

Manual de control de Libros

1. inicio

Este manual tiene como objetivo guiar al usuario en el uso de la página web para la gestión de libros en una sala de lectura. También ofrece una descripción detallada de los códigos implementados para facilitar futuras modificaciones.

2. Requisitos del Sistema

- **Navegador Web:** Google Chrome, Firefox, Safari, Edge (última versión).
- **Editor de Código:** Visual Studio Code (recomendado).
- **Servidor Web (Opcional):** Si se despliega en un servidor, se recomienda Node.js con Express o cualquier otro servidor HTTP.

3. Estructura del Proyecto

El proyecto está organizado de la siguiente manera:

```
bash
Copiar código
/mi-proyecto
├── /css
│   └── estilos.css           # Estilos de la página
├── /js
│   └── app.js               # Lógica de la aplicación
├── /img
│   └── (imágenes)          # Imágenes utilizadas en la página
└── index.html              # Estructura principal de la página
```

4. Funcionamiento de la Página

Inicio

Al abrir la página `index.html`, el usuario verá la interfaz principal con opciones para:

- **Buscar libros en el catálogo.**
- **Ver y gestionar los pedidos.**

Catálogo de Libros

En esta sección, el usuario puede:

- **Añadir nuevos libros al catálogo:** incluyendo imagen, editorial, y cantidad en existencia.

- **Buscar libros:** usando la barra de búsqueda.
- **Actualizar el estado de un libro:** marcarlo como "prestado" o "en la biblioteca".
- **Marcar libros agotados:** si la cantidad en existencia es cero.

Gestión de Pedidos

Aquí, el usuario puede:

- **Agregar información de pedidos:** nombre del estudiante, fecha del pedido, ciudad, teléfono y colegio.
- **Actualizar el estado del pedido:** marcarlo como "prestado" o "en la biblioteca".

5. Especificaciones del Código

HTML

El archivo `index.html` contiene la estructura básica de la página, organizada en secciones (`div`) para cada función principal:

- **Encabezado y Barra de Navegación:**

```
html
Copiar código
<header>
  <h1>Sala de Lectura - Gestión de Libros</h1>
  <nav>
    <button onclick="mostrarCatalogo()">Catálogo de
Libros</button>
    <button onclick="mostrarPedidos()">Pedidos</button>
  </nav>
</header>
```

- **Formulario para Añadir Libros:**

```
html
Copiar código
<section id="catalogo" style="display:none;">
  <h2>Catálogo de Libros</h2>
  <form id="addBookForm">
    <input type="text" id="titulo" placeholder="Título del
Libro" required>
    <input type="text" id="autor" placeholder="Autor" required>
    <input type="text" id="editorial" placeholder="Editorial">
    <input type="number" id="cantidad" placeholder="Cantidad en
Existencia" required>
    <input type="file" id="imagen" accept="image/*">
    <button type="submit">Añadir Libro</button>
  </form>
</section>
```

CSS

El archivo `estilos.css` define el diseño visual de la página.

- **Estilo de la Barra de Navegación:**

```
css
Copiar código
nav button {
  margin: 5px;
  padding: 10px 20px;
  background-color: #4CAF50;
  color: white;
  border: none;
  cursor: pointer;
}
```

- **Estilo del Formulario de Libros:**

```
css
Copiar código
form#addBookForm {
  display: flex;
  flex-direction: column;
  margin-top: 20px;
}

form#addBookForm input, form#addBookForm button {
  margin-bottom: 10px;
}
```

JavaScript

El archivo `app.js` maneja la lógica de la aplicación, incluyendo la interacción con el DOM y la gestión de eventos.

- **Mostrar y Ocultar Secciones:**

```
javascript
Copiar código
function mostrarCatalogo() {
  document.getElementById('catalogo').style.display = 'block';
  document.getElementById('pedidos').style.display = 'none';
}

function mostrarPedidos() {
  document.getElementById('catalogo').style.display = 'none';
  document.getElementById('pedidos').style.display = 'block';
}
```

- **Añadir Nuevo Libro:**

```
javascript
Copiar código
document.getElementById('addBookForm').addEventListener('submit',
function(e) {
    e.preventDefault();
    // Capturar datos del formulario y añadir al catálogo
});
```

Conexión a la Base de Datos

Actualmente, los datos se gestionan localmente en el navegador. Para un despliegue real, puedes conectar el frontend a un backend usando APIs y una base de datos remota.

6. Modificaciones Futuras

Este manual también te permitirá entender dónde hacer cambios específicos si decides modificar o ampliar la funcionalidad:

- **Agregar nuevos campos:** Puedes añadir nuevos campos en el formulario de libros o pedidos.
- **Mejorar la interfaz:** Modificando el archivo `estilos.css` para adaptar el diseño visual.
- **Integrar una base de datos:** Puedes conectar el formulario al backend para almacenar datos en una base de datos externa.

7. Solución de Problemas Comunes

Problema: La página no carga correctamente

- **Solución:** Asegúrate de que todos los archivos (`index.html`, `estilos.css`, `app.js`) estén en las rutas correctas.

Problema: Los libros no se añaden al catálogo

- **Solución:** Verifica que el JavaScript no tiene errores. Puedes abrir la consola del navegador (F12) para buscar mensajes de error.

Problema: No se muestran las imágenes de los libros

- **Solución:** Asegúrate de que las imágenes estén en la carpeta correcta (`/img`) y que los nombres de los archivos coincidan con los nombres utilizados en el código.

Este manual debe proporcionar a la persona que reciba el proyecto toda la información necesaria para entender cómo funciona la página, cómo está estructurado el código y cómo

hacer futuras modificaciones. Puedes adaptar y expandir este manual según sea necesario para cubrir todas las particularidades de tu proyecto.

40

You said:

puedes crear un formulario para el mismo código de ingreso que tenga correo y contraseña
2/2

ChatGPT said:

ChatGPT

Aquí tienes un ejemplo de cómo agregar un formulario de inicio de sesión que incluya campos para correo electrónico y contraseña al mismo código que ya tienes:

Formulario de Inicio de Sesión

Puedes agregar el siguiente código HTML al archivo `index.html` para crear un formulario de inicio de sesión:

```
html
Copiar código
<section id="login" style="display: block;">
  <h2>Inicio de Sesión</h2>
  <form id="loginForm">
    <input type="email" id="email" placeholder="Correo Electrónico"
required>
    <input type="password" id="password" placeholder="Contraseña"
required>
    <button type="submit">Iniciar Sesión</button>
  </form>
</section>
```

Estilos CSS para el Formulario de Inicio de Sesión

Añade el siguiente código en tu archivo `estilos.css` para darle estilo al formulario:

```
css
Copiar código
#login {
  max-width: 300px;
  margin: 0 auto;
  padding: 20px;
  border: 1px solid #ddd;
  border-radius: 5px;
  box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}

#loginForm input {
```

```

        width: 100%;
        padding: 10px;
        margin: 10px 0;
        border: 1px solid #ddd;
        border-radius: 5px;
    }

    #loginForm button {
        width: 100%;
        padding: 10px;
        background-color: #4CAF50;
        color: white;
        border: none;
        border-radius: 5px;
        cursor: pointer;
    }

    #loginForm button:hover {
        background-color: #45a049;
    }

```

Lógica JavaScript para el Inicio de Sesión

Agrega la siguiente lógica a tu archivo `app.js` para manejar el evento de inicio de sesión:

```

javascript
Copiar código
document.getElementById('loginForm').addEventListener('submit',
function(e) {
    e.preventDefault();

    // Captura los valores del correo electrónico y la contraseña
    const email = document.getElementById('email').value;
    const password = document.getElementById('password').value;

    // Aquí podrías añadir la lógica para validar el usuario contra una
    base de datos o lista de usuarios
    if (email === "usuario@example.com" && password === "123456") {
        alert('Inicio de sesión exitoso');
        // Ocultar el formulario de inicio de sesión y mostrar el
        catálogo o los pedidos
        document.getElementById('login').style.display = 'none';
        document.getElementById('catalogo').style.display = 'block';
    } else {
        alert('Correo o contraseña incorrectos');
    }
});

```

Integración del Formulario con el Resto de la Página

1. **Colocar el formulario al inicio:** Para que el formulario de inicio de sesión sea lo primero que ve el usuario, asegúrate de que la sección de inicio de sesión (`<section id="login">`) esté antes de las otras secciones (`#catalogo`, `#pedidos`) en tu archivo `index.html`.

2. **Mostrar la sección adecuada tras el inicio de sesión:** En el código JavaScript, después de validar las credenciales, puedes ocultar la sección de inicio de sesión y mostrar la sección de catálogo o pedidos, dependiendo de tu flujo de trabajo.

Este formulario de inicio de sesión es simple y se basa en la validación básica en el frontend. Para una aplicación real, deberías considerar integrar un backend que maneje la autenticación segura, incluyendo el cifrado de contraseñas y el manejo de sesiones o tokens de autenticación.

Si quieres que el login proteja la aplicación completamente y funcione en un entorno de producción, necesitarías implementar medidas de seguridad más avanzadas en el backend.