



## Instructivo: Crear cliente frontend para consumir json-server

- **Prerrequisitos:**

Tener el json server encendido. Para defecto de este taller es el <http://localhost:3001/eventos>

### 1. Crear carpeta del proyecto

```
Terminal

mkdir cliente-crud
cd cliente-crud
code .
```

### 2. Crear archivo index.html

```
Terminal

touch index.html
```



Agrega el siguiente contenido:

```
index.html

<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>CRUD Eventos (Prompt)</title>
</head>
<body>
  <script src="crud_eventos.js"></script>
</body>
</html>
```

Este archivo carga el JavaScript que ejecutará el menú interactivo.

### 3. Crear archivo crud\_eventos.js

#### 3.1 crear la variable que representa la conexión al Json web Server

```
crud_eventos.js

// ✓ URL del json-server
const API_URL = "http://localhost:3001/eventos";
```

#### 3.2 crear un método de verificación de la respuesta

```
crud_eventos.js

// ✓ Verificar respuesta HTTP y convertir a JSON
function checkResponse(res) {
  if (!res.ok) throw new Error(`Error ${res.status}:
  ${res.statusText}`);
  return res.json();
}
```

</Riwi>





### 3.3 crear metodos del crud

```
// ✅ GET: Listar todos los eventos
async function listar() {
  try {
    const res = await fetch(API_URL);
    const data = await checkResponse(res);
    console.log("📅 Eventos:", data);
  } catch (e) {
    console.error("❌ Error al listar:", e.message);
  }
}

// ✅ POST: Crear un nuevo evento
async function crear() {
  const nombre = prompt("Nombre del evento:");
  const lugar = prompt("Lugar del evento:");
  const fecha = prompt("Fecha (YYYY-MM-DD):");

  try {
    const res = await fetch(API_URL, {
      method: "POST",
      headers: { "Content-Type": "application/json" },
      body: JSON.stringify({ nombre, lugar, fecha })
    });
    const data = await checkResponse(res);
    console.log("✅ Evento creado:", data);
  } catch (e) {
    console.error("❌ Error al crear:", e.message);
  }
}
```



```
// ✅ PUT: Actualizar un evento existente
async function actualizar() {
  const id = prompt("ID del evento:");
  const nombre = prompt("Nuevo nombre:");
  const lugar = prompt("Nuevo lugar:");
  const fecha = prompt("Nueva fecha (YYYY-MM-DD):");

  try {
    const res = await fetch(`${API_URL}/${id}`, {
      method: "PUT",
      headers: { "Content-Type": "application/json" },
      body: JSON.stringify({ nombre, lugar, fecha })
    });
    const data = await checkResponse(res);
    console.log("📝 Evento actualizado:", data);
  } catch (e) {
    console.error("❌ Error al actualizar:", e.message);
  }
}
```



```
// ✓ DELETE: Eliminar evento por ID
async function eliminar() {
  const id = prompt("ID del evento a eliminar:");
  try {
    const res = await fetch(`${API_URL}/${id}`, { method: "DELETE" });
    if (res.ok) {
      console.log("🗑️ Evento eliminado.");
    } else {
      throw new Error(`Error ${res.status}`);
    }
  } catch (e) {
    console.error("❌ Error al eliminar:", e.message);
  }
}
```



### 3.4 crear la función el menú

```
// ✅ Menú interactivo con prompt()
async function menu() {
  let opcion;
  do {
    opcion = prompt(
      📖 MENÚ CRUD EVENTOS\\n` +
      `1. Listar\\n` +
      `2. Crear\\n` +
      `3. Actualizar\\n` +
      `4. Eliminar\\n` +
      `0. Salir\\n` +
      `Elige una opción:`
    );

    switch (opcion) {
      case "1": await listar(); break;
      case "2": await crear(); break;
      case "3": await actualizar(); break;
      case "4": await eliminar(); break;
      case "0": console.log("👋 Adiós."); break;
      default: console.warn("❗ Opción inválida.");
    }
  } while (opcion !== "0");
}
```



3.5 Llamar a la funcion menu(;

```
// ✓ Ejecutar menú  
menu();
```

4. Levantar el servicio en visual studio code

