

He decidido hacer el trabajo con [CouchDB](#). Es una base de datos NoSQL orientada a documentos que guarda datos en formato JSON como documentos individuales. Al contrario de bases de datos relacionales como MySQL o PostgreSQL.


Mi trabajo consiste en una lista de tareas que se pueden organizar en completadas o no, con la capacidad de eliminarlas.

Lista de Tareas

Añadir




Completadas ▾

☒ ~~jugar a la play~~



PASO 1.

Lo primero que hice fue instalar en mi ordenador CouchDB, al hacerlo es necesario establecer un usuario y contraseña para poder acceder a la interfaz en <http://127.0.0.1:5984/> [utils](#). Dentro, creé una base de datos llamada tareas.

Databases					Database name ▾	Create Database	{ } JSON		
	Name	Size	# of Docs	Partitioned	Actions				
	tareas	6.2 KB	1 0	No	  				

PASO 2.

Aquí empecé con el código; lo primero fue añadir [PouchDB](#), una librería JS que me permite trabajar con CouchDB de manera local y luego sincronizarlo:

```
const db = new PouchDB('tareas_local');  
const remoteDB = new PouchDB ('http://user:***@localhost:5984/tareas');
```

En la segunda línea, es importante especificar el nombre y la contraseña de CouchDB, así como el nombre de la base de datos creada (tareas).

Lo siguiente fue sincronizar las dos BD

```
db.sync(remoteDB, {  
  live: true,  
  retry: true  
}).on('change', mostrarTareas);
```

En la última línea se actualiza la vista cada vez que haya un cambio.

PASO 3.

Después, decidí empezar a escribir las funciones:

Función que elimina las tareas por ID y versión.

```
function eliminarTarea(id, rev) {  
  db.remove(id, rev);  
}
```

Función que cambia el estado de las tareas y las actualiza.

```
function marcarTarea(tarea) {  
  tarea.completada = !tarea.completada;  
  db.put(tarea).then(() => mostrarTareas()).catch(console.error);  
}
```

Función que permite cambiar el texto de las tareas (es decir, el nombre).

```
function editarTexto(id, rev, nuevoTexto, tarea) {  
  if (nuevoTexto.trim() && nuevoTexto !== tarea.texto) {  
    tarea.texto = nuevoTexto;  
    db.put(tarea).then(() => mostrarTareas()).catch(console.error);  
  }  
}
```

Función que muestra todas las tareas.

```
function mostrarTareas() {  
  const filtroTexto =  
document.getElementById("filtroBusqueda").value.toLowerCase();  
  const filtroEstado = document.getElementById("filtroEstado").value;
```

Dentro de esta función se cogen todos los documentos y se filtran por texto y estado.

```
db.allDocs({ include_docs: true, descending: false }).then((result) => {  
  const lista = document.getElementById("listaTareas");  
  lista.innerHTML = "";
```

```
result.rows  
  .map(r => r.doc)  
  .filter(t => t.texto.toLowerCase().includes(filtroTexto))  
  .filter(t =>  
    filtroEstado === "todas" ||  
    (filtroEstado === "completadas" && t.completada) ||  
    (filtroEstado === "pendientes" && !t.completada)  
  )
```

Para cada tarea, crea una tarjeta;

```
.forEach(tarea => {  
  const card = document.createElement("div");  
  card.className = "card card-tarea fade-in";  
  
  const cardBody = document.createElement("div");  
  cardBody.className = "card-body d-flex justify-content-between align-items-center";
```

Un checkbox,

```
const checkbox = document.createElement("input");  
checkbox.type = "checkbox";  
checkbox.checked = tarea.completada;  
checkbox.className = "form-check-input me-3";  
checkbox.onchange = () => marcarTarea(tarea);
```

Un texto editable,

```
const textoInput = document.createElement("input");  
textoInput.type = "text";  
textoInput.value = tarea.texto;  
textoInput.className = "input-editable" + (tarea.completada ? " tarea-completada" : "");  
textoInput.onblur = () => editarTexto(tarea._id, tarea._rev, textoInput.value, tarea);  
textoInput.onkeypress = (e) => {  
  if (e.key === "Enter") {  
    e.preventDefault();  
    textoInput.blur(); // Se guarda al pulsar enter
```

```

    }
  };

```

Los botones,

```

const btnEliminar = document.createElement("button");
    btnEliminar.className = "btn btn-sm btn-outline-danger";
    btnEliminar.innerHTML = '<i class="bi bi-trash"></i>';
    btnEliminar.onclick = () => eliminarTarea(tarea._id,
tarea._rev);

```

Y se agrupa el contenido.

```

const content = document.createElement("div");
    content.className = "d-flex align-items-center flex-grow-1";
    content.appendChild(checkbox);
    content.appendChild(textoInput);

    cardBody.appendChild(content);
    cardBody.appendChild(btnEliminar);
    card.appendChild(cardBody);
    lista.appendChild(card);
  });
});
}

```

Por último, se muestran los cambios al cargar la pagina.

```

db.changes({ since: 'now', live: true }).on('change', mostrarTareas);
mostrarTareas();

```

PASO 4.

En este punto decidí hacer el html y css:

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Lista de Tareas</title>
    <script
src="https://cdn.jsdelivr.net/npm/pouchdb@7.3.1/dist/pouchdb.min.js"></
script>
    <link rel="stylesheet" type="text/css" href="style.css">
    <!-- Bootstrap -->

```

```

    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">
    <!-- Bootstrap Icons -->
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.10.5/font/bootstrap-icons.css" rel="stylesheet">
</head>
<body>
    <div class="container" style="max-width: 700px;">
        <h2 class="text-center mb-4">Lista de Tareas</h2>

        <div class="input-group mb-3">
            <input id="nuevaTarea" type="text" class="form-control"
placeholder="Escribe una tarea..." />
            <button class="btn btn-primary"
onclick="añadirTarea()">Añadir</button>
        </div>

        <div class="mb-3 d-flex gap-2">
            <input type="text" id="filtroBusqueda" class="form-control"
placeholder="Buscar tarea..." oninput="mostrarTareas()" />
            <select class="form-select w-auto" id="filtroEstado"
onchange="mostrarTareas()">
                <option value="todas">Todas</option>
                <option value="completadas">Completadas</option>
                <option value="pendientes">Pendientes</option>
            </select>
        </div>

        <div id="listaTareas" class="vstack gap-3"></div>
    </div>

    <script
src="https://cdn.jsdelivr.net/npm/pouchdb@7.3.1/dist/pouchdb.min.js"></script>
    <script src="app.js"></script>
</body>
</html>

```

CSS:

```
body {
  background: #f4f6f9;
  padding-top: 50px;
  font-family: "Segoe UI", sans-serif;
}

.card-tarea {
  border: none;
  border-radius: 1rem;
  box-shadow: 0 2px 10px rgba(0, 0, 0, 0.05);
  transition: transform 0.2s;
}

.card-tarea:hover {
  transform: scale(1.01);
}

.tarea-completada {
  text-decoration: line-through;
  opacity: 0.6;
}

.fade-in {
  animation: fadeIn 0.4s ease-in;
}

@keyframes fadeIn {
  from {
    opacity: 0;
    transform: translateY(-5px);
  }

  to {
    opacity: 1;
    transform: translateY(0);
  }
}

.input-editable {
  border: none;
  background: transparent;
  width: 100%;
  outline: none;
}
```

PASO 5.

Aquí simplemente abrí el html en el navegador y comprobé que funcionaba correctamente.