

UNIVERSIDAD NACIONAL DE SAN AGUSTÍN
FACULTAD DE INGENIERIA DE PRODUCCIÓN Y SERVICIOS
ESCUELA PROFESIONAL DE INGENIERIA DE SISTEMAS



CURSO:
PROGRAMACION WEB 2

DOCENTE:
CORRALES DELGADO CARLOS JOSE

TEMA:
SQLite

PRESENTADO POR:
SANTA CRUZ VILLA DIEGO SEBASTIAN

GIT HUB:
<https://github.com/DiegoSantaC/LaboratoriosPW2/tree/TareasTeoriaPW2>

RESULTADOS: Tabla de Peliculas dinamica

Películas

- GatoConBotas (2024)
- Dead y Wolverine (2024)
- Elementos (2023)
- Mario Bros (2023)

☰ Rows: 4

	id 🔑 # ↕ Filtrar [iconos]	titulo 📄 # ↕ Filtrar [iconos]	anio # ↕ Filtrar [iconos]
1	2	GatoConBotas	2024
2	3	Dead y Wolverine	2024
3	5	Elementos	2023
4	6	Mario Bros	2023
5			

ERRORES:

1) Falla de nombres

Expression

GET http://localhost:3000/script.js net::ERR_ABORTED localhost/:18 404 (NOT FOUND)

Esto es debido a una falla de nombres de los archivos al conectar con el servidos python (script-→ scrips)

```
20 @app.route("/script.js")
21 def js():
22     return send_from_directory('.', 'scrips.js')
```

2) Error de conexion

```
✖ GET http://localhost:3000/script.js net::ERR_ABORTED localhost/:18
404 (NOT FOUND)

24 conn = sqlite3.connect("imdb.db")
25
26 @app.route("/peliculas")
27 def obtener_peliculas():
28     try:
29         cursor = conn.cursor()
30         cursor.execute("SELECT titulo, anio FROM Pelicula LIMIT 10")
31         resultados = cursor.fetchall()
32         return jsonify([{"titulo": r[0], "anio": r[1]} for r in resultados])
33     except Exception as e:
34         return jsonify({"error": str(e)})
```

Se crea la conexión fuera de la función lo cual causa error en los hilos.

CODIGO:

a)Index.html

```
1 <!DOCTYPE html>
2 <html lang="es">
3 <head>
4     <meta charset="UTF-8">
5     <title>Películas</title>
6 </head>
7 <body>
8     <h1>Películas</h1>
9
10    <form id="formulario">
11        <input type="text" id="titulo" placeholder="Título" required>
12        <input type="number" id="anio" placeholder="Año" required>
13        <button type="submit">Agregar</button>
14    </form>
15
16    <ul id="lista"></ul>
17
18    <script src="script.js"></script>
19 </body>
20 </html>
```

b) Script.js(AJAX)

```
1  function cargarPelículas() {
2      fetch("/películas")
3          .then(res => res.json())
4          .then(data => {
5              const lista = document.getElementById("lista");
6              lista.innerHTML = "";
7              data.forEach(p => {
8                  const li = document.createElement("li");
9                  li.textContent = `${p.titulo} (${p.anio})`;
10                 const btnEliminar = document.createElement("button");
11                 btnEliminar.textContent = "✖";
12                 btnEliminar.onclick = () => eliminarPelicula(p.id);
13                 li.appendChild(btnEliminar);
14                 lista.appendChild(li);
15             });
16         });
17  }
18  --
19  function agregarPelicula(evento) {
20      evento.preventDefault();
21      const titulo = document.getElementById("titulo").value;
22      const anio = document.getElementById("anio").value;
23
24      fetch("/películas", {
25          method: "POST",
26          headers: { "Content-Type": "application/json" },
27          body: JSON.stringify({ titulo, anio })
28      })
29      .then(() => {
30          document.getElementById("formulario").reset();
31          cargarPelículas();
32      });
33  }
34
35  function eliminarPelicula(id) {
36      fetch(`/películas/${id}`, { method: "DELETE" })
37          .then(() => cargarPelículas());
38  }
39
40  document.getElementById("formulario").addEventListener("submit", agregarPelicula);
41
42  window.onload = cargarPelículas;
```

c) Servidor.py

```
24 @app.route("/peliculas", methods=["GET"])
25 def obtener_peliculas():
26     with sqlite3.connect("imdb.db") as conn:
27         peliculas = conn.execute("SELECT * FROM Pelicula").fetchall()
28         return jsonify([{"id": p[0], "titulo": p[1], "anio": p[2]} for p in peliculas])
29
30 @app.route("/peliculas", methods=["POST"])
31 def agregar_pelicula():
32     datos = request.get_json()
33     titulo = datos.get("titulo")
34     anio = datos.get("anio")
35     with sqlite3.connect("imdb.db") as conn:
36         conn.execute("INSERT INTO Pelicula (titulo, anio) VALUES (?, ?)", (titulo, anio))
37         return jsonify({"mensaje": "Película agregada correctamente"})
38
39 @app.route("/peliculas/<int:id>", methods=["DELETE"])
40 def eliminar_pelicula(id):
41     with sqlite3.connect("imdb.db") as conn:
42         conn.execute("DELETE FROM Pelicula WHERE id = ?", (id,))
43         return jsonify({"mensaje": "Película eliminada"})
44
45 if __name__ == "__main__":
46     inicializar_bd()
47     app.run(debug=True, host="localhost", port=3000)

```



```
1 from flask import Flask, request, jsonify, send_from_directory
2 import sqlite3
3
4 app = Flask(__name__)
5
6 def inicializar_bd():
7     with sqlite3.connect("imdb.db") as conn:
8         conn.execute("""
9             CREATE TABLE IF NOT EXISTS Pelicula (
10                 id INTEGER PRIMARY KEY AUTOINCREMENT,
11                 titulo TEXT NOT NULL,
12                 anio INTEGER NOT NULL
13             )
14         """)
15
16 @app.route("/")
17 def index():
18     return send_from_directory('.', 'index.html')
19
20 @app.route("/script.js")
21 def js():
22     return send_from_directory('.', 'script.js')
23
```

CONCLUSIONES:

Este proyecto me sirvió para enseñarme a cómo crear una página web que conecta con un programa en Python para mostrar información de una base de datos. Aprendí a guardar los datos en una base pequeña llamada SQLite y a hacer que el servidor entregue esos datos cuando el usuario los pide. También pude ver cómo el navegador puede pedir esos datos usando JavaScript y mostrarlos en la página. Durante el desarrollo, entendí la importancia de organizar bien el código, abrir y cerrar la base de datos correctamente, y manejar errores comunes para que todo funcione sin problemas. Así, el proyecto sirve para entender cómo trabajan juntos el servidor, la base de datos y la página web para que el usuario vea la información que quiere.