Mecanismos de almacenamiento local

1. Métodos principales
2. `localStorage`
3. `sessionStorage`
4. Cookies
    Cookies seguras
5. IndexedDB (requiere programación asíncrona)

1. Métodos principales

Método	Persistencia	Tamaño	Datos complejos	Seguridad
`localStorage`	Permanente	~5-10 MB	No	No encriptado
`sessionStorage`	Temporal	~5-10 MB	No	No encriptado
Cookies	Temporal/Permanente	~4 KB	No	Encriptable
IndexedDB	Permanente	~50 MB o más	Sí	Mejor seguridad

2. `localStorage`

- **Características:**
    - Datos persistentes (no se eliminan al cerrar el navegador).
    - Almacena datos en formato clave-valor (strings).
- **Métodos:**

```
localStorage.setItem("key", "value"); // Guardar un valor
let value = localStorage.getItem("key"); // Leer un valor
localStorage.removeItem("key"); // Eliminar un valor
localStorage.clear(); // Eliminar todo
```

- **Ejemplo práctico:**

```
localStorage.setItem("theme", "dark");
console.log(localStorage.getItem("theme")); // "dark"
```

3. `sessionStorage`

- **Características:**
  - Datos temporales (se eliminan al cerrar la pestaña o ventana).
  - También usa formato clave-valor.
- **Métodos:**

```
sessionStorage.setItem("key", "value"); // Guardar un valor
let value = sessionStorage.getItem("key"); // Leer un valor
sessionStorage.removeItem("key"); // Eliminar un valor
sessionStorage.clear(); // Eliminar todo
```

- **Ejemplo práctico:**

```
sessionStorage.setItem("cart", JSON.stringify(["item1", "item
2"]));
console.log(JSON.parse(sessionStorage.getItem("cart"))); //
["item1", "item2"]
```

4. Cookies

- **Características:**
  - Tamaño limitado (~4 KB).
  - Se envían automáticamente al servidor con cada solicitud HTTP.
  - Se pueden configurar con expiración.
- **Crear y leer cookies:**

```
// Crear una cookie
document.cookie = "username=JohnDoe; expires=Fri, 31 Dec 2024
23:59:59 GMT; path=/";

// Leer cookies
console.log(document.cookie);
```

```
// Eliminar una cookie
document.cookie = "username=; expires=Thu, 01 Jan 1970 00:00:
00 GMT; path=/";
```

Cookies seguras

- **Configuraciones clave:**
  - `HttpOnly`: Evita que las cookies sean accesibles por JavaScript.
  - `Secure`: Asegura que las cookies solo se envíen a través de HTTPS.
  - `SameSite`: Controla si las cookies se envían con solicitudes de otros sitios:
    - `Strict`: Solo en el mismo dominio.
    - `Lax`: En solicitudes GET de navegación.
    - `None`: Siempre (requiere `Secure`).
- **Ejemplo práctico: Configuración de cookies seguras (servidor):**

```
const express = require('express');
const app = express();

app.get('/', (req, res) => {
    res.cookie('sessionID', 'abc123', {
        httpOnly: true,
        secure: true,
        sameSite: 'Strict',
        maxAge: 3600000, // 1 hora
    });
    res.send('Cookie segura configurada');
});

app.listen(3000, () => console.log('Servidor en http://localh
ost:3000'));
```

5. IndexedDB (requiere programación asíncrona)

- **Características:**
  - Base de datos NoSQL para almacenar grandes cantidades de datos.
  - Ideal para datos complejos y aplicaciones offline.
- **Uso básico:**

```
let request = indexedDB.open("MyDatabase", 1);

request.onupgradeneeded = function (event) {
    let db = event.target.result;
    db.createObjectStore("users", { keyPath: "id" });
};

request.onsuccess = function (event) {
    let db = event.target.result;
    let transaction = db.transaction(["users"], "readwrite");
    let store = transaction.objectStore("users");

    store.add({ id: 1, name: "John Doe" });
    console.log("User added to IndexedDB.");
};
```