

**MIT Art, Design and Technology University**

**MIT School of Computing, Pune**

**Department of Computer Science and Engineering**

|  |
| --- |
| **Lab File** |

# **Practical - Web Programming**

# **Class - L.Y. (SEM-II) SMAD**

# **Name of the Practical Coordinator**

# **Prof. Rohini Bhosale**

**A.Y. 2024 – 2025 (SEM-II)**

**COURSE STRUCTURE**

|  |  |  |  |
| --- | --- | --- | --- |
| **Full Stack Development**  **SEMESTER – VII** | | | |
| **Course Code:** |  | **Course Credits:** | **04** |
| **Teaching Hours / Week (L:T:P):** | **2:0:4** | **CA Marks:** | **40** |
| **Total Number of Teaching Hours:** | **--** | **END-SEM Marks:** | **60** |
| **Course Pre-requisites:** Web Technology | | | |
| **Course Description:**   1. This course offers a comprehensive foundation in full stack web development using Node.js, Express.js, EJS, and MySQL. 2. Students will learn to build dynamic and responsive web applications by integrating front-end and back-end technologies. 3. The course emphasizes hands-on project-based learning and real-world application development. 4. Learners will design RESTful APIs and manage databases for seamless client-server interaction. 5. Through practical exercises, students will gain industry-ready skills for scalable web application development. | | | |
| **Course Learning Objectives:** This course will enable the students to:   1. Introduce students to server-side development using Node.js and its ecosystem. 2. Equip learners with the ability to develop scalable and efficient web servers using Express.js. 3. Enable the use of EJS for dynamic web page rendering and data binding. 4. Provide hands-on experience in integrating relational databases like MySQL with Node.js. 5. Facilitate the development of full stack applications through project-based learning using REST APIs and client-server integration. | | | |
| **Course Outcome:**   1. CO1: Build responsive front-end user interfaces using Bootstrap layout and component system. 2. CO2: Analyze the core principles, architecture, and modular structure of Node.js for robust server-side application development. 3. CO3: Construct and deploy scalable web servers using Node.js and Express.js to efficiently manage HTTP protocols and client-server communication. 4. CO4: Design and synthesize dynamic, data-driven user interfaces using EJS templating integrated with backend services. 5. CO5: Implement and manage CRUD operations by interfacing Node.js with MySQL and engineer RESTful APIs for structured data exchange. | | | |
| **UNIT – I** | **Responsive Interfaces with Bootstrap Components** | | **14 Hours** |
| **Main Topic-1: Introduction to Media Queries, Responsive Web Design Using Media Queries, Syntax, Uses of Media Queries, Benefits of Responsive Design, Elements of a Responsive Design, Working of Responsive Design**  **Main Topic-2: Introduction to Bootstrap: Bootstrap Concept, Uses of Bootstrap, Bootstrap library source, Bootstrap Basics- container, jumbotron, page header, buttons, tables;**  **Main Topic-3: Bootstrap Grid System, Bootstrap Images, Bootstrap Forms, Bootstrap Flexbox- horizontal direction, vertical direction.** | | | |
| **Pedagogy** | **ICT Teaching / Power Point Presentation and Videos** | | |
| **Self-study / Do it yourself** | | |
| **Experiential Learning Topics:** | | |
| **Case Study / PBL - Project Based Learning** | | |
| **UNIT – II** | **Backend Development Fundamentals with Node.js** | | **10 Hours** |
| **Main Topic-1: Introduction to Node.js: What is Node.js? Why use it? Capabilities and file structure**  **Main Topic-2: Setting up Node.js Development Environment – Installing Node.js and npm, NPM and project creation, Node.js modules, Creating a basic web server, Node.js console** | | | |
| **Pedagogy** | **ICT Teaching / Power Point Presentation and Videos** | | |
| **Self-study / Do it yourself** | | |
| **Experiential Learning Topics:** | | |
| **Case Study / PBL - Project Based Learning** | | |
| **UNIT – III** | **Building Web Servers using Express.js** | | **12 Hours** |
| **Main Topic-1: Introduction to Express.js – Features, benefits, use cases**  **Main Topic-2: Express Server Setup – Installing Express.js, project setup, configuration, building Express - based web server** | | | |
| **Pedagogy** | **ICT Teaching / Power Point Presentation and Videos** | | |
| **Self-study / Do it yourself** | | |
| **Experiential Learning Topics:** | | |
| **Case Study / PBL - Project Based Learning** | | |
| **UNIT – IV** | **Templating and Dynamic Views with EJS** | | **12 Hours** |
| **Main Topic-1: Integrating EJS Templates, Creating EJS partials and views, Passing data to views, rendering variables, Looping through data in templates**  **Main Topic-2: Building dynamic front-ends using EJS with Express** | | | |
| **Pedagogy** | **ICT Teaching / Power Point Presentation and Videos** | | |
| **Self-study / Do it yourself** | | |
| **Experiential Learning Topics:** | | |
| **Case Study / PBL - Project Based Learning** | | |
| **UNIT – V** | **Database Integration and REST API Development** | | **12 Hours** |
| **Main Topic-1: Connecting Node.js with MySQL, Installing MySQL, setting up the connection, Performing CRUD operations using Node.js and MySQL**  **Main Topic-2: Building RESTful APIs using Express.js, HTTP methods (GET, POST, PUT, DELETE), Handling requests and responses using Express, Using body-parser for form data** | | | |
| **Pedagogy** | **ICT Teaching / Power Point Presentation and Videos** | | |
| **Self-study / Do it yourself** | | |
| **Experiential Learning Topics:** | | |
| **Case Study / PBL - Project Based Learning** | | |

**Text Books:**

1. Andrew Mead, |Learning Node.js Development: Learn the fundamentals of Node.js, and deploy and test Node.js applications on the web|, Packt Publishing, ISBN-10 ‏ : ‎ 1788395549, ISBN-13 ‏ : ‎ 978-1788395540
2. Ethan Brown, |Web Development with Node and Express, 2e: Leveraging the JavaScript Stack|, O′Reilly; 2nd edition, ISBN-10 ‏ : ‎ 1492053511, ISBN-13 ‏ : ‎ 978-1492053514

**Reference Books:**

1. Mario Casciaro and Luciano Mammino, |Node.js Design Patterns|, Packt Publishing, ISBN: 978-1839214110
2. Russell Dyer, |Learning MySQL and MariaDB|, O'Reilly Media, ISBN: 978-1449362904
3. Leonard Richardson and Mike Amundsen,|RESTful Web APIs|, O’Reilly Media, ISBN: 978-1449358068
4. Ben Frain, Responsive Web Design with HTML5 and CSS (3e), Packt Publishing, ISBN: 978-1839211560

**URLs (Optional) - List of Online Courses**

1. [**Full-Stack Web Development with React Specialization – Coursera (by The Hong Kong University of Science and Technology)**](https://www.coursera.org/specializations/full-stack-react) Covers: Node.js, Express.js, MongoDB, React, REST APIs  
    Platform: Coursera  
    Level: Intermediate – Project-based
2. **The Complete Node.js Developer Course (3rd Edition) – Udemy (by Andrew Mead & Rob Percival)** Covers: Node.js, Express, MongoDB, REST APIs, Authentication  
    Platform: Udemy  
    Level: Beginner to Advanced
3. [**Web Development with Node.js and Express – Coursera (by University of London)**](https://www.coursera.org/learn/web-app-development-nodejs) Covers: Node.js, Express.js, REST APIs  
    Platform: Coursera  
    Level: Intermediate
4. **MySQL for Data Analytics and Business Intelligence – Udemy** Covers: MySQL Basics to Advanced (joins, CRUD, queries)  
    Platform: Udemy  
    Level: Beginner to Intermediate
5. **Full Stack Open 2024 – University of Helsinki (Free)** Covers: Modern web app development with React, Node.js, REST APIs, MongoDB  
    Platform: Independent (Free)  
    Level: Advanced

**List of Assignments:**

|  |  |
| --- | --- |
| **Sr. No.** | **Assignment Title** |
| 1. | **Create a Static Homepage Layout Using Bootstrap**  1.1 Design a responsive homepage using Bootstrap components like container, jumbotron, headers, and buttons.  1.2 Use the grid system to create a structured and adaptive layout suitable for any web app. |
| 2. | **Design a Grid-Based Information Page**  2.1 Develop a multi-column content layout using Bootstrap Grid.  2.2 Display items such as cards, product previews, or blog entries with consistent alignment and spacing. |
| 3. | **Set Up a Node.js Project with Express Server**  3.1  Initialize a new Node.js project using npm init.  3.2 Install Express.js and set up a basic web server with routes like Home (/), About (/about), and Contact (/contact). |
| 4. | **Implement Modular Routing and Middleware Logging**  4.1 Organize server logic using separate route modules (e.g., userRoutes, productRoutes).  4.2 Add middleware to log request URLs, methods, and timestamps for all incoming traffic. |
| 5. | **Create a Data Submission Form and Handle POST Requests**  5.1 Build an HTML form (e.g., for adding items or submitting feedback) and handle its POST request using Express.  5.2 Display the submitted data dynamically on a confirmation view. |
| 6. | **Serve Dynamic Content Using Server-Side Rendering**  6.1 Render a list of items (e.g., products, posts, services) dynamically using EJS with hardcoded or temporary in-memory data.  6.2 Route should accept data from the server and render it in the view using loops. |
| 7. | **Create Dynamic Detail Pages Using URL Parameters**  7.1 Use route parameters to render individual item details dynamically via EJS.  7.2 Display properties like title, description, and date using passed route data. |
| 8. | **Build a Reusable Layout Using EJS Partials**  8.1 Create common EJS partials (header, footer, navbar) and include them across views.  8.2 Maintain consistency across all pages by using a shared layout template. |
| 9. | **Connect Node.js to MySQL and Perform Basic CRUD**  9.1 Create MySQL tables and connect to them using a MySQL Node.js package.  9.2 Implement basic CRUD operations from Express routes to the database. |
| 10. | **Develop RESTful APIs for Data Access and Manipulation**  10.1 Build a set of RESTful APIs using Express.js for performing operations.  10.2 Test the endpoints using Postman and ensure they return structured JSON responses. |

|  |  |  |
| --- | --- | --- |
| ASSESSMENT AND EVALUATION PATTERN | | |
| WEIGHTAGE | Continuous Assessment (CA) | End Semester Assessment (ESA) |
| CA - Assignments (PBL / Case Study / Presentation / Seminar / Group Discussions / Quiz / Test | | |
| ESA - Bloom’s Taxonomy Levels: Remembering, Understanding, Applying, Analyzing, Evaluating, and Creating | | |
| Experiential Learning - Internship (Summer / Winter) / Industry Visit / Site Visit / Field Trips | | |
| Project Based Learning – Mini (Minor) Project / Major Project | | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Course Articulation Matrix (CO-PO Mapping) [Subject Title – Subject Code]** | | | | | | | | | | | | | | |
| COs | PO-1 | PO-2 | PO-3 | PO-4 | PO-5 | PO-6 | PO-7 | PO-8 | PO-9 | PO-10 | PO-11 | PO-12 | PSO-1 | PSO-2 |
| CO-1 | 3 | 3 | 2 | 3 | 3 |  |  |  |  | 3 |  | 3 | 2 |  |
| CO-2 | 3 | 2 |  |  | 2 |  |  |  |  | 3 |  | 3 | 2 |  |
| CO-3 | 3 | 3 | 2 | 3 | 3 |  |  |  |  | 2 |  | 3 | 2 |  |
| CO-4 |  |  |  |  | 3 |  |  |  | 2 | 3 |  | 2 |  | 3 |
| CO-5 |  |  |  |  | 3 |  |  |  | 2 | 3 |  | 2 |  | 3 |
| **3** – HIGH,  **2** – MEDIUM, **1** - LOW | | | | | | | | | | | | | | |

## **Experiment No.1**

## 

**Problem Statement:** **Create a Static Homepage Layout Using Bootstrap**

1 Design a responsive homepage using Bootstrap components like container, jumbotron, headers, and buttons.

2 Use the grid system to create a structured and adaptive layout suitable for any web app.

## **Objective:**

To create a visually structured and mobile-friendly homepage using Bootstrap’s responsive grid system and components.

## **Theory:**

Bootstrap is a front-end CSS framework that simplifies responsive web design. By using components like .container, .row, .col, and .btn, developers can create layouts that adjust automatically across devices. The homepage typically includes a hero section, navigation bar, and call-to-action buttons styled with Bootstrap classes.

## **Code:**

A. Home page:

code:

<!-- Hero Section -->

<section class="hero">

    <div class="container">

        <div class="row justify-content-center">

            <div class="col-lg-8 text-center">

                <h1 class="display-4 fw-bold mb-4">Discover Stunning Photography</h1>

                <p class="lead mb-5">Explore our curated collection of high-quality photographs from talented artists around the world. Find the perfect image for your next project or decoration.</p>

                <div class="d-flex justify-content-center gap-3">

                    <a href="/gallery" class="btn btn-primary btn-lg px-4">Browse Gallery</a>

                    <a href="#featured" class="btn btn-outline-light btn-lg px-4">Featured Photos</a>

                </div></div></div</div>

</section>

<!-- Featured Photos -->

<section id="featured" class="py-5">

    <div class="container">

        <h2 class="section-title mb-5">Featured Photos</h2>

        <div class="row g-4">

            <% featuredPhotos.forEach(photo => { %>

                <div class="col-md-6 col-lg-3">

                    <div class="photo-card h-100">

                        <div class="photo-img-container">

                            <img src="/images/<%= photo.filename %>" alt="<%= photo.title %>" class="photo-img"loading="lazy">

                            <div class="photo-overlay">

                                <a href="/gallery" class="btn btn-outline-light">View More</a>

                            </div></div>

                        <div class="card-body">

                            <h5 class="card-title"><%= photo.title %></h5>

                            <p class="card-text text-muted"><%= photo.category %></p>

                            <div class="d-flex justify-content-between align-items-center">

                                <span class="h5 mb-0">$<%= photo.price.toFixed(2) %></span>

                                <button class="btn btn-sm btn-primary add-to-cart" data-id="<%= photo.id %>">

                                    <i class="fas fa-cart-plus me-1"></i> Add to Cart

                                </button>

                            </div></div></div></div>

            <% }); %>

        </div>

        <div class="text-center mt-5">

            <a href="/gallery" class="btn btn-outline-primary">View All Photos</a>

        </div></div>

</section>

<!-- Features Section -->

<section class="bg-light py-5">

    <div class="container">

        <div class="row g-4">

            <div class="col-md-4">

                <div class="text-center p-4">

                    <div class="bg-primary bg-opacity-10 d-inline-flex align-items-center justify-content-center rounded-circle mb-3" style="width: 80px; height: 80px;">

                        <i class="fas fa-truck fa-2x text-primary"></i>

                    </div>

                    <h4>Fast Delivery</h4>

                    <p class="text-muted mb-0">Quick and reliable shipping to your doorstep.</p>

                </div></div>

            <div class="col-md-4">

                <div class="text-center p-4">

                    <div class="bg-primary bg-opacity-10 d-inline-flex align-items-center justify-content-center rounded-circle mb-3" style="width: 80px; height: 80px;">

                        <i class="fas fa-medal fa-2x text-primary"></i>

                    </div>

                    <h4>Premium Quality</h4>

                    <p class="text-muted mb-0">High-resolution photos from professional photographers.</p>

                </div></div>

            <div class="col-md-4">

                <div class="text-center p-4">

                    <div class="bg-primary bg-opacity-10 d-inline-flex align-items-center justify-content-center rounded-circle mb-3" style="width: 80px; height: 80px;">

                        <i class="fas fa-headset fa-2x text-primary"></i>

                    </div>

                    <h4>24/7 Support</h4>

                    <p class="text-muted mb-0">Our team is always here to help you.</p>

                </div></div></div></div>

</section>

<!-- Call to Action -->

<section class="py-5 bg-primary text-white">

    <div class="container text-center">

        <h2 class="mb-4">Ready to find your perfect photo?</h2>

        <p class="lead mb-4">Join thousands of satisfied customers who found their perfect photograph with us.</p>

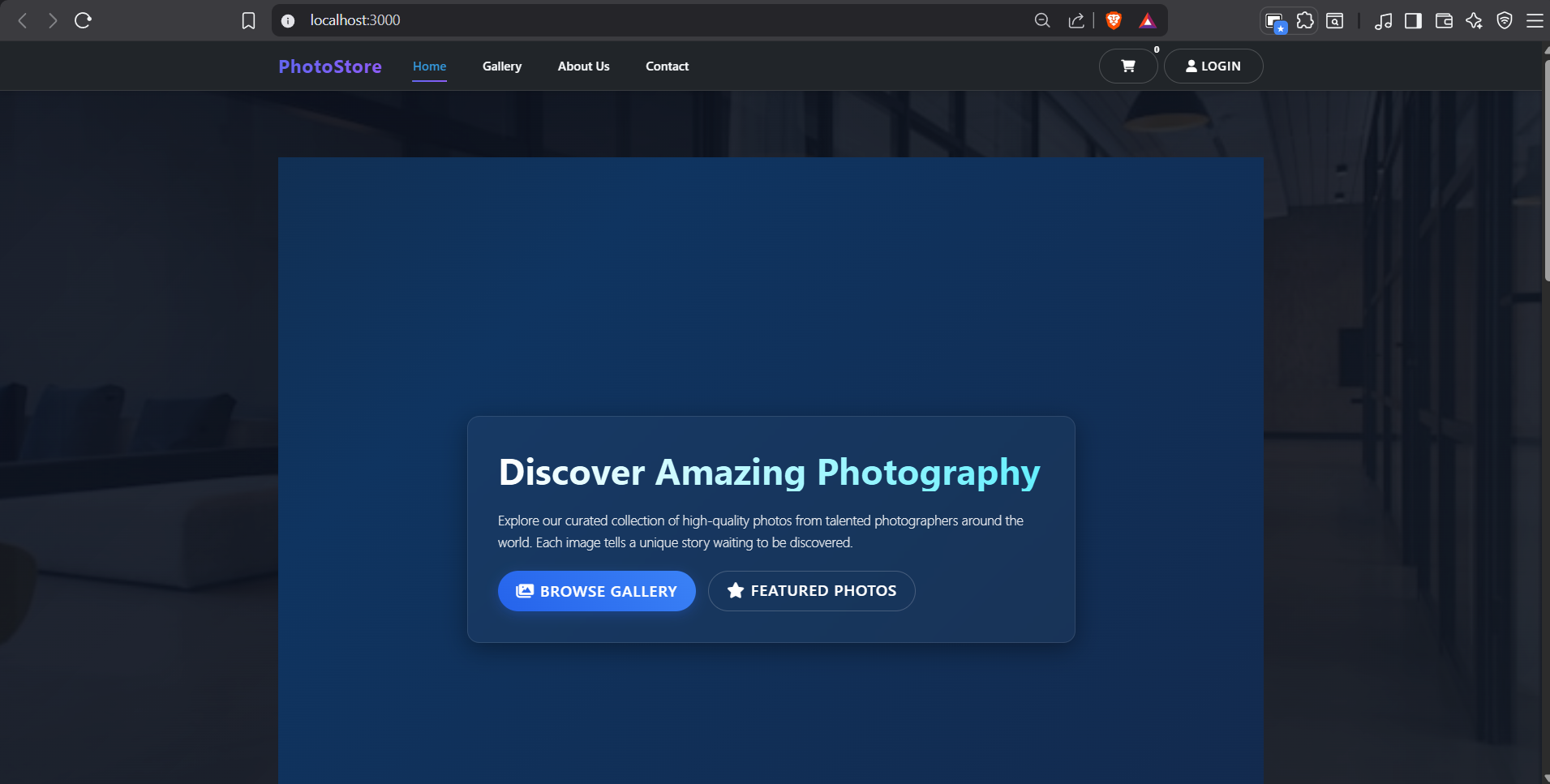
        <a href="/gallery" class="btn btn-light btn-lg px-5">Start Exploring</a>

    </div>

</section>

## **Output:**

* 1. Index/Home page output:



## **Conclusion:**

A responsive and visually appealing homepage was successfully developed using Bootstrap. The layout adapts to all screen sizes while maintaining a clean structure suitable for the PhotoStore application.

## **Experiment No.2**

**Problem Statement:** **Design a Grid-Based Information Page**

2.1 Develop a multi-column content layout using Bootstrap Grid.

2.2 Display items such as cards, product previews, or blog entries with consistent alignment and spacing.

## **Objective:**

To use Bootstrap’s 12-column grid system to design a structured photo gallery layout with equal spacing and alignment.

## **Theory:**

Bootstrap’s grid system uses containers, rows, and columns to arrange content. The .col-md-\* and .col-lg-\* classes allow defining how many columns each element occupies at various screen sizes. Cards and images can be embedded within grid columns to create responsive galleries.

## **Code:**

A. Home page:

code:

<!-- Hero Section with Dynamic Background -->

<style>

    @keyframes gradientBG {0% { background-position: 0% 50%; }50% { background-position: 100% 50%; }100% { background-position: 0% 50%; }}.hero-section {position: relative;overflow: hidden;min-height: 100vh;display: flex;align-items: center;color: #fff;background: linear-gradient(-45deg, #1a1a2e, #16213e, #0f3460, #1a1a1a);background-size: 400% 400%;animation: gradientBG 15s ease infinite;padding: 100px 0;box-sizing: border-box;}

    @media (max-width: 992px) {.hero-section {min-height: 80vh;padding: 80px 0; }}

    @media (max-width: 768px) {.hero-section {min-height: 70vh;padding: 60px 0;text-align: center;}}

    @media (max-width: 576px) {.hero-section {min-height: 60vh;padding: 40px 0; }.hero-content h1 {font-size: 2.2rem;}

        .hero-buttons {flex-direction: column;gap: 15px !important;}

        .hero-buttons .btn {width: 100%;}}

    .particles {position: absolute;top: 0;left: 0;width: 100%;height: 100%;overflow: hidden;z-index: 0;}

    .particle {position: absolute;display: block;background: rgba(255, 255, 255, 0.1);border-radius: 50%;pointer-events: none;}

    .particle:nth-child(1) {width: 300px;height: 300px;top: 10%;left: 10%;

        background: radial-gradient(circle, rgba(100, 240, 255, 0.1) 0%, rgba(100, 240, 255, 0) 70%);

        animation: float 15s infinite ease-in-out; }

    .particle:nth-child(2) {width: 200px;height: 200px;bottom: 10%;right: 15%;

        background: radial-gradient(circle, rgba(59, 130, 246, 0.1) 0%, rgba(59, 130, 246, 0) 70%);

        animation: float 12s infinite ease-in-out reverse;}

    .particle:nth-child(3) {width: 150px;height: 150px;top: 40%;right: 20%;

        background: radial-gradient(circle, rgba(139, 92, 246, 0.1) 0%, rgba(139, 92, 246, 0) 70%);

        animation: float 18s infinite ease-in-out; }

    .particle:nth-child(4) {width: 250px;height: 250px;bottom: 20%;left: 15%;

        background: radial-gradient(circle, rgba(99, 102, 241, 0.1) 0%, rgba(99, 102, 241, 0) 70%);

        animation: float 20s infinite ease-in-out reverse;}

    .particle:nth-child(5) {width: 180px;height: 180px;top: 30%;left: 30%;

        background: radial-gradient(circle, rgba(168, 85, 247, 0.1) 0%, rgba(168, 85, 247, 0) 70%);

        animation: float 14s infinite ease-in-out;}

    @keyframes float {0%, 100% { transform: translateY(0) translateX(0); }25% { transform: translateY(-20px) translateX(10px); }50% { transform: translateY(10px) translateX(-10px); }75% { transform: translateY(-10px) translateX(-5px); }}

    @media (max-width: 1024px) {

        .hero-bg {background-attachment: scroll;}}

    .hero-bg.active {opacity: 1; }

    .hero-content {position: relative;z-index: 1;max-width: 800px;margin: 0 auto;padding: 40px;

        background: rgba(255, 255, 255, 0.03);border-radius: 16px;backdrop-filter: blur(10px);border: 1px solid rgba(255, 255, 255, 0.1);box-shadow: 0 8px 32px 0 rgba(0, 0, 0, 0.36);}

    @media (max-width: 768px) {

        .hero-content {max-width: 90%;padding: 30px 20px;}

    }

    @keyframes fadeIn {from { opacity: 0; transform: translateY(20px); }to { opacity: 1; transform: translateY(0); }

    }

    .hero-content h1 {font-size: 3.5rem;font-weight: 700;margin-bottom: 20px;

        background: linear-gradient(90deg, #fff, #64f0ff);

        -webkit-background-clip: text;

        -webkit-text-fill-color: transparent;

        line-height: 1.2;

    }

    .hero-content p {font-size: 1.25rem;line-height: 1.6;color: rgba(255, 255, 255, 0.9);margin-bottom: 30px;}

    .hero-buttons {animation: fadeIn 0.8s ease-out forwards;}

    .hero-content p { animation-delay: 0.2s;font-size: 1.2rem;line-height: 1.6;}

    .hero-buttons { animation-delay: 0.4s;justify-content: center;flex-wrap: wrap;gap: 15px;}

    .btn-primary {background: linear-gradient(45deg, #2563eb, #3b82f6);border: none;padding: 12px 30px;font-weight: 600;border-radius: 50px;transition: all 0.3s ease;box-shadow: 0 4px 15px rgba(37, 99, 235, 0.3);}

    .btn-outline-light {border: 2px solid rgba(255, 255, 255, 0.2);color: #fff;padding: 10px 28px;font-weight: 600;border-radius: 50px;transition: all 0.3s ease; }

    .btn-primary:hover {transform: translateY(-2px);box-shadow: 0 6px 20px rgba(37, 99, 235, 0.4);}

    .btn-outline-light:hover {background: rgba(255, 255, 255, 0.1);border-color: rgba(255, 255, 255, 0.3);transform: translateY(-2px);}

</style>

<section class="hero-section">

    <!-- Animated Background Elements -->

    <div class="particles"><div class="particle"></div><div class="particle"></div><div class="particle"></div><div class="particle"></div><div class="particle"></div>

    </div>

    <div class="container">

        <div class="hero-content">

            <h1 class="display-3 fw-bold mb-4">Discover Amazing Photography</h1>

            <p class="lead mb-4">Explore our curated collection of high-quality photos from talented photographers around the world. Each image tells a unique story waiting to be discovered.</p>

            <div class="d-flex gap-3 hero-buttons">

                <a href="/gallery" class="btn btn-primary btn-lg px-4">

                    <i class="fas fa-images me-2"></i>Browse Gallery</a>

                <a href="#featured" class="btn btn-outline-light btn-lg px-4">

                    <i class="fas fa-star me-2"></i>Featured Photos</a>

            </div></div></div></section>

<script>

    document.addEventListener('DOMContentLoaded', function() {const buttons = document.querySelectorAll('.btn');buttons.forEach(button => {button.addEventListener('mousemove', (e) => {const rect = button.getBoundingClientRect();const x = e.clientX - rect.left;const y = e.clientY - rect.top;button.style.setProperty('--x', `${x}px`);button.style.setProperty('--y', `${y}px`);});});});

</script>

<!-- Featured Photos -->

<section id="featured" class="py-5">

    <div class="container">

        <h2 class="section-title mb-5">Featured Photos</h2>

        <div class="row g-4">

            <% featuredPhotos.forEach(photo => { %>

                <div class="col-md-6 col-lg-3">

                    <div class="photo-card h-100">

                        <div class="photo-img-container">

                            <img src="/images/<%= photo.filename %>" alt="<%= photo.title %>" class="photo-img"loading="lazy">

                            <div class="photo-overlay">

                                <a href="/gallery" class="btn btn-outline-light">View More</a>

                            </div></div>

                        <div class="card-body">

                            <h5 class="card-title"><%= photo.title %></h5>

                            <p class="card-text text-muted"><%= photo.category %></p>

                            <div class="d-flex justify-content-between align-items-center">

                                <span class="h5 mb-0">$<%= photo.price.toFixed(2) %></span>

                                <button class="btn btn-sm btn-primary add-to-cart" data-id="<%= photo.id %>">

                                    <i class="fas fa-cart-plus me-1"></i> Add to Cart

                                </button>

                            </div></div></div></div>

            <% }); %>

        </div>

        <div class="text-center mt-5">

            <a href="/gallery" class="btn btn-outline-primary">View All Photos</a>

        </div></div>

</section>

<!-- Features Section -->

<section class="bg-light py-5">

    <div class="container"><div class="row g-4"><div class="col-md-4">

                <div class="text-center p-4">

                    <div class="bg-primary bg-opacity-10 d-inline-flex align-items-center justify-content-center rounded-circle mb-3" style="width: 80px; height: 80px;">

                        <i class="fas fa-truck fa-2x text-primary"></i>

                    </div>

                    <h4>Fast Delivery</h4>

                    <p class="text-muted mb-0">Quick and reliable shipping to your doorstep.</p>

                </div></div>

            <div class="col-md-4">

                <div class="text-center p-4">

                    <div class="bg-primary bg-opacity-10 d-inline-flex align-items-center justify-content-center rounded-circle mb-3" style="width: 80px; height: 80px;">

                        <i class="fas fa-medal fa-2x text-primary"></i></div>

                    <h4>Premium Quality</h4>

                    <p class="text-muted mb-0">High-resolution photos from professional photographers.</p>

                </div></div>

            <div class="col-md-4">

                <div class="text-center p-4">

                    <div class="bg-primary bg-opacity-10 d-inline-flex align-items-center justify-content-center rounded-circle mb-3" style="width: 80px; height: 80px;">

                        <i class="fas fa-headset fa-2x text-primary"></i>

                    </div>

                    <h4>24/7 Support</h4>

                    <p class="text-muted mb-0">Our team is always here to help you.</p>

                </div></div></div></div>

</section>

<!-- Call to Action -->

<section class="py-5 bg-primary text-white">

    <div class="container text-center">

        <h2 class="mb-4">Ready to find your perfect photo?</h2>

        <p class="lead mb-4">Join thousands of satisfied customers who found their perfect photograph with us.</p>

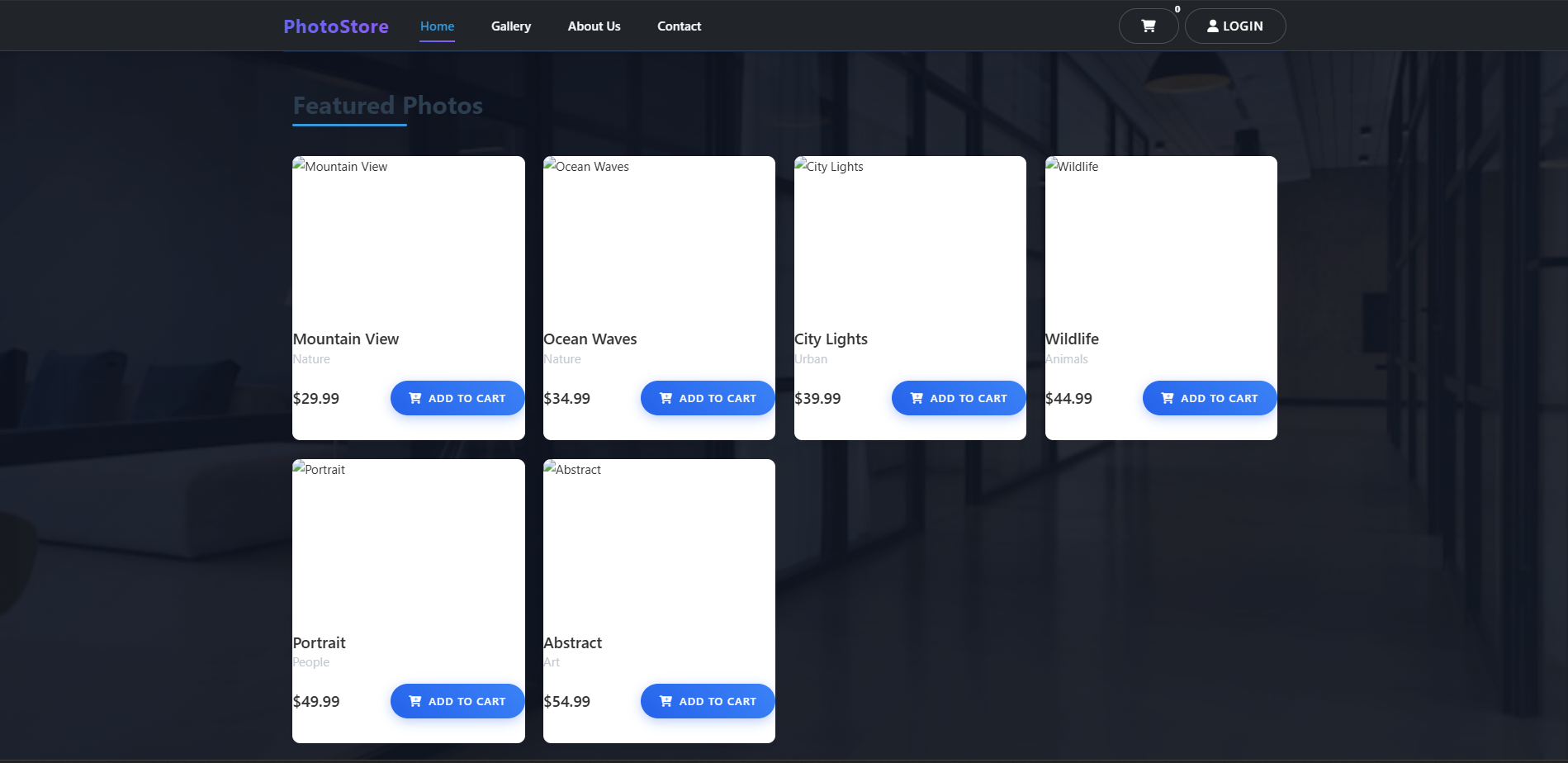
        <a href="/gallery" class="btn btn-light btn-lg px-5">Start Exploring</a>

    </div>

</section>

## **Output:**

* 1. Index/Home page output:



## **Conclusion:**

A grid-based photo gallery page was created using Bootstrap’s grid and card components. The layout dynamically adjusts to different device widths, ensuring uniform alignment of all images.

## 

## **Experiment No.3**

**Problem Statement:** **Set Up a Node.js Project with Express Server**

3.1  Initialize a new Node.js project using npm init.

3.2 Install Express.js and set up a basic web server with routes like Home (/), About (/about), and Contact (/contact).

## **Objective:**

To configure a Node.js environment, install Express.js, and build a simple web server handling multiple routes.

## **Theory:**

Node.js enables server-side JavaScript execution. Express.js is a lightweight web framework for handling routes, requests, and responses. Using app.get() and app.listen(), developers can define endpoints and start the server.

## **Code:**

* 1. About page:

<!-- About Header -->

<div class="text-center py-5">

<h1 class="display-5 fw-bold mb-3">About PhotoStore</h1>

<p class="lead">Capturing moments, creating memories, and sharing the beauty of the world through photography.</p>

</div>

<!-- Our Story -->

<section class="container py-4">

<div class="row align-items-center">

<div class="col-lg-6 mb-4 mb-lg-0">

<img src="/images/about-1.jpg" alt="Our Story" class="img-fluid rounded shadow">

</div>

<div class="col-lg-6 ps-lg-4">

<h2 class="mb-3">Our Story</h2>

<p>Founded in 2023, PhotoStore began as a passion project between photography enthusiasts celebrating the art of photography.</p>

<p>From a small collection of nature photographs to a global marketplace, we connect photographers with art lovers worldwide.</p>

<p>Our mission is to make beautiful photography accessible while supporting the creative community behind the lens.</p></div></div>

</section>

<!-- Our Team -->

<section class="py-5 bg-light">

<div class="container">

<div class="text-center mb-5">

<h2>Our Team</h2>

<p class="lead">The passionate individuals behind PhotoStore</p>

</div>

<div class="row g-4">

<div class="col-md-4">

<div class="card h-100 border-0 shadow-sm">

<div class="team-img-container">

<img src="/images/team-1.jpg" alt="Team Member" class="card-img-top">

<div class="team-overlay">

<h5>....</h5>

<p>Founder & Photographer</p>

</div></div>

<div class="card-body text-center">

<p>15+ years in landscape photography, featured in National Geographic.</p>

<div class="social-links">

<a href="#"><i class="fab fa-instagram"></i></a>

<a href="#"><i class="fab fa-500px"></i></a>

</div></div></div></div>

<div class="col-md-4">

<div class="card h-100 border-0 shadow-sm">

<div class="team-img-container">

<img src="/images/team-2.jpg" alt="Team Member" class="card-img-top">

<div class="team-overlay">

<h5>..</h5>

<p>Tech Lead</p>

</div></div>

<div class="card-body text-center">

<p>Ensuring our platform delivers the best experience for photographers and customers.</p>

<div class="social-links">

<a href="#"><i class="fab fa-github"></i></a>

<a href="#"><i class="fab fa-linkedin"></i></a></div></div></div></div>

<div class="col-md-4">

<div class="card h-100 border-0 shadow-sm">

<div class="team-img-container">

<img src="/images/team-3.jpg" alt="Team Member" class="card-img-top">

<div class="team-overlay">

<h5>....</h5>

<p>Community Manager</p>

</div></div>

<div class="card-body text-center">

<p>Connecting our community of photographers and customers for exceptional experiences.</p>

<div class="social-links">

<a href="#"><i class="fab fa-instagram"></i></a>

<a href="#"><i class="fab fa-twitter"></i></a>

</div></div></div></div></div></div>

</section>

<!-- Our Values -->

<section class="py-5">

<div class="container">

<div class="text-center mb-5">

<h2>Our Values</h2>

<p class="lead">What drives us every day</p>

</div>

<div class="row g-4"><div class="col-md-4"><div class="text-center p-3">

<div class="value-icon">

<i class="fas fa-camera-retro"></i>

</div>

<h4>Quality</h4>

<p class="text-muted">Showcasing only the highest quality photographs from talented artists worldwide.</p></div></div>

<div class="col-md-4">

<div class="text-center p-3">

<div class="value-icon">

<i class="fas fa-users"></i>

</div>

<h4>Community</h4>

<p class="text-muted">Building a supportive space for photographers and art lovers to connect.</p></div>

</div>

<div class="col-md-4">

<div class="text-center p-3">

<div class="value-icon">

<i class="fas fa-leaf"></i></div>

<h4>Sustainability</h4>

<p class="text-muted">Committed to eco-friendly practices and supporting environmental causes.</p>

</div></div></div></div>

</section>

<!-- Join Our Community -->

<section class="py-5 bg-primary text-white text-center">

<div class="container">

<h2 class="mb-3">Join Our Community</h2>

<p class="lead mb-4">Showcase and sell your photography with us!</p>

<a href="/contact" class="btn btn-light btn-lg px-4">Get in Touch</a></div>

</section>

<style>

.team-img-container {position: relative;height: 250px;overflow: hidden;}

.team-img-container img {width: 100%; height: 100%;object-fit: cover;}

.team-overlay {position: absolute;bottom: 0;left: 0;right: 0;background: rgba(0,0,0,0.7);color: white;padding: 1rem;}

.value-icon {width: 80px;height: 80px;margin: 0 auto 1rem;display: flex;align-items: center;justify-content: center;background: rgba(13, 110, 253, 0.1);border-radius: 50%;}

.value-icon i {font-size: 2rem;color: #0d6efd;}

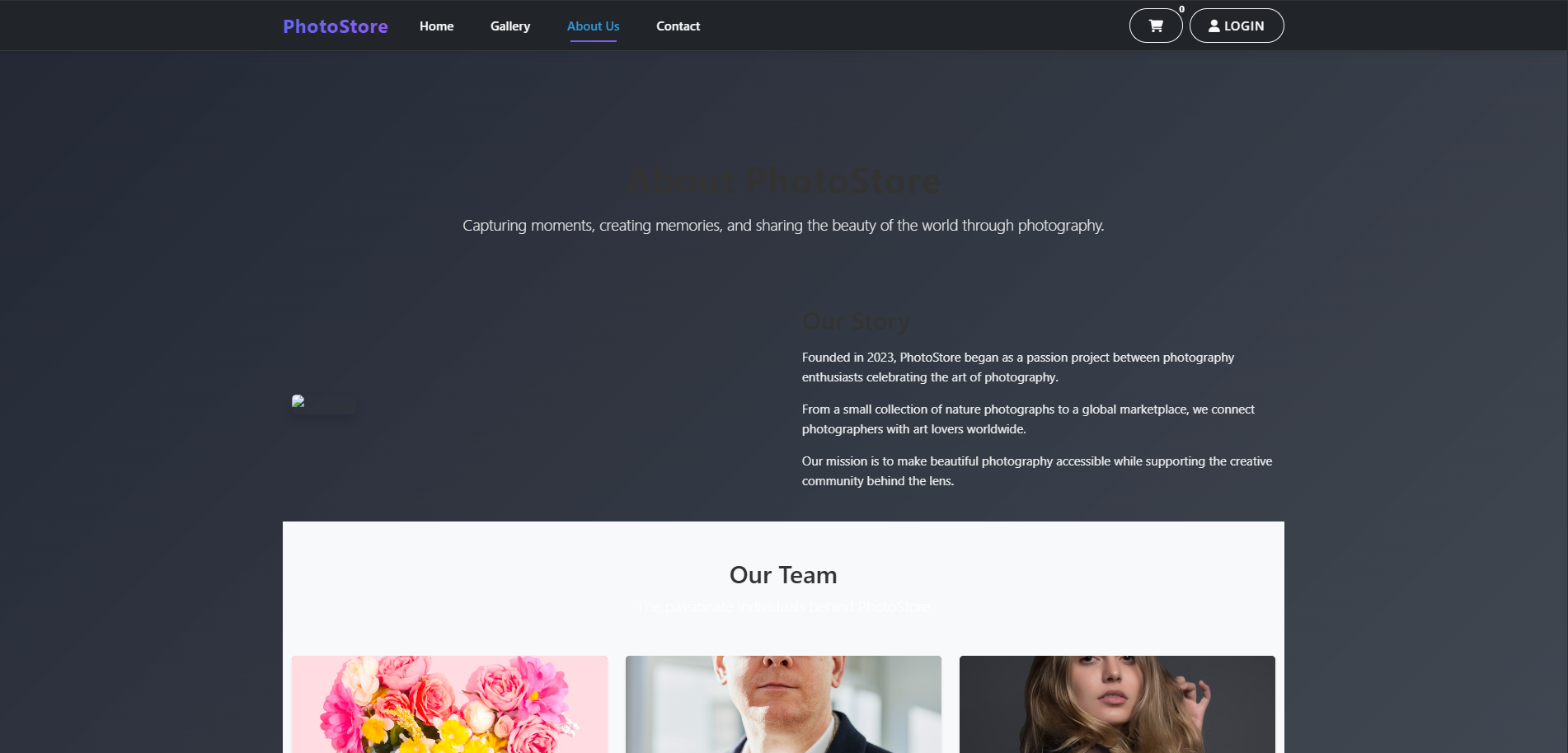
.social-links a {color: #6c757d;margin: 0 8px;font-size: 1.2rem;transition: color 0.3s;}

.social-links a:hover {color: #0d6efd;}

</style>

## **Output:**

* 1. About page output:



## **Code:**

* 1. Contact page:

<!-- Contact Header -->

<div class="row mb-5">

<div class="col-lg-8 mx-auto text-center">

<h1 class="display-5 fw-bold mb-3">Get In Touch</h1>

<p class="lead">Have questions or feedback? We'd love to hear from you. Fill out the form below or reach out through our social media channels.</p>

</div></div>

<div class="row g-5">

<!-- Contact Form -->

<div class="col-lg-7">

<div class="contact-form"><h3 class="mb-4">Send Us a Message</h3>

<form id="contact-form" novalidate>

<div id="form-message" class="alert d-none" role="alert"></div>

<div class="row g-3">

<div class="col-md-6">

<label for="name" class="form-label">Your Name <span class="text-danger">\*</span></label>

<input type="text" class="form-control" id="name" name="name" required>

<div class="invalid-feedback">Please enter your name.</div></div>

<div class="col-md-6">

<label for="email" class="form-label">Email Address <span class="text-danger">\*</span></label>

<input type="email" class="form-control" id="email" name="email" required>

<div class="invalid-feedback">Please enter a valid email address.</div>

</div></div>

<div class="mb-3 mt-3">

<label for="subject" class="form-label">Subject</label>

<input type="text" class="form-control" id="subject" name="subject">

</div>

<div class="mb-3">

<label for="message" class="form-label">Your Message <span class="text-danger">\*</span></label>

<textarea class="form-control" id="message" name="message" rows="5" required></textarea>

<div class="invalid-feedback">Please enter your message.</div>

</div>

<div class="d-grid">

<button type="submit" class="btn btn-primary btn-lg">

<span class="submit-text">Send Message</span>

<span class="spinner-border spinner-border-sm d-none" role="status" aria-hidden="true"></span>

</button></div>

</form></div></div>

<!-- Contact Information -->

<div class="col-lg-5">

<div class="bg-light p-4 rounded h-100">

<h3 class="mb-4">Contact Information</h3>

<p>We're here to help and answer any questions you might have. We look forward to hearing from you!</p>

<ul class="list-unstyled mt-4">

<li class="mb-4">

<div class="d-flex">

<div class="me-3">

<div class="bg-primary bg-opacity-10 d-flex align-items-center justify-content-center rounded-circle" style="width: 50px; height: 50px;">

<i class="fas fa-map-marker-alt text-primary"></i>

</div></div>

<div>

<h5 class="mb-1">Our Location</h5>

<p class="text-muted mb-0">123 Photography Street, City, Country</p>

</div></div></li>

<li class="mb-4">

<div class="d-flex">

<div class="me-3">

<div class="bg-primary bg-opacity-10 d-flex align-items-center justify-content-center rounded-circle" style="width: 50px; height: 50px;">

<i class="fas fa-envelope text-primary"></i>

</div></div>

<div>

<h5 class="mb-1">Email Us</h5>

<p class="text-muted mb-0">info@photostore.com<br>support@photostore.com</p>

</div></div></li>

<li class="mb-4">

<div class="d-flex">

<div class="me-3">

<div class="bg-primary bg-opacity-10 d-flex align-items-center justify-content-center rounded-circle" style="width: 50px; height: 50px;">

<i class="fas fa-phone-alt text-primary"></i>

</div></div>

<div>

<h5 class="mb-1">Call Us</h5>

<p class="text-muted mb-0">+1 (555) 123-4567<br>Mon - Fri, 9:00 AM - 6:00 PM</p>

</div></div></li></ul>

<div class="mt-5">

<h5 class="mb-3">Follow Us</h5>

<div class="social-links">

<a href="#" class="text-primary me-3" data-bs-toggle="tooltip" title="Facebook">

<i class="fab fa-facebook-f fa-lg"></i>

</a>

<a href="#" class="text-primary me-3" data-bs-toggle="tooltip" title="Instagram">

<i class="fab fa-instagram fa-lg"></i>

</a>

<a href="#" class="text-primary me-3" data-bs-toggle="tooltip" title="Twitter">

<i class="fab fa-twitter fa-lg"></i>

</a>

<a href="#" class="text-primary me-3" data-bs-toggle="tooltip" title="Pinterest">

<i class="fab fa-pinterest-p fa-lg"></i>

</a>

<a href="#" class="text-primary" data-bs-toggle="tooltip" title="YouTube">

<i class="fab fa-youtube fa-lg"></i>

</a></div></div></div></div></div>

<!-- Map Section -->

<div class="row mt-5">

<div class="col-12">

<div class="map-container rounded overflow-hidden" style="height: 400px; background-color: #f8f9fa;">

<!-- You can embed a Google Map here -->

<div class="h-100 d-flex align-items-center justify-content-center bg-light">

<div class="text-center">

<i class="fas fa-map-marked-alt fa-3x text-muted mb-3"></i>

<h5>Our Location</h5>

<p class="text-muted">123 Photography Street, City, Country</p>

</div></div></div></div></div>

<!-- FAQ Section -->

<section class="py-5 mt-5 bg-light">

<div class="container">

<div class="text-center mb-5">

<h2>Frequently Asked Questions</h2>

<p class="lead">Find answers to common questions about our services</p>

</div>

<div class="row justify-content-center">

<div class="col-lg-8">

<div class="accordion" id="faqAccordion">

<div class="accordion-item mb-3 border-0 shadow-sm">

<h3 class="accordion-header" id="headingOne">

<button class="accordion-button" type="button" data-bs-toggle="collapse" data-bs-target="#collapseOne">

How do I purchase a photo?

</button>

</h3>

<div id="collapseOne" class="accordion-collapse collapse show" data-bs-parent="#faqAccordion">

<div class="accordion-body">

Purchasing a photo is easy! Simply browse our gallery, select the photo you love, and click the "Add to Cart" button. You can review your selection in the cart before proceeding to checkout. We accept various payment methods including credit cards and PayPal.</div></div></div>

<div class="accordion-item mb-3 border-0 shadow-sm">

<h3 class="accordion-header" id="headingTwo">

<button class="accordion-button collapsed" type="button" data-bs-toggle="collapse" data-bs-target="#collapseTwo">

What file formats are available for download?

</button>

</h3>

<div id="collapseTwo" class="accordion-collapse collapse" data-bs-parent="#faqAccordion">

<div class="accordion-body">

We offer high-resolution JPEG files for all our photos. Some photos may also be available in RAW format upon request. After purchase, you'll receive an email with a download link where you can access your purchased photos in the available formats.

</div></div></div>

<div class="accordion-item mb-3 border-0 shadow-sm">

<h3 class="accordion-header" id="headingThree">

<button class="accordion-button collapsed" type="button" data-bs-toggle="collapse" data-bs-target="#collapseThree">

Can I request a custom photo shoot?

</button>

</h3>

<div id="collapseThree" class="accordion-collapse collapse" data-bs-parent="#faqAccordion"><div class="accordion-body">Yes! We offer custom photo shoots based on your specific requirements. Please use the contact form to provide details about your project, and our team will get back to you with availability and pricing information.</div></div></div>

<div class="accordion-item border-0 shadow-sm">

<h3 class="accordion-header" id="headingFour">

<button class="accordion-button collapsed" type="button" data-bs-toggle="collapse" data-bs-target="#collapseFour">

What is your return policy?

</button>

</h3>

<div id="collapseFour" class="accordion-collapse collapse" data-bs-parent="#faqAccordion">

<div class="accordion-body">Due to the digital nature of our products, we don't offer refunds for downloaded photos. However, if you encounter any issues with your download or the quality of the photos, please contact our support team within 7 days of purchase, and we'll be happy to assist you.</div></div></div></div>

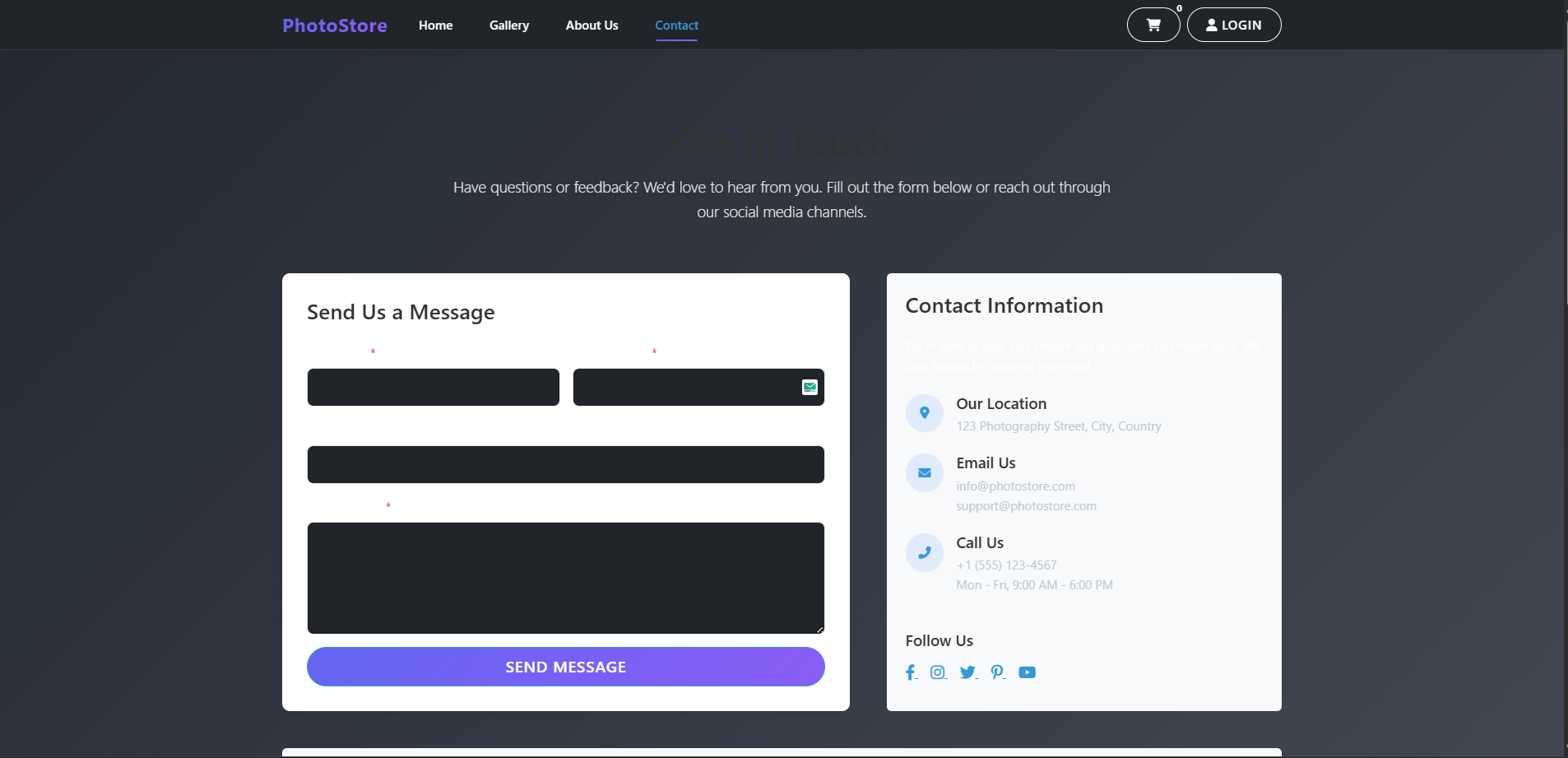
<div class="text-center mt-5">

<p class="mb-0">Still have questions? <a href="#contact-form" class="text-primary">Contact us</a> and we'll get back to you as soon as possible.</p></div></div></div></div>

</section>

## **Output:**

* 1. Contact page output:



## **Conclusion:**

A Node.js project was initialized and an Express.js server was successfully configured. Routes for Home, About, and Contact pages were created and rendered in the browser.

## **Experiment No.4**

**Problem Statement: Implement Modular Routing and Middleware Logging**

4.1 Organize server logic using separate route modules (e.g., userRoutes, productRoutes).

4.2 Add middleware to log request URLs, methods, and timestamps for all incoming traffic.

## **Objective:**

To modularize Express routes for better project structure and implement a middleware function for logging incoming requests.

## **Theory:**

Modular routing helps separate application logic into smaller files (e.g., userRoutes.js, productRoutes.js). Middleware in Express can intercept requests and perform tasks such as logging, authentication, or parsing. Logging includes capturing URL, method, and timestamp.

## **Code:**

PhotoStore/

│

├── routes/

│ ├── userRoutes.js

│ └── productRoutes.js

│

├── public/

│ ├── css/

│ │ └── style.css

│

├── views/

│ ├── partials/

│ │ ├── header.ejs

│ │ └── footer.ejs

│ ├── home.ejs

│ ├── gallery.ejs

│ ├── about.ejs

│ ├── contact.ejs

│ ├── login.ejs

│ └── cart.ejs

│

├── app.js

└── package.json

require('dotenv').config();

const express = require('express');

const expressLayouts = require('express-ejs-layouts');

const path = require('path');

const bodyParser = require('body-parser');

const mongoose = require('mongoose');

const session = require('express-session');

const MongoStore = require('connect-mongo');

const app = express();

const PORT = process.env.PORT || 3000;

// MongoDB Connection

mongoose.connect(process.env.MONGODB\_URI || 'mongodb://localhost:27017/photostore', {

    useNewUrlParser: true,

    useUnifiedTopology: true

})

.then(() => console.log('Connected to MongoDB'))

.catch(err => console.error('MongoDB connection error:', err));

// Session configuration

app.use(session({

    secret: process.env.SESSION\_SECRET || 'your-secret-key',

    resave: false,

    saveUninitialized: false,

    store: MongoStore.create({

        mongoUrl: process.env.MONGODB\_URI || 'mongodb://localhost:27017/photostore',

        ttl: 24 \* 60 \* 60 // 1 day

    }),

    cookie: {

        maxAge: 24 \* 60 \* 60 \* 1000 // 1 day

    }

}));

// Make user data available to all templates

app.use((req, res, next) => {

    res.locals.user = req.session.user || null;

    next();

});

// Sample data (in-memory storage)

const photos = [

  { id: 1, title: 'Mountain View', category: 'Nature', price: 29.99, filename: 'mountain.jpg' },

  { id: 2, title: 'Ocean Waves', category: 'Nature', price: 34.99, filename: 'ocean.jpg' },

  { id: 3, title: 'City Lights', category: 'Urban', price: 39.99, filename: 'city.jpg' },

  { id: 4, title: 'Wildlife', category: 'Animals', price: 44.99, filename: 'wildlife.jpg' },

  { id: 5, title: 'Portrait', category: 'People', price: 49.99, filename: 'portrait.jpg' },

  { id: 6, title: 'Abstract', category: 'Art', price: 54.99, filename: 'abstract.jpg' },

];

let cart = [];

// Middleware

app.use(express.static('public'));

app.use(express.json());

app.use(express.urlencoded({ extended: true }));

// EJS setup

app.use(expressLayouts);

app.set('view engine', 'ejs');

app.set('views', path.join(\_\_dirname, 'views'));

app.set('layout', 'layouts/main');

// Background images configuration

const backgroundImages = {

    home: '/images/hero-bg.jpg',

    gallery: '/images/gallery-bg.jpg',

    about: '/images/about-bg.jpg',

    contact: '/images/contact-bg.jpg',

    login: '/images/login-bg.jpg',

    cart: '/images/cart-bg.jpg',

    default: '/images/default-bg.jpg'

};

// Set background image middleware

app.use((req, res, next) => {

    // Get the current route name

    let route = req.path.substring(1) || 'home';

    route = route.split('/')[0]; // Get the first part of the path

    // Set the background image based on the route

    res.locals.backgroundImage = backgroundImages[route] || backgroundImages.default;

    next();

});

// Import User model

const User = require('./models/User');

// Authentication Middleware

const isAuthenticated = (req, res, next) => {

    if (req.session.user) {

        return next();

    }

    res.redirect('/login');

};

// Authentication Routes

app.post('/register', async (req, res) => {

    console.log('=== NEW REGISTRATION ATTEMPT ===');

    console.log('Request body:', JSON.stringify(req.body, null, 2));

    try {const { firstName, lastName, email, password, confirmPassword } = req.body;

        // Basic validation

        if (!firstName || !lastName || !email || !password || !confirmPassword) {

            const missingFields = [];

            if (!firstName) missingFields.push('firstName');

            if (!lastName) missingFields.push('lastName');

            if (!email) missingFields.push('email');

            if (!password) missingFields.push('password');

            if (!confirmPassword) missingFields.push('confirmPassword');

            console.log('Validation failed - Missing fields:', missingFields);

            return res.status(400).json({

                success: false,

                message: 'All fields are required',

                missingFields

            });

        }

        if (password !== confirmPassword) {

            console.log('Validation failed - Passwords do not match');

            return res.status(400).json({

                success: false,

                message: 'Passwords do not match'

            });

        }

        // Check if user exists

        console.log('Checking if user exists:', email);

        const existingUser = await User.findOne({ email });

        if (existingUser) {

            console.log('Registration failed - Email already exists:', email);

            return res.status(400).json({

                success: false,

                message: 'Email already registered'

            });

        }        // Create new user

        console.log('Creating new user...');

        const user = new User({

            firstName: firstName.trim(),

            lastName: lastName.trim(),

            email: email.trim().toLowerCase(),

            password: password

        });

        // Save user to database

        console.log('Saving user to database...');

        await user.save();

        console.log('User created successfully:', user.email);

        // Set user session

        req.session.user = {

            id: user.\_id,

            firstName: user.firstName,

            lastName: user.lastName,

            email: user.email

        };

        console.log('Session created, registration successful');

        return res.json({

            success: true,

            message: 'Registration successful',

            redirect: '/',

            userId: user.\_id

        });

    } catch (error) {

        console.error('=== REGISTRATION ERROR ===');

        console.error('Error name:', error.name);

        console.error('Error message:', error.message);

        console.error('Error code:', error.code);

        console.error('Error stack:', error.stack);

        if (error.name === 'ValidationError') {

            const messages = Object.values(error.errors).map(val => val.message);

            console.error('Validation errors:', messages);

            return res.status(400).json({

                success: false,

                message: 'Validation error',

                errors: messages

            });

        }

        if (error.code === 11000) {

            console.error('Duplicate key error - Email already exists');

            return res.status(400).json({

                success: false,

                message: 'Email already registered',

                duplicate: true

            });

        }

        console.error('Unexpected error during registration:', error);

        return res.status(500).json({

            success: false,

            message: 'An unexpected error occurred. Please try again.',

            error: process.env.NODE\_ENV === 'development' ? error.message : undefined

        });

    }

});

app.post('/login', async (req, res) => {

    try {

        console.log('Login attempt:', req.body.email);

        const { email, password } = req.body;

        if (!email || !password) {

            console.log('Email and password are required');

            return res.status(400).json({

                success: false,

                message: 'Email and password are required'

            });

        }

        // Find user by email

        const user = await User.findOne({ email });

        if (!user) {

            console.log('User not found:', email);

            return res.status(400).json({

                success: false,

                message: 'Invalid email or password'

            });

        }

        // Check password

        const isMatch = await user.comparePassword(password);

        if (!isMatch) {

            console.log('Invalid password for user:', email);

            return res.status(400).json({

                success: false,

                message: 'Invalid email or password'

            });

        }

        // Set user session

        req.session.user = {

            id: user.\_id,

            firstName: user.firstName,

            lastName: user.lastName,

            email: user.email

        };

        console.log('Login successful, redirecting to /');

        res.json({ success: true, redirect: '/' });

    } catch (error) {

        console.error('Login error:', error);

        res.status(500).json({

            success: false,

            message: 'An error occurred during login. Please try again.'

        });

    }

});

app.get('/logout', (req, res) => {

    req.session.destroy(err => {

        if (err) {

            console.error('Error destroying session:', err);

            return res.redirect('/');

        }

        res.redirect('/login');

    });

});

// Routes

app.get('/', (req, res) => {

    const featuredPhotos = photos.slice(0, 8);

    res.render('index', {

        title: 'Home',

        featuredPhotos,

        backgroundImage: backgroundImages.home

    });

});

app.get('/gallery', (req, res) => {

    res.render('gallery', {

        title: 'Gallery',

        photos,

        categories: [...new Set(photos.map(photo => photo.category))],

        backgroundImage: backgroundImages.gallery

    });

});

app.get('/about', (req, res) => {

    res.render('about', {

        title: 'About Us',

        backgroundImage: backgroundImages.about

    });

});

app.get('/contact', (req, res) => {

    res.render('contact', {

        title: 'Contact Us',

        backgroundImage: backgroundImages.contact

    });

});

app.get('/login', (req, res) => {

    res.render('login', {

        title: 'Login / Register',

        backgroundImage: backgroundImages.login

    });

});

app.get('/cart', (req, res) => {

  const cartItems = cart.map(item => {

    const photo = photos.find(p => p.id === item.photoId);

    return { ...photo, quantity: item.quantity };

  });

  const total = cartItems.reduce((sum, item) => sum + (item.price \* item.quantity), 0);

  res.render('cart', {

    title: 'Shopping Cart',

    cartItems,

    total: total.toFixed(2)

  });

});

// API Routes

app.post('/api/cart/add', (req, res) => {

  const { photoId } = req.body;

  const existingItem = cart.find(item => item.photoId === parseInt(photoId));

  if (existingItem) {

    existingItem.quantity += 1;

  } else {

    cart.push({ photoId: parseInt(photoId), quantity: 1 });

  }

  res.json({ success: true, cartCount: getCartCount() });

});

app.post('/api/cart/remove', (req, res) => {

  const { photoId } = req.body;

  cart = cart.filter(item => item.photoId !== parseInt(photoId));

  res.json({ success: true, cartCount: getCartCount() });

});

app.post('/api/contact', (req, res) => {

  // In a real app, you would save this to a database

  console.log('Contact form submitted:', req.body);

  res.json({ success: true, message: 'Thank you for your message! We will get back to you soon.' });

});

// Helper function

function getCartCount() {

  return cart.reduce((count, item) => count + item.quantity, 0);

}

// Start server

app.listen(PORT, () => {

  console.log(`Server is running on http://localhost:${PORT}`);

});

## **Conclusion:**

Server routes were modularized, and a middleware logger was implemented. All HTTP requests are now logged with relevant details, improving debugging and maintainability.

## **Experiment No.5**

**Problem Statement:** **Create a Data Submission Form and Handle POST Requests**

5.1 Build an HTML form (e.g., for adding items or submitting feedback) and handle its POST request using Express.

5.2 Display the submitted data dynamically on a confirmation view.

## **Objective:**

To implement an HTML form that sends user input through POST requests handled by Express.js routes.

## **Theory:**

In Express.js, form data is handled using body-parser middleware. The server reads form inputs from req.body and processes them accordingly. EJS can dynamically render submitted data for confirmation or acknowledgment messages.

## **Code:**

<!-- Contact Header -->

<div class="row mb-5">

    <div class="col-lg-8 mx-auto text-center">

        <h1 class="display-5 fw-bold mb-3">Get In Touch</h1>

        <p class="lead">Have questions or feedback? We'd love to hear from you. Fill out the form below or reach out through our social media channels.</p></div></div>

<div class="row g-5">

    <!-- Contact Form -->

    <div class="col-lg-7">

        <div class="contact-form">

            <h3 class="mb-4">Send Us a Message</h3>

            <form id="contact-form" novalidate>

                <div id="form-message" class="alert d-none" role="alert"></div>

                <div class="row g-3">

                    <div class="col-md-6">

                        <label for="name" class="form-label">Your Name <span class="text-danger">\*</span></label>

                        <input type="text" class="form-control" id="name" name="name" required>

                        <div class="invalid-feedback">Please enter your name.</div></div>

                    <div class="col-md-6">

                        <label for="email" class="form-label">Email Address <span class="text-danger">\*</span></label>

                        <input type="email" class="form-control" id="email" name="email" required>

                        <div class="invalid-feedback">Please enter a valid email address.</div></div></div>

                <div class="mb-3 mt-3">

                    <label for="subject" class="form-label">Subject</label>

                    <input type="text" class="form-control" id="subject" name="subject"></div>

                <div class="mb-3">

                    <label for="message" class="form-label">Your Message <span class="text-danger">\*</span></label>

                    <textarea class="form-control" id="message" name="message" rows="5" required></textarea>

                    <div class="invalid-feedback">Please enter your message.</div></div>

                <div class="d-grid">

                    <button type="submit" class="btn btn-primary btn-lg">

                        <span class="submit-text">Send Message</span>

                        <span class="spinner-border spinner-border-sm d-none" role="status" aria-hidden="true"></span>

                    </button></div>

            </form></div></div>

    <!-- Contact Information -->

    <div class="col-lg-5">

        <div class="bg-light p-4 rounded h-100">

            <h3 class="mb-4">Contact Information</h3>

            <p>We're here to help and answer any questions you might have. We look forward to hearing from you!</p>

            <ul class="list-unstyled mt-4">

                <li class="mb-4">

                    <div class="d-flex">

                        <div class="me-3">

                            <div class="bg-primary bg-opacity-10 d-flex align-items-center justify-content-center rounded-circle" style="width: 50px; height: 50px;">

                                <i class="fas fa-map-marker-alt text-primary"></i></div></div>

                        <div>

                            <h5 class="mb-1">Our Location</h5>

                            <p class="text-muted mb-0">123 Photography Street, City, Country</p>

                        </div></div></li>

                <li class="mb-4">

                    <div class="d-flex">

                        <div class="me-3">

                            <div class="bg-primary bg-opacity-10 d-flex align-items-center justify-content-center rounded-circle" style="width: 50px; height: 50px;">

                                <i class="fas fa-envelope text-primary"></i></div></div>

                        <div>

                            <h5 class="mb-1">Email Us</h5>

                            <p class="text-muted mb-0">info@photostore.com<br>support@photostore.com</p>

                        </div></div></li>

                <li class="mb-4">

                    <div class="d-flex">

                        <div class="me-3">

                            <div class="bg-primary bg-opacity-10 d-flex align-items-center justify-content-center rounded-circle" style="width: 50px; height: 50px;">

                                <i class="fas fa-phone-alt text-primary"></i></div></div>

                        <div>

                            <h5 class="mb-1">Call Us</h5>

                            <p class="text-muted mb-0">+1 (555) 123-4567<br>Mon - Fri, 9:00 AM - 6:00 PM</p>

                        </div></div></li></ul>

            <div class="mt-5">

                <h5 class="mb-3">Follow Us</h5>

                <div class="social-links">

                    <a href="#" class="text-primary me-3" data-bs-toggle="tooltip" title="Facebook">

                        <i class="fab fa-facebook-f fa-lg"></i></a>

                    <a href="#" class="text-primary me-3" data-bs-toggle="tooltip" title="Instagram">

                        <i class="fab fa-instagram fa-lg"></i></a>

                    <a href="#" class="text-primary me-3" data-bs-toggle="tooltip" title="Twitter">

                        <i class="fab fa-twitter fa-lg"></i></a>

                    <a href="#" class="text-primary me-3" data-bs-toggle="tooltip" title="Pinterest">

                        <i class="fab fa-pinterest-p fa-lg"></i></a>

                    <a href="#" class="text-primary" data-bs-toggle="tooltip" title="YouTube">

                        <i class="fab fa-youtube fa-lg"></i>

                    </a></div></div></div></div></div>

<!-- Map Section -->

<div class="row mt-5">

    <div class="col-12">

        <div class="map-container rounded overflow-hidden" style="height: 400px; background-color: #f8f9fa;">

            <!-- You can embed a Google Map here -->

            <div class="h-100 d-flex align-items-center justify-content-center bg-light">

                <div class="text-center">

                    <i class="fas fa-map-marked-alt fa-3x text-muted mb-3"></i>

                    <h5>Our Location</h5>

                    <p class="text-muted">123 Photography Street, City, Country</p>

                </div></div></div></div></div>

<!-- FAQ Section -->

<section class="py-5 mt-5 bg-light">

    <div class="container">

        <div class="text-center mb-5">

            <h2>Frequently Asked Questions</h2>

            <p class="lead">Find answers to common questions about our services</p></div>

        <div class="row justify-content-center">

            <div class="col-lg-8">

                <div class="accordion" id="faqAccordion">

                    <div class="accordion-item mb-3 border-0 shadow-sm">

                        <h3 class="accordion-header" id="headingOne">

                            <button class="accordion-button" type="button" data-bs-toggle="collapse" data-bs-target="#collapseOne">How do I purchase a photo? </button></h3>

                        <div id="collapseOne" class="accordion-collapse collapse show" data-bs-parent="#faqAccordion">

                            <div class="accordion-body">

                                Purchasing a photo is easy! Simply browse our gallery, select the photo you love, and click the "Add to Cart" button. You can review your selection in the cart before proceeding to checkout. We accept various payment methods including credit cards and PayPal. </div></div></div>

                    <div class="accordion-item mb-3 border-0 shadow-sm">

                        <h3 class="accordion-header" id="headingTwo">

                            <button class="accordion-button collapsed" type="button" data-bs-toggle="collapse" data-bs-target="#collapseTwo">What file formats are available for download? </button>

                        </h3>

                        <div id="collapseTwo" class="accordion-collapse collapse" data-bs-parent="#faqAccordion">

                            <div class="accordion-body">We offer high-resolution JPEG files for all our photos. Some photos may also be available in RAW format upon request. After purchase, you'll receive an email with a download link where you can access your purchased photos in the available formats. </div></div></div>

                    <div class="accordion-item mb-3 border-0 shadow-sm">

                        <h3 class="accordion-header" id="headingThree">

                            <button class="accordion-button collapsed" type="button" data-bs-toggle="collapse" data-bs-target="#collapseThree">Can I request a custom photo shoot? </button>

                        </h3>

                        <div id="collapseThree" class="accordion-collapse collapse" data-bs-parent="#faqAccordion">

                            <div class="accordion-body">

                                Yes! We offer custom photo shoots based on your specific requirements. Please use the contact form to provide details about your project, and our team will get back to you with availability and pricing information. </div></div></div>

                    <div class="accordion-item border-0 shadow-sm">

                        <h3 class="accordion-header" id="headingFour">

                            <button class="accordion-button collapsed" type="button" data-bs-toggle="collapse" data-bs-target="#collapseFour">What is your return policy? </button></h3>

                        <div id="collapseFour" class="accordion-collapse collapse" data-bs-parent="#faqAccordion">

                            <div class="accordion-body">

                                Due to the digital nature of our products, we don't offer refunds for downloaded photos. However, if you encounter any issues with your download or the quality of the photos, please contact our support team within 7 days of purchase, and we'll be happy to assist you.

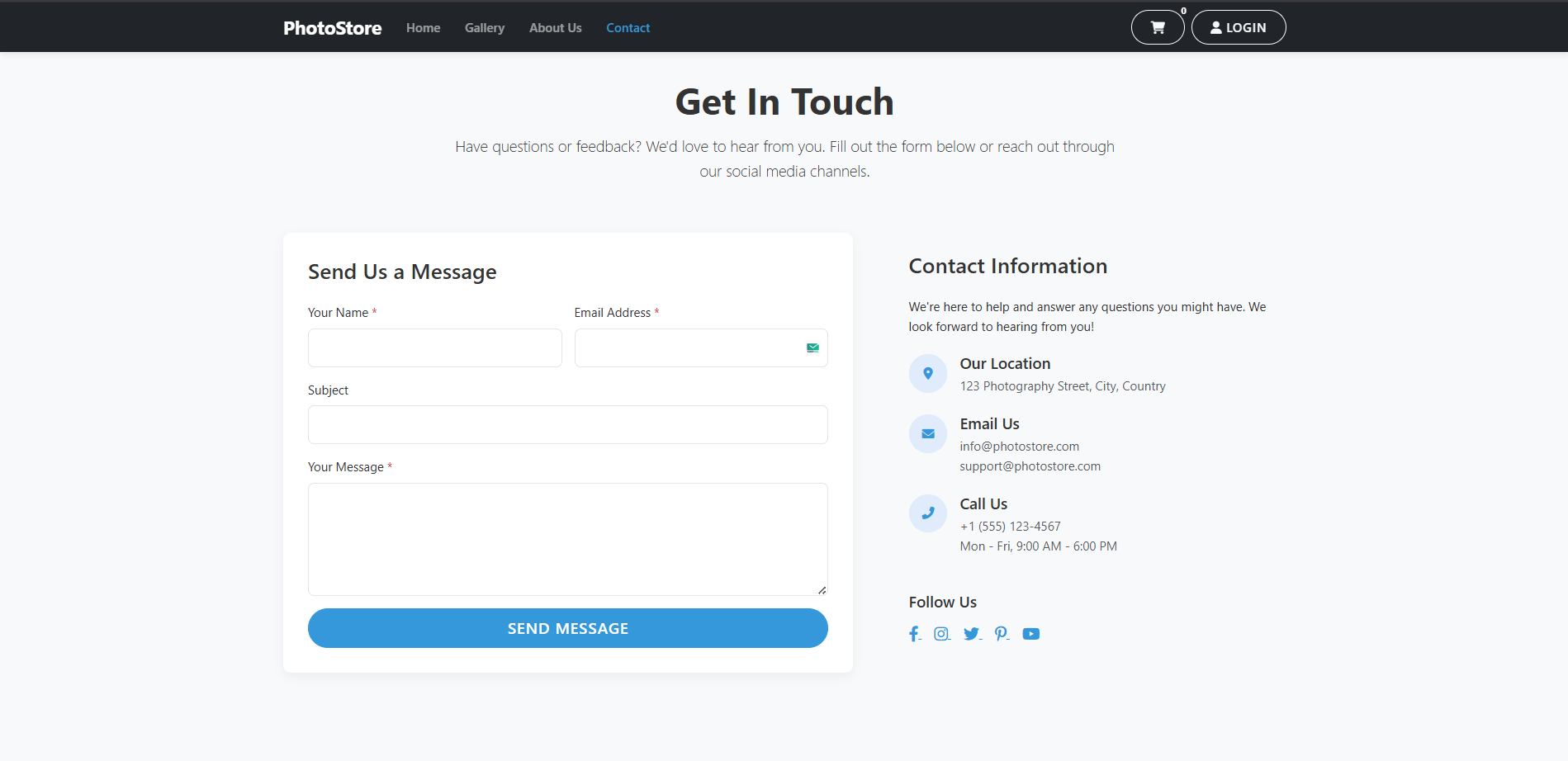
                            </div> </div></div></div>

          <div class="text-center mt-5">

                    <p class="mb-0">Still have questions? <a href="#contact-form" class="text-primary">Contact us</a> and we'll get back to you as soon as possible.</p></div></div></div></div>

</section>

Output:



## **Conclusion:**

A functional form was created for collecting user feedback. The POST request was handled successfully, and the submitted data was displayed on a confirmation page.

## **Experiment No.6**

**Problem Statement:** **Serve Dynamic Content Using Server-Side Rendering**

6.1 Render a list of items (e.g., products, posts, services) dynamically using EJS with hardcoded or temporary in-memory data.

6.2 Route should accept data from the server and render it in the view using loops.

## **Objective:**

To use EJS templates and in-memory data arrays to display dynamic content on the web page.

## **Theory:**

EJS (Embedded JavaScript Templates) allows embedding server-side data directly into HTML. Using <% %> and <%= %> tags, dynamic data (e.g., photo titles, descriptions) can be looped and rendered using res.render().

## **Code:**

* 1. Product page:

<!-- Gallery Header -->

<div class="row mb-5">

<div class="col-lg-8 mx-auto text-center">

<h1 class="display-5 fw-bold mb-3">Our Photo Gallery</h1>

<p class="lead">Discover our curated collection of stunning photographs. Filter by category or browse through all our photos.</p>

</div>

</div>

<!-- Filter Buttons -->

<div class="filter-buttons justify-content-center mb-5">

<button class="filter-btn active" data-filter="all">All Photos</button>

<% categories.forEach(category => { %>

<button class="filter-btn" data-filter="<%= category.toLowerCase() %>"><%= category %></button>

<% }); %>

</div>

<!-- Photo Grid -->

<div class="row g-4">

<% photos.forEach(photo => { %>

<div class="col-md-6 col-lg-4 col-xl-3 photo-item" data-categories="<%= photo.category.toLowerCase() %>">

<div class="photo-card h-100">

<div class="photo-img-container">

<img

src="/images/<%= photo.filename %>"

alt="<%= photo.title %>"

class="photo-img"

loading="lazy"

data-bs-toggle="modal"

data-bs-target="#photoModal"

data-title="<%= photo.title %>"

data-category="<%= photo.category %>"

data-price="<%= photo.price.toFixed(2) %>"

data-src="/images/<%= photo.filename %>"

>

<div class="photo-overlay">

<button class="btn btn-outline-light view-photo"

data-bs-toggle="modal"

data-bs-target="#photoModal"

data-title="<%= photo.title %>"

data-category="<%= photo.category %>"

data-price="<%= photo.price.toFixed(2) %>"

data-src="/images/<%= photo.filename %>">

<i class="fas fa-search-plus me-2"></i> View

</button>

</div>

</div>

<div class="card-body">

<h5 class="card-title"><%= photo.title %></h5>

<p class="card-text text-muted"><%= photo.category %></p>

<div class="d-flex justify-content-between align-items-center">

<span class="h5 mb-0">$<%= photo.price.toFixed(2) %></span>

<button class="btn btn-sm btn-primary add-to-cart" data-id="<%= photo.id %>">

<i class="fas fa-cart-plus me-1"></i> Add to Cart

</button>

</div>

</div>

</div>

</div>

<% }); %>

</div>

<!-- Empty State -->

<div id="no-results" class="text-center py-5 d-none">

<div class="mb-4">

<i class="fas fa-images fa-4x text-muted mb-3"></i>

<h3>No photos found</h3>

<p class="text-muted">Try adjusting your search or filter to find what you're looking for.</p>

<button class="btn btn-primary mt-2 filter-btn" data-filter="all">Show All Photos</button>

</div>

</div>

<!-- Photo Modal -->

<div class="modal fade" id="photoModal" tabindex="-1" aria-hidden="true">

<div class="modal-dialog modal-dialog-centered modal-lg">

<div class="modal-content">

<div class="modal-header">

<h5 class="modal-title" id="photoModalTitle">Photo Title</h5>

<button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>

</div>

<div class="modal-body p-0">

<div class="row g-0">

<div class="col-md-8">

<img src="" alt="" class="img-fluid" id="modalPhoto">

</div>

<div class="col-md-4 p-4">

<div class="d-flex flex-column h-100">

<div class="mb-3">

<span class="badge bg-primary mb-2" id="modalCategory">Category</span>

<h4 id="modalTitle">Photo Title</h4>

<p class="text-muted mb-0">High-resolution digital download</p>

</div>

<div class="mt-auto">

<div class="d-flex justify-content-between align-items-center mb-3">

<h3 class="mb-0" id="modalPrice">Rs.0.00</h3>

<button class="btn btn-primary add-to-cart" id="modalAddToCart">

<i class="fas fa-cart-plus me-1"></i> Add to Cart

</button>

</div>

<div class="d-grid">

<a href="/cart" class="btn btn-outline-primary">

<i class="fas fa-shopping-cart me-1"></i> View Cart

</a>

</div></div></div></div></div></div></div></div></div>

<!-- Add this script at the bottom of the file -->

<script>

document.addEventListener('DOMContentLoaded', function() {

// Initialize photo modal

const photoModal = document.getElementById('photoModal');

if (photoModal) {

photoModal.addEventListener('show.bs.modal', function(event) {

const button = event.relatedTarget;

const title = button.getAttribute('data-title');

const category = button.getAttribute('data-category');

const price = button.getAttribute('data-price');

const src = button.getAttribute('data-src');

const modalTitle = photoModal.querySelector('.modal-title');

const modalCategory = photoModal.querySelector('#modalCategory');

const modalPrice = photoModal.querySelector('#modalPrice');

const modalPhoto = photoModal.querySelector('#modalPhoto');

const addToCartBtn = photoModal.querySelector('#modalAddToCart');

modalTitle.textContent = title;

modalCategory.textContent = category;

modalPrice.textContent = `Rs.${price}`;

modalPhoto.src = src;

modalPhoto.alt = title;

// Set up add to cart button

const photoId = button.closest('.photo-card')?.querySelector('.add-to-cart')?.dataset.id;

if (photoId) {

addToCartBtn.onclick = function() {

addToCart(photoId);

const toast = new bootstrap.Toast(document.querySelector('.toast'));

toast.show();

};

}

});

}

// Handle filter buttons

const filterButtons = document.querySelectorAll('.filter-btn');

const photoItems = document.querySelectorAll('.photo-item');

const noResults = document.getElementById('no-results');

filterButtons.forEach(button => {

button.addEventListener('click', function() {

const filter = this.getAttribute('data-filter');

let visibleCount = 0;

photoItems.forEach(item => {

const categories = item.getAttribute('data-categories');

if (filter === 'all' || categories.includes(filter)) {

item.style.display = 'block';

item.classList.add('animate\_\_animated', 'animate\_\_fadeIn');

visibleCount++;

} else {

item.style.display = 'none';

}

});

// Show/hide no results message

if (noResults) {

if (visibleCount === 0) {

noResults.classList.remove('d-none');

} else {

noResults.classList.add('d-none');

}

}

// Update active button

filterButtons.forEach(btn => btn.classList.remove('active'));

this.classList.add('active');

// Scroll to top of gallery

document.querySelector('.gallery-container')?.scrollIntoView({ behavior: 'smooth' });

});

});

});

</script>

## **Output:**

* 1. Contact page output:



## **Conclusion:**

Dynamic photo data was successfully rendered on the gallery page using EJS. This approach enables efficient server-side rendering without requiring a database.

## **Experiment No.7**

**Problem Statement:** **Create Dynamic Detail Pages Using URL Parameters**

7.1 Use route parameters to render individual item details dynamically via EJS.

7.2 Display properties like title, description, and date using passed route data.

## **Objective:**

To use route parameters in Express to display details for specific photos dynamically.

## **Theory:**

Express allows defining route parameters using a colon syntax (e.g., /photo/:id). The value can be accessed using req.params.id. By matching this ID with in-memory photo data, EJS can render details like title, description, and price.

## **Code:**

* 1. Product page:

<!-- Gallery Header -->

<div class="row mb-5">

<div class="col-lg-8 mx-auto text-center">

<h1 class="display-5 fw-bold mb-3">Our Photo Gallery</h1>

<p class="lead">Discover our curated collection of stunning photographs. Filter by category or browse through all our photos.</p></div></div>

<!-- Filter Buttons -->

<div class="filter-buttons justify-content-center mb-5">

<button class="filter-btn active" data-filter="all">All Photos</button>

<% categories.forEach(category => { %>

<button class="filter-btn" data-filter="<%= category.toLowerCase() %>"><%= category %></button>

<% }); %>

</div>

<!-- Photo Grid -->

<div class="row g-4">

<% photos.forEach(photo => { %>

<div class="col-md-6 col-lg-4 col-xl-3 photo-item" data-categories="<%= photo.category.toLowerCase() %>">

<div class="photo-card h-100">

<div class="photo-img-container">

<img src="/images/<%= photo.filename %>" alt="<%= photo.title %>"

class="photo-img" loading="lazy" data-bs-toggle="modal"

data-bs-target="#photoModal" data-title="<%= photo.title %>"

data-category="<%= photo.category %>" data-price="<%= photo.price.toFixed(2) %>"

data-src="/images/<%= photo.filename %>">

<div class="photo-overlay">

<button class="btn btn-outline-light view-photo"

data-bs-toggle="modal" data-bs-target="#photoModal"

data-title="<%= photo.title %>" data-category="<%= photo.category %>"

data-price="<%= photo.price.toFixed(2) %>" data-src="/images/<%= photo.filename %>">

<i class="fas fa-search-plus me-2"></i> View

</button></div></div>

<div class="card-body">

<h5 class="card-title"><%= photo.title %></h5>

<p class="card-text text-muted"><%= photo.category %></p>

<div class="d-flex justify-content-between align-items-center">

<span class="h5 mb-0">$<%= photo.price.toFixed(2) %></span>

<button class="btn btn-sm btn-primary add-to-cart" data-id="<%= photo.id %>">

<i class="fas fa-cart-plus me-1"></i> Add to Cart

</button></div></div></div></div><% }); %></div>

<!-- Empty State -->

<div id="no-results" class="text-center py-5 d-none">

<div class="mb-4">

<i class="fas fa-images fa-4x text-muted mb-3"></i>

<h3>No photos found</h3>

<p class="text-muted">Try adjusting your search or filter to find what you're looking for.</p>

<button class="btn btn-primary mt-2 filter-btn" data-filter="all">Show All Photos</button>

</div></div>

<!-- Photo Modal -->

<div class="modal fade" id="photoModal" tabindex="-1" aria-hidden="true">

<div class="modal-dialog modal-dialog-centered modal-lg">

<div class="modal-content">

<div class="modal-header">

<h5 class="modal-title" id="photoModalTitle">Photo Title</h5>

<button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button></div>

<div class="modal-body p-0">

<div class="row g-0">

<div class="col-md-8">

<img src="" alt="" class="img-fluid" id="modalPhoto"></div>

<div class="col-md-4 p-4">

<div class="d-flex flex-column h-100">

<div class="mb-3">

<span class="badge bg-primary mb-2" id="modalCategory">Category</span>

<h4 id="modalTitle">Photo Title</h4>

<p class="text-muted mb-0">High-resolution digital download</p></div>

<div class="mt-auto">

<div class="d-flex justify-content-between align-items-center mb-3">

<h3 class="mb-0" id="modalPrice">Rs.0.00</h3>

<button class="btn btn-primary add-to-cart" id="modalAddToCart">

<i class="fas fa-cart-plus me-1"></i> Add to Cart</button></div>

<div class="d-grid">

<a href="/cart" class="btn btn-outline-primary">

<i class="fas fa-shopping-cart me-1"></i> View Cart</a></div></div></div></div></div></div></div></div></div>

<!-- Add this script at the bottom of the file -->

<script>

document.addEventListener('DOMContentLoaded', function() {

// Initialize photo modal

const photoModal = document.getElementById('photoModal');

if (photoModal) {

photoModal.addEventListener('show.bs.modal', function(event) {

const button = event.relatedTarget;

const title = button.getAttribute('data-title');

const category = button.getAttribute('data-category');

const price = button.getAttribute('data-price');

const src = button.getAttribute('data-src');

const modalTitle = photoModal.querySelector('.modal-title');

const modalCategory = photoModal.querySelector('#modalCategory');

const modalPrice = photoModal.querySelector('#modalPrice');

const modalPhoto = photoModal.querySelector('#modalPhoto');

const addToCartBtn = photoModal.querySelector('#modalAddToCart');

modalTitle.textContent = title;

modalCategory.textContent = category;

modalPrice.textContent = `Rs.${price}`;

modalPhoto.src = src;

modalPhoto.alt = title;

// Set up add to cart button

const photoId = button.closest('.photo-card')?.querySelector('.add-to-cart')?.dataset.id;

if (photoId) {

addToCartBtn.onclick = function() {

addToCart(photoId);

const toast = new bootstrap.Toast(document.querySelector('.toast'));

toast.show();

};}});}

// Handle filter buttons

const filterButtons = document.querySelectorAll('.filter-btn');

const photoItems = document.querySelectorAll('.photo-item');

const noResults = document.getElementById('no-results');

filterButtons.forEach(button => {

button.addEventListener('click', function() {

const filter = this.getAttribute('data-filter');

let visibleCount = 0;

photoItems.forEach(item => {

const categories = item.getAttribute('data-categories');

if (filter === 'all' || categories.includes(filter)) {

item.style.display = 'block';

item.classList.add('animate\_\_animated', 'animate\_\_fadeIn');

visibleCount++;

} else {item.style.display = 'none';}});

// Show/hide no results message

if (noResults) {

if (visibleCount === 0) {

noResults.classList.remove('d-none');

} else {noResults.classList.add('d-none');}}

// Update active button

filterButtons.forEach(btn => btn.classList.remove('active'));

this.classList.add('active');

// Scroll to top of gallery

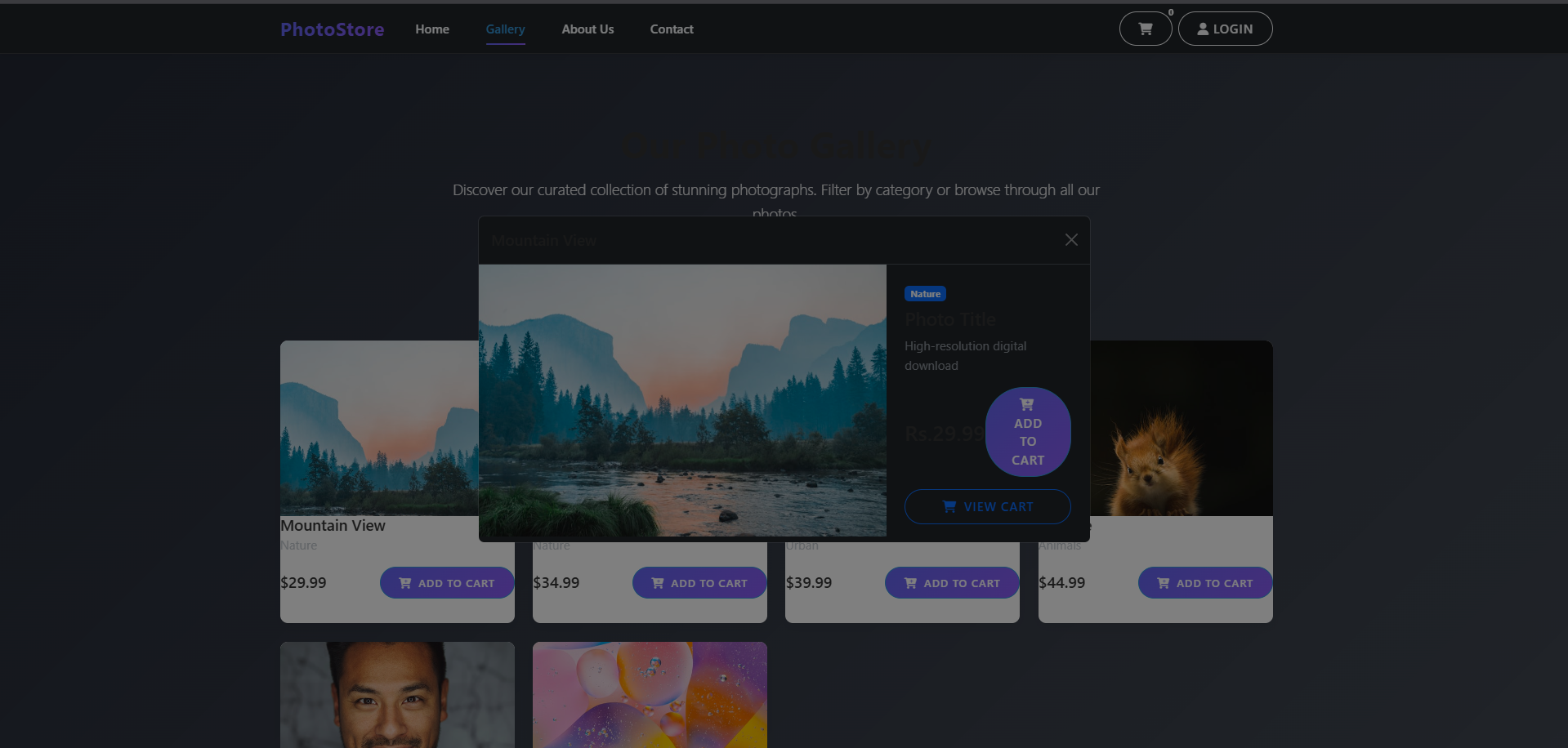
document.querySelector('.gallery-container')?.scrollIntoView({ behavior: 'smooth' });

});})});

</script>

## **Output:**

* 1. Contact page output:



## **Conclusion:**

Dynamic detail pages were created successfully using route parameters. Each photo page displays its unique information, improving interactivity and user engagement.

## **Experiment No.8**

**Problem Statement:** **Build a Reusable Layout Using EJS Partials**

8.1 Create common EJS partials (header, footer, navbar) and include them across views.

8.2 Maintain consistency across all pages by using a shared layout template.

## **Objective:**

To enhance maintainability and consistency by using EJS partials in multiple views.

## **Theory:**

EJS partials are smaller template files that can be included in multiple views using <%- include('partials/header') %>. This allows reusing common UI sections like navigation and footers, ensuring consistent design.

## **Code:**

Main page:

Code:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>PhotoStore - <%= title %></title>

    <!-- Bootstrap CSS -->

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">

    <!-- Font Awesome -->

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css">

    <!-- Custom CSS -->

    <link rel="stylesheet" href="/css/style.css">

</head>

<body>

    <!-- Navigation -->

    <nav class="navbar navbar-expand-lg navbar-dark bg-dark fixed-top">

        <div class="container">

            <a class="navbar-brand" href="/">PhotoStore</a>

            <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav">

                <span class="navbar-toggler-icon"></span>

            </button>

            <div class="collapse navbar-collapse" id="navbarNav">

                <ul class="navbar-nav me-auto">

                    <li class="nav-item">

                        <a class="nav-link <%= title === 'Home' ? 'active' : '' %>" href="/">Home</a>

                    </li>

                    <li class="nav-item">

                        <a class="nav-link <%= title === 'Gallery' ? 'active' : '' %>" href="/gallery">Gallery</a>

                    </li>

                    <li class="nav-item">

                        <a class="nav-link <%= title === 'About Us' ? 'active' : '' %>" href="/about">About Us</a>

                    </li>

                    <li class="nav-item">

                        <a class="nav-link <%= title === 'Contact Us' ? 'active' : '' %>" href="/contact">Contact</a>

                    </li>

                </ul>

                <div class="d-flex">

                    <a href="/cart" class="btn btn-outline-light position-relative me-2">

                        <i class="fas fa-shopping-cart"></i>

                        <span class="position-absolute top-0 start-100 translate-middle badge rounded-pill bg-danger" id="cart-count">

                            <%- typeof cartCount !== 'undefined' ? cartCount : '0' %>

                        </span>

                    </a>

                    <a href="/login" class="btn btn-outline-light">

                        <i class="fas fa-user"></i> Login

                    </a></div></div></div></nav>

    <!-- Main Content -->

    <main class="container py-5 mt-5">

        <%- body %>

    </main>

    <!-- Footer -->

    <footer class="bg-dark text-white py-4 mt-5">

        <div class="container">

            <div class="row">

                <div class="col-md-6">

                    <h5>PhotoStore</h5>

                    <p>Your destination for beautiful photography.</p>

                </div>

                <div class="col-md-3">

                    <h5>Quick Links</h5>

                    <ul class="list-unstyled">

                        <li><a href="/" class="text-white">Home</a></li>

                        <li><a href="/gallery" class="text-white">Gallery</a></li>

                        <li><a href="/about" class="text-white">About Us</a></li>

                        <li><a href="/contact" class="text-white">Contact</a></li>

                    </ul>

                </div>

                <div class="col-md-3">

                    <h5>Connect</h5>

                    <div class="social-links">

                        <a href="#" class="text-white me-2"><i class="fab fa-facebook-f"></i></a>

                        <a href="#" class="text-white me-2"><i class="fab fa-instagram"></i></a>

                        <a href="#" class="text-white me-2"><i class="fab fa-twitter"></i></a>

                        <a href="#" class="text-white"><i class="fab fa-pinterest"></i></a>

                    </div>

                </div>

            </div>

            <hr class="my-4">

            <div class="text-center">

                <p class="mb-0">&copy; 2025 PhotoStore. All rights reserved.</p>

            </div>

        </div>

    </footer>

    <!-- Bootstrap JS and dependencies -->

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>

    <!-- Custom JS -->

    <script src="/js/main.js"></script>

</body>

</html>

## **Conclusion:**

Reusable EJS partials were implemented successfully. All pages in the PhotoStore project now share a common layout and design structure, improving consistency and reducing redundancy.

## **Experiment No.9**

**Problem Statement:** **Connect Node.js to MySQL and Perform Basic CRUD**

9.1 Create MySQL tables and connect to them using a MySQL Node.js package.

9.2 Implement basic CRUD operations from Express routes to the database.

## **Objective:**

To connect to a MySQL database and perform Create, Read, Update, and Delete operations using Express routes.

## **Theory:**

MySQL can be connected to Node.js using libraries like mysql2 or sequelize. CRUD operations are performed through SQL queries inside Express route handlers, enabling persistent storage of photo and user data.

## **Code:**

* 1. App page:

const express = require('express');

const expressLayouts = require('express-ejs-layouts');

const path = require('path');

const bodyParser = require('body-parser');

const mysql = require('mysql2');

const bcrypt = require('bcryptjs');

const session = require('express-session');

const app = express();

const PORT = process.env.PORT || 3000;

// Database connection

const connection = mysql.createConnection({

host: 'localhost',

user: 'localapp',

password: 'Admin@123',

database: 'testapp'

});

connection.connect((err) => {

if (err) {

console.error('Error connecting to database:', err);

process.exit(1);

}

console.log('Connected to MySQL database');

// Set the views directory

app.set('views', path.join(\_\_dirname, 'views'));

app.set('view engine', 'ejs');

// Middleware

app.use(express.static(path.join(\_\_dirname, 'public')));

app.use(expressLayouts);

app.use(express.urlencoded({ extended: true }));

app.use(express.json());

// Session configuration

app.use(session({

secret: process.env.SESSION\_SECRET || 'change\_this\_secret',

resave: false,

saveUninitialized: false,

cookie: { secure: false }

}));

// Make user and cartCount available to all views

app.use((req, res, next) => {

res.locals.user = req.session.user || null;

res.locals.cartCount = req.session.cart ? req.session.cart.length : 0;

next();

});

// Static photos data

const staticPhotos = [

{ id: 1, title: 'Mountain View', category: 'Nature', price: 29.99, filename: 'mountain.jpg' },

{ id: 2, title: 'Ocean Waves', category: 'Nature', price: 34.99, filename: 'ocean.jpg' },

{ id: 3, title: 'City Lights', category: 'Urban', price: 39.99, filename: 'city.jpg' },

{ id: 4, title: 'Wildlife', category: 'Animals', price: 44.99, filename: 'wildlife.jpg' },

{ id: 5, title: 'Portrait', category: 'People', price: 49.99, filename: 'portrait.jpg' },

{ id: 6, title: 'Abstract', category: 'Art', price: 54.99, filename: 'abstract.jpg' },

];

// Routes

app.get('/', (req, res) => {

res.render('index', {

title: 'Home',

user: req.session.user || null,

cartCount: req.session.cart ? req.session.cart.length : 0,

photos: staticPhotos

});

});

app.get('/gallery', (req, res) => {

res.render('gallery', {

title: 'Gallery',

user: req.session.user || null,

cartCount: req.session.cart ? req.session.cart.length : 0

});

});

app.get('/about', (req, res) => {

res.render('about', {

title: 'About Us',

user: req.session.user || null,

cartCount: req.session.cart ? req.session.cart.length : 0

});

});

app.get('/contact', (req, res) => {

res.render('contact', {

title: 'Contact Us',

user: req.session.user || null,

cartCount: req.session.cart ? req.session.cart.length : 0

});

});

app.get('/register', (req, res) => {

res.render('register', {

title: 'Register',

message: null

});

});

app.post('/register', (req, res) => {

const { first, last, username, pass } = req.body;

if (!first || !last || !username || !pass) {

return res.render('register', {

title: 'Register',

message: 'All fields are required'

});

}

// Password validation can be added here if needed

connection.query(

'INSERT INTO user\_tl (firstname, lastname, email, password) VALUES (?, ?, ?, ?)',

[first, last, username, pass],

(err, results) => {

if (err) {

console.error('Database error:', err);

return res.render('register', {

title: 'Register',

message: 'Database error occurred.'

});

}

// Store user in session

req.session.user = {

id: results.insertId,

email: username,

firstname: first,

lastname: last

};

return res.redirect('/');

}

);

});

app.get('/login', (req, res) => {

res.render('login', {

title: 'Login',

message: null

});

});

app.post('/login', (req, res) => {

const { username, pass } = req.body;

if (!username || !pass) {

return res.render('login', {

title: 'Login',

message: 'Please enter both email and password.'

});

}

connection.query(

'SELECT \* FROM user\_tl WHERE email = ? AND password = ?',

[username, pass],

(err, results) => {

if (err) {

console.error('Database error:', err);

return res.render('login', { title: 'Login', message: 'Database error occurred.' });}

if (results.length > 0) {

// Store user in session

req.session.user = {

id: results[0].id,

email: results[0].email,

firstname: results[0].firstname,

lastname: results[0].lastname

};

return res.redirect('/');

} else {

return res.render('login', {

title: 'Login',

message: 'Invalid email or password'

});}});});

// Start the server only after database connection is established

app.listen(3000, '0.0.0.0', () => {

console.log('Server is running on http://localhost:3000');

}).on('error', (err) => {

console.error('Server Error:', err);

process.exit(1);

});

});

app.get('/cart', (req, res) => {

const cartItems = cart.map(item => {

const photo = photos.find(p => p.id === item.photoId);

return { ...photo, quantity: item.quantity };

});

const total = cartItems.reduce((sum, item) => sum + (item.price \* item.quantity), 0);

res.render('cart', {

title: 'Shopping Cart',

cartItems,

total: total.toFixed(2)

});

});

// API Routes

app.post('/api/cart/add', (req, res) => {

const { photoId } = req.body;

const existingItem = cart.find(item => item.photoId === parseInt(photoId));

if (existingItem) {

existingItem.quantity += 1;

} else {

cart.push({ photoId: parseInt(photoId), quantity: 1 });

}

res.json({ success: true, cartCount: getCartCount() });

});

app.post('/api/cart/remove', (req, res) => {

const { photoId } = req.body;

cart = cart.filter(item => item.photoId !== parseInt(photoId));

res.json({ success: true, cartCount: getCartCount() });

});

app.post('/api/contact', (req, res) => {

// In a real app, you would save this to a database

console.log('Contact form submitted:', req.body);

res.json({ success: true, message: 'Thank you for your message! We will get back to you soon.' });

});

// Helper function

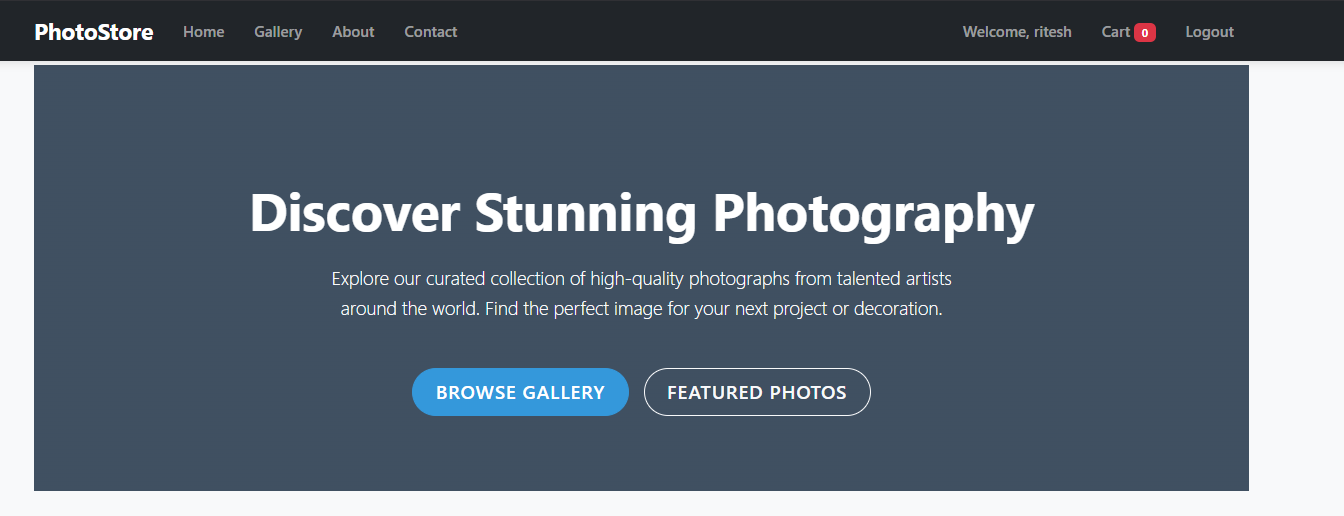
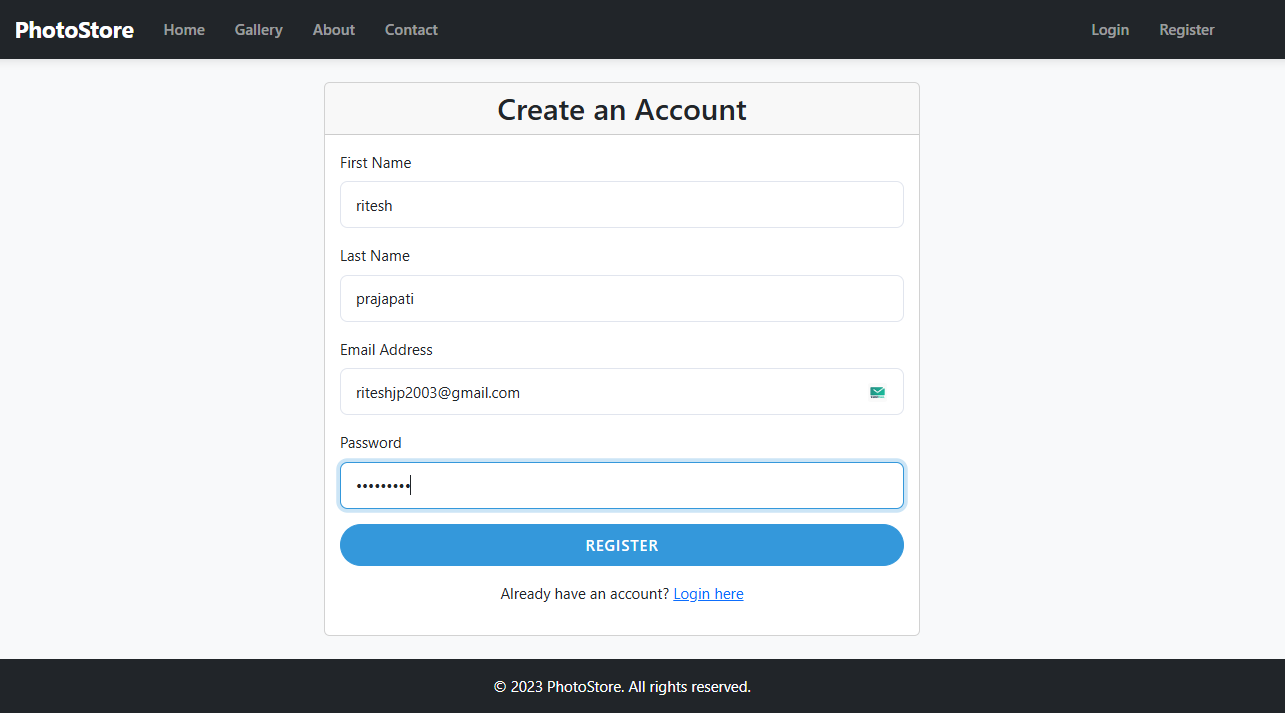
function getCartCount() {

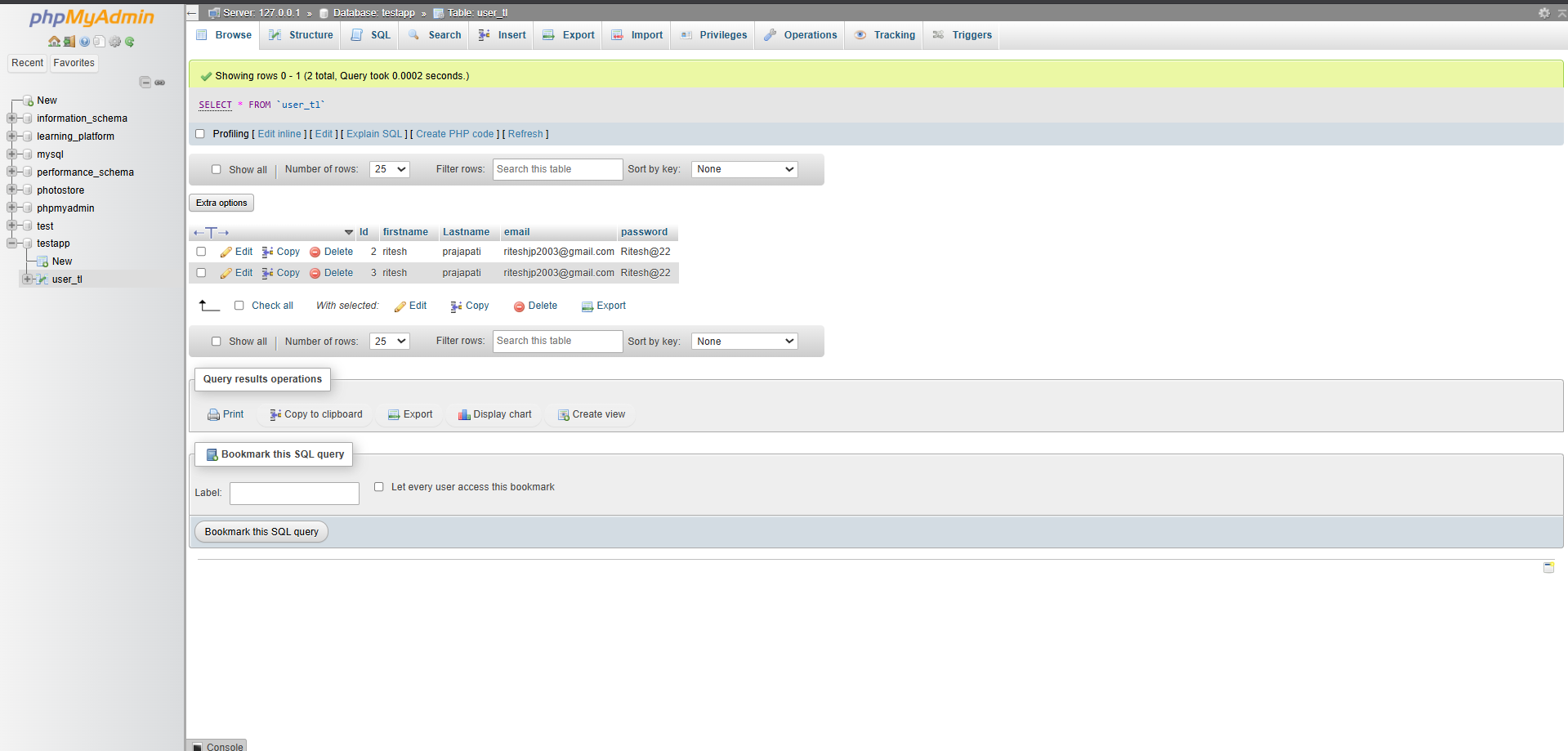
return cart.reduce((count, item) => count + item.quantity, 0);

}

## **Output:**

output:





## **Conclusion:**

A successful connection was established between Node.js and MySQL. CRUD operations were executed from Express routes, allowing the PhotoStore application to manage data dynamically.

## **Experiment No.10**

**Problem Statement:** **Develop RESTful APIs for Data Access and Manipulation**

10.1 Build a set of RESTful APIs using Express.js for performing operations.

10.2 Test the endpoints using Postman and ensure they return structured JSON responses.

## **Objective:**

To create and test RESTful API routes for PhotoStore using Express.js, returning structured JSON responses.

## **Theory:**

RESTful APIs follow HTTP methods (GET, POST, PUT, DELETE) to perform operations on resources. Express routes can be configured to send JSON responses using res.json(). Tools like Postman are used for testing API endpoints.

## **Code:**

* 1. JSON:

{

"name": "photostore",

"version": "1.0.0",

"description": "A photography gallery and marketplace web application",

"main": "app.js",

"scripts": {

"start": "node app.js",

"dev": "nodemon app.js"

},

"dependencies": {

"bcryptjs": "^3.0.3",

"body-parser": "^1.20.2",

"ejs": "^3.1.9",

"express": "^4.18.2",

"express-ejs-layouts": "^2.5.1",

"express-session": "^1.18.2",

"mysql": "^2.18.1",

"mysql2": "^3.15.3"

},

"devDependencies": {

"nodemon": "^3.0.2"

},

"engines": {

"node": ">=14.0.0"

}

}

## **Conclusion:**

RESTful APIs for PhotoStore were successfully developed and tested. The APIs return well-structured JSON responses, providing a foundation for integrating front-end and mobile clients in future versions.