👷 Day 10 Lab Sheet – jQuery, AJAX, JSON & Database Concepts

Objective

By the end of this lab, you will:

- Manipulate the DOM using **jQuery**
- Fetch data using **AJAX (Fetch API)**
- Work with **JSON** data
- Understand basic **database operations (SQL & NoSQL)**

Setup

Add jQuery in all HTML files:

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

Task 1: DOM Manipulation using jQuery

1. Create day10_jquery.html.
2. Add:

<h2 id="heading">Welcome to jQuery</h2>

<button id="btn">Click Me</button>

<script>

$("#btn").click(function() {

$("#heading").text("Text changed using jQuery!");

$("#heading").css("color", "blue");

});

</script>

✅ On click → text and color change.

Task 2: jQuery Effects

Add below the existing button:

<button id="toggleBtn">Toggle Box</button>

```
<script>
 $("#toggleBtn").click(function() {
   $("#box").fadeToggle(1000);
 });
</script>
```

✅ Button toggles visibility of the box with a fade animation.

Task 3: AJAX with Fetch API

1. Create day10_fetch.html.
2. Add:

```
<button id="loadBtn">Load Users</button>
<ul id="userList"></ul>


<script>
document.getElementById("loadBtn").addEventListener("click", async () => {
 try {
   let response = await fetch("https://jsonplaceholder.typicode.com/users");
   let users = await response.json();
   let list = document.getElementById("userList");
   list.innerHTML = "";
   users.forEach(u => {
    list.innerHTML += `<li>${u.name} (${u.email})</li>`;
   });
 } catch (error) {
   console.error("Error loading users:", error);
 }
});
</script>
```

✅ Click → displays list of user names and emails.

Task 4: Working with JSON

1. Create day10_json.html.
2. Inside <script>:

```
let student = {
 name: "Alice",
 age: 22,
 course: "Web Development"
};

// Convert object → JSON
let jsonStr = JSON.stringify(student);
console.log("JSON String:", jsonStr);

// Convert JSON → object
let parsed = JSON.parse(jsonStr);
console.log("Parsed Object:", parsed);
console.log("Student Name:", parsed.name);
```

✅ Console logs both JSON string and parsed object.

Task 5: Mini Project – Display Posts Using Fetch

1. Create day10_posts.html.
2. Add:

```
<h2>Posts</h2>
<button id="fetchPosts">Load Posts</button>
<div id="posts"></div>

<script>
document.getElementById("fetchPosts").addEventListener("click", async () => {
 let response = await fetch("https://jsonplaceholder.typicode.com/posts?_limit=5");
 let posts = await response.json();
 let output = "";
 posts.forEach(p => {
```

```
  output += `<h4>${p.title}</h4><p>${p.body}</p><hr>`;
 });
 document.getElementById("posts").innerHTML = output;
});
</script>
```

✅ On click → displays first 5 posts with titles and body text.

Task 6: Basic Database Concepts (SQL Simulation)

📘 Write the following SQL queries in a text editor (for practice):

Create a table:

```
CREATE TABLE students (
 id INT PRIMARY KEY,
 name VARCHAR(50),
 age INT
);
```

Insert records:

```
INSERT INTO students VALUES (1, 'Alice', 22);
INSERT INTO students VALUES (2, 'Bob', 23);
```

Retrieve data:

```
SELECT * FROM students;
```

Update record:

```
UPDATE students SET age = 24 WHERE id = 2;
```

Delete record:

```
DELETE FROM students WHERE id = 1;
```

✅ These queries simulate basic CRUD operations.

Task 7: MongoDB (NoSQL) Basics (Optional Practice)

If you have MongoDB installed or using an online playground:

```
// Insert
db.students.insertOne({ name: "Alice", age: 22 });
```

```
// Find
db.students.find();

// Update
db.students.updateOne({ name: "Alice" }, { $set: { age: 23 } });

// Delete
db.students.deleteOne({ name: "Alice" });
```

✅ Observe JSON-like syntax instead of tables.

✅ **Deliverables**

- day10_jquery.html (DOM manipulation)
- day10_fetch.html (AJAX + Fetch)
- day10_json.html (JSON handling)
- day10_posts.html (Mini project)
- SQL & MongoDB practice queries (text file or screenshots)

All files should run correctly in the browser and produce expected results.