BIT 224 Practical CAT 2(P100, P101, P103, P106)

To be attempted in groups. The due date is between the 14th and 18th. Be prepared for physical marking and the awarding of marks on any day.

Subject: E-Commerce Web Application

The **goal** is to create a basic e-commerce web application where users can interact with a database through AJAX-driven interactions. The application will include:

1. A front-end interface for user input.
2. A back-end PHP script to handle requests and interact with a database.
3. Use of JSON for data exchange.
4. AJAX: Allows asynchronous communication between client and server without page reloads

Technologies Used

- **Frontend:** HTML, CSS, JavaScript (AJAX, Fetch API or XMLHttpRequest).
- **Backend:** PHP.
- **Database:** MySQL (optional for advanced features).

Core Requirements

Requirements

Part 1: Front-End Development

1. **Create the User Interface:**
   - **Design a simple HTML page displaying a list of products. Each product should include:**
     - A product name.
     - A brief description.
     - An image.
     - A price.
     - An "Add to Cart" button.
2. **Build a Shopping Cart Section:**
   - Show the items that users have added to their cart.
   - Display the total price of items in the cart.

- Include a "Checkout" button.

3. **Use CSS for Styling:**
    - Ensure your application is visually appealing and easy to use. Utilize basic CSS styling.
4. **Implement AJAX Functionality:**
    - Use either the Fetch API or XMLHttpRequest to handle AJAX requests to the server.

      // Fetch API to send data from the form to the server. Sample code

      ```
      document.getElementById('userForm').addEventListener('submit',
      function(e) { e.preventDefault(); const name =
      document.getElementById('name').value; // Using Fetch API
      fetch('server.php', { method: 'POST', headers: { 'Content-Type':
      'application/json', }, body: JSON.stringify({ name }), }) .then(response
      => response.json()) .then(data => {
      document.getElementById('response').innerText = `Server
      ```
    - Fetch the product data dynamically from the server when the page loads.

Part 2: Back-End Development

1. **Set Up the PHP Server**:
    - Create a PHP script (api.php) to handle incoming AJAX requests.
    - Utilize PHP to retrieve product data from a MySQL database and send it back as JSON.
2. **Database Interaction**:
    - Set up a simple MySQL database:
        - Create a table to store products with fields for id, name, description, image_url, and price.
        - Insert at least 5 sample products into the database.
3. **AJAX Endpoints in PHP**:
    - Create endpoints in the api.php script:
        - An endpoint to fetch all products (e.g., /api.php?action=getProducts).

          // products.php (example)

```
header ('Content-Type: application/json');
echo json_encode([
  ['id' => 1, 'name' => 'Product A', 'price' => 29.99],
  ['id' => 2, 'name' => 'Product B', 'price' => 49.99]
]);
```

- An endpoint to handle adding items to the cart (e.g., /api.php?action=addToCart).
- **Cart management system with:**
    1. Add/remove items
    2. Quantity updates
- An endpoint to handle checkout (e.g., /api.php?action=checkout).

Deliverables

A complete web application hosted on a web server or local environment.

Grading Criteria

Functionality (20%): Application works as intended with functional AJAX requests and responsive UI.

Code Quality (10%): Code is clean, well-structured, and properly commented.

User Interface (10%): Attractive and user-friendly design.