**Base de datos Biblioteca**

Código necesario para implementar una base de datos en IndexedDB que maneje las tablas socios, libros y préstamos, y permita realizar altas, bajas y modificaciones en las tres tablas.

Se incluye la estructura básica de la base de datos, junto con funciones para agregar, modificar y eliminar registros.

// Nombre de la base de datos y versión

const DB\_NAME = "BibliotecaDB";

const DB\_VERSION = 1;

let db;

// Inicialización de la base de datos

function initDB() {

const request = indexedDB.open(DB\_NAME, DB\_VERSION);

request.onupgradeneeded = function (event) {

db = event.target.result;

// Crear la tabla "socios"

if (!db.objectStoreNames.contains("socios")) {

const sociosStore = db.createObjectStore("socios", { keyPath: "id", autoIncrement: true });

sociosStore.createIndex("nombre", "nombre", { unique: false });

}

// Crear la tabla "libros"

if (!db.objectStoreNames.contains("libros")) {

const librosStore = db.createObjectStore("libros", { keyPath: "id", autoIncrement: true });

librosStore.createIndex("titulo", "titulo", { unique: false });

}

// Crear la tabla "préstamos"

if (!db.objectStoreNames.contains("prestamos")) {

const prestamosStore = db.createObjectStore("prestamos", { keyPath: "id", autoIncrement: true });

prestamosStore.createIndex("socioId", "socioId", { unique: false });

prestamosStore.createIndex("libroId", "libroId", { unique: false });

}

console.log("Base de datos inicializada");

};

request.onsuccess = function (event) {

db = event.target.result;

console.log("Conexión exitosa a la base de datos");

};

request.onerror = function (event) {

console.error("Error al abrir la base de datos", event.target.errorCode);

};

}

// Función genérica para agregar un registro

function agregarRegistro(storeName, data, callback) {

const transaction = db.transaction(storeName, "readwrite");

const store = transaction.objectStore(storeName);

const request = store.add(data);

request.onsuccess = function () {

console.log(`Registro agregado en ${storeName}`, data);

if (callback) callback(true);

};

request.onerror = function (event) {

console.error(`Error al agregar registro en ${storeName}`, event.target.error);

if (callback) callback(false);

};

}

// Función genérica para modificar un registro

function modificarRegistro(storeName, data, callback) {

const transaction = db.transaction(storeName, "readwrite");

const store = transaction.objectStore(storeName);

const request = store.put(data);

request.onsuccess = function () {

console.log(`Registro modificado en ${storeName}`, data);

if (callback) callback(true);

};

request.onerror = function (event) {

console.error(`Error al modificar registro en ${storeName}`, event.target.error);

if (callback) callback(false);

};

}

// Función genérica para eliminar un registro

function eliminarRegistro(storeName, id, notificarResultado) {

const transaction = db.transaction(storeName, "readwrite");

const store = transaction.objectStore(storeName);

const request = store.delete(id);

request.onsuccess = function () {

console.log(`Registro eliminado en ${storeName}`, id);

if (notificarResultado) notificarResultado(true);

};

request.onerror = function (event) {

console.error(`Error al eliminar registro en ${storeName}`, event.target.error);

if (callback) callback(false);

};

}

// Ejemplo de función callback

function notificarResultado(exito) {

if (exito) {

console.log("El registro se eliminó correctamente.");

} else {

console.error("Hubo un problema al eliminar el registro.");

}

}

// Uso de la función eliminarRegistro con el callback

const storeName = "miAlmacen";

const id = 123;

eliminarRegistro(storeName, id, notificarResultado);

// Ejemplo de uso

initDB();

// Agregar un socio

// agregarRegistro("socios", { nombre: "Juan Pérez", edad: 30 }, (success) => {});

// Modificar un libro

// modificarRegistro("libros", { id: 1, titulo: "El Principito", autor: "Antoine de Saint-Exupéry" }, (success) => {});

// Eliminar un préstamo

// eliminarRegistro("prestamos", 1, (success) => {});

// En este caso la función callback se sustituye por una función anónima a la que le // pasamos el parámetro success que indica verdadero y la operación ha sido //correcta y falso en caso contrario

agregarRegistro("socios", { nombre: "Juan Pérez", edad: 30 }, (success) => {

if (success) {

console.log("Registro agregado exitosamente.");

} else {

console.error("Error al agregar el registro.");

}

});

**Detalles del código:**

1. **Estructura de la base de datos:**
   * socios contiene registros de los socios con un id autoincrementado.
   * libros contiene información sobre los libros disponibles.
   * préstamos asocia un socioId con un libroId.
2. **Funciones genéricas:**
   * **agregarRegistro**: Para añadir un nuevo registro en cualquier tabla.
   * **modificarRegistro**: Para actualizar un registro existente.
   * **eliminarRegistro**: Para borrar un registro según su id.
3. **Indices**:
   * Se crean índices para búsquedas rápidas en las columnas clave como nombre, titulo, socioId, y libroId.

**Cómo usar el código:**

* Llama a initDB() al cargar la página.
* Usa las funciones **agregarRegistro**, **modificarRegistro** y **eliminarRegistro** pasando como parámetros el nombre de la tabla y los datos correspondientes.