**Introducción**

Nuestro proyecto se ha basado en crear una biblioteca online, en la era digital en la que nos encontramos, este método de transmisión de información se ha convertido en una herramienta invaluable para acceder y difundir conocimiento de manera ágil, y como objetivo, de una manera gratuita. En este proyecto, nos hemos propuesto crear una biblioteca virtual que cumpla con esa misión fundamental: extender el conocimiento a todos, sin barreras económicas o geográficas.

En primer lugar, es importante entender qué queremos conseguir con esta web. Una biblioteca online es un repositorio en línea que almacena una amplia variedad de recursos, tales como libros, artículos, revistas y documentos académicos, que se encuentran disponibles para su consulta y descarga a través de internet. Esta modalidad de biblioteca nos permite superar las limitaciones físicas de una biblioteca tradicional, ya que se puede acceder a ella desde cualquier lugar y en cualquier momento.

La elección de desarrollar una biblioteca virtual como proyecto surge de nuestra convicción de que el acceso al conocimiento debe ser un derecho universal. Creemos firmemente que todas las personas, independientemente de su origen, situación económica o ubicación geográfica, deben tener la oportunidad de enriquecerse intelectualmente y acceder a recursos educativos de calidad de forma gratuita.

Nuestra misión es derribar las barreras que limitan el acceso al conocimiento. Queremos brindar una plataforma que permita a estudiantes, académicos, investigadores y entusiastas de diversas áreas encontrar información relevante y actualizada, impulsando así el aprendizaje continuo y el crecimiento personal

**Objetivos**

El objetivo principal de este proyecto es proporcionar un acceso fácil, seguro y gratuito a una amplia variedad de libros a través de un repositorio en línea. Nos proponemos crear un espacio virtual donde las personas puedan descargar y leer libros de manera sencilla, eliminando las barreras físicas y económicas que suelen limitar el acceso a la literatura.

Además de ofrecer una amplia selección de títulos, otro objetivo fundamental es fomentar la participación activa de los usuarios. Queremos que este proyecto no solo sea un lugar para descargar y leer libros, sino también un espacio donde los usuarios puedan compartir y expresar sus opiniones sobre los títulos que han leído. Consideramos que el intercambio de ideas y perspectivas enriquece la experiencia de lectura y promueve un diálogo constructivo entre los usuarios.

Con este propósito, hemos incorporado funcionalidades en la plataforma que permiten a los usuarios calificar y dejar reseñas de los libros que han disfrutado. Creemos en la importancia de brindar una voz a los lectores y generar un espacio interactivo donde puedan compartir sus experiencias, recomendaciones y críticas constructivas. De esta manera, aspiramos a crear una comunidad de lectores comprometidos y colaborativos.

Al facilitar el acceso a los libros y promover la participación de los usuarios, esperamos lograr dos objetivos secundarios. En primer lugar, buscamos fomentar el hábito de lectura y despertar el interés por la literatura en un público amplio. Creemos que al hacer que los libros estén al alcance de todos, sin restricciones económicas, estaremos incentivando el placer de la lectura y el aprendizaje constante.

En segundo lugar, aspiramos a crear un espacio inclusivo donde se promueva el intercambio de ideas y la diversidad de opiniones. Valoramos la importancia de la libertad de expresión y el respeto por las diferentes perspectivas. Al permitir que los usuarios compartan sus opiniones, generaremos un entorno en el que se fomente el debate constructivo y se promueva la reflexión crítica.

**Marco / Requisitos Funcionales para el Desarrollo de la Plataforma**

En esta sección, detallaremos los requisitos funcionales que deberá cumplir la web que se desarrollará para la biblioteca virtual. Los requisitos funcionales son las acciones o funciones específicas que la solución técnica debe ser capaz de realizar.

**Registro de usuarios**: La plataforma deberá contar con una página de registro que permita a los usuarios crear una cuenta. La información proporcionada durante el registro se almacenará en una base de datos para su posterior uso y verificación.

**Inicio de sesión (login):** Una vez registrados, los usuarios podrán acceder a la plataforma a través de un inicio de sesión. Esto garantizará la seguridad y privacidad de su información personal.

**Base de datos de libros en formato PDF**: La plataforma deberá contar con una base de datos que almacene los libros en formato PDF. Esta base de datos permitirá la organización y gestión eficiente de los libros disponibles para su descarga.

**Funcionalidad de descarga:** Los usuarios podrán descargar los libros de forma sencilla y directa desde la plataforma. Deberá haber un sistema que facilite la descarga de los archivos PDF de manera rápida y eficiente.

**Valoración y comentarios de los usuarios**: Los usuarios registrados podrán valorar y comentar los libros que han leído. Esta funcionalidad permitirá la interacción y participación activa de los usuarios, así como la generación de opiniones y recomendaciones que enriquecerán la experiencia de la comunidad.

**Visualización de valoraciones y comentarios:** Todos los usuarios, incluso aquellos que no estén registrados, podrán ver las valoraciones y comentarios dejados por otros usuarios. Esto fomentará la retroalimentación y ayudará a los usuarios a tomar decisiones informadas al seleccionar los libros que deseen leer.

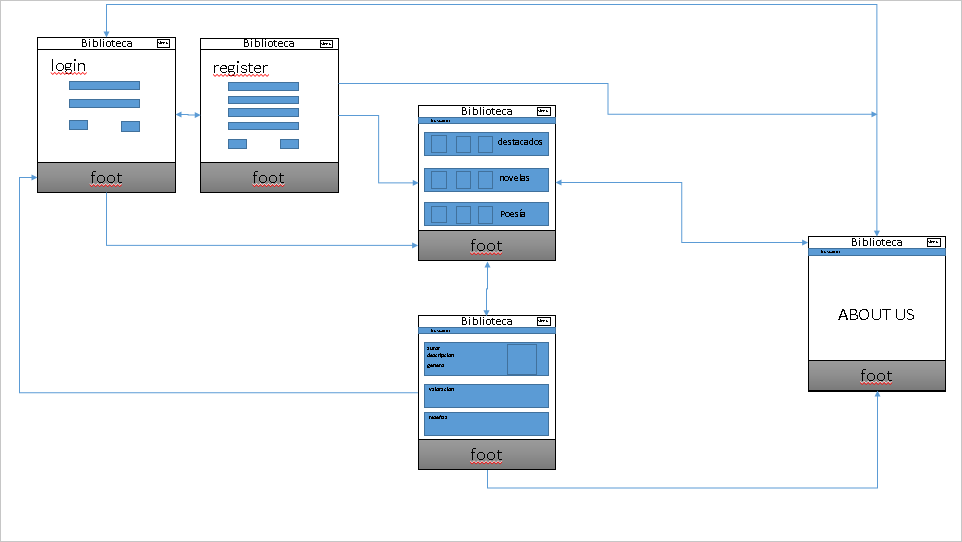
**Diseño moderno y intuitivo:** La plataforma se distinguirá por un diseño moderno y disruptivo que ofrezca una experiencia visual agradable y cómoda para los usuarios. Se evitará el diseño cuadrado, aburrido y poco intuitivo que suele encontrarse en otras bibliotecas online, y se buscará sorprender al usuario con una interfaz innovadora y atractiva.

Con estos requisitos, buscamos ofrecer a los usuarios una plataforma completa y amigable que cumpla con sus necesidades de acceso a libros, interacción con la comunidad y una experiencia visual agradable.

Es importante tener en cuenta que estos requisitos funcionales son fundamentales para el desarrollo de la plataforma y se enfocan en las acciones y funciones clave que se espera que la solución técnica pueda realizar.

**Diseño de la solución /proyecto**

**Mapa de navegación**



**Index:** Se encarga de mostrar todos los libros disponibles en la web, estan divididos por categorias, aparte tiene un buscador para encontrar libros en concreto, desde esta pagina puedes moverte al login, about us y a la pagina de libro.

Login: La pagina Index se muestra hayas iniciado sesion o no, es la pagina principal de la web, pero para poder usar todas las funcionalidades de la web, necesitaras iniciar sesion.

About us: Es una pagina sencilla para cualquier usuario de la web, en la que habla sobre la misma y los creadores.

Pagina de libro: Se encarga de mostrar el libro seleccionado por el usuario, solo se puede acceder desde el index, o desde el buscador al buscar un libro en concreto.

**Pagina de libro:** Se encarga de mostrar el libro elegido, desde el Index o el buscador, muestra la portada del libro, descripcion, autores y genero. Tambien tienes la opción de descargarte el pdf del libro, de manera libre y gratuita.

Si eres usuario de la web, se te habilitara la opcion de valorar el libro, mediante una valoracion por estrellas, esa puntuacion se registrara y se hara la media por cada libro para los libros mejores valorados, mostrarlos en libros destacados dentro de Index.

Tambien podras dejar una reseña con esa puntuacion, podras usarala para contar tu experiencia con el libro que te ha parecido, y si lo recomendarias, tambien comentar que te parece la pagina, las reseñas son unicas por libro e usuario, una vez dejaes una reseña a un libro no podra añadir mas en ese libro. Si no has iniciado sesion , no se te mostrara la opcion de dejar la valoracion y reseña, pero si se te mostraran las reseñas ya publicadas por otros usuarios, se te mostrara un mensaje para que inicies sesion en caso de que quieras dejar una opinion.

Desde esta pagina puedes moverte a la pagina principal Index, About us y login en caso de no iniciar sesion.

**About us:** Se encarga de contar un poco sobre nosotros y la pagina, es una pagina sencilla, que su funcionamiento es simplemente informativo, se compone por html y css en la gran mayoria , tiene un poco de java y php, pero solo por el buscador y algunas acciones de menu desplegable.

Desde esta pagina se puede viajar a la pagina principal Index y a la pagina de login de no haber iniciado sesion.

**Login:** Esta pagina se encarga de verificar las credenciales de un visitante de la web, para iniciar sesion en caso de tener un usuario validado,(la validacion se hace tanto en front como en back), esto da acceso a mas funciones dentro de la web, es una pagina sencilla para iniciar sesion, aunque no es obligatorio para usar la web.

Desde esta web se puede viajar a Register, Index y a About us.

**Register:** Esta pagina se encarga de registrar usuarios, para visitantes nuevos de la web o visitantes que hayan decidio resgistrarse en la web, para acceder a mas funcionalidades de la misma, esta pagina validara los datos introducidos por el visitante y si cumple los requisitos se le enviara a la pagina inicio sesion, ahi ya iniciara sesion con sus credenciales y entrara como usuario a nuestra web.

Desde esta pagina se podra acceder al login, al Index y a About us.

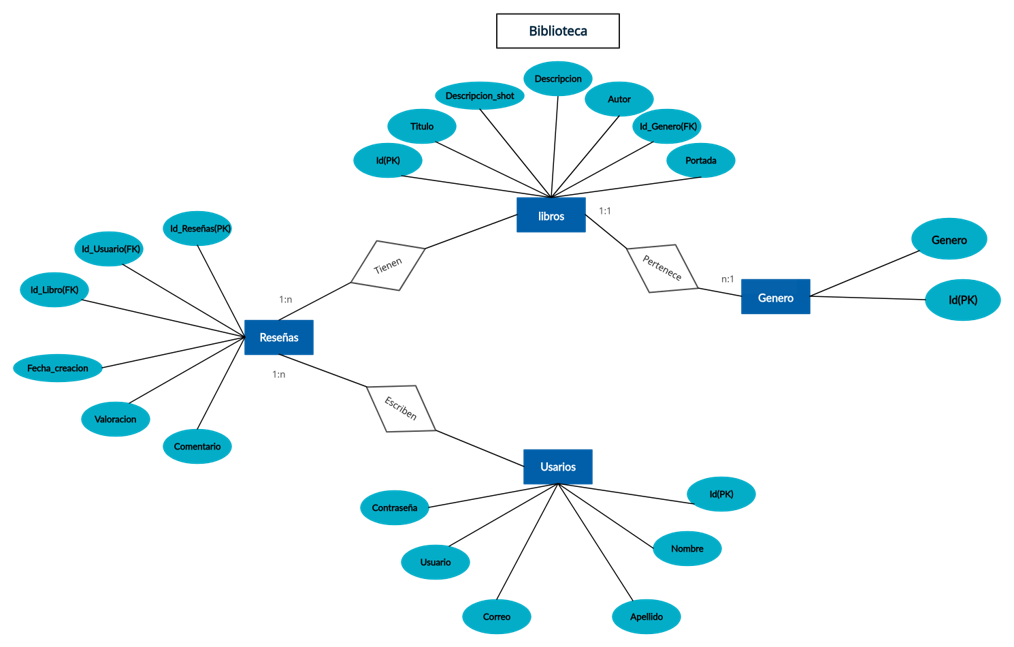
**Modelo de base de datos**

**Libros:**  Aqui se almacenan todos los libros de la web, se utilizara para recoger los datos y mostrarlos en la web segun su genero, desde aqui sacamos titulo ,autores, portada etc... Esta en relacion con genero.

**Genero:**  Gracias a esta tabla, podremos diferenciar los libros segun el tipo, la unica funcion de la tabla es filtrar los libros por genero

**Reseñas:**  Se utiliza para almacenar las opinones de todos los usuarios por cada libro, un libro tiene muchas reseñas y un usuario puede poner muchas reseñas, gracias a esta tabla mostramos las opiniones y la puntuacion de cada usuario segun el libro.

**Usuarios:**  Aqui se almacenaran los usuarios registrados en la web, esta tabla se utilizara para validar usuarios y guardar sus credenciales y validarlos, tambien servira para analizar si tiene reseñas mostrarlas y si puede dejar reseñas en los libros, tambien para mostrar sus datos segun en el apartado web de reseñas



**Diseño de la arquitectura**

**Back-end:**

**Proceso\_inicio:** Aqui validaremos todos los campos del formulario de incio de sesion , se encuentra en la misma pagina que el html, se reciben los datos por post, se hace una consulta a la tabla usuario para comparar con los datos introducidos.

**Proceso\_registro:** Aqui validaremos todos los campos del formulario de registro, que todos los caracteres intro ducidos sean validos, este procesado lo tenemos en un archivo aparte del html, el metodo utulizado es el get, se hace una query para recoger los datos introducidos y comparar email y usuario para que no coincida con ninguno existente y despues se introducen en la base de datos para que haya registro del nuevo usuario.

**Proceso\_reseña:** Este proceso se utiliza para incluir la reseña en la base de datos, se encuentra en un archivo aparte, y el metodo utilizado es post, se recogen todos los datos del formulario y se insertan en la base de datos.

**Proceso\_libro:** Este proceso se utiliza para la pagina añadir libro, solo accesible por el usuario admin, una solucion para añadir libros a la web con un formulario, este proceso se encuentra en un archivo aparte, y con metodo post, se hace un insert en base de datos de todos los campos necesarios para la tabla libro, tambien se puede subir la portada del libro y el pdf, todo automatizado para mostrarse en web al momento de subirlo.

**Buscador:** En este proceso nos encargamos de darle funcionalidad a la barra de buscador de todas las paginas, hacemos una query para recoger los titulos de todos los libros, entra a funcionar cuando empiezas a escribir algo y muestra todos los resultados relacionados, el resultado que selecciones te llevara a la pagina del libro en cuestion. El metodo utilizado es el post y se encuentra en un archivo aparte.

**Conexion:** Hacemos la conexion con la base de datos, y la establecemos con el servidor, hacemos una funccion para cuando establezca conexion y por si falla.

**Generico:** Aqui nos referimos a todo el php que contiene cualquier pagina de la web, como la funcion session\_start, que empieza en el login y se arrastra a todas las paginas para verificar si eres un usuario registrado o un visitante de la web sin loguear, tambien se usa para algunas validaciones. Tambien tenemos un poco de php para alguna query como la de mostrar los libros tanto en index como en pagina\_libro para mostrar todos los datos, tambien un poco de codig oque completa el archivo buscador y los includes.

**Front-end**

**register.js:** Esta es una validacion de front, la utilizamos en el login y en el register, se encuentra en un archivo aparte, es una validacion visual, para que los usuarios sepan si han introducido los campos de forma correcta y si no, les marque un mensaje explicando su error, antes de pasar a la validacion de back.

**General.js:** Nos referimos al javascript que contienen todas las paginas, en nuestra web tenemos un boton menu que muestra unas opciones ,y se esconde cuando volvemos a pulsar, es un breve contenido de javascript que tienen todas las paginas,que se compone de dos funciones.

**CSS:** Hemos realizado un archivo llamado style, el cual contiene todo el css de la web, enlazado con todo el html de las paginas, actua sobre sections, inputs, botones etc..

**Html:** La estructura de html se compone del body, con tres apartados el header, contenido, y el footer, en el header tenemos lo principal de todas las paginas nombre de la web buscador y boton menu, en el contenido, segun la pagina se trabaja de una forma u otra pero siempre todo contenido en sections,

Puede contener formularios, o datos que se muestran en paginas gracias a php.

Y el footer, donde dejamos informacion adicional de la pagina, redes sociales etc...

**Servidor de aplicaciones**

En nuestra arquitectura, hemos seleccionado el servidor de aplicaciones Apache para ejecutar y gestionar nuestras aplicaciones web. Apache es una opción popular y confiable que nos permite alojar y ejecutar nuestro back-end y servidor web de manera eficiente. Proporciona un entorno seguro y escalable para ejecutar nuestras aplicaciones PHP y gestionar las solicitudes de los clientes.

**Servidor web**

El servidor web que hemos elegido en nuestra arquitectura es Apache. Apache actúa como el intermediario entre los clientes y nuestras páginas web. Recibe las solicitudes HTTP de los usuarios y responde con las páginas web correspondientes. Utilizando Apache, podemos alojar y servir nuestras páginas HTML, CSS y JavaScript de manera eficiente, brindando una experiencia fluida a nuestros visitantes.

**Lenguajes utilizados**

**Php:** Lo hemos utilizado para hacer todas las querys y la conexion con la base de datos, tambien para hacer procesos de back y validaciones, todos los libros se muestran por php tambien.

**Css:** Con este lenguaje hemos dado todo el estilo a la pagina web

**Js:** Se ha usado para algunas validaciones en front y algunos eventos de escucha.

**Html:** Es toda la estructura de la web, aqui se ha aplicado el estilo realizado. Con html hemos hecho toda la colocacion y disposicion de la web.

**PhpmyAdmin:** Este lenguaje es el que hemos usado para realizar toda la base de datos, ya que nos lo facilita el servidor de apache, aqui se han hecho las querys y tablas, con la finalidaz de almacenar datos recibidos de la web, y tambien de enviarlos a la web.

**Diseño tecnico de la solución /proyecto**

**Libro de estilo**

**Rejilla:** En este sitio web, no se ha utilizado un sistema de rejilla específico. Los elementos y componentes se han organizado utilizando propiedades de posicionamiento y diseño flexbox para lograr la distribución y alineación deseadas en función del diseño. Esta decisión se tomó para mantener la flexibilidad y adaptabilidad del diseño sin restringirse a una estructura de rejilla rígida. Se ha priorizado la usabilidad y la visualización óptima en diferentes dispositivos y tamaños de pantalla.

**Logotipo:** No hemos desarollado ningun logotipo, en su lugar hemos utilizado el nombre de la web el cual se a posicionado siguiendo la estructura de html, dentro del header alineado a la izquierda

**Tipografia:**

Body: dentro del body el color de letra es #ffffff ,la fuente utilizada es Arial y el tamaño es de 16px, el espacio de linea es de 1.5.

Header H1: aquí cambiamos la fuente a Verdana y el tamaño a 32px.

A: el estilo de los links en general de la pagina es de color #eee y text decoration none , para cuando este sin visitar o visitado.

Nav a: para modificar el link se cambia el tamaño a 1.2rem y se le quita la decoracion al texto del link, se mantiene el color. Cuando hacemos hover cambia de color a #a9997df.

Input buscador: el buscador tiene un tamaño de fuente de 1.2 rem.

Menu desplegable: la fuente tiene un tamaño de 1.2 rem y el color de la letra es #333.

H3 Libro.a: en la pagina index los libros que se muestran cuando haces hover tienen un titulo que es un link para mandarte a su pagina, ese link tiene text decoration underline, y es un tamaño de letra de 1.5 rem.

Libro .descripcion p: tiene un tamaño de fuente de 16 px.

Input.login y register : la fuente es de un tamaño de 1.2 rem.

Login y register.button: la fuente es de un tamaño de 1 rem.

Detalles libro h2: es el titulo del libro que se muestra en la pagina libro una vez seleccionas un libro de la index o por el buscador, el tamaño de la fuente es de 28px.

Detalles libro datos: los demas datos tienen el mismo tamaño 18px.

**Paleta de colores:**

Colores primarios: dentro el body #292929, en el header #401c47 y

footer #401c47, el section se compone por una imagen dentro de la carpeta src con el nombre de section.png y una sombra rgba (0,0,0,0.3).

Colores secundarios: el navegador el fondo es #lalala, el borde de los resultados de busqueda #ccc, background en los detalles de la pagina libro #555 y las estrellas color gold.

**Imágenes y graficos:**

La pagina web no tiene ningun grafico y las imágenes que tiene son 3 tipos la del section que es una imagen de fondo que se adapta al tamaño.

La de portada de libros en el index que las medidas son 300x400.

La portada del libro seleccionado en la pagina de libro que es 300x450.

**Definicion de componentes**

**Etiquetas:** Se adaptan al estilo que este definido anteriormente.

**Botones:**

Boton de buscador: margin-left: 1rem; padding: 0.5rem 1rem; font-size: 1.2rem;

background-color: #6C5CE7; color: #fff; border: none; border-radius: 5px;

cursor: pointer; estilo hover background color #4834d4.

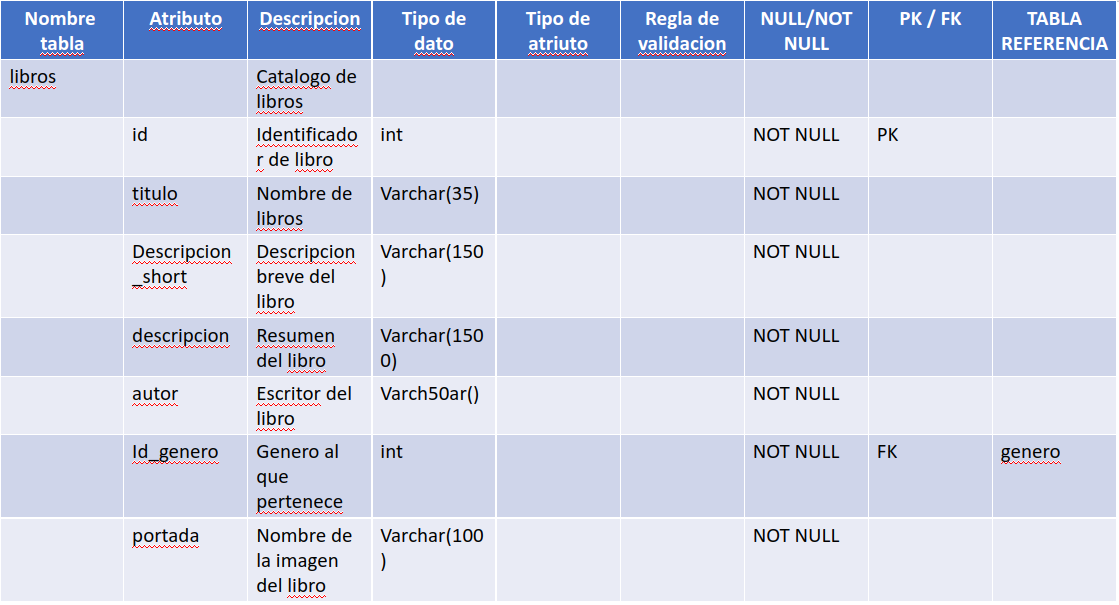
Boton menu desplegable: padding: 0.5rem; font-size: 1.2rem; background-color: #6C5CE7; color: #fff; border: none; border-radius: 5px; cursor: pointer; transition: all 0.3s ease; estilo hover background color #4834d4 y transiction all 0.3s ease.

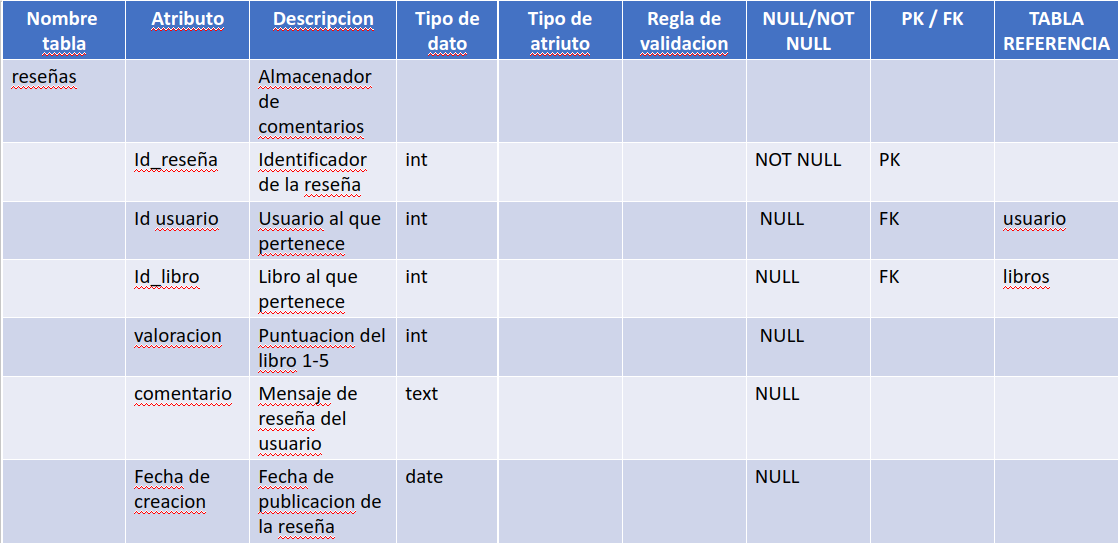
Boton de los formularios: margin-left: 1rem; padding: 0.5rem 0.8rem; font-size: 1rem; background-color:#660066; color: #fff; border: none; border-radius: 5px; cursor: pointer; margin-top: 30px; estilo hover backgorund color #4a014a.

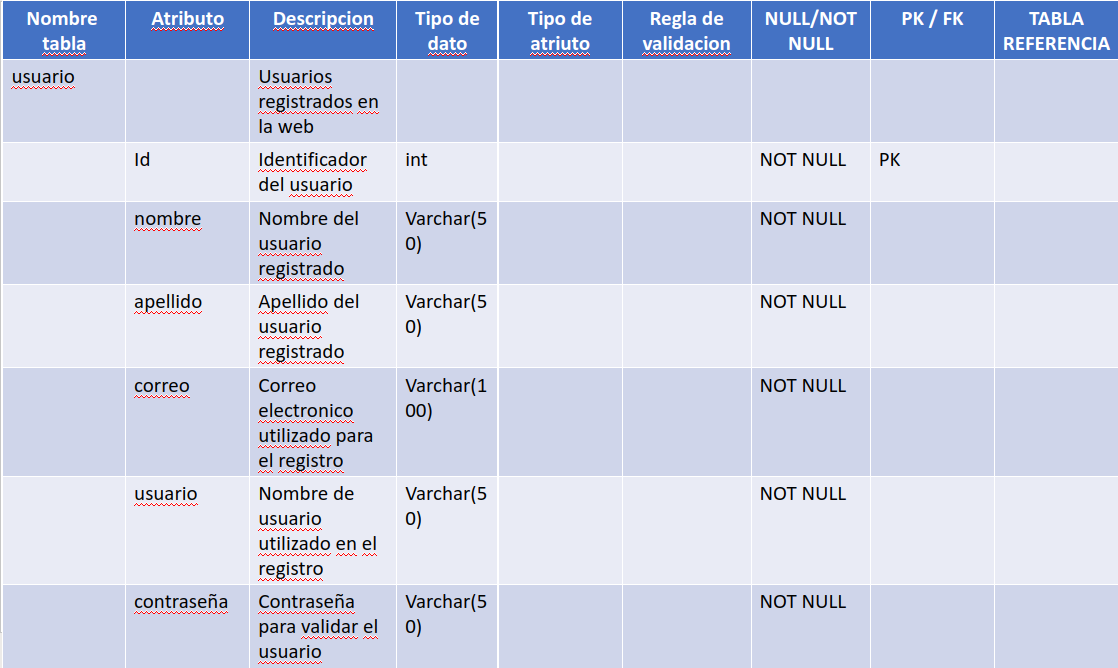
**Inputs:**

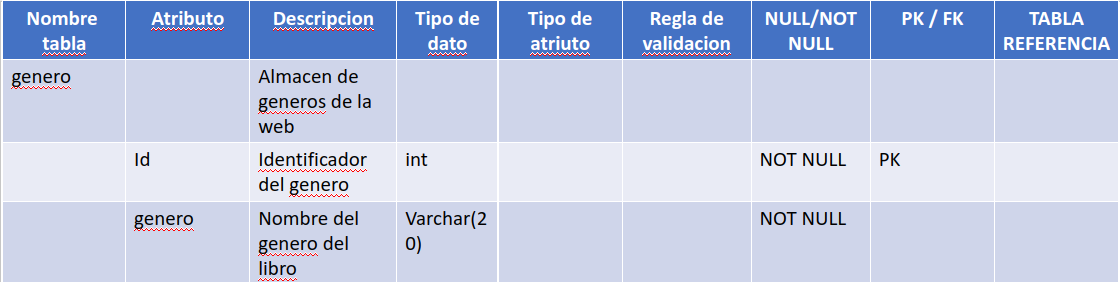
Input formularios: margin-left: 1rem; padding: 0.5rem 0.8rem; font-size: 1rem; color: #fff; border: none; border-radius: 5px; cursor: pointer; margin-top: 30px;

**Diseño de tablas de la base de datos**









**Demo**

Para arrancar el proyecto primero que nada deberemos tener instalados el xamp o cualquier otra aplicación que nos pueda dar un soporte web y de base de datos, una vez lo tengamos instalado, deberemos extraer el .rar(o .zip) y tendremos que colocar la carpeta ya extraída en el directorio htdocs de nuestro xamp (que estará instalado por defecto en la raíz de c). Después de esto importaremos al Mysql del xamp el archivo SQL que viene en la carpeta, esto nos creara la base de datos junto a toda la estructura de tablas y los inserts necesarios. Una vez hagamos esto encenderemos los servicios de apache (servidor web) y MySQL (base de datos) y arrancaremos poniendo el localhoist/ y el nombre del proyecto (o la carpeta), en este caso ‘biblioteca’.

Esto nos llevara por defecto al index del proyecto y ya estaremos listos para usarlo y testearlo adecuadamente.

**Referencias**  **/ Bibliografía**

Las referencias que hemos tomado para la realización del proyecto han sido principalmente el contenido dado durante el año, tanto apuntes como ejercicios prácticos que realizamos que nos han servido para refrescar términos y conceptos que teníamos olvidado.

También cabe destacar la tarea de investigación y aprendizaje que hemos llevado a cabo para los campos que eran totalmente nuevos para nosotros, entre estas fuentes podemos destacar:

* <https://learn.microsoft.com/es-es/docs/> la página oficial de Microsoft con toda la documentación ordenada por lenguajes.
* <https://es.stackoverflow.com/> Un famoso foro de programación que se dedica a solucionar las dudas de sus usuarios en conjunto con el apoyo de su comunidad.
* <https://desarrolloweb.com/> una pagina web pensada para la formación en desarrollo web y lenguajes de programación.
* <https://developer.mozilla.org/es/> la página de formación y documentación o, también conocido como la famosa MDN.

Elegimos estas webs para buscar todos los recursos que necesitábamos puesto a su fiabilidad y la gran confianza que le tenemos.

**Anexos**

* 1. **Código fuente**

En este punto, pondremos los códigos del proyecto separado por archivos y explicaremos su funcionalidad sin entrar en el absoluto detalle de la función de cada línea para no hacer interminable el trabajo

* **Añadir\_libro.php:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<link rel="stylesheet" href="style.css">

<title>Añadir Libro</title>

<?php

error\_reporting(E\_ALL);

ini\_set('display\_errors', 1);

session\_start();

if (isset($\_POST['logout'])) {

session\_unset();

session\_destroy();

header('Location: login.php');

exit;

}

if (isset($\_SESSION['loggedin']) && $\_SESSION['loggedin'] === true) {

$username = $\_SESSION['usuario'];

echo "Bienvenido, $username.";

} else {

echo "Bienvenido, visitante.";

}

?>

</head>

Hasta aquí la principal funcionalidad se encarga de comprobar si la sesión esta iniciada y que usuario es el que ha iniciado sesión.

<body>

<header>

<a href="index.php" class="logo"><h1>Biblioteca </h1></a>

<div class="menu" >

<button id="menu-desplegable">Menú</button>

<ul id="opciones-menu">

<li><a><?php if (isset($\_SESSION['loggedin'])){echo $username;}else{echo $username='Visitante';} ?></a></li>

<li><a href="index.php">Inicio</a></li>

<li><a href="about\_us.php">About Us</a></li>

<?php if ($username=='admin'){

echo "<li><a href='añadir\_libro.php'>Añadir libros</a></li>";

} ?>

<li><?php if(isset($\_SESSION['loggedin'])){echo" <form method='post' action=''>

<input type='submit' name='logout' value='Cerrar sesión' style=' border: 1; background-color: #fff; border: 2px solid #ccc; border-radius: 5px; padding: 0.5rem; margin-top: 0.5rem; font-size: 1.2rem; color: #333; cursor: pointer;'>

</form>";}else{ echo "<li><a href='login.php'>Iniciar sesion</a>";}?> </li>

</ul>

</div>

<script>

var botonMenu = document.getElementById("menu-desplegable");

var opcionesMenu = document.getElementById("opciones-menu");

botonMenu.addEventListener("click", function() {

if (opcionesMenu.style.display === "none") {

opcionesMenu.style.display = "block";

} else {

opcionesMenu.style.display = "none";

}

});

</script>

</header>

Aquí tenemos el menú que usaremos en la web, con un apartado oculto que solo se muestra si somos el usuario admin, y otro apartado que varía en función de si estas con la sesión iniciada o no y se vasa en mostrar un botón de login o de cerrar sesión respectivamente, también tenemos un script que muestra el menú desplegable al hacer click en el botón.

<main>

<form id="buscador">

<input autocomplete="off" type="text" id="barra" name="barra" placeholder="Buscar libros...">

<br>

</form>

<div id="resultados" class="resultados-container"></div>

<script>

var formulario = document.getElementById("buscador");

var barra = document.getElementById("barra");

var resultadosDiv = document.getElementById("resultados");

barra.addEventListener("keyup", function (event) {

event.preventDefault(); // Evitar que se envíe el formulario

var term = barra.value.trim().toLowerCase();

if (term === "") {

resultadosDiv.innerHTML = "";

} else {

var xhr = new XMLHttpRequest();

xhr.open("GET", "buscador.php?barra=" + term, true);

xhr.onreadystatechange = function () {

if (xhr.readyState === XMLHttpRequest.DONE && xhr.status === 200) {

var resultados = JSON.parse(xhr.responseText);

var resultadosHTML = "";

if (resultados.length > 0) {

for (var i = 0; i < resultados.length; i++) {

var resultado = resultados[i];

var titulo = resultado.titulo.toLowerCase();

var genero = resultado.genero.toLowerCase();

var autor = resultado.autor.toLowerCase();

if (

titulo.indexOf(term) === 0 ||

genero.indexOf(term) === 0 ||

autor.indexOf(term) === 0

) {

resultadosHTML += "<p><a href='pagina\_libro.php?id=" +resultado.titulo + "'>" + resultado.titulo + "</a></p>";

resultadosHTML += "<hr>";

}

}

} else {

resultadosHTML = "No se encontraron resultados.";

}

resultadosDiv.innerHTML = resultadosHTML;

}

};

xhr.send();

}

});

</script>

Aquí tenemos el buscador de la pagina, consta de la parte en html, y el script, este script principalmente se encarga de recoger el valor que introducimos en la barra en tiempo real, acto seguido le pasa al archivo buscador.php (lo explicaremos mas adelante) para que haga las consultas en la base de datos y acto seguido imprima debajo un href que te redirija a la pagina del libro.

<section>

<div class="formulario">

<h1>Añadir Libro</h1>

<form class="add" id="register" action="procesar\_libro.php" method="POST" enctype="multipart/form-data">

<label for="titulo"><h3>Título:</h3></label>

<input type="text" name="titulo" id="titulo" required><br>

<label for="descripcion\_short"><h3>Descripción Corta:</h3></label>

<input type="text" name="descripcion\_short" id="descripcion\_short" required><br>

<label for="descripcion"><h3>Descripción:</h3></label>

<textarea class="area" name="descripcion" id="descripcion" rows="6" cols="50" required></textarea><br>

<label for="autor"><h3>Autor:</h3></label>

<input type="text" name="autor" id="autor" required><br>

<label for="id\_genero"><h3>ID Género: 1, 2, 3 o 4</h3></label>

<input type="number" name="id\_genero" id="id\_genero" required><br>

<label for="portada"><h3>Portada:</h3></label>

<input type="file" name="portada" id="portada" accept="image/jpeg, image/png" required><br>

<label for="pdf"><h3>PDF:</h3></label>

<input type="file" name="pdf" id="pdf" accept="application/pdf" required><br>

<input type="submit" value="Añadir Libro" class="register-button">

</form>

</div>

</section>

</main>

</body>

</html>

Por ultimo aquí tenemos el formulario para añadir libro, el cual hace una llamada y envía toda la información que le demos al archivo procesar\_libro.php.

* **Procesar\_libro.php:**

Se encarga de recibir la información que se le ha proporcionado al form de añadir\_libro.php , procesarla y añadirlo de manera adecuada a la base de datos.

<?php

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

$titulo = $\_POST["titulo"];

$descripcion\_short = $\_POST["descripcion\_short"];

$descripcion = $\_POST["descripcion"];

$autor = $\_POST["autor"];

$id\_genero = $\_POST["id\_genero"];

$portada\_nombre = $\_FILES["portada"]["name"];

$portada\_extension = pathinfo($portada\_nombre, PATHINFO\_EXTENSION);

$directorio\_img = "src/";

$portada\_nuevo\_nombre = $directorio\_img . $titulo . '.' . $portada\_extension;

if (move\_uploaded\_file($\_FILES["portada"]["tmp\_name"], $portada\_nuevo\_nombre)) {

$conn = new mysqli('127.0.0.1', 'root', '', 'biblioteca');

if ($conn->connect\_error) {

die("Error en la conexión a la base de datos: " . $conn->connect\_error);

}

$sql = "INSERT INTO libros (titulo, descripcion\_short, descripcion, autor, id\_genero, portada)

VALUES ('$titulo', '$descripcion\_short', '$descripcion', '$autor', $id\_genero, '$portada\_nuevo\_nombre')";

if ($conn->query($sql) === TRUE) {

echo "El libro se añadió correctamente.";

} else {

echo "Error al añadir el libro: " . $conn->error;

}

$conn->close();

} else {

echo "Error al subir la portada.";

}

}

header("Location: añadir\_libro.php");

exit();

?>

Aquí vemos como recibimos vía post la información del libro a añadir, definimos en que ruta se van a guardar los archivos, que nombre tendrán y su extensión (PDF y JPG). Por ultimo se añaden los datos a la base de datos con un insert.

Como ya he explicado elementos generales de la web, como el menú o el buscador y son elementos que se repiten en la mayoría de paginas apartir de aquí pegare el código que sea único por pagina.

* **Login.php:**

<?php

if ($\_SERVER['REQUEST\_METHOD'] === 'POST') {

$usuario = $\_POST['usuario'];

$contrasena = $\_POST['password'];

$hostnameBDD = 'localhost';

$usernameBDD = 'admin';

$passwordBDD = '123';

$databaseBDD = 'biblioteca';

$conexion = new mysqli('127.0.0.1', 'root','' , 'biblioteca');

if ($conexion->connect\_error) {

die('Error al conectar con la base de datos: ' . $conexion->connect\_error);

}

$usuario = $conexion->real\_escape\_string($usuario);

$contrasena = $conexion->real\_escape\_string($contrasena);

$consulta = "SELECT \* FROM usuario WHERE usuario = '$usuario'";

$resultado = $conexion->query($consulta);

if ($resultado && $resultado->num\_rows === 1) {

$fila = $resultado->fetch\_assoc();

if ($contrasena === $fila['contraseña']) {

$\_SESSION['loggedin'] = true;

$\_SESSION['usuario'] = $usuario;

header('Location: index.php');

exit;

} else {

$error = 'Usuario o contraseña incorrectos';

}

} else {

$error = 'Usuario o contraseña incorrectos';

}

$conexion->close();

}

?>

* **Procesar\_resena.php:**

Este archivo se encarga de toda la lógica y programación para subir las reseñas a la base de datos.

<?php

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

$titulo = $\_POST["titulo"];

$id\_libro = $\_POST["id\_libro"];

$id\_usuario = $\_POST["id\_user"];

$valoracion = $\_POST["valoracion"];

$resena = $\_POST["resena"];

$conn = new mysqli('127.0.0.1', 'root','' , 'biblioteca');

if ($conn->connect\_error) {

die("Error en la conexión a la base de datos: " . $conn->connect\_error);

}

$sql = "INSERT INTO reseñas (id\_usuario, id\_libro, valoracion, comentario, fecha\_creacion)

VALUES ('" . $id\_usuario . "', '" . $id\_libro . "', " . $valoracion . ", '" . $resena . "', CURDATE())";

if ($conn->query($sql) === TRUE) {

header("Location: pagina\_libro.php?id=$titulo");

exit();

} else {

echo "Error al añadir la reseña: " . $conn->error;

}  
 $conn->close();

}

?>

Primero recuperamos via POST los datos del formulario, hacemos la conexión a la base de datos, y generamos una sentencia de insert, al que le introduciremos los valores proporcionados. Después de unos mensajes de error por si la conexión falla o si hubo algún error en relación a la inserción de la reseña, cerramos la conexión con la base de datos y redirigimos a la pagina de libro en cuestión.

* **Register.js:**

Este archive se encarga de hacer todas las validaciones del lado de cliente del registro, usando javascript podemos comprobar en el momento si el nombre o la información enviada contiene algún tipo de error o están intentando hacernos alguna inyección SQL.

* window.addEventListener("load", function () {
* document.getElementById("aceptar").disabled = true;
* document.getElementById("fecha").valueAsDate = new Date();
* });

* function validarEmail() {
* email = document.getElementById('correo');
* expresion = /^\w+([.-\_+]?\w+)\*@\w+([.-]?\w+)\*(\.\w{2,10})+$/;
* if (!expresion.test(email.value)) {
* return false;
* } else {
* return true;
* }
* }

* function validarCampos() {
* var nombre = document.getElementById('nombre').value;
* var apellido1 = document.getElementById('apellido1');
* var correo = document.getElementById('correo').value;
* var usuario = document.getElementById('usuario').value;
* var password = document.getElementById('password').value;
* if (nombre.length == 0 || apellido1.value === '' || apellido2.value === '' || correo.length == 0 || telefono.length == 0 ||
* fecha.length == 0 || usuario.length == 0 || password.length == 0 ) {
* return false;
* } else {
* return true;
* }
* }

* function comprobar() {

* if (!validarCampos()) {
* document.getElementById("error").innerHTML = '<p>se deben completar todos los campos<p>';
* }
* else if (!validarTelefono()) {
* document.getElementById("error").innerHTML = '<p>no es un telefono valido<p>';
* }
* else if (!validarEmail()) {
* document.getElementById("error").innerHTML = '<p>no es un email valido<p>';
* }
* }

Principalmente comprueba que todos los campos estén rellenos, y comprueba la forma de algunos campos, como el correo, que debe llevar una @ y el dominio al que pertenece el correo seguido de un . y la extensión adecuada.

* **Register.php**

Esta pagina corresponde a toda la lógica y funcionalidad detrás de las validaciones del lado de servidor, la información proveniente es recibida de registe.php (el formulario anterior)

* <?php
* include('conexion.php');

* function verificarExistenciaUsuarioCorreo($usuario, $correo)
* {
* $conexion = mysqli\_connect(hostnameBDD, usernameBDD, passwordBDD, databaseBDD) or die("Error: " . mysqli\_error($conexion));

* $usuario = mysqli\_real\_escape\_string($conexion, $usuario);
* $correo = mysqli\_real\_escape\_string($conexion, $correo);



Primero creamos una función para verificar si ya esta registrado el usuario con ese imsmo correo o nombre de usuario, para ellos usamos la función mysqli\_real\_escape\_string(),para crear un cadena o sentencia SQL segura para nuestra base de datos, como primer parámetro de entrada le damos los datos de conexión y como segundo parámetro la cadena que usaremos más adelante en la sentencia. Esta función es utilizada principalmente para evitar inyecciones sql a la base de datos.

* $sql = "SELECT COUNT(\*) as total FROM usuario WHERE usuario = '$usuario' OR correo = '$correo'";
* $resultado = mysqli\_query($conexion, $sql);

* $fila = mysqli\_fetch\_assoc($resultado);
* $total = $fila['total'];

* mysqli\_close($conexion);
* return $total > 0;
* }

A continuación añadimos la sentencia SQL que usamos para comprobar si ya existen en la base de datos, se comprueba si el SELECT devuelve algún registro. Si no devuelve ninguno esta comprobación estará hecha y te dejara continuar con el registro.

* function is\_password(string $pwd): bool
* {
* if (
* mb\_strlen($pwd) >= 1 and preg\_match('/[A-Z]/', $pwd)
* and preg\_match('/[a-z]/', $pwd)
* and preg\_match('/[0-9]/', $pwd)
* and !preg\_match('/\\s/', $pwd)
* ) {
* return true;
* }
* return false;
* }
* function is\_mail($str)
* {
* return (false !== filter\_var($str, FILTER\_VALIDATE\_EMAIL));
* }

Ahora verificamos en la función is\_password() si la contraseña cumple los requisitos mínimos de caracteres y con otra función miramos si el correo también es un correo correcto.

* $enviado = $\_GET['nombre'] ?? false;

* if ($enviado) {
* $valido = true;
* $msgerror = '';
* $pwd = filter\_input(INPUT\_GET, 'password');
* $todos = filter\_input\_array(INPUT\_GET);

* if ($valido) {
* if (!is\_password($todos['password'])) {
* $msgerror = 'password no valida';
* $valido = false;
* }

* if (!is\_mail($todos['correo'])) {
* $msgerror = 'correo no valido';
* $valido = false;
* }
* if (!str\_text($todos['usuario'], 0, 15)) {
* $msgerror = 'usuario no valido';
* $valido = false;
* }
* }

* if (verificarExistenciaUsuarioCorreo($todos['usuario'], $todos['correo'])) {
* $msgerror = 'El usuario o el correo ya están registrados.';
* $valido = false;
* }

Ahora pasamos a comprobar los campos usando las funciones anteriores y si todos los campos son validos y pasan todas las validaciones de parte de servidor se procederá a hacer el registro de usuario, creando una sentencia INSERT a la base de datos.

* if ($valido) {
* $conexion = mysqli\_connect(hostnameBDD, usernameBDD, passwordBDD, databaseBDD) or die("Error: " . mysqli\_error($conexion));
* $nombre = mysqli\_real\_escape\_string($conexion, $todos['nombre']);
* $apellido = mysqli\_real\_escape\_string($conexion, $todos['apellido']);
* $correo = mysqli\_real\_escape\_string($conexion, $todos['correo']);
* $usuario = mysqli\_real\_escape\_string($conexion, $todos['usuario']);
* $password = mysqli\_real\_escape\_string($conexion, $todos['password']);

* $sql = "INSERT INTO `usuario` (`id`, `nombre`, `apellido`, `correo`, `usuario`, `contraseña`) VALUES (NULL,
* '$nombre',
* '$apellido',
* '$correo',
* '$usuario',
* '$password')";
* mysqli\_query($conexion, $sql);

* mysqli\_close($conexion);

* header('Location: login.php');
* exit;
* } else {
* print '<script>alert("' . $msgerror . '");history.go(-1);</script>';
* die();
* }
* }
* ?>

La función mysqli\_real\_escape\_string() se utiliza para poder usar caracteres especiales de forma segura en una sentencia SQL.

**Sentencias SQL**

**CREATE TABLE `genero` (**

**`id` int(11) NOT NULL,**

**`genero` varchar(20) NOT NULL**

**)**

**CREATE TABLE `libros` (**

**`id` int(11) NOT NULL,**

**`titulo` varchar(35) NOT NULL,**

**`descripcion\_short` varchar(150) NOT NULL,**

**`descripcion` varchar(1500) NOT NULL,**

**`autor` varchar(50) NOT NULL,**

**`id\_genero` int(11) NOT NULL,**

**`portada` varchar(100) NOT NULL**

**)**

**ALTER TABLE `genero`**

**ADD PRIMARY KEY (`id`);**

**ALTER TABLE `libros`**

**ADD PRIMARY KEY (`id`),**

**ADD KEY `id\_genero` (`id\_genero`);**

**CREATE TABLE `reseñas` (**

**`id\_resena` int(11) NOT NULL,**

**`id\_usuario` int(11) DEFAULT NULL,**

**`id\_libro` int(11) DEFAULT NULL,**

**`valoracion` int(11) DEFAULT NULL,**

**`comentario` text DEFAULT NULL,**

**`fecha\_creacion` date DEFAULT NULL**

**)**

**CREATE TABLE `usuario` (**

**`id` int(11) NOT NULL,**

**`nombre` varchar(50) NOT NULL,**

**`apellido` varchar(50) NOT NULL,**

**`correo` varchar(100) NOT NULL,**

**`usuario` varchar(50) NOT NULL,**

**`contraseña` varchar(50) NOT NULL**

**)**

**ALTER TABLE `reseñas`**

**ADD PRIMARY KEY (`id\_resena`),**

**ADD UNIQUE KEY `un\_usuario\_un\_libro` (`id\_usuario`,`id\_libro`),**

**ADD KEY `id\_libro` (`id\_libro`);**

**ALTER TABLE `usuario`**

**ADD PRIMARY KEY (`id`);**

Obviando el html, que no tiene nada realmente salvo la al mismo archivo(codigo expuesto arriba).

Podemos ver como hago la conexión a la base de datos, comprobamos si la conexión se estableció con éxito, y usamos la función real\_escape\_string() en los campos de usuario y contraseña que recibimos por el form para otorgar a los caracteres que la base de datos podría considerar como caracteres especiales la cualidad de carácter regular, por así decirlo que lo detecte como un string y no como un elemento sintactico propio de la base de datos. De esta forma nos aseguramos que no hay ningún problema con la consulta que hacemos justo después para buscar al usuario en la base de datos. Si existe nos devolverá un un registro el cual usaremos para comprobar la contraseña que introdujo en el form, si ambas contraseñas coinciden (la introducida y la de la BBDD) iniciara sesión y se le guardara el usuario de sesión.