sus títulos favoritos y alquilarlos de manera sencilla. Además pueden mantener un control total sobre tus préstamos, con acceso a la información sobre cuándo deben devolver los títulos alquilados.

Para los administradores de bibliotecas, BookHub es la herramienta perfecta para llevar un seguimiento de todos los aspectos de la gestión de préstamos y usuarios. Pueden ver la lista de usuarios registrados y gestionar su estado. También tendrán acceso completo al registro de préstamos y podrán marcar los libros como devueltos una vez que los usuarios realicen la devolución.

BookHub ha sido ideada para que la administración de bibliotecas sea más eficiente y libre de complicaciones, uniendo a lectores y administradores en una plataforma amigable e intuitiva. A lo largo de las siguientes páginas, analizaremos de manera más detallada la estructura de la aplicación y los pormenores de su funcionamiento.

2. ESTRUCTURA GENERAL

A nivel de estructura general, la web se ha planteado como una SPA (Single Page Application). La elección de desarrollar BookHub como una SPA se basa en la búsqueda de una experiencia de usuario fluida y eficiente. Las SPAs son una opción ideal cuando se pretende proporcionar una navegación rápida y sin interrupciones a través de la aplicación.

Al optar por una SPA, BookHub carga inicialmente todos los recursos necesarios (en este caso HTML, PHP y CSS) durante la primera visita del usuario. A partir de ese momento, las interacciones subsiguientes se gestionan de manera dinámica, actualizando solo las secciones específicas de la página que cambian, en lugar de recargar toda la página. Esto se traduce en tiempos de carga más rápidos y una experiencia de usuario más fluida, ya que se minimiza la latencia al no requerir recargas completas de la página.

Además, la arquitectura de una SPA también simplifica la gestión del estado de la aplicación, ya que puede manejarse de manera centralizada, facilitando el desarrollo y la depuración.

En resumen, la elección de desarrollar BookHub como una SPA se fundamenta en tres pilares: la optimización de la velocidad de carga, la mejora de la experiencia del usuario y la simplificación del desarrollo. Todos ellos contribuyen a proporcionar un entorno más atractivo y eficiente para los usuarios de la aplicación.

En las imágenes que se muestran a continuación, se puede observar de manera más detallada cómo funciona la estructura del index del fichero html que va por debajo de la aplicación. Vemos como las vistas de la cabecera y el footer siempre se incluyen con la directiva “include”. Sin embargo, el “include” del resto de las vistas que conforman la estructura de la web sólo se realiza en función de la existencia de la ruta introducida en la petición y la existencia o no de una sesión.

```
<!DOCTYPE html>

<html lang="es">

<head>
  <link rel="stylesheet" href="css/estilo.css?v=<?php echo time();?>" />
  <style>
    @import url('https://fonts.googleapis.com/css2?family=MuseoModerno:ital,wght@0,100;0,300;0,400;0,600;1,100;1,200;1,400&display=swap');
  </style>
  <title>BookHub</title>
</head>

<body>
  <!--La cabecera siempre se va a ver-->
  <?php include_once "estructuraWeb/cabecera.inc.php";?>

  <?php

  //Si no existe la sesión ni la ruta...
  if (isset($_SESSION ["usuario"]) && isset($_GET["ruta"])){
    include_once "estructuraWeb/menuSuperior.inc.php";
    include_once "estructuraWeb/login.inc.php";
  }
```

```

//Si no existe la sesión pero existe la ruta...
if (!isset($_SESSION ["usuario"]) && isset($_GET["ruta"])) {

    if ($_GET["ruta"] == "principal" || $_GET["ruta"] == "perfil" ||
        $_GET["ruta"] == "catalogo" || $_GET["ruta"] == "alquileres" ||
        $_GET["ruta"] == "prestamos" || $_GET["ruta"] == "gestionUsuarios") {

        header("Location:".$_SERVER["PHP_SELF"]."?ruta=login");
    }

    if ($_GET["ruta"] == "registro"){
        include_once "estructuraWeb/registro.inc.php";
    }

    if ($_GET["ruta"] == "info"){
        include_once "estructuraWeb/info.inc.php";
    }

    if ($_GET["ruta"] == "login"){
        include_once "estructuraWeb/menuSuperior.inc.php";
        include_once "estructuraWeb/login.inc.php";
    }
}

```

```

//Si existe la sesión...
if (isset($_SESSION ["usuario"])) {

    if (isset($_GET["ruta"]) && $_GET["ruta"] == "principal") {

        include_once "estructuraWeb/principal.inc.php";

    }

    if (isset($_GET["ruta"]) && $_GET["ruta"] == "perfil") {

        include_once "estructuraWeb/perfil.inc.php";

    }

    if (isset($_GET["ruta"]) && $_GET["ruta"] == "logout") {

        include_once "estructuraWeb/logout.inc.php";

    }

    if (isset($_GET["ruta"]) && $_GET["ruta"] == "catalogo") {

        include_once "estructuraWeb/catalogo.inc.php";

    }

    if (isset($_GET["ruta"]) && $_GET["ruta"] == "alquileres") {

        include_once "estructuraWeb/alquileres.inc.php";

    }

    if (isset($_GET["ruta"]) && $_GET["ruta"] == "prestamos") {

        include_once "estructuraWeb/prestamos.inc.php";

    }

    if (isset($_GET["ruta"]) && $_GET["ruta"] == "gestionUsuarios") {

        include_once "estructuraWeb/gestionUsuarios.inc.php";

    }
}

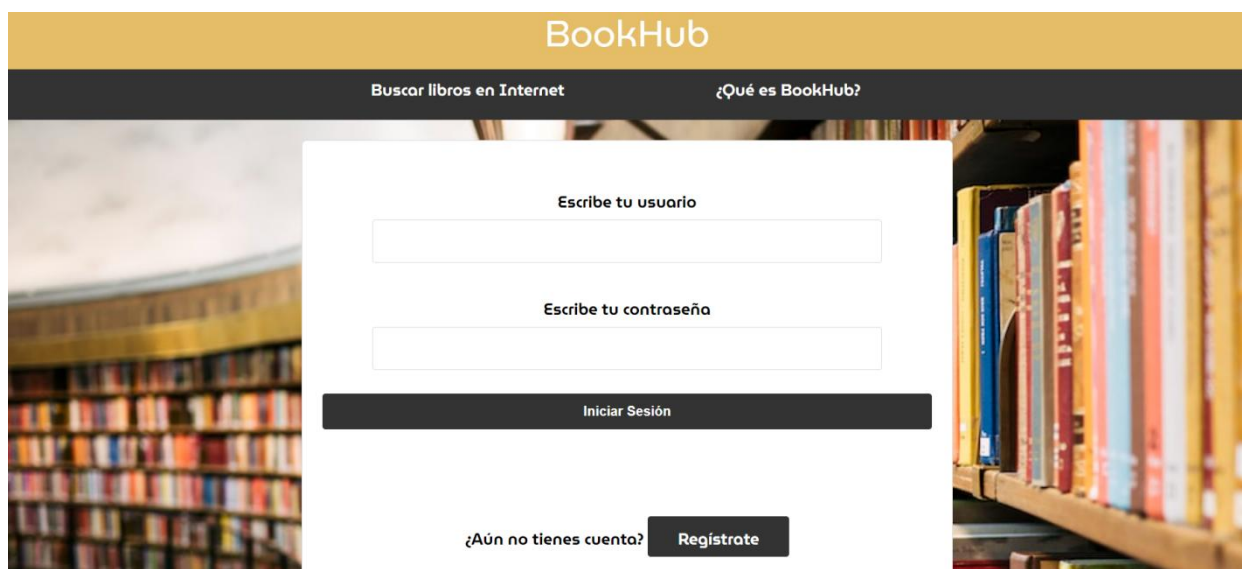
```

3. FUNCIONALIDAD DE LA APLICACIÓN

3.1. Login

Al acceder a la página principal de BookHub, los usuarios son recibidos con una interfaz clara y directa que enfatiza la facilidad de uso y la navegación intuitiva. En la parte superior de la página, el título distintivo "BookHub" y un menú horizontal con dos botones destacados. El primer botón invita a los usuarios a explorar una amplia variedad de libros mediante el acceso a una web especializada de búsqueda de libros en Internet. Al hacer clic en este botón, se abrirá una nueva pestaña o ventana del navegador, facilitando recursos literarios adicionales. El segundo botón, ubicado junto al primero, dirige a los usuarios hacia una vista dedicada a la presentación del proyecto. Al hacer clic en este botón, se carga una página adicional que ofrece información básica sobre BookHub, su propósito y características destacadas. Esta función proporciona a los usuarios una visión más concreta del proyecto.

Justo debajo, se presenta un formulario de inicio de sesión con campos para ingresar nombre de usuario y contraseña. Este diseño minimalista permite a los usuarios con cuentas existentes acceder fácilmente a sus perfiles personales. Para aquellos que aún no tienen una cuenta, se incluye un botón de enlace para el registro, guiándolos hacia el proceso de creación de cuenta de manera rápida y sencilla.



¿Cómo se realiza esta validación de usuario y contraseña? Con el la función php correspondiente (**ComprobarLogin**), la cual encuentra en un fichero independiente en el que se almacenan todas las funciones php que requieren conexión con la base de datos (con el

objetivo de modularizar las diferentes tareas que se realizan en el código de la aplicación) llamado “bibliotecaFunciones.php”.

```
function ComprobarLogin ($usuario, $passwd):bool {  
  
    $conexion = EstablecerConexion();  
  
    $consulta = mysqli_query($conexion, "SELECT password FROM usuarios WHERE usuario='$usuario' AND borrado=0");  
  
    if ($consulta && mysqli_num_rows($consulta) == 1) {  
        $fila = mysqli_fetch_assoc($consulta);  
        $passwordAlmacenado = $fila['password'];  
  
        if (password_verify($passwd, $passwordAlmacenado)) {  
            CerrarConexion($conexion);  
            return true;  
        }  
    }  
  
    CerrarConexion($conexion);  
    return false;  
}
```

La función **ComprobarLogin** realiza lo siguiente:

- Parámetros de entrada:

- \$usuario: Nombre de usuario proporcionado por el usuario durante el intento de inicio de sesión.

- \$passwd: Contraseña proporcionada por el usuario durante el intento de inicio de sesión.

Establecimiento de la conexión:

Se inicia la función llamando a EstablecerConexion(), una función auxiliar que establece y devuelve una conexión a la base de datos. La conexión se almacena en la variable \$conexion.

- Consulta SQL:

Se realiza una consulta SQL en la base de datos utilizando mysqli_query. La consulta busca una fila en la tabla de usuarios donde el nombre de usuario coincida con el proporcionado y la cuenta no esté marcada como borrada (borrado=0).

A continuación, la consulta selecciona el campo de contraseña (password).

- Verificación de resultados:

Se verifica si la consulta fue exitosa (\$consulta && mysqli_num_rows(\$consulta) == 1), lo que significa que se encontró exactamente una coincidencia para el nombre de usuario proporcionado.

- Verificación de la contraseña:

Si la consulta fue exitosa, se obtiene la contraseña almacenada en la base de datos. Se utiliza la función password_verify para comparar la contraseña proporcionada por el usuario con la

contraseña almacenada después de haber sido cifrada (aclararemos esto a continuación). Si la verificación es exitosa, significa que las contraseñas coinciden y la función devuelve true, indicando un inicio de sesión válido.

- Cierre de conexión:

Se llama a `CerrarConexion($conexion)` para cerrar la conexión a la base de datos, independientemente del resultado de la verificación.

```
login.inc.php X
estructuraWeb > login.inc.php
1  <section>
2
3  <?php if (isset ($errValidacion) && $errValidacion == true) { ?>
4
5      <p id="txtError"> Usuario y/o contraseña incorrectas </p>
6
7  <?php
8  }
9  ?>
10
11 |
12 <form id="formValidacion" action="<?php echo $_SERVER["PHP_SELF"];?>" method="post">
13 <br>
14 <label for="usuario"> Nombre de usuario: </label>
15 <input type="text" id="usuario" name="usuario" value="<?php if (isset($usuario)) echo $usuario; ?>"><br>
16 <br>
17
18 <label for="password"> Contraseña: </label>
19 <input type="password" id="password" name="password">
20 <br>
21 <br>
22 <input type="submit" id="enviar" name="enviar" value="Iniciar Sesión">
23 <br>
24 <p id="txtQuieroRegistrarme">¿Aún no tienes cuenta? <a href="index.php?ruta=registro" id="enlaceRegistro"> Regístrate </a> </p>
25 </form>
26
27 </section>
28
```

login.inc.php, donde se encuentra el formulario en el que se introducen los datos


```

index.php
1  <?php
2  include_once "../BBDD/bibliotecaFunciones.php";
3  /**
4   *Aquí vamos a realizar la lógica de la página del login
5   *
6   */
7
8  session_start();
9
10 //En la página del login, funcionalidad del botón "Iniciar sesión"
11
12 if (isset ($_POST["enviar"])) {
13
14 //Comprobamos que el usuario y la contraseña son válidas y, de ser así, le asignamos una sesión con usuario y rol
15
16 if (isset($_POST["usuario"]) && !empty ($_POST["usuario"]) && isset($_POST["password"]) && !empty ($_POST["password"])) {
17
18 if (ComprobarLogin($_POST["usuario"],$_POST["password"])){
19
20     $_SESSION ["usuario"] = $_POST["usuario"];
21
22     $array = RecuperaDatosUsuario($_POST["usuario"]);
23
24     $_SESSION ["rol"] = $array ["rol"];
25
26     header ("Location: ". $_SERVER["PHP_SELF"] ."?ruta=principal");
27
28     exit;
29
30 } else {
31     $errValidacion = true;
32 }
33 }
34 }

```

index.php, donde se recoge la lógica de la comprobación del login

Si la validación del usuario es correcta, asignamos dos variables de sesión con el nombre y el rol del usuario registrado, gracias a las cuales haremos persistentes estos valores en el uso de toda la aplicación. La variable de sesión “nombre” es de gran utilidad para recuperar de los datos de ese usuario concreto desde la base de datos. La variable “rol”, para mostrar según qué opciones al usuario que esté manejando la aplicación, pudiendo ser un usuario estándar o uno con funciones de administrador.

3.2. Registro

Aunque la pantalla de registro no está visible para el usuario nada más acceder a la web, el registro está estrechamente relacionado con el login del usuario, y es la otra vista a la que el usuario puede acceder antes de utilizar la aplicación. Por esto, se ha decidido recoger la lógica del registro de usuario, al igual que la del login, en el archivo index.

La lógica del registro se realiza utilizando la función **RegistroUsuario()** que, como ya sabemos, se encuentra en el fichero “bibliotecaFunciones.php”:

```

function RegistroUsuario($nombre,$apellido1,$apellido2,$usuario,$password,$correo) {
    $conexion = EstablecerConexion();

    if (comprobarCorreo($correo) && comprobarContrasena($password)) {

        $passwordHash = password_hash($password, PASSWORD_BCRYPT);

        try {
            mysqli_query($conexion, "INSERT INTO usuarios (`nombre`, `apellido1`, `apellido2`, `usuario`, `password`, `correo`, `fecha_registro`, `rol`) VALUES ('".$nombre."', '".$apellido1."', '".$apellido2."', '".$usuario."', '".$passwordHash."', '".$correo."', NOW(), 'usuario');");
            return true;
        } catch (Exception $e) {
            echo $e->getMessage();
            return false;
        }
    }

    CerrarConexion($conexion);
}

```

¿Cómo trabaja esta función?

- Parámetros de entrada:

- \$nombre, \$apellido1, \$apellido2: Información personal del usuario (nombre y apellidos).
- \$usuario: Nombre de usuario elegido por el nuevo usuario.
- \$password: Contraseña proporcionada por el nuevo usuario (sin cifrar).
- \$correo: Dirección de correo electrónico del nuevo usuario.

- Establecimiento de la conexión:

Se inicia la función llamando a EstablecerConexion(). La conexión se almacena en la variable \$conexion.

- Comprobación de formato de correo y contraseña:

Se comprueba si las cadenas introducidas por el usuario en los campos de correo electrónico y contraseña cumplen con el formato requerido con las funciones comprobarCorreo() y comprobarContrasena(). De ser así, ambas funciones devolverán "true" y el programa entrará en el bloque de la condición para seguir ejecutando el programa y realizar el hashing de la contraseña.

- Hashing de la contraseña:

Se utiliza password_hash para cifrar la contraseña y almacenarla en la base de datos de forma segura. El resultado del cifrado se almacena en la variable \$passwordHash.

- Inserción en la base de datos:

Se ejecuta una consulta SQL con `mysqli_query` para insertar un nuevo registro en la tabla de usuarios. La información del usuario, incluyendo el nombre, apellidos, nombre de usuario, contraseña cifrada, correo electrónico, fecha de registro y rol, se inserta en la base de datos.

La fecha de registro se establece utilizando `NOW()` para obtener la fecha y hora actuales.

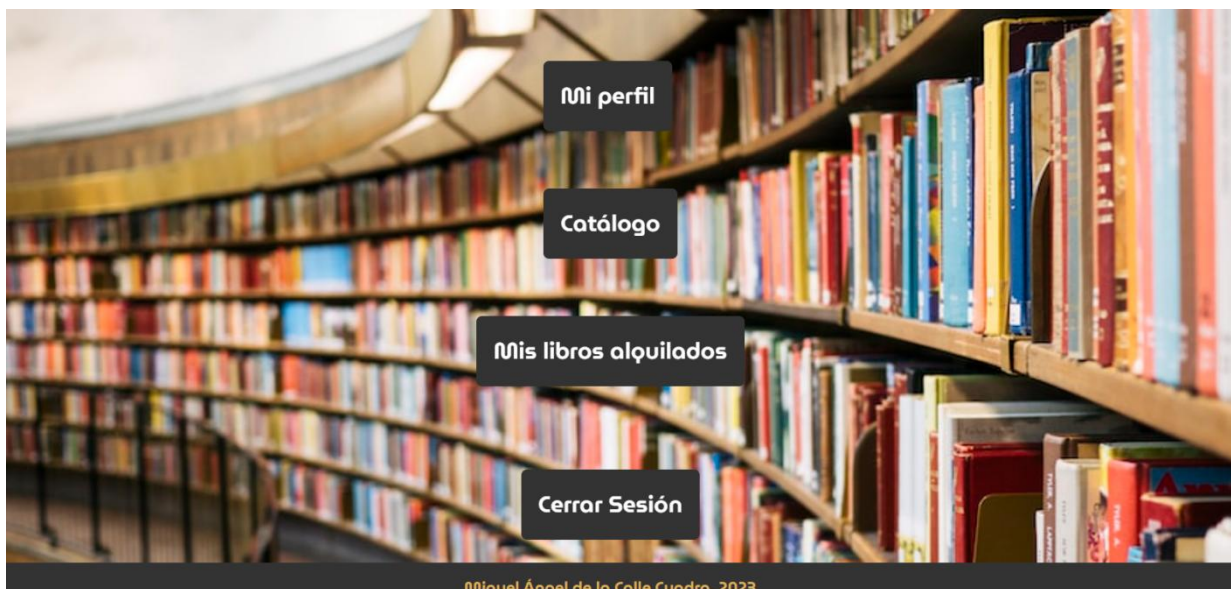
Si la inserción es exitosa, se imprime "Usuario insertado". En caso de error, se imprime un mensaje de error que incluye detalles específicos sobre el error.

- Cierre de conexión:

Se llama a `CerrarConexion($conexion)` para cerrar la conexión a la base de datos, independientemente del resultado de la inserción.

3.3. Opciones del usuario estándar

Cuando el usuario ya dispone de una cuenta, tal como ya se indicaba en la sección correspondiente, realiza el login y se le asignan dos variables de nombre de usuario y sesión. Estas serán indispensables para mostrar las opciones de las que dispondrá el usuario en su pantalla principal según sea un usuario estándar o el administrador. Si es un usuario estándar, su pantalla principal tiene este aspecto:



El usuario dispone de las opciones que se muestran en la imagen: revisar los datos de su cuenta, acceder al catálogo de libros disponibles en la biblioteca (más tarde se tratará con mayor profundidad de este apartado) para alquilarlos, y ver los libros que tiene alquilados actualmente. Todas estas vistas se sirven de funciones del archivo `bibliotecaFunciones.php` para mostrar los datos necesarios en cada una de ellas, como veremos a continuación.

3.3.1. Sección “Mi perfil”

En esta sección el usuario puede comprobar los datos de su cuenta. Para poder mostrar la tabla HTML que contiene esta vista, la aplicación recupera los datos de la base a través de la función `RecuperaDatosUsuario()` y los utiliza para mostrar la información correspondiente en dicha tabla:



```
index.php  perfil.inc.php x
estructuraWeb > perfil.inc.php
1  <section>
2
3  <?php
4  $arrayDatos = RecuperaDatosUsuario($_SESSION["usuario"]);
5
6  // Mapeo de nombres de campo a textos personalizados
7  $campoAMensaje = array(
8      "nombre" => "Nombre:",
9      "apellido1" => "Primer Apellido:",
10     "apellido2" => "Segundo Apellido:",
11     "usuario" => "Nombre de Usuario:",
12     "correo" => "Correo Electrónico:"
13 );
14
15 //Creamos la tabla con datos obtenidos de la tabla Usuarios como array asociativo
16 echo "<table id='tablaDatosUsuario'>";
17 foreach ($arrayDatos as $campo => $val) {
18     if (isset($campoAMensaje[$campo])) {
19         $mensaje = $campoAMensaje[$campo];
20         echo "<tr><td>".$mensaje."</td>";
21         echo "<td>".$val."</td></tr>";
22     }
23 }
24 echo "</table>";
25 ?>
26
27 <a href="<?php echo $_SERVER["PHP_SELF"]."?ruta=principal";?>" class="btnesPerfil">Atrás</a>
28 <a href="<?php echo $_SERVER["PHP_SELF"]."?ruta=logout";?>" class="btnesPerfil">Cerrar Sesión</a>
29 </section>
```

```

2 references
function RecuperaDatosUsuario ($nombreUser):array|bool {
    $conexion = EstablecerConexion();

    if ($datos = mysqli_query($conexion, "SELECT * FROM usuarios WHERE usuario='".$nombreUser."'")){
        //Guardamos cada uno de los resultados en un array asociativo llamado $arrayDatos con la funcion mysqli_fetch
        $arrayDatos = mysqli_fetch_assoc($datos);

        if ($arrayDatos){
            return $arrayDatos;
        }
    }
    mysqli_free_result ($datos);
    CerrarConexion($conexion);
}

```

- **Parámetro de entrada:**

La función toma un parámetro llamado \$nombreUser, que se espera sea el nombre de usuario del cual se desean recuperar los datos.

- **Establecimiento de conexión:**

Llama a la función EstablecerConexion() para obtener una conexión a la base de datos.

- **Consulta SQL:**

Realiza una consulta SQL utilizando el nombre de usuario proporcionado para seleccionar todos los datos asociados a ese usuario desde la tabla "usuarios" en la base de datos.

- **Recuperación de datos:**

Utiliza la función mysqli_fetch_assoc para obtener la fila de resultados como un array asociativo llamado \$arrayDatos.

- **Verificación de existencia de datos:**

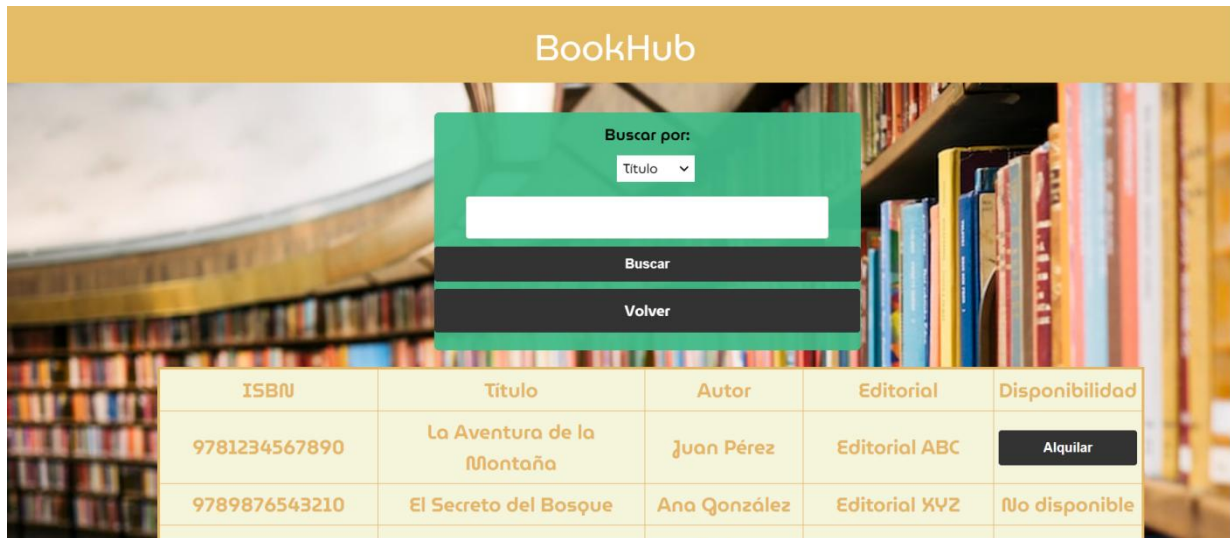
Comprueba si se han encontrado datos para el usuario. Si \$arrayDatos existe, significa que se encontraron datos y los devuelve.

- **Liberación de resultados y cierre de conexión:**

Libera los resultados de la consulta y cierra la conexión a la base de datos utilizando las funciones mysqli_free_result y CerrarConexion.

3.3.2. Sección “Catálogo”

Esta sección funciona de manera similar a la anterior, con la excepción de que utiliza llamadas a dos funciones distintas en función de los libros que deba mostrar, pues la sección catálogo incluye un filtro para que el usuario pueda buscar los libros de manera más eficiente. De esta forma, el usuario puede elegir si buscar un libro por su ISBN, título, autor o editorial.



Cuando el usuario accede al apartado de catálogo, se le muestra por defecto un registro con todos los libros disponibles en la aplicación, ya que el usuario aún no ha filtrado nada. Esta información la muestra llamando a la función `ListarLibros()`:

```
198 /**
199  *ListarLibros function: devuelve un array con los datos de todos los libros de la BBDD
200  *
201  */
202 2 references
203 function ListarLibros ():array|bool {
204     $conexion = EstablecerConexion();
205     $arrayLibros = array(); // Array vacío para almacenar resultados
206
207     //en la variable $totalLibros guardamos todo lo que contiene la tabla libros
208     if ($totalLibros = mysqli_query($conexion, "SELECT * FROM libros;")) {
209         // Recorremos $totalLibros y vamos guardando cada resultado como array asociativo, que vamos añadiendo
210         //al array original
211         while ($fila = mysqli_fetch_assoc($totalLibros)) {
212             $arrayLibros[] = $fila;
213         }
214         // Liberamos el conjunto de resultados
215         mysqli_free_result($totalLibros);
216         return $arrayLibros;
217     } else {
218         // Si la consulta falla
219         return false;
220     }
221 }
222 //Cerramos la conexión con la base de datos
223 CerrarConexion($conexion);
224
225 }
```

- Establecimiento de conexión:

Se inicia estableciendo una conexión a la base de datos utilizando la función `EstablecerConexion()`.

- Inicialización del array:

Se crea un array vacío llamado `$arrayLibros`, que se utilizará para almacenar los resultados de la consulta.

- Consulta SQL:

Se realiza una consulta SQL para seleccionar todos los datos de la tabla "libros" en la base de datos.

- Recorrido de resultados:

Se utiliza un bucle `while` para recorrer todos los resultados de la consulta, fila por fila. Cada fila se guarda como un array asociativo llamado `$fila`. Este array se agrega al array `$arrayLibros`.

- Liberación de resultados:

Después de recorrer todos los resultados, se liberan los recursos del conjunto de resultados utilizando `mysqli_free_result`.

- Retorno de datos:

Se devuelve el array `$arrayLibros`, que ahora contiene todos los libros de la tabla.

- Manejo de errores:

Si la consulta SQL falla, la función devuelve `false`.

- Cierre de conexión:

Finalmente, se cierra la conexión a la base de datos utilizando la función `CerrarConexion()`.



Sin embargo, si el usuario elige el campo por el que quiere buscar e introduce un texto en el filtro buscador, el programa utilizará la función LibrosFiltrados():

```
1 reference
function LibrosFiltrados ($campo, $valor):array|bool {
    $conexion = EstablecerConexion();
    $arrayLibrosFiltrados = array(); // Array vacío para almacenar resultados

    //en la variable $totalLibros guardamos todo lo que contiene la tabla libros
    if ($totalLibros = mysqli_query($conexion, "SELECT * FROM libros WHERE $campo LIKE '%$valor%'")) {
        // Recorremos $totalLibros y vamos guardando cada resultado como array asociativo, que vamos añadiendo
        //al array original
        while ($fila = mysqli_fetch_assoc($totalLibros)) {
            $arrayLibrosFiltrados[] = $fila;
        }
        // Liberamos el conjunto de resultados
        mysqli_free_result($totalLibros);
        return $arrayLibrosFiltrados;
    } else {
        // Si la consulta falla
        return false;
    }
}

//Cerramos la conexión con la base de datos
CerrarConexion($conexion);
}
```

Esta función realiza el mismo proceso que la función anterior. Lo único que varía es la consulta que se realiza a la base de datos, que se modifica para obtener solo los elementos que coincidan con lo introducido por el usuario en el buscador. En ambos casos, eso sí, podemos observar que no hay rastro de código HTML en estas funciones php. Solo tenemos código php (con sentencias SQL), y a partir de esta función se coge el array que devuelve para poder rellenar la tabla HTML que se crea en la vista de este apartado. La separación de la lógica de negocio (PHP) y la lógica de presentación (HTML) es una práctica que mejora la modularidad, el mantenimiento, la reusabilidad y la escalabilidad del código.

Es en esta misma sección donde los usuarios pueden realizar, como vemos en las imágenes, el alquiler de los libros (que no estén ya alquilados por otro usuario o por ellos mismos antes). Para ello, solo deben pulsar en el botón “Alquilar” que se encuentra situado a la derecha de cada uno de los títulos. La funcionalidad de dicho botón se realiza llamando a función `alquilar()`, que trabaja de la siguiente manera:

```

*/
1 reference
function alquilar ($idLibro, $nombreUsuario) {

    //Establecemos conexión con la base de datos
    $conexion = EstablecerConexion();

    //Obtenemos el id del usuario que ha alquilado el libro con la función dameidUsuario()
    $idUser = dameidUsuario($nombreUsuario);

    //Obtenemos el título con el id del libro que está siendo alquilado (se lo hemos pasado a la función) usando la función dameTitulo()
    $tituloLibro = dameTitulo($idLibro);

    //Actualizamos en la tabla libros el campo "alquilado" a TRUE en el libro cuyo id es el que le hemos pasado a la función
    mysqli_query($conexion, "UPDATE libros SET alquilado = TRUE WHERE id = '$idLibro'");

    //Creamos una variable con la fecha del sistema más dos semanas
    $fechaDevolucion = date('Y-m-d', strtotime('+2 weeks'));

    //Realizamos el insert en la tabla préstamos con los datos correspondientes
    mysqli_query($conexion, "INSERT INTO prestamos (id_usuario, id_libro, nombre_libro, fin_prestamo)
    VALUES ('$idUser', '$idLibro', '$tituloLibro', '$fechaDevolucion')");

    CerrarConexion($conexion);
}

```

- Conexión a la base de datos:

Se inicia una conexión a la base de datos utilizando la función `EstablecerConexion()`. Esta función probablemente esté definida en otro lugar del código y es responsable de realizar la conexión y devolver el objeto de conexión.

Obtención del ID de usuario:

- Obtención del ID de usuario:

Se llama a la función `dameidUsuario()` para obtener el ID de usuario correspondiente al nombre de usuario proporcionado. Este ID se utilizará más adelante en la inserción de datos en la tabla de préstamos. La función `dameidUduario()` se encuentra recogida en el mismo archivo `bibliotecaFunciones()`, y funciona de la siguiente manera:

```

2 references
function dameidUsuario($nombreUser):mixed {
    $conexion = EstablecerConexion();

    $query = "SELECT id FROM usuarios WHERE usuario='$nombreUser'";
    $result = mysqli_query($conexion, $query);

    if ($result && $row = mysqli_fetch_assoc($result)) {
        $id = $row['id'];
    }

    mysqli_free_result($result);
    CerrarConexion($conexion);

    return $id;
}

```

- Obtención del título del libro:

Similar al paso anterior, se llama a la función `dameTitulo()` para obtener el título del libro correspondiente al ID de libro proporcionado. Este título se utilizará en la inserción de datos

en la tabla de préstamos. La función dameTitulo() se encuentra recogida en el mismo archivo bibliotecaFunciones(), y funciona de la siguiente manera:

```
function dameTitulo($idLibro):mixed {  
    $conexion = EstablecerConexion();  
  
    $query = "SELECT titulo FROM libros WHERE id='$idLibro'";  
    $result = mysqli_query($conexion, $query);  
  
    if ($result && $row = mysqli_fetch_assoc($result)) {  
        $titulo = $row['titulo'];  
    }  
  
    mysqli_free_result($result);  
    CerrarConexion($conexion);  
  
    return $titulo;  
}
```

- Actualización del estado de alquiler del libro:

Se ejecuta una consulta SQL para actualizar el campo "alquilado" a TRUE en la tabla de libros para el libro específico identificado por el ID proporcionado.

- Generación de la fecha de devolución:

Se genera una fecha de devolución establecida como la fecha actual más dos semanas utilizando la función date() y strtotime().

- Inserción de datos en la tabla de préstamos:

Se ejecuta una consulta SQL para insertar una nueva fila en la tabla de préstamos con los datos del usuario, el libro, el título del libro y la fecha de devolución.

- Cierre de la conexión a la base de datos:

Se cierra la conexión a la base de datos utilizando la función CerrarConexion().

Si el proceso de alquiler ha sido exitoso, se muestra un mensaje de confirmación y se utiliza un pequeño efecto visual de "spinner" realizado con CSS para indicar que se está realizando la operación. Luego, se realiza una redirección a la página principal (?ruta=principal) después de 2 segundos con un elemento "header" que incorpora un "refresh".

Además, a diferencia de las vistas de Registro y Login, donde los datos recogidos en el formulario se enviaban al index para allí tratar la lógica que trata los datos que le llegan de los formularios de estas dos vistas, la funcionalidad de esta vista se recoge en el mismo fichero, para mayor claridad y comprensión del código, al unir el formulario y la lógica php que trata los datos que llegan del mismo en un mismo archivo. Vemos todo esto a continuación:

```

estructuraWeb > catalogo.inc.php
1  <?php
2  include_once "../BBDD/bibliotecaFunciones.php";
3
4  // Funcionalidad del botón "Alquilar"
5  if ($_SERVER["REQUEST_METHOD"] === "POST") {
6      $arrayLibros = ListarLibros(); // Cargar los libros nuevamente
7      foreach ($arrayLibros as $libro) {
8          if (isset($_POST[$libro['id']])) {
9              alquilar($libro['id'], $_SESSION['usuario']);
10             echo "<h1 id='avisoAlquilado'>Procesando alquiler del libro. Serás redirigido en unos segundos.</h1>";
11             echo "<br>";
12             echo "<div class= 'spinner'>
13                 <div></div>
14                 <div></div>
15                 <div></div>
16                 <div></div>
17                 <div></div>
18                 <div></div>
19                 </div>";
20             header("refresh:2;url=" . $_SERVER["PHP_SELF"] . "?ruta=principal");
21             exit();
22         }
23     }
24 }
25

```

```

26 // Funcionalidad del botón "Buscar por filtro"
27 if (isset($_POST["btnBuscarFiltro"]) && !empty($_POST['valorFiltro'])) {
28     $campo = $_POST['select'];
29     $valor = $_POST['valorFiltro'];
30     $arrayLibros = LibrosFiltrados($campo, $valor);
31 } else {
32     // Si no se ha pulsado el botón de filtro, mostrar todos los libros
33     $arrayLibros = ListarLibros();
34 }
35 ?>
36
37 <section id="sectionCatalogo">
38     <form id="formCatalogo" action="<?php echo $_SERVER["PHP_SELF"] . "?ruta=catalogo"; ?>" method="post">
39         <div id="divFiltro">
40             <form id="formFiltro" action="<?php echo $_SERVER["PHP_SELF"] . "?ruta=catalogo"; ?>" method="post">
41                 <label for="select">Buscar por:</label>
42                 <select name="select" id="select">
43                     <option value="titulo" selected>Titulo</option>
44                     <option value="autor">Autor</option>
45                     <option value="isbn">ISBN</option>
46                     <option value="editorial">Editorial</option>
47                 </select>
48                 <input type="text" name="valorFiltro" id="valorFiltro">
49                 <input type="submit" name="btnBuscarFiltro" value="Buscar">
50                 <a href="<?php echo $_SERVER["PHP_SELF"] . "?ruta=principal"; ?>">Volver</a>
51             </div>
52             <table id='tablaCatalogo'>
53                 <tr>
54                     <td>ISBN</td>
55                     <td>Titulo</td>
56                     <td>Autor</td>
57                     <td>Editorial</td>
58                     <td>Disponibilidad</td>
59                 </tr>
60                 <?php foreach ($arrayLibros as $libro) { ?>
61                     <tr>
62                         <td><?php echo $libro["isbn"]; ?></td>
63                         <td><?php echo $libro["titulo"]; ?></td>
64                         <td><?php echo $libro["autor"]; ?></td>
65                         <td><?php echo $libro["editorial"]; ?></td>
66                         <td>
67                             <?php
68                             if ($libro["alquilado"] == TRUE) {
69                                 echo "No disponible";
70                             } else { ?>
71                                 <input type="submit" name="<?php echo $libro['id']; ?>" value="Alquilar">
72                             <?php }
73                             ?>
74                         </td>
75                     </tr>
76                 <?php } ?>
77             </table>
78         </form>
79     </section>

```

catalogo.inc.php

3.3.3. Sección “Mis libros alquilados”

La funcionalidad de esta vista es muy similar a la vista en el apartado “Mi perfil”. En la sección “Mis libros alquilados”, como su propio nombre indica, el usuario puede llevar un control de los libros que tiene alquilados actualmente, los cuales se recuperan mediante la función `RecuperarAlquilerUsuario()`:

BookHub

Atrás

Libro alquilado	Fecha fin de préstamo
El Secreto del Bosque	2023-11-30
Los Secretos del Universo	2023-11-30

```

estructuraWeb >  alquileres.inc.php
1  <section id="sectionAlquileres">
2
3  <?php
4  $arrayDatos = RecuperarAlquilerUsuario($_SESSION["usuario"]);
5
6  if ($arrayDatos == []) {
7  echo "<h1 id='txtNoAlquiler'> En estos momentos no tienes ningún libro alquilado.</h1>";
8  } else{
9  //Creamos la tabla con datos obtenidos de la tabla Usuarios como array asociativo
10 echo "<table id='tablaAlquileres'>";
11 echo "<tr>
12 |         |         |         |         |         |         |
13 |         |         |         |         |         |         |
14 |         |         |         |         |         |         |
15 |         |         |         |         |         |         |
16 |         |         |         |         |         |         |
17 |         |         |         |         |         |         |
18 |         |         |         |         |         |         |
19 |         |         |         |         |         |         |
20 |         |         |         |         |         |         |
21 |         |         |         |         |         |         |
22 |         |         |         |         |         |         |
23 |         |         |         |         |         |         |
24 |         |         |         |         |         |         |
25 |         |         |         |         |         |         |
26 |         |         |         |         |         |         |
27 |         |         |         |         |         |         |
28 |         |         |         |         |         |         |
29 |         |         |         |         |         |         |
30 |         |         |         |         |         |         |
31 |         |         |         |         |         |         |
32 |         |         |         |         |         |         |
33 |         |         |         |         |         |         |
34 |         |         |         |         |         |         |
35 |         |         |         |         |         |         |
36 |         |         |         |         |         |         |
37 |         |         |         |         |         |         |
38 |         |         |         |         |         |         |
39 |         |         |         |         |         |         |
40 |         |         |         |         |         |         |
41 |         |         |         |         |         |         |
42 |         |         |         |         |         |         |
43 |         |         |         |         |         |         |
44 |         |         |         |         |         |         |
45 |         |         |         |         |         |         |
46 |         |         |         |         |         |         |
47 |         |         |         |         |         |         |
48 |         |         |         |         |         |         |
49 |         |         |         |         |         |         |
50 |         |         |         |         |         |         |
51 |         |         |         |         |         |         |
52 |         |         |         |         |         |         |
53 |         |         |         |         |         |         |
54 |         |         |         |         |         |         |
55 |         |         |         |         |         |         |
56 |         |         |         |         |         |         |
57 |         |         |         |         |         |         |
58 |         |         |         |         |         |         |
59 |         |         |         |         |         |         |
60 |         |         |         |         |         |         |
61 |         |         |         |         |         |         |
62 |         |         |         |         |         |         |
63 |         |         |         |         |         |         |
64 |         |         |         |         |         |         |
65 |         |         |         |         |         |         |
66 |         |         |         |         |         |         |
67 |         |         |         |         |         |         |
68 |         |         |         |         |         |         |
69 |         |         |         |         |         |         |
70 |         |         |         |         |         |         |
71 |         |         |         |         |         |         |
72 |         |         |         |         |         |         |
73 |         |         |         |         |         |         |
74 |         |         |         |         |         |         |
75 |         |         |         |         |         |         |
76 |         |         |         |         |         |         |
77 |         |         |         |         |         |         |
78 |         |         |         |         |         |         |
79 |         |         |         |         |         |         |
80 |         |         |         |         |         |         |
81 |         |         |         |         |         |         |
82 |         |         |         |         |         |         |
83 |         |         |         |         |         |         |
84 |         |         |         |         |         |         |
85 |         |         |         |         |         |         |
86 |         |         |         |         |         |         |
87 |         |         |         |         |         |         |
88 |         |         |         |         |         |         |
89 |         |         |         |         |         |         |
90 |         |         |         |         |         |         |
91 |         |         |         |         |         |         |
92 |         |         |         |         |         |         |
93 |         |         |         |         |         |         |
94 |         |         |         |         |         |         |
95 |         |         |         |         |         |         |
96 |         |         |         |         |         |         |
97 |         |         |         |         |         |         |
98 |         |         |         |         |         |         |
99 |         |         |         |         |         |         |
100 |         |         |         |         |         |         |
101 |         |         |         |         |         |         |
102 |         |         |         |         |         |         |
103 |         |         |         |         |         |         |
104 |         |         |         |         |         |         |
105 |         |         |         |         |         |         |
106 |         |         |         |         |         |         |
107 |         |         |         |         |         |         |
108 |         |         |         |         |         |         |
109 |         |         |         |         |         |         |
110 |         |         |         |         |         |         |
111 |         |         |         |         |         |         |
112 |         |         |         |         |         |         |
113 |         |         |         |         |         |         |
114 |         |         |         |         |         |         |
115 |         |         |         |         |         |         |
116 |         |         |         |         |         |         |
117 |         |         |         |         |         |         |
118 |         |         |         |         |         |         |
119 |         |         |         |         |         |         |
120 |         |         |         |         |         |         |
121 |         |         |         |         |         |         |
122 |         |         |         |         |         |         |
123 |         |         |         |         |         |         |
124 |         |         |         |         |         |         |
125 |         |         |         |         |         |         |
126 |         |         |         |         |         |         |
127 |         |         |         |         |         |         |
128 |         |         |         |         |         |         |
129 |         |         |         |         |         |         |
130 |         |         |         |         |         |         |
131 |         |         |         |         |         |         |
132 |         |         |         |         |         |         |
133 |         |         |         |         |         |         |
134 |         |         |         |         |         |         |
135 |         |         |         |         |         |         |
136 |         |         |         |         |         |         |
137 |         |         |         |         |         |         |
138 |         |         |         |         |         |         |
139 |         |         |         |         |         |         |
140 |         |         |         |         |         |         |
141 |         |         |         |         |         |         |
142 |         |         |         |         |         |         |
143 |         |         |         |         |         |         |
144 |         |         |         |         |         |         |
145 |         |         |         |         |         |         |
146 |         |         |         |         |         |         |
147 |         |         |         |         |         |         |
148 |         |         |         |         |         |         |
149 |         |         |         |         |         |         |
150 |         |         |         |         |         |         |
151 |         |         |         |         |         |         |
152 |         |         |         |         |         |         |
153 |         |         |         |         |         |         |
154 |         |         |         |         |         |         |
155 |         |         |         |         |         |         |
156 |         |         |         |         |         |         |
157 |         |         |         |         |         |         |
158 |         |         |         |         |         |         |
159 |         |         |         |         |         |         |
160 |         |         |         |         |         |         |
161 |         |         |         |         |         |         |
162 |         |         |         |         |         |         |
163 |         |         |         |         |         |         |
164 |         |         |         |         |         |         |
165 |         |         |         |         |         |         |
166 |         |         |         |         |         |         |
167 |         |         |         |         |         |         |
168 |         |         |         |         |         |         |
169 |         |         |         |         |         |         |
170 |         |         |         |         |         |         |
171 |         |         |         |         |         |         |
172 |         |         |         |         |         |         |
173 |         |         |         |         |         |         |
174 |         |         |         |         |         |         |
175 |         |         |         |         |         |         |
176 |         |         |         |         |         |         |
177 |         |         |         |         |         |         |
178 |         |         |         |         |         |         |
179 |         |         |         |         |         |         |
180 |         |         |         |         |         |         |
181 |         |         |         |         |         |         |
182 |         |         |         |         |         |         |
183 |         |         |         |         |         |         |
184 |         |         |         |         |         |         |
185 |         |         |         |         |         |         |
186 |         |         |         |         |         |         |
187 |         |         |         |         |         |         |
188 |         |         |         |         |         |         |
189 |         |         |         |         |         |         |
190 |         |         |         |         |         |         |
191 |         |         |         |         |         |         |
192 |         |         |         |         |         |         |
193 |         |         |         |         |         |         |
194 |         |         |         |         |         |         |
195 |         |         |         |         |         |         |
196 |         |         |         |         |         |         |
197 |         |         |         |         |         |         |
198 |         |         |         |         |         |         |
199 |         |         |         |         |         |         |
200 |         |         |         |         |         |         |
201 |         |         |         |         |         |         |
202 |         |         |         |         |         |         |
203 |         |         |         |         |         |         |
204 |         |         |         |         |         |         |
205 |         |         |         |         |         |         |
206 |         |         |         |         |         |         |
207 |         |         |         |         |         |         |
208 |         |         |         |         |         |         |
209 |         |         |         |         |         |         |
210 |         |         |         |         |         |         |
211 |         |         |         |         |         |         |
212 |         |         |         |         |         |         |
213 |         |         |         |         |         |         |
214 |         |         |         |         |         |         |
215 |         |         |         |         |         |         |
216 |         |         |         |         |         |         |
217 |         |         |         |         |         |         |
218 |         |         |         |         |         |         |
219 |         |         |         |         |         |         |
220 |         |         |         |         |         |         |
221 |         |         |         |         |         |         |
222 |         |         |         |         |         |         |
223 |         |         |         |         |         |         |
224 |         |         |         |         |         |         |
225 |         |         |         |         |         |         |
226 |         |         |         |         |         |         |
227 |         |         |         |         |         |         |
228 |         |         |         |         |         |         |
229 |         |         |         |         |         |         |
230 |         |         |         |         |         |         |
231 |         |         |         |         |         |         |
232 |         |         |         |         |         |         |
233 |         |         |         |         |         |         |
234 |         |         |         |         |         |         |
235 |         |         |         |         |         |         |
236 |         |         |         |         |         |         |
237 |         |         |         |         |         |         |
238 |         |         |         |         |         |         |
239 |         |         |         |         |         |         |
240 |         |         |         |         |         |         |
241 |         |         |         |         |         |         |
242 |         |         |         |         |         |         |
243 |         |         |         |         |         |         |
244 |         |         |         |         |         |         |
245 |         |         |         |         |         |         |
246 |         |         |         |         |         |         |
2
```

alquileres.inc.php

Como podemos observar, si el array recuperado con la función `RecuperaAlquilerUsuario()` es un array vacío (lo que significa que el usuario logueado no tiene libros alquilados actualmente), simplemente se muestra un texto avisando de que no tiene ningún libro alquilado y, por lo tanto, no hay información que mostrar. Si el array devuelto no está vacío, se recuperan los libros alquilados por ese usuario de la tabla `Préstamos` de la base de datos, y muestran en la vista. `RecuperaAlquilerUsuario()` funciona de la siguiente manera:

```
function RecuperarAlquilerUsuario($usuario):array {
    $conexion = EstablecerConexion();
    $idUserio = dameidUsuario($usuario);
    $LibrosAlquilados = mysqli_query($conexion, "SELECT nombre_libro, fin_prestamo FROM prestamos
    WHERE id_usuario='$idUserio' AND fin_prestamo IS NOT NULL");

    $arrayLibrosAlquilados = array(); // Inicializamos un array para almacenar los resultados

    while ($fila = mysqli_fetch_assoc($LibrosAlquilados)) {
        $arrayLibrosAlquilados[] = $fila; // Agregamos cada registro al array
    }

    mysqli_free_result($LibrosAlquilados);
    CerrarConexion($conexion);

    return $arrayLibrosAlquilados; // Devolvemos el array de resultados
}
```

- Conexión a la base de datos:

Inicia una conexión a la base de datos utilizando la función `EstablecerConexion()`.

- Obtención del ID de usuario:

Llama a la función `dameidUsuario()` para obtener el ID de usuario correspondiente al nombre de usuario proporcionado. Este ID se utilizará en la consulta SQL para recuperar los libros alquilados por ese usuario.

- Consulta SQL para obtener libros alquilados:

Ejecuta una consulta SQL para seleccionar el nombre del libro y la fecha de devolución de la tabla de préstamos para un usuario específico (`id_usuario`) cuya fecha de devolución (`fin_prestamo`) no sea nula.

- Recopilación de resultados en un array:

Inicializa un array para almacenar los resultados y utiliza un bucle `while` para recorrer todas las filas resultantes de la consulta SQL. Cada fila (libro alquilado) se agrega al array `$arrayLibrosAlquilados`.

- Devolución del array de resultados:

`return $arrayLibrosAlquilados;`

Devuelve el array que contiene la información de los libros alquilados por el usuario.

- Liberación de memoria y cierre de conexión:

Libera la memoria asociada con el resultado de la consulta utilizando `mysqli_free_result()` y cierra la conexión a la base de datos utilizando la función `CerrarConexion()`.

3.4. Opciones del usuario administrador

Si el usuario logueado en la aplicación tiene privilegios de administrador (en la base de datos su rol es “admin”), se le asigna una variable de sesión “rol” que es igual a “admin”. Debido a esto, su pantalla principal se cargará con las siguientes opciones disponibles:



La funcionalidad del apartado “Mi perfil” ya se ha detallado anteriormente, pues es exactamente la misma que podemos encontrar en el apartado correspondiente del usuario estándar. Empezamos, pues, con el apartado “Gestión de usuarios”.

3.4.1. Gestión de usuarios

BookHub							Volver
Nombre	Primer apellido	Segundo apellido	Nombre de usuario	Correo Electrónico	Fecha de registro	Borrar usuario	
Anónimo	García	García	usuarioEstandar1	usuarioEstandar1@gmail.com	2023-11-16	Dar de baja	
AnónimoDos	Fernández	Fernández	usuarioEstandar2	usuarioEstandar2@gmail.com	2023-11-16	Dar de baja	

En la pantalla “Gestión de usuarios”, el administrador de la aplicación puede ver la información de los usuarios que están actualmente logueados en la misma . Al final de cada uno de ellos, como se puede observar, dispone del botón “Dar de baja”, con el cual, como su propio nombre indica, puede dar de baja en el sistema al usuario correspondiente. Primero, vemos la estructura general de esta vista que, como ya sucedía con la vista “Catálogo”, contiene tanto el formulario como la lógica php dentro del mismo fichero:

```

1 <?php
2 include_once "../BBDD/bibliotecaFunciones.php";
3
4 // Verificar si se ha enviado el formulario antes de mostrar cualquier contenido
5 //Funcionalidad del botón "Borrar usuario"
6 if ($_SERVER["REQUEST_METHOD"] === "POST") {
7     $arrayUsuarios = RecuperaTotalUsuarios(); // Cargar los usuarios nuevamente
8
9     foreach ($arrayUsuarios as $usuario) {
10         if (isset($_POST[$usuario['id']])) {
11             borrarUsuario($usuario['id']);
12             echo "<h1 id='avisoUserBorrado'>Eliminando usuario de la base de datos. Serás redirigido en unos segundos.</h1>";
13             echo "<br>";
14             echo "<div class= 'spinner'>
15                 <div></div>
16                 <div></div>
17                 <div></div>
18                 <div></div>
19                 <div></div>
20                 <div></div>
21             </div>";
22             header("refresh:2;url=".$_SERVER["PHP_SELF"]."?ruta=principal");
23             exit;
24         }
25     }
26 }
27 ?>
28
29 <section>
30 <form id="formUsuarios" action="<?php echo $_SERVER["PHP_SELF"] . "?ruta=gestionUsuarios";?>" method="post">
31     <?php
32     $arrayUsuarios = RecuperaTotalUsuarios();
33     ?>
34     <table id='tablaUsuarios'>
35         <tr>
36             <td>Nombre</td>
37             <td>Primer apellido</td>
38             <td>Segundo apellido</td>
39             <td>Nombre de usuario</td>
40             <td>Correo Electrónico</td>
41             <td>Fecha de registro</td>
42             <td>Borrar usuario</td>
43         </tr>
44         <?php foreach ($arrayUsuarios as $usuario) {
45             if ($usuario["borrado"] == 0 && $usuario["rol"] == "usuario") { ?>
46                 <tr>
47                     <td><?php echo $usuario["nombre"]; ?></td>
48                     <td><?php echo $usuario["apellido1"]; ?></td>
49                     <td><?php echo $usuario["apellido2"]; ?></td>
50                     <td><?php echo $usuario["usuario"]; ?></td>
51                     <td><?php echo $usuario["correo"]; ?></td>
52                     <td><?php echo $usuario["fecha_registro"]; ?></td>
53                     <td>
54                         <input type="submit" name="<?php echo $usuario['id']; ?>" value="Dar de baja">
55                     </td>
56                 </tr>
57             <?php } ?>
58         </table>
59         <a href="<?php echo $_SERVER["PHP_SELF"] . "?ruta=principal";?>" id="btnVolverUsu">Volver</a>
60     </form>
61 </section>

```

Justo después de realizar el borrado, si ha sido exitoso, se muestra un mensaje de confirmación y se utiliza un pequeño efecto visual de "spinner" antes de hacer la redirección a la página principal, tal y como sucede cuando se alquila un libro.

```

28
29 <section>
30 <form id="formUsuarios" action="<?php echo $_SERVER["PHP_SELF"] . "?ruta=gestionUsuarios";?>" method="post">
31     <?php
32     $arrayUsuarios = RecuperaTotalUsuarios();
33     ?>
34     <table id='tablaUsuarios'>
35         <tr>
36             <td>Nombre</td>
37             <td>Primer apellido</td>
38             <td>Segundo apellido</td>
39             <td>Nombre de usuario</td>
40             <td>Correo Electrónico</td>
41             <td>Fecha de registro</td>
42             <td>Borrar usuario</td>
43         </tr>
44         <?php foreach ($arrayUsuarios as $usuario) {
45             if ($usuario["borrado"] == 0 && $usuario["rol"] == "usuario") { ?>
46                 <tr>
47                     <td><?php echo $usuario["nombre"]; ?></td>
48                     <td><?php echo $usuario["apellido1"]; ?></td>
49                     <td><?php echo $usuario["apellido2"]; ?></td>
50                     <td><?php echo $usuario["usuario"]; ?></td>
51                     <td><?php echo $usuario["correo"]; ?></td>
52                     <td><?php echo $usuario["fecha_registro"]; ?></td>
53                     <td>
54                         <input type="submit" name="<?php echo $usuario['id']; ?>" value="Dar de baja">
55                     </td>
56                 </tr>
57             <?php } ?>
58         </table>
59         <a href="<?php echo $_SERVER["PHP_SELF"] . "?ruta=principal";?>" id="btnVolverUsu">Volver</a>
60 </form>
61 </section>

```

La lógica para mostrar la tabla de usuarios registrados es exactamente la misma que se utiliza en anteriores apartados, llamando a la función `RecuperaTotalUsuarios()`, como vemos a continuación:

```

2 references
123 function RecuperaTotalUsuarios():array|bool {
124
125     $conexion = EstablecerConexion();
126     $arrayUsuarios = array(); // Array vacío para almacenar resultados
127
128     //en la variable $totalUsuarios guardamos todo lo que contiene la tabla libros
129     if ($totalUsuarios = mysqli_query($conexion,"SELECT * FROM usuarios")) {
130         // Recorremos $totalUsuarios y vamos guardando cada resultado como array asociativo, que vamos añadiendo
131         //al array original
132         while ($fila = mysqli_fetch_assoc($totalUsuarios)) {
133             $arrayUsuarios[] = $fila;
134         }
135         // Liberamos el conjunto de resultados
136         mysqli_free_result($totalUsuarios);
137         return $arrayUsuarios;
138     } else {
139         // Si la consulta falla
140         return false;
141     }
142 }
143 //Cerramos la conexión con la base de datos
144 CerrarConexion($conexion);
145
146 }
147

```

Más interesante resulta la funcionalidad del botón para dar de baja a los usuarios que, como vemos, se sirve de la función `borrarUsuario()`, que realiza lo siguiente:

- Conexión a la base de datos:

Inicia una conexión a la base de datos utilizando la función `EstablecerConexion()`.

- Actualización de la columna borrado en la tabla usuarios:

Ejecuta una consulta SQL para actualizar la columna borrado a 1 en la tabla usuarios para el usuario específico identificado por el ID proporcionado.

- Manejo de errores en la actualización de la tabla usuarios:

Verifica si hubo errores en la actualización de la tabla usuarios. Si hay un error, muestra un mensaje de error, cierra la conexión y devuelve false.

- Actualización de la columna fin_prestamo en la tabla prestamos:

Ejecuta una consulta SQL para actualizar la columna fin_prestamo a NULL en la tabla prestamos para todos los libros alquilados por el usuario identificado por el ID proporcionado.

- Manejo de errores en la actualización de la tabla prestamos:

Verifica si hubo errores en la actualización de la tabla prestamos. Si hay un error, muestra un mensaje de error, cierra la conexión y devuelve false.

- Obtención de IDs de libros devueltos por el usuario:

Ejecuta una consulta SQL para obtener los IDs de los libros que el usuario ha devuelto y almacena estos IDs en un array llamado \$arrayIdLibrosDevueltos.

- Actualización de la columna alquilado en la tabla libros:

Utiliza un bucle foreach para recorrer el array de IDs de libros devueltos y ejecuta consultas SQL para actualizar la columna alquilado a 0 en la tabla libros para cada libro correspondiente al ID.

- Cierre de la conexión a la base de datos:

Devuelve true indicando que la operación fue exitosa y cierra la conexión con la base de datos.

```

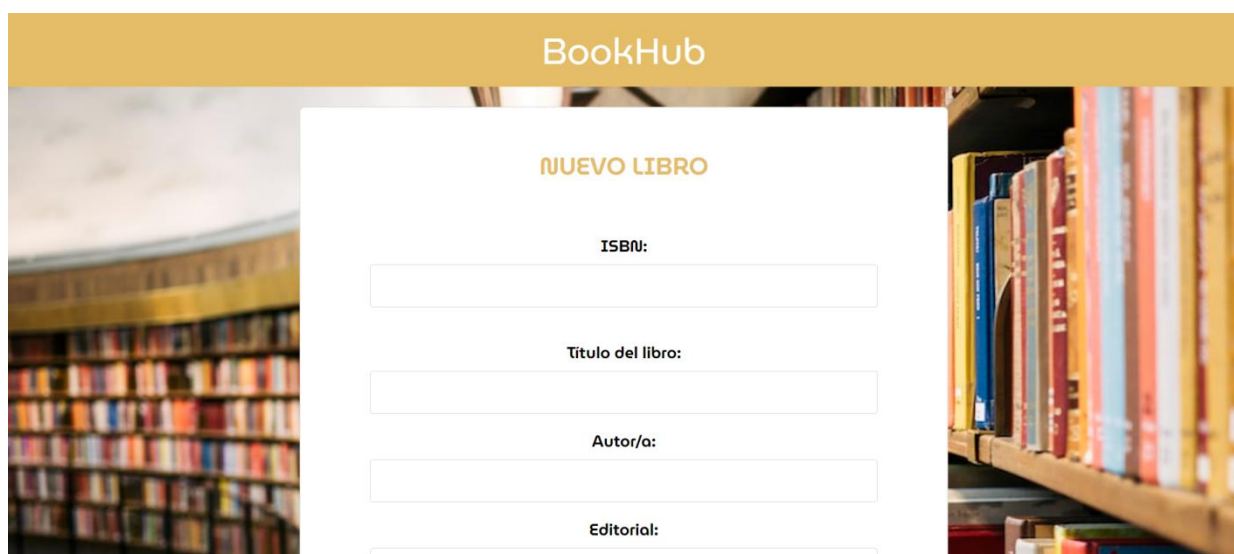
77 function borrarUsuario($idUser): bool {
78
79     $conexion = EstablecerConexion();
80
81     //Actualizamos en la tabla USUARIOS poniendo a TRUE la columna borrado del usuario correspondiente
82     $borradoUsuario = mysqli_query($conexion, "UPDATE usuarios SET borrado = 1 WHERE id = '$idUser'");
83
84     if (!$borradoUsuario) {
85         echo "Error al actualizar la tabla de usuarios: " . mysqli_error($conexion);
86         CerrarConexion($conexion);
87         return false;
88     }
89
90     //Actualizamos la tabla PRESTAMOS poniendo a NULL la fecha de los libros que tuviese alquilados ese usuario
91     $borradoAlquileres = mysqli_query($conexion, "UPDATE prestamos SET fin_prestamo = NULL WHERE id_usuario = '$idUser'");
92
93     if (!$borradoAlquileres) {
94         echo "Error al actualizar la tabla de usuarios: " . mysqli_error($conexion);
95         CerrarConexion($conexion);
96         return false;
97     }
98
99     //Seleccionamos desde la tabla PRESTAMOS los id de los libros que tuviese alquilados el usuario borrado y los guardamos en un array
100    $arrayIdLibrosDevueltos = array ();
101
102    if ($idsLibrosDevueltos = mysqli_query($conexion, "SELECT id_libro FROM prestamos WHERE id_usuario = '$idUser'")) {
103        // Recorremos $idsLibrosDevueltos y vamos guardando cada resultado (id) en el array vacío $arrayIdLibrosDevueltos
104        while ($fila = mysqli_fetch_array($idsLibrosDevueltos)) {
105            $arrayIdLibrosDevueltos[] = $fila['id_libro'];
106        }
107    }
108
109    //Recorremos $arrayIdLibrosDevueltos y por cada id de los libros actualizamos la tabla LIBROS poniendo a FALSE la columna alquilado
110    foreach ($arrayIdLibrosDevueltos as $id) {
111        mysqli_query($conexion, "UPDATE libros SET alquilado=0 WHERE id = '$id'");
112    }
113
114    return true;
115    CerrarConexion($conexion);
116 }
117

```

De esta manera, cuando un usuario es dado de baja en la aplicación, se asegura también la anulación del alquiler de los libros que tuviese alquilados y que estos estén de nuevo disponibles en el catálogo para el resto de usuarios.

Como vemos, al usuario que se le da de baja en la aplicación no se le elimina por completo de la base de datos, puesto que esta no es una práctica recomendable. Marcar el estado del usuario como "borrado" en lugar de eliminarlo por completo proporciona flexibilidad, conserva la integridad referencial, permite la auditoría, facilita la recuperación de datos y puede ser necesario para cumplir con ciertos requisitos legales o normativos.

3.4.2. Agregar libro nuevo



BookHub

NUEVO LIBRO

ISBN:

Título del libro:

Autor/a:

Editorial:

En la pestaña “Agregar libro nuevo” el administrador puede añadir a la base de datos los libros nuevos que lleguen a la biblioteca con una sencilla interfaz. Su funcionamiento es el mismo que el de la pestaña de registro de usuario.

```
estructuraWeb > agregarLibros.inc.php
1  <?php
2
3  if (isset($_POST["btnAñadirLibro"])) {
4      if ((isset($_POST["isbn"]) && !empty($_POST["isbn"])) && (isset($_POST["titulo"])
5          && !empty($_POST["titulo"])) && (isset($_POST["autor"])
6          && !empty($_POST["autor"])) && (isset($_POST["editorial"]) && !empty($_POST["editorial"]))) {
7          RegistroLibro($_POST["isbn"], $_POST["titulo"], $_POST["autor"], $_POST["editorial"]);
8          echo "<h1 id='avisoAñadirLibro'>Añadiendo libro a la base de datos. Serás redirigido en unos segundos.</h1>";
9          echo "<br>";
10         echo "<div class='spinner'>
11             <div></div>
12             <div></div>
13             <div></div>
14             <div></div>
15             <div></div>
16             <div></div>
17         </div>";
18         header("refresh:2;url=".$_SERVER["PHP_SELF"]."?ruta=principal");
19         exit;
20     } else if (empty($_POST["nombre"]) || empty($_POST["apellido1"]) || empty($_POST["apellido2"])
21         || empty($_POST["username"]) || empty($_POST["correo"]) || empty($_POST["password"])) {
22         $errorDatos = true;
23     }
24 }
25 ?>
26
27
```

```

28 <section id="sectionAgregarLibro">
29
30 <form id="formularioAgregarLibro" action="<?php echo $_SERVER["PHP_SELF"] . "?ruta=agregarLibros"; ?>" method="post">
31
32 <?php if (isset ($errorDatos) && $errorDatos == true) { ?>
33 <p id="txtError"> Faltan datos por introducir </p>
34 <?php
35 }
36 ?>
37
38 <h1 id="tituloLibro"> NUEVO LIBRO </h1>
39 <br>
40 <label for="isbn"> ISBN: </label>
41 <input type="text" name="isbn" value=""><br>
42 <br>
43
44 <label for="titulo"> Título del libro: </label>
45 <input type="text" name="titulo">
46 <br>
47
48 <label for="autor"> Autor/a: </label>
49 <input type="text" name="autor">
50 <br>
51
52 <label for="editorial"> Editorial: </label>
53 <input type="text" name="editorial">
54 <br>
55 <br>
56 <input type="submit" id="btnRegistro" name="btnAnadirLibro" value="Añadir Libro">
57 <br>
58 <br>
59 <a href="<?php echo $_SERVER["PHP_SELF"] . "?ruta=principal"; ?>" id="btnVolverAgregarLibros">Volver</a>
60
61 </form>
62
63 </section>

```

Como vemos, de nuevo a diferencia de las vistas “Login” y “Registro”, el formulario html y la lógica php que recoge los datos que le envía se sitúan en el mismo fichero, como ya sucedía en las otras pestañas.

Es la función RegistroLibro() la que realiza la inserción del mismo en la base de datos:

```

function RegistroLibro($isbn,$titulo,$autor,$editorial) {

    $conexion = EstablecerConexion();

    if (mysqli_query($conexion, "INSERT INTO libros (`isbn`,`titulo`,`autor`,`editorial`,`alquilado`)
VALUES ('".$isbn."','".$titulo."','".$autor."','".$editorial."',0);") === TRUE) {
        // La consulta se ejecutó con éxito, no mostramos nada
    } else {
        echo "Error al insertar el libro: " . mysqli_error($conexion);
    }

    CerrarConexion($conexion);
}

```

Justo después de realizar la inserción, si ha sido exitosa, se muestra un mensaje de confirmación y se utiliza un pequeño efecto visual de "spinner" antes de hacer la redirección a la página principal, tal y como sucede cuando se alquila un libro o se borra un usuario de la aplicación.

3.4.3. Gestión de préstamos

BookHub					
ID Usuario	ID Libro	Nombre Libro	Fecha fin de préstamo	Devolución	Volver
38	27	El Señor de los Anillos: Las Dos Torres	2023-11-30	Devolver	
38	23	El día que se perdió la cordura	2023-11-30	Devolver	
40	25	Watchmen	2023-11-30	Devolver	

```
estructuraWeb > prestamos.inc.php
1  <?php
2  include_once "../BBDD/bibliotecaFunciones.php";
3
4  // Verificar si se ha enviado el formulario antes de mostrar cualquier contenido
5  //Funcionalidad del botón devolver
6  if ($_SERVER["REQUEST_METHOD"] === "POST") {
7      $arrayDatos = RecuperarAlquileres(); // Cargar los prestamos nuevamente
8
9      foreach ($arrayDatos as $alquiler) {
10         if (isset($_POST[$alquiler['id_libro']])) {
11             devolver($alquiler['id_libro']);
12             echo "<h1 id='avisoDevuelto'>Devolviendo libro a la biblioteca. Serás redirigido en unos segundos.</h1>";
13             echo "<br>";
14             echo "<div class= 'spinner'>
15                 <div></div>
16                 <div></div>
17                 <div></div>
18                 <div></div>
19                 <div></div>
20                 <div></div>
21             </div>";
22             header("refresh:2;url=".$_SERVER["PHP_SELF"]."?ruta=principal");
23             exit;
24         }
25     }
26 }
27 ?>
28
```

```
29
30 <section id="sectionPrestamos">
31 <form id="formPrestamos" action="<?php echo $_SERVER["PHP_SELF"] . "?ruta=prestamos";?>" method="post">
32 <?php
33 $arrayDatos = RecuperarAlquileres();
34 if ($arrayDatos===[]){
35     echo "<h1 id='txtNoPrestamos'> En estos momentos hay ningún libro prestado. </h1>";
36 } else {
37     ?>
38     <table id='tablaPrestamos'>
39 <tr>
40 <td>ID Usuario</td>
41 <td>ID Libro </td>
42 <td>Nombre Libro </td>
43 <td>Fecha fin de préstamo</td>
44 <td>Devolución</td>
45 </tr>
46 <?php foreach ($arrayDatos as $alquiler) { ?>
47 <tr>
48 <td><?php echo $alquiler['id_usuario']; ?></td>
49 <td><?php echo $alquiler['id_libro']; ?></td>
50 <td><?php echo $alquiler['nombre_libro']; ?></td>
51 <td><?php echo $alquiler['fin_prestamo']; ?></td>
52 <td><input type="submit" name="<?php echo $alquiler['id_libro']; ?>" value="Devolver"></td>
53 </tr>
54 <?php }?>
55 </table>
56 <a href="<?php echo $_SERVER["PHP_SELF"] . "?ruta=principal";?>" id="btnVolverPres">Volver</a>
57 </form>
58
59 </section>
```

En la pestaña “Gestión de préstamos” el administrador puede ver los libros que están actualmente prestados y a qué usuarios se les han prestado, así como la fecha en la que deben efectuar la devolución. Además, pueden gestionar la devolución del libro cuando los usuarios lo devuelven a la biblioteca. Solo tienen que pulsar el botón “Devolver”, disponible en esta vista, y el libro volverá a estar disponible para su alquiler en la aplicación.

De nuevo, el formulario y la lógica php quedan integrados en el mismo fichero, y tenemos de nuevo mensaje con el spinner mostrados antes de redirigir a la pantalla principal cuando la devolución de un libro se realiza satisfactoriamente.

La función devolver() integra la lógica para devolver libros, y es la siguiente:

```
3 function devolver($idLibro):bool {
4
5     $conexion = EstablecerConexion();
6
7     // Actualizar el campo "alquilado" en la tabla libros
8     $resultLibro = mysqli_query($conexion,"UPDATE libros SET alquilado = 0 WHERE id = '$idLibro';" );
9
10    if (!$resultLibro) {
11        echo "Error al actualizar el estado del libro: " . mysqli_error($conexion);
12        CerrarConexion($conexion);
13        return false;
14    }
15
16    // Actualizar la tabla préstamos
17    $resultPrestamos = mysqli_query($conexion, "UPDATE prestamos SET fin_prestamo = NULL WHERE id_libro = '$idLibro';");
18
19    if (!$resultPrestamos) {
20        echo "Error al actualizar la tabla de préstamos: " . mysqli_error($conexion);
21        CerrarConexion($conexion);
22        return false;
23    }
24
25    return true;
26    CerrarConexion($conexion);
27 }
28
```

- Establecimiento de la conexión a la base de datos:

Inicia una conexión a la base de datos utilizando la función EstablecerConexion().

- Actualización del campo "alquilado" en la tabla "libros":

Ejecuta una consulta SQL para actualizar el campo "alquilado" a 0 en la tabla "libros" para el libro específico identificado por el ID proporcionado.

- Manejo de errores en la actualización de la tabla "libros":

Verifica si hubo errores en la actualización de la tabla "libros". Si hay un error, muestra un mensaje de error, cierra la conexión y devuelve false.

- Actualización de la tabla "préstamos":

Ejecuta una consulta SQL para actualizar la columna "fin_prestamo" a NULL en la tabla "préstamos" para el libro específico identificado por el ID proporcionado. Esto indica que el libro ha sido devuelto y no tiene una fecha de devolución asociada.

- Manejo de errores en la actualización de la tabla "préstamos":

Verifica si hubo errores en la actualización de la tabla "préstamos". Si hay un error, muestra un mensaje de error, cierra la conexión y devuelve false.

- Cierre de la conexión a la base de datos:

Llamada a la función CerrarConexion() para cerrar la conexión con la base de datos.

3.5. Seguridad y tratamiento de errores

3.5.1. Logout

Cuando un usuario desea cerrar sesión, dispone de dos botones para hacerlo: uno situado en su ventana principal, desde la cual puede acceder al resto de funcionalidades que le proporciona la aplicación, y otro situado en la ventana "Mi perfil", en la que puede revisar los datos de su cuenta. En ambos casos, su funcionalidad es la misma.

```
<a href="<?php echo $_SERVER["PHP_SELF"]."?ruta=logout";?>" class="btnePerfil">Cerrar Sesión</a>
```

Este botón es, simplemente, un elemento html enlace "<a>" que hace una redirección a "logout.inc.php", archivo que incluye el siguiente código php:

```
1 <?php
2 //eliminamos las variables de sesión
3 session_unset();
4 //eliminamos la sesión
5 session_destroy();
6 //redireccionamos a index
7 header("Location:".$_SERVER["PHP_SELF"]);
8 ?>
```

Como vemos, este sencillo código php realiza tres acciones:

- session_unset(): Elimina las dos variables de sesión (nombre de usuario y rol), pero no destruye la sesión en sí.
- session_destroy(): Destruye las sesiones, con lo que elimina toda la información asociada con las ellas en el servidor.
- Por último redirige al usuario al index de nuevo, donde verá la pantalla inicial de login porque ya no existe la sesión.

Al hacer la redirección a esta ruta, el index realiza la inclusión de este código con la directiva "include_once" y lo ejecuta, realizando así la destrucción de la información guardada del usuario.

3.5.2. Tratamiento de inserción de rutas indebidas

Es importante tratar los casos en los que un usuario malicioso intente acceder a partes de la aplicación a las que no debería tener acceso mediante peticiones al servidor a través de URL insertadas manualmente. También lo es manejar el error en caso de que el usuario introduzca una ruta equivocada.

Las rutas equivocadas son gestionadas por la aplicación de la siguiente manera:

```
86 //Si no existen sesión ni ruta...
87 if (!isset($_SESSION ["usuario"]) && !isset($_GET["ruta"])){
88     include_once "estructuraWeb/menuSuperior.inc.php";
89     include_once "estructuraWeb/login.inc.php";
90 }
91
92 //Si no existe la sesión pero existe la ruta...
93 if (!isset($_SESSION ["usuario"]) && isset($_GET["ruta"])) {
94
95     //Caso de introducir una ruta inventada
96     if ($_GET["ruta"] != "agregarLibros" && $_GET["ruta"] != "alquileres" &&
97         $_GET["ruta"] != "cabecera" && $_GET["ruta"] != "catalogo" &&
98         $_GET["ruta"] != "gestionUsuarios" && $_GET["ruta"] != "info" &&
99         $_GET["ruta"] != "login" && $_GET["ruta"] != "logout" &&
100         $_GET["ruta"] != "perfil" && $_GET["ruta"] != "prestamos" &&
101         $_GET["ruta"] != "principal" && $_GET["ruta"] != "registro") {
102
103         include_once "estructuraWeb/rutaError.inc.php";
104     }
105 }
106
107
108 //Si existe la sesión
109 if (isset($_SESSION ["usuario"])) {
110
111     //Si se introduce una ruta inventada...
112     if ($_GET["ruta"] != "agregarLibros" && $_GET["ruta"] != "alquileres" &&
113         $_GET["ruta"] != "cabecera" && $_GET["ruta"] != "catalogo" &&
114         $_GET["ruta"] != "gestionUsuarios" && $_GET["ruta"] != "info" &&
115         $_GET["ruta"] != "login" && $_GET["ruta"] != "logout" &&
116         $_GET["ruta"] != "perfil" && $_GET["ruta"] != "prestamos" &&
117         $_GET["ruta"] != "principal" && $_GET["ruta"] != "registro") {
118
119         include_once "estructuraWeb/rutaError.inc.php";
120     }
121 }
122
```

Tanto si existe la sesión como si no, si el usuario introduce una ruta que no corresponde con ninguna de las que incluye esta aplicación, se realiza un “include_once” de “rutaError.inc.php”. Este archivo contiene un sencillo código php que realiza la siguiente función:

- Se imprime un mensaje de error en formato HTML (<h1>) indicando que la dirección no existe y que el usuario será redirigido en unos segundos, acompañado de un efecto visual spinner que indica que la página está cargando.

- Se utiliza una estructura condicional:

- Si no existe una sesión de usuario (!isset(\$_SESSION["usuario"])), se programa una redirección a la página principal con el login después de 2 segundos.
- Si existe una sesión de usuario, se redirige a la página principal del usuario (\$_SERVER["PHP_SELF"] . "?ruta=principal") después de 2 segundos.

```

estructuraWeb > rutaError.inc.php
1  <?php
2  echo "<h1 id='avisoAlquilado'>Error 404. Esta dirección no existe.Serás redirigido en unos segundos.</h1>";
3  echo "<br>";
4  echo "<div class= 'spinner'>
5      <div></div>
6      <div></div>
7      <div></div>
8      <div></div>
9      <div></div>
10     <div></div>
11     </div>";
12
13  if (isset($_SESSION ["usuario"])) {
14      header("refresh:2;url=".$_SERVER["PHP_SELF"]);
15  } else {
16      header("refresh:2;url=" . $_SERVER["PHP_SELF"] . "?ruta=principal");
17  }
18  ?>

```

Por otra parte, la inserción maliciosa de rutas a mano se trata de la siguiente forma:

```

92  //Si no existe la sesión pero existe la ruta...
93  if (!isset($_SESSION ["usuario"]) && isset($_GET["ruta"])) {
94
95      //Caso de introducir una ruta inventada
96      if ($_GET["ruta"] != "agregarLibros" && $_GET["ruta"] != "alquileres" &&
97          $_GET["ruta"] != "cabecera" && $_GET["ruta"] != "catalogo" &&
98          $_GET["ruta"] != "gestionUsuarios" && $_GET["ruta"] != "info" &&
99          $_GET["ruta"] != "login" && $_GET["ruta"] != "logout" &&
100         $_GET["ruta"] != "perfil" && $_GET["ruta"] != "prestamos" &&
101         $_GET["ruta"] != "principal" && $_GET["ruta"] != "registro") {
102
103         include_once "estructuraWeb/rutaError.inc.php";
104     }
105
106     if ($_GET["ruta"] == "principal" || $_GET["ruta"] == "perfil" ||
107         $_GET["ruta"] == "catalogo" || $_GET["ruta"] == "alquileres" ||
108         $_GET["ruta"] == "prestamos" || $_GET["ruta"] == "gestionUsuarios") {
109
110         ?> <script>alert("Acceso restringido.");</script> <?php
111         header("refresh:0.1;url=" . $_SERVER["PHP_SELF"] . "?ruta=login");
112     }

```

En el caso de no existir la sesión, si introduce cualquiera de las rutas de la aplicación que condicen con partes de esta accesibles solo para usuarios previamente registrados, se ejecuta un script que le avisa al usuario de que tiene el acceso restringido a esa ruta.

Por otro lado, si la sesión existe:

```

143 //tratamos el acceso indebido a rutas
144
145 //si el usuario es admin...
146
147 if ($_SESSION["rol"]=="admin") {
148
149     if(isset($_GET["ruta"]) && $_GET["ruta"] == "prestamos") {
150
151         include_once "estructuraWeb/prestamos.inc.php";
152
153     }
154
155     else if (isset($_GET["ruta"]) && $_GET["ruta"] == "perfil") {
156
157         include_once "estructuraWeb/perfil.inc.php";
158
159     }
160
161     else if (isset($_GET["ruta"]) && $_GET["ruta"] == "logout") {
162
163         include_once "estructuraWeb/logout.inc.php";
164
165     }
166
167     else if (isset($_GET["ruta"]) && $_GET["ruta"] == "gestionUsuarios") {
168
169         include_once "estructuraWeb/gestionUsuarios.inc.php";
170
171     }
172
173     else if (isset($_GET["ruta"]) && $_GET["ruta"] == "agregarLibros") {
174
175         include_once "estructuraWeb/agregarLibros.inc.php";
176
177     }
178
179     else if (isset($_GET["ruta"]) && $_GET["ruta"] == "principal") {
180
181         include_once "estructuraWeb/principal.inc.php";
182
183     }
184
185     else if (isset($_GET["ruta"]) && $_GET["ruta"] == "catalogo") {
186         ?> <script>alert("Acceso restringido.");</script> <?php
187         header("refresh:0.1;url=" . $_SERVER["PHP_SELF"] . "?ruta=principal");
188     }
189
190     else if (isset($_GET["ruta"]) && $_GET["ruta"] == "alquileres") {
191         ?> <script>alert("Acceso restringido.");</script> <?php
192         header("refresh:0.1;url=" . $_SERVER["PHP_SELF"] . "?ruta=principal");
193     }
194 }
195

```

Si el rol del usuario es administrador, solo puede introducir las rutas a las que tiene acceso el administrador. En caso de introducir alguna de las rutas del usuario estándar, se ejecuta el script de aviso.


```

195
196 //si el usuario es estándar...
197
198 if ($_SESSION["rol"]=="usuario") {
199
200     if (isset($_GET["ruta"]) && $_GET["ruta"] == "perfil") {
201
202         include_once "estructuraWeb/perfil.inc.php";
203
204     }
205
206     else if (isset($_GET["ruta"]) && $_GET["ruta"] == "logout") {
207
208         include_once "estructuraWeb/logout.inc.php";
209
210     }
211
212     else if (isset($_GET["ruta"]) && $_GET["ruta"] == "principal") {
213
214         include_once "estructuraWeb/principal.inc.php";
215
216     }
217
218     else if (isset($_GET["ruta"]) && $_GET["ruta"] == "catalogo") {
219
220         include_once "estructuraWeb/catalogo.inc.php";
221
222     }
223
224     else if (isset($_GET["ruta"]) && $_GET["ruta"] == "alquileres") {
225
226         include_once "estructuraWeb/alquileres.inc.php";
227
228     }
229
230     else if (isset($_GET["ruta"]) && $_GET["ruta"] == "gestionUsuarios") {
231
232         ?> <script>alert("Acceso restringido.");</script> <?php
233         | header("refresh:0.1;url=" . $_SERVER["PHP_SELF"] . "?ruta=principal");
234         }
235
236     else if (isset($_GET["ruta"]) && $_GET["ruta"] == "prestamos") {
237
238         ?> <script>alert("Acceso restringido.");</script> <?php
239         | header("refresh:0.1;url=" . $_SERVER["PHP_SELF"] . "?ruta=principal");
240         }
241
242     else if (isset($_GET["ruta"]) && $_GET["ruta"] == "agregarLibros") {
243
244         ?> <script>alert("Acceso restringido.");</script> <?php
245         | header("refresh:0.1;url=" . $_SERVER["PHP_SELF"] . "?ruta=principal");
246         }
247
248
249     }
250

```

El programa sigue la misma lógica con el usuario estándar.

3.5.3. Filtrado de email y contraseña

Cuando un usuario nuevo introduce sus datos para registrarse en la aplicación, los campos de contraseña y correo tienen doble comprobación. Por un lado, la del propio HTML en el cliente, al declarar los esos campos como inputs de tipo “password” y “email”, respectivamente. Esto implica que, en el caso del campo contraseña, los caracteres que se van introduciendo se sustituyan automáticamente por símbolos cuando un usuario la está introduciendo, para que el texto de la misma no sea visible directamente. En cuanto al correo electrónico, al ser un campo input del tipo email, obliga a que el texto introducido por el usuario cumpla con el formato de correo electrónico.

Por la parte del servidor, en el PHP, estos campos también son comprobados previamente antes de realizar el registro del usuario en la base de datos. Estas comprobaciones se realizan con las funciones comprobarCorreo() y comprobarContraseña(), las cuales vemos a continuación:

```
46 function comprobarCorreo($correo):bool {
47     //verifica si el correo introducido tiene un formato válido
48     if(filter_var($correo, FILTER_VALIDATE_EMAIL)){
49         return true;
50     } else {
51         return false;
52     }
53 }
54
```

```
1 function comprobarContraseña($contrasena): bool {
2     // Verifica si la contraseña tiene entre 8 y 16 caracteres
3     $longitudValida = strlen($contrasena) >= 8 && strlen($contrasena) <= 16;
4
5     // Verifica si hay al menos una mayúscula y un número en la contraseña
6     $tieneMayuscula = preg_match('/[A-Z]/', $contrasena) === 1;
7     $tieneNumero = preg_match('/[0-9]/', $contrasena) === 1;
8
9     // Retorna true si la contraseña cumple con los requisitos, false en caso contrario
10    if($longitudValida && $tieneMayuscula && $tieneNumero) {
11        return true;
12    } else {
13        return false;
14    }
15 }
16
```

Como vemos, la función comprobarCorreo() se sirve a su vez de la función de php “FILTER_VALIDATE_EMAIL”, la cual nos sirve para comprobar que una cadena corresponde, efectivamente, con el formato de una cadena de tipo correo electrónico. Si es así devuelve “true”, y si no, “false”.

La función `comprobarContraseña()` se sirve, por su parte, de expresiones regulares. En este caso, se imponen tres condiciones: que la longitud de la contraseña se encuentre entre los 8 y 16 caracteres, que contenga al menos una letra mayúscula, y que contenga al menos un carácter numérico. Si estas tres condiciones se cumplen, la función devolverá "true". En caso contrario, devolverá "false".

```
function RegistroUsuario($nombre,$apellido1,$apellido2,$usuario,$password,$correo) {  
    $conexion = EstablecerConexion();  
  
    if (comprobarCorreo($correo) && comprobarContraseña($password)) {  
        $passwordHash = password_hash($password, PASSWORD_BCRYPT);  
  
        try {  
            mysqli_query($conexion, "INSERT INTO usuarios (`nombre`, `apellido1`, `apellido2`, `usuario`, `password`,  
            `correo`, `fecha_registro`, `rol`) VALUES ('".$nombre."', '".$apellido1."', '".$apellido2."', '".$usuario."',  
            '".$passwordHash."', '".$correo."', NOW(), 'usuario');");  
            return true;  
        } catch (Exception $e) {  
            echo $e->getMessage();  
            return false;  
        }  
    }  
  
    CerrarConexion($conexion);  
}
```

En la función que realiza el registro del usuario nuevo en la base de datos, como vimos, se comprueba previamente con las dos funciones anteriores que los datos introducidos son válidos. Si ambas funciones devuelven "true", se procede a la inserción del nuevo usuario en la base de datos con los datos que se introdujeron en el formulario.

3.5.5. Cifrado de contraseñas

La encriptación de contraseñas se lleva a cabo utilizando la función `password_hash()` de PHP con el algoritmo BCRYPT, como ya vimos en las primeras páginas en el documento y en el punto anterior. Es importante destacar que, siguiendo las mejores prácticas de seguridad, las contraseñas de los usuarios no se almacenan en texto claro en la base de datos.

En lugar de eso, lo que se guarda es una cadena resultante de la encriptación de la contraseña que el usuario proporciona al registrarse en la aplicación. Este enfoque garantiza un nivel adicional de seguridad, ya que incluso si se produjera un acceso no autorizado a la base de datos, las contraseñas permanecerían protegidas y no estarían expuestas en su forma original. Así, cuando realizamos el login, y como ya vimos también en las primeras secciones de la documentación, la función `comprobarLogin()` debe hacer el proceso inverso sirviéndose de la función de PHP `password_verify()`, cual realiza la desencriptación de la cadena almacenada en

la base de datos compara el resultado con el texto introducido por el usuario en la base de datos:

```
1 reference
367 v function ComprobarLogin ($usuario, $passwd):bool {
368
369     $conexion = EstablecerConexion();
370
371     $consulta = mysqli_query($conexion, "SELECT password FROM usuarios WHERE usuario='$usuario' AND borrado=0");
372
373 v     if ($consulta && mysqli_num_rows($consulta) == 1) {
374         $fila = mysqli_fetch_assoc($consulta);
375         $passwordAlmacenado = $fila['password'];
376
377 v         if (password_verify($passwd, $passwordAlmacenado)) {
378             CerrarConexion($conexion);
379             return true;
380         }
381     }
382
383     CerrarConexion($conexion);
384     return false;
385 }
386
```

4. CONCLUSIÓN

Para finalizar este documento, y a modo de pequeña conclusión, el proyecto BookHub se destaca como una experiencia enriquecedora que ha permitido la aplicación práctica de conocimientos en HTML, CSS y PHP para desarrollar una plataforma funcional de gestión de bibliotecas. A través de este proceso, se logró diseñar una interfaz atractiva y fácil de usar, proporcionando a los usuarios una herramienta eficiente para gestionar el catálogo de libros.

La incorporación de PHP como lenguaje de backend ha demostrado ser crucial para la manipulación dinámica de datos. Gracias a ello, se ha logrado almacenar y recuperar información de manera efectiva, así como ofrecer a usuarios y administradores la capacidad de gestionar libros y usuarios de manera fácil y rápida.

Por último, se han fortalecido las habilidades en el diseño de bases de datos, asegurando una estructura que permita la integridad y disponibilidad de la información, lo que ha favorecido a su vez a la comprensión de la interacción cliente-servidor, vital para el funcionamiento fluido de BookHub y cualquier aplicación web.