

**MIT Art, Design and Technology University**

**MIT School of Computing, Pune**

**Department of Information Technology**

|  |
| --- |
| **Lab Manual** |

# **Practical - Web Programming**

# **Class - S.Y. (SEM-II), DA**

# **Batch - DA-I/II**

# **Utkarsh Saste**

# **Mr./Ms.**

**A.Y. 2024 – 2025 (SEM-II)**

File Index page given in the stationary

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Web Programming**  **SEMESTER – IV** | | | | | |
| **Course Code:** | | 23IT2008 | **Course Credits:** | 02 | |
| **Teaching Hours / Week (L:T:P):** | | 0:0:4 | **CA Marks:** | 25 | |
| **Total Number of Teaching Hours:** | |  | **END-SEM Marks:** | 25 | |
| **Course Pre-requisites:** | | | | | |
| **Course Description:**  This course provides a comprehensive introduction to web technology, designed to help students develop a strong foundation in building and managing websites and web applications. The curriculum covers key topics such as HTML, CSS, and JavaScript,PHP, MySQL, which are essential for creating interactive, well-designed web pages. Students will also explore the principles of responsive design, ensuring that web applications are optimized for different devices and screen sizes.  The course dives deeper into server-side technologies, including HTTP, web servers, and databases, allowing students to understand how websites function behind the scenes. Emphasis is placed on practical learning, and students will gain hands-on experience by working on projects that showcase their ability to design, develop, and deploy websites.  By the end of the course, students will be proficient in using modern web technologies to create web applications. They will understand how to handle client-server interactions, manage user data, and implement various web technologies to enhance the functionality of their applications. | | | | | |
| **Course Learning Objectives:** This course will enable the students to:   1. Understand fundamental concepts of front-end web development. 2. Enable students to create basic web pages incorporating essential elements such as images, hyperlinks, lists, tables, and forms. 3. Teach students how to use CSS to manage fonts, lists, colors, text alignment, and background images for a cohesive and aesthetically pleasing web design. 4. Develop an understanding of JavaScript scopes to manage the visibility and lifetime of variables and functions effectively. 5. Equip students with the skills to implement and handle JavaScript events, enabling enhanced user interactions through event-driven programming. 6. Apply comprehensive knowledge of HTML, CSS, and JavaScript to develop a complete front-end application. Utilize project-based learning to showcase problem-solving skills and creativity in web development projects. 7. Configure server environments with Apache/TOMCAT. 8. Set up a PHP development environment and write basic PHP scripts. 9. Master PHP programming constructs for web development tasks. 10. Create and process HTML forms, and manage MySQL database operations. 11. Develop comprehensive back-end applications using PHP and MySQL. | | | | | |
| **Course Outcome:** After taking this course, Students will be able to :   1. Apply knowledge of HTML to create the structure of the webpage and CSS to style and layout the elements, making the application visually appealing. 2. Apply comprehensive knowledge of HTML, CSS, and JavaScript to develop a complete front-end application and utilize project-based learning to showcase problem-solving skills and creativity in web development projects. 3. Set up and configure a server environment using tools like Apache or TOMCAT and set up a PHP development environment. Write & execute simple PHP scripts, understanding PHP syntax and basic features, create HTML forms to collect user data and integrate with PHP for processing. 4. Design and develop a back-end application using PHP and MySQL, implementing CRUD operations to manage data effectively. | | | | | |
| **UNIT – I** | **Introduction to HTML and Cascading Style Sheet** | | | | **09 Hours** |
| Module 1 - Markup Language (HTML): Introduction to HTML, Formatting and Fonts, Commenting Code, Anchors, Backgrounds, Images, Hyperlinks, Lists, Tables, Frames, HTML Forms  Module 2 - CSS: Need for CSS, introduction to CSS, basic syntax and structure, Levels of style sheets, Style specification formats, BOX Model, Selector forms, Property value forms, Font properties, List properties, Color, Alignment of text, Background images | | | | | |
| **Pedagogy** | **ICT Teaching / PowerPoint Presentation and Videos:**  **Use tools like Visual Studio Code (free).**  **Videos:**  [**https://www.coursera.org/learn/html-css-javascript-for-web-developers**](https://www.coursera.org/learn/html-css-javascript-for-web-developers) | | | | |
| **Self-study / Do it yourself /:**  **Practice creating basic HTML pages and enhancing them using CSS.** | | | | |
| **Experiential Learning Topics:**  **Design a simple webpage for coffee shop website** | | | | |
| **PBL - Project Based Learning:**  **Create a multi-page website (e.g., coffee shop website) using HTML and CSS.** | | | | |
|  | | | | | |
| **UNIT – II** | **Front-End Development** | | | | **09 Hours** |
| Module 3 - Overview of JavaScript, including JS in an HTML (Embedded, External), Basic JS syntax, basic interaction with HTML  Module 4 - Core features of JavaScript: Data types, Control Structures, Arrays, Functions and Scopes | | | | | |
| **Pedagogy** | **ICT Teaching / PowerPoint Presentation and Videos:**  **Use tools like Visual Studio Code (free).**  **Videos:**  [**https://www.coursera.org/learn/javascript-basics**](https://www.coursera.org/learn/javascript-basics) | | | | |
| **Self-study / Do it yourself /:**  **Solve exercises on JavaScript syntax, control structures, and functions** | | | | |
| **Experiential Learning Topics:**  **Build a web page with interactive elements (e.g., a simple calculator).** | | | | |
| **PBL - Project Based Learning:**  **Develop an interactive webpage that uses JavaScript to validate form inputs or perform basic calculations.** | | | | |
|  | | | | | |
| **UNIT – III** | **Advanced Front-End Development** | | | | **09 Hours** |
| Module 5 - DOM: DOM levels, DOM Objects and their properties and methods, Manipulating DOM  Module 6 - JavaScript Events: JavaScript Events, Types of JavaScript Events, Objects in JS, Event Handling | | | | | |
| **Pedagogy** | **ICT Teaching / PowerPoint Presentation and Videos:**  [**https://www.coursera.org/learn/building-interactive-web-pages-using-javascript**](https://www.coursera.org/learn/building-interactive-web-pages-using-javascript)  **Use tools like Visual Studio Code (free).** | | | | |
| **Self-study / Do it yourself /:**  **Practice exercises on DOM traversal and event handling.** | | | | |
| **Experiential Learning Topics:**  **Add dynamic behavior to a webpage using DOM and events (e.g., a to-do list app).** | | | | |
| **PBL - Project Based Learning:**  **Develop a web page with dynamic content (e.g., a task manager or interactive quiz) using DOM manipulation and event handling.** | | | | |
|  | | | | | |
| **UNIT – IV** | **Server Side Scripting** | | | | **09 Hours** |
| Module 7 - Set up and configure a server environment using tools like Apache or TOMCAT, set up a PHP development environment.  Module 8 -Introduction to PHP: : Introduction to PHP, Server side scripting Vs Client side scripting, Basic Development Concepts (Mixing PHP with HTML), Creating, Writing & Running First PHP Script, PHP syntax, conditions & Loops, Functions, String manipulation, Arrays & Functions,  Module 9 - Form handling with HTML and PHP: Designing of Forms using HTML, Form Handling using GET and POST methods of Form | | | | | |
| **Pedagogy** | **ICT Teaching / PowerPoint Presentation and Videos:**  [**https://www.coursera.org/learn/web-applications-php**](https://www.coursera.org/learn/web-applications-php)  **Use tools like Visual Studio Code (free), XAMPP/WAMP for PHP server setup, and MySQL Workbench for database management** | | | | |
| **Self-study / Do it yourself /:**  **Practice exercises on form handling and server-side scripting with PHP.** | | | | |
| **Experiential Learning Topics:**  **Create a basic form for data submission and handle it using PHP (e.g., feedback form).** | | | | |
| **PBL - Project Based Learning:**  **Develop a small server-side application (e.g., a contact form with email validation and submission).** | | | | |
|  | | | | | |
| **UNIT – V** | **Working with Databases and Web Application Development** | | | | **09 Hours** |
| Module 10 - Working with databases using MySQL with PHP: MySQL database, create database, create table, primary key with AUTO\_INCREMENT setting, Insert Data Into a Database Table, Select Data From a Database Table, Open or close a Connection to the MySQL Server.  Module 11 - Web Application Development (Project): Develop the web application to handle client-server interactions, manage user data, and implement various web technologies to enhance the functionality of their applications. Example: Website for a Coffee Shop | | | | | |
| **Pedagogy** | **ICT Teaching / PowerPoint Presentation and Videos:**  **Use tools like Visual Studio Code (free), XAMPP/WAMP for PHP server setup, and MySQL Workbench for database management**  **Videos:**  [**https://www.coursera.org/learn/web-app**](https://www.coursera.org/learn/web-app) | | | | |
| **Self-study / Do it yourself /:**  **Exercises on creating and manipulating databases using PHP and MySQL.** | | | | |
| **Experiential Learning Topics:**  **Create a database and design a webpage to display its data dynamically.** | | | | |
| **PBL - Project Based Learning:**  **Develop a fully functional web application (e.g., a Coffee Shop website or e-commerce platform) that integrates database functionality for data management.** | | | | |

**Experiment No.1**

**Problem Statement:**

1. Create the basic structure of the Library Management System website, including the home page layout with a header, main content area, and footer.

Prepare a common project website design and plan document for all assignments. Consider the following points:

1. Brief information about the project.
2. Set the goals & deliverables.
3. Finalize the modules of the project.
4. Define the audience.
5. Describe pain points & the ideal experience (On the basis of existing systems)
6. Set the visual direction.
7. Map out the Project structure.
8. Plan the content for each page.
9. Add ideas for content, images & layout.
10. Determine your site structure or create content for your core website pages:  
    a. Home page  
    b. About page  
    c. Book Catalog / Services page  
    d. Testimonial/Review page  
    e. Support/Contact page  
    f. Starter blog posts (optional)
11. Create and collect design elements
12. These design elements define your brand personality and help users feel what your brand represents through the use of:  
    a. Colors  
    b. Fonts and typography  
    c. Logos  
    d. Images and photos

**Objective:**  
To design the basic structure of a Library Management System website by planning its layout, content, and visual elements, ensuring it meets user needs and effectively represents the institution.

**Theory:**  
**Project Design and Plan Document for Library Website**

1. **Brief Information about the Project**  
   The project is to create a user-friendly and informative website for a Library Management System. It aims to attract students, researchers, and readers by highlighting available resources, services, library policies, and member functionalities. The website will support login and registration for user personalization and secure access to digital features.
2. **Goals and Deliverables**  
   **Goals:**

* Develop an engaging and functional library website.
* Showcase the library's catalog, policies, team, and contact information.
* Enable users to register, log in, borrow books, and access account data.
* Create a responsive design adaptable to mobile, tablet, and desktop devices.

**Deliverables:**

* Website Pages:
  + Home Page
  + About Page
  + Catalog/Services Page
  + Testimonials Page
  + Contact Page
  + Login Page
  + Registration Page
  + Optional Blog section for updates and news
* Core Features:
  + Header and footer with consistent navigation
  + Functional login and registration system
  + Responsive design and accessible layout
  + Professional branding with colors, fonts, and images

1. **Finalize the Modules of the Project**  
   **Website Modules:**
2. Home Page Module
   * Hero section, announcements, featured books
   * Navigation links to all pages
   * Footer with contact info and links
3. About Page Module
   * Library history, mission, values
   * Meet the librarians section
4. Catalog/Services Page Module
   * Categories: Fiction, Non-fiction, Journals, eBooks, etc.
   * Search and filter options
   * Book cards with image, title, author, availability
5. Testimonials Page Module
   * User reviews of library experience
   * Submit testimonial form
6. Contact Page Module
   * Inquiry form (name, email, subject, message)
   * Embedded map and contact details
7. Login Page Module
   * Email and password fields
   * Link to registration page and "Forgot Password"
8. Registration Page Module
   * Name, email, password form
   * Terms acceptance
9. Footer Module
   * Social links, privacy policy, contact, quick links
10. **Define the Audience**  
    **Target Audience:**

* Students: Seek academic resources, quick book search, user-friendly layout
* Faculty: Access research materials, journals, and manage borrowed items
* Researchers: Need extensive archives, database access
* Visitors/New Users: Require intuitive navigation and basic info

**Mapped Features:**

* Online catalog with search and filter
* Personalized user login for borrowing and history
* Responsive design for mobile users
* Dedicated sections for faculty and students
* Blog and announcement panel

1. **Describe Pain Points &the Ideal Experience**  
   **Pain Points of Existing Systems:**

* Cluttered navigation
* No mobile optimization
* Poor catalog search options
* Incomplete book or policy information
* No digital reservation system
* No user personalization

**Ideal Experience:**

* Clean UI with sticky navbar
* Responsive design
* Easy book lookup with filters
* Personalized user dashboard
* Real-time notifications
* Direct communication via contact form

**Ideal User Journey:**

1. Arrive at a clean, welcoming homepage
2. Navigate to catalog and find desired book
3. Register/Login to borrow
4. Submit request
5. Receive confirmation and updates
6. Contact support or read latest blog/news

**6. Set the visual direction**

1. Visual Design Goals

The visual design of the second-hand gaming console store website should reflect its personality, build trust, and create an inviting experience for customers. It should align with the following principles:

* Welcoming and Comfortable: The website should feel cozy and approachable, much like the product presentation of the second-hand gaming console store itself.
* Modern and Minimalistic: Clean layouts and modern design elements create a professional and user-friendly aesthetic.
* Brand Representation: The visual elements, including colors, typography, and images, should communicate the second-hand gaming console store's values and target audience.

2. Defining the Core Visual Elements

a. Color Palette

A warm and earthy color palette inspired by console and natural tones creates a visually consistent and soothing experience.

|  |  |  |
| --- | --- | --- |
| Color | Hex Code | Usage |
| Console Brown | #6F4E37 | Header, footer, buttons, and highlights. |
| Creamy Beige | #F5F5DC | Background to create warmth and contrast. |
| Deep Espresso | #3C2F2F | Text and important accents for legibility. |
| Latte White | #FAF3E0 | Secondary backgrounds and subtle contrasts. |
| Olive Green | #556B2F | Call-to-action buttons for natural harmony. |

b. Typography

Fonts should be easy to read while reflecting the warm and inviting atmosphere of the second-hand gaming console store.

* Primary Font: *Poppins* or *Roboto* (Sans-serif) – For headings and call-to-action text.
* Secondary Font: *Open Sans* or *Lora* – For body text and descriptions.
* Attributes: Use bold headings for emphasis and lighter weights for readability.

c. Logos and Branding

A sleek, memorable logo based on the console theme is essential. For instance:

* Use a stylized console device, bean, or sportable consolesm motif in the logo design.
* The logo should include the second-hand gaming console store's name in the selected typography.
* A monochrome version of the logo can be created for simplicity in headers or footers.

d. Imagery and Icons

High-quality visuals can make the website feel alive and inviting.

* Photography:
  + Pictures of freshly useed console, gaming consoles, cozy seating spaces, and happy customers.
  + Showcase specialty console consoles, desserts, and in-store product presentation.
* Icons:
  + Minimalistic icons for categories like catalog, location, testimonials, and contact.
* Hero Images:
  + Use a carousel or static hero banner on the homepage featuring key products or seasonal promotions.

3. Applying Visual Design to Pages

a. Home Page

* Banner Area: Hero image with text overlay showcasing a featured product or promo.
* Color Scheme: Warm tones for text and buttons, cream or PlayStation shades for the background.
* Typography: Bold for headlines like "Welcome to [Console Store Name]."

b. About Page

* Use authentic imagery of staff, customers, or the console preparation process.
* Soft, inviting colors to match the tone of storytelling.

c. Product/Service Page

* Ensure product cards display item images prominently with prices and descriptions.
* Add hover effects to highlight product interactions.

d. Testimonial Page

* Use customer photos and quotes with clean card layouts.
* A slider element for seamless scrolling through reviews.

e. Contact Page

* Embed Google Maps with the store’s location using the green-toned call-to-action buttons.

f. Login and Registration Pages

* Keep the form layout minimal, with soft color backgrounds and visible input fields.
* Buttons in console brown or green for consistent branding.

4. Layout and Design Hierarchy

The visual hierarchy ensures an easy and intuitive flow through the website:

1. Headers and Banners: Prominent for branding and immediate engagement.
2. Navigation Bar: Sticky and unobtrusive for easy exploration.
3. Sections and Grids: Structured with clear breaks using background shades or bpurchases.
4. Call-to-Action: Buttons prominently styled to encourage actions like "Purchase Now" or "Sign Up."

5. Expected Impact of Visual Direction

1. Enhanced Engagement: A warm design encourages users to explore further.
2. Stronger Branding: Consistency in colors and typography strengthens identity.
3. Better Retention: User-friendly layouts and aesthetic appeal retain visitors.
4. Higher Conversions: Effective call-to-action placement drives purchases or registrations.

**7. Map out the Project structure**

console\_store\_website/

│

├── index.html             # Home page

├── about.html             # About page

├── products.html          # Products/Services page

├── testimonials.html      # Testimonials page

├── contact.html           # Support page

├── login.html             # Login page

├── register.html          # Registration page

├── blog.html              # Optional blog page

│

├── assets/

│   ├── css/

│   │   ├── style.css        # Global CSS

│   │   ├── responsive.css   # Media queries for mobile optimization

│   │

│   ├── js/

│   │   ├── main.js          # Interactive scripts

│   │   ├── formValidation.js # Scripts for login and registration validation

│   │

│   ├── images/

│       ├── logo.png         # Website logo

│       ├── homepage\_banner.jpg  # Hero banner for home

│       ├── console\_catalog/     # Images for catalog items

│       ├── portable consolesm\_photos/     # Images for the about page

│       ├── icons/           # Icons for UI elements

│

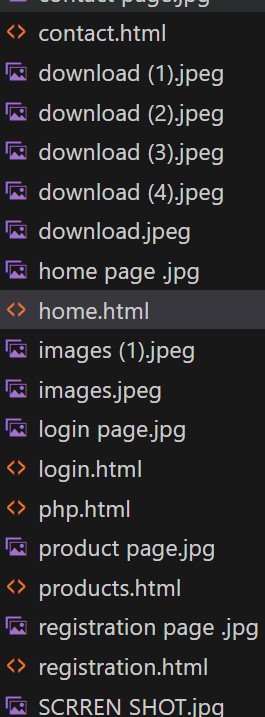
├── fonts/

│   ├── Poppins/            # Primary font

│   ├── OpenSans/           # Secondary font

│

└── README.md               # Project documentation file



**8. Plan the content for each page**

The website will include a minimum of 5 core pages, along with additional Login and Registration pages. This plan details the content for each page.

1. Home Page

**1. Home Page**

**Purpose:**

* Welcome visitors.
* Highlight the Library Management System’s features and resources

**Content Plan:**

* **Header:**
  + Logo on the left.
  + Navigation menu: Home, About, Resources, Testimonials, Contact.
  + Login/Sign-Up button on the top right
  + A high-quality banner image (second-hand gaming console store product presentation, a featured console).
  + A tagline like: *“Brewed to Perfection!”*.
  + CTA button: *“Explore Our Menu”*.
* **2. About Page**
* **Purpose:**
* Share the story, vision, and people behind the Library Management System.
* **Content Plan:**
* **Header:** (same as home page).
* **About Us Section:**
* A short history of the library or institution.
* Emphasis on values like accessibility, lifelong learning, and community service.
* **Meet the Team Section:**
* Photos and bios of librarians, administrators, or tech team
* Meet the Team Section:
  + Photos and short bios of the founders or portable consolesm members.

**Special Features Section:**

* “Why Choose Our Library?”
* Highlight USPs like digital resources, research support, and 24/7 access.

**Footer:** (same as home page).

**3. Resources Page (Product/Service Page)**

**Purpose:**

* Showcase available books, journals, digital resources, and services.

**Content Plan:**

* **Header:** (same as home page).
* **Resource Categories:**
  + Categories like Fiction, Non-Fiction, Research Journals, Digital Archives.
  + Resource cards with title, image, short description, availability.
* **Highlight Section:**
  + “Top Picks” or “Most Borrowed Items.”
* **CTA Section:**
  + Button: “Borrow Now” linking to login/registration.
* **Footer:** (same as home page).

**5. Contact Page**

**Purpose:**

* Allow users to connect with library staff for support, suggestions, or inquiries.

**Content Plan:**

* **Header:** (same as home page).
* **Contact Form:**
  + Name, Email, Subject, Message.
* **Location Section:**
  + Embedded map of the library location.
* **Operating Hours Section:**
  + Clearly list timings for physical visits.
* **Footer:** (same as home page)

**6. Login Page**

**Purpose:**

* Allow registered users to log in and access personalized features.

**Content Plan:**

* **Form:**
  + Email and Password fields.
  + Submit button.
* **Forgot Password Link:**
  + Link to password recovery.
* **CTA:**
  + Link to Registration: “Don’t have an account? Sign up now!

**7. Registration Page**

**Purpose:**

* Let new users register and access the system.

**Content Plan:**

* **Form Fields:**
  + Full Name, Email, Password, Confirm Password.
* **Validation:**
  + Password strength indicators and validation checks.
* **CTA:**
  + “Register and Start Borrowing Today!”

**1. Home Page**

**Layout Ideas:**

* Fixed header with navigation and login.
* Hero image of a library space or reading room.
* Section with book tiles (e.g., “Book of the Month”).

**Content Ideas:**

* Welcome message like “Welcome to the Digital Library Portal.”
* Updates on events like reading challenges or book fairs.

**Image Ideas:**

* Shelves with books, people reading, quiet study areas.

**2. About Page**

**Layout Ideas:**

* Timeline of library history.
* Grid for team members.

**Content Ideas:**

* Mission: Promote education and equal access to information.
* Values: Inclusivity, knowledge, innovation.

**Image Ideas:**

* Photos of team, heritage building, tech-enabled reading areas

**3. Resources Page**

* **Layout Ideas:**
* Tabbed interface for print vs digital resources.
* Filters: Genre, author, publication date.

**Content Ideas:**

* Resource descriptions, download links (if digital), availability.

**Image Ideas:**

* High-res covers, shelves, e-library visuals.

**4. Testimonials Page**

**Layout Ideas:**

* Carousel for rotating feedback.
* Star ratings and reviewer names.

**Content Ideas:**

* Quotes about helpfulness of staff, rich collection, user interface.

**Image Ideas:**

* Smiling readers, workshops, study groups

**5. Contact Page**

**Layout Ideas:**

* Side-by-side form and map.
* Icons for phone, email, location.

**Content Ideas:**

* Simple, friendly messaging: “We're here to help!”

**Image Ideas:**

* Icon illustrations, front view of the library.

**6. Login Page**

**Layout Ideas:**

* Clean form UI with visual consistency.

**Content Ideas:**

* Field labels, login prompt, and link to reset password.

**Image Ideas:**

* Book icons or abstract knowledge visuals.

**7. Registration Page**

**Layout Ideas:**

* Centered form with optional sidebar content.

**Content Ideas:**

* Invitation to join: “Become a Member Today!”

**Image Ideas:**

* Icons for learning, study, book stack.

**8. Visual Design Ideas for All Pages**

* Colors: Book brown, parchment cream, scholarly green.
* Typography: Headers in Playfair Display, body in Lato.
* Logos: Book or library emblem with elegant serif font.
* Images: Realistic shots, soft lighting, minimal filters**.**

**9. Determine your site structure or Create content for your core website pages:**

1. **Home page**
2. **About page**
3. **Product/Service page**
4. **Testimonial/review page**
5. **Support page**
6. **Starter blog posts**

**1. Home Page**

The home page serves as the introduction to the second-hand gaming console store and provides navigation to the key sections of the website.

Here's a revised version of the website outline, reflecting the focus on a second-hand gaming console store:

**1. Home Page**

**Header:**

* **Logo:** A simple, bold representation of the second-hand gaming console store (e.g., a stylized gaming console or controllers).
* **Navigation Links:** Menu, About, Products, Testimonials, Contact.
* **Call-to-Action Button:** "Buy Now" or "Join Us" (links to the purchase or registration page).

**Hero Section:**

* **Background Image:** A full-width image of popular second-hand gaming consoles or the store's gaming console collection.
* **Text Overlay:** “Power Up Your Gaming Experience with Pre-Loved Consoles!”
* **Call-to-Action Button:** "Explore Our Consoles" or "Shop Now."

**About Section (Teaser):**

* A short paragraph introducing the store, inviting visitors to learn more about its mission to provide quality second-hand gaming consoles.
* Link to the About page.

**Product Highlights Section:**

* **Featured Products:** Grid showcasing 3-4 key products like "Console of the Week," "Featured Accessories," etc.
* Images and short descriptions with an option to learn more or make a purchase.

**Social Proof Section (Testimonial Teaser):**

* Snippets from customer reviews with a “See More” button linking to full testimonials.

**Footer:**

* Quick links to catalog, store hours, locations, FAQs.
* Social media icons (Facebook, Instagram, Twitter).
* Store address with Google Map embed.

**2. About Page**

**Introduction:**

* Overview of the library’s history, mission, and values.
* Brief history: “Founded in 2025 to empower lifelong learners and promote reading culture…”

**Meet the Team:**

* Grid layout with names, photos, and bios of librarians, tech admins, and support staff.

**Our Promise:**

* Information about how resources are curated, managed, and made accessible.
* Bullet points or icons showing services like digital archives, special access for differently-abled users, and sustainability efforts (e.g., paper recycling)

**3. Resources Page (Product/Service Page)**

**Resource Categories:**

* **Categories: “New Arrivals,” “eBooks,” “Research Journals,” “Fiction & Non-Fiction.”**
* **Each resource should include a cover image, description, availability, and category.**
* **“Borrow” or “Read Online” button linking to the borrowing system.**

**Popular Resources & Featured Collections:**

* **Carousel or grid showcasing limited-time featured books, events, or digital bundles.**

**Access Options:**

* **Details on how to borrow books, access eBooks, or reserve materials online or in-person.**

**Footer:**

* **(Same as Home Page)**

**4. Testimonials/Review Page**

**User Reviews:**

* Carousel/grid showing reader testimonials with names and star ratings.
* “Submit Your Review” button for feedback.

**Featured Testimonials:**

* Pull in reviews from institutional feedback forms, social media, or review platforms.

**Reviewing Process Section:**

* Brief explanation of how user feedback is verified and shared.

**5. Contact Page**

**Contact Form:**

* Fields: Name, Email, Message.
* Submit button with validation.

**Social Media & Address Section:**

* Facebook, Instagram, Twitter icons.
* Library address, phone, email.

**Interactive Map:**

* Google Maps embed to show physical locations.

**Support Information:**

* Quick FAQs and contact details for specific queries (e.g., membership, renewals)

**6. Starter Blog Posts**

**Blog Categories:**

* **Library Insights:** Articles like “How to Use Our Digital Library” or “Maximize Research with JSTOR.”
* **Behind the Shelves:** Features on library operations, staff interviews, or resource curation.
* **Sustainability & Tech:** Posts on digitization, eco-friendly practices, and archiving.
* **Community Stories:** Library events, reading competitions, or social outreach highlights.

**7. Login & Registration Pages**

**Login Page:**

* Fields: Username/email, password.
* “Forgot Password?” link.

**Registration Page:**

* Fields: Name, Email, Password.
* Option to opt into newsletters or event notifications.

**8. Overall Website Structure Map**

* **Home Page:** Intro with featured resources and announcements.
* **About Page:** Library background, team, and core values.
* **Resources Page:** Collection categories and borrowing options.
* **Testimonials Page:** Feedback and user experiences.
* **Contact Page:** Inquiry form and location info.
* **Blog Page:** Posts on learning, literacy, and library tech.
* **Login/Registration Page:** User accounts for personal tracking.

**9. Design Elements**

**Colors:**

* **Primary Colors:** Book Brown (#6F4F37), Beige (#D8CAB8), Cream (#F1E0C6)
* **Accent Colors:** Ink Black (#2B1B1D), Knowledge Green (#6DBF3A)

**Fonts and Typography:**

* **Heading Font:** Playfair Display or Lora (serif)
* **Body Font:** Open Sans or Roboto (sans-serif)

**Logo:**

* A simple logo incorporating a book, feather quill, or open scroll—symbolizing knowledge and tradition.

**Imagery and Photos:**

* High-quality photos of library interiors, readers, book covers, study spaces.
* Lifestyle imagery: students studying, researchers reading, children in reading programs.

**Interactive Elements:**

* **Buttons:** Knowledge Green or Ink Black for primary actions like “Borrow Now,” “Search Catalog.”
* **Icons:** Clean symbols for catalog, user login, support, location, and blog/news.

**Psychological Impact:**  
These colors and design elements promote a sense of warmth, trust, intellectual curiosity, and approachability. Green adds a layer of eco-consciousness and growth, perfect for a library’s educational and community-oriented mission.

**2. Fonts and Typography**

Typography conveys tone and supports the user’s reading experience.

* **Heading Font:**  
  *Playfair Display* or *Lora* creates a scholarly, trustworthy look — ideal for headers and banners.
* **Body**   
  *Open Sans* or *Roboto* ensures easy reading across all devices for resource descriptions, blog articles, and user instructions.
* **Font Weights:**
  + Bold for headings to draw attention
  + Regular or light for paragraphs to maintain flow

**Impact:**  
The combination of serif and sans-serif fonts strikes a balance between formality and readability, mirroring the library’s role as both a heritage institution and a modern knowledge provider.

**3. Logo**

* **Logo**   
  A minimal, iconic logo representing an open book, stylized pages, or a torch of knowledge.
* **Color**   
  Primary palette colors with slight green accents (e.g., for a leaf on a book or torch flame).
* **Logo Usage:**
  + Top of every page (in the header)
  + In print materials, social media, newsletters, and library cards

**Impact:**  
The logo becomes the face of the library’s digital and physical identity—instantly conveying values of learning, accessibility, and credibility.

**4. Imagery and Photos**

* **Resource Photography:**  
  Crisp images of featured books, digital devices, and journals.
* **Ambiance Shots:**  
  Cozy reading spaces, community events, collaborative study zones.
* **Lifestyle Photography:**
  + Students, teachers, and researchers in action
  + Children participating in library programs or storytelling sessions

**Impact:**  
Real-world visuals foster a strong emotional connection and show the library as a vibrant, welcoming space.

**5. Interactive Elements and Buttons**

* **Buttons:**
  + Clear actions like "Borrow Now", "Join the Library", "Submit a Request"
  + Hover effects (shadows, background shift) to guide users
* **Icons:**
  + Open book (catalog), map pin (location), envelope (contact), lightbulb (new arrivals or suggestions)

**Conclusion**

A well-structured, user-centered interface not only simplifies access but also enhances the library experience by aligning design with user intent. Whether someone is browsing for leisure reading, researching academic papers, or managing their digital loans, the interface guides them seamlessly through their journey.

By integrating:

* Consistent design patterns
* Smart iconography
* Responsive layouts
* Personalized user dashboards
* Interactive tools (like search, filters, and bookmarks)

…this system transforms a traditional library into a smart, inclusive, and engaging digital platform.

Such a design approach:

* Reduces cognitive load for new users
* Encourages exploration for curious readers
* Builds loyalty by offering features that feel both familiar and empowering
* Supports the broader mission of the library: democratizing knowledge and fostering community learning

In short, the Library Management System isn’t just a website—it’s a gateway to lifelong learning in the digital age.

**Experiment No.2**

**Problem Statement:**

•Create a detailed home page for the Library Management System (LMS).  
•Create a detailed catalog page listing all the available books and resources, categorized appropriately.  
• Create a dashboard or user account page that allows users to manage borrowed books, view due dates, and renew items.  
• Create an About Us page that provides detailed information about the library’s mission, history, and staff.  
• Create a Contact page that allows users to reach out to library staff via a form.  
• Design and implement an admin/user registration form for the Library Management System.  
• Design and implement an admin/user login form for secure access to personalized features

**Objective:**

To create a **Library Management System** webpage using HTML, which allows users to search, borrow, and manage access to physical and digital library resources

**Introduction:**

In today's academic and institutional environments, libraries must evolve to offer convenient digital access to information. This project focuses on creating a responsive and functional web interface for managing library operations and enabling users to interact with available resources.

**The platform serves:**

* Students and Readers – for searching, borrowing, and renewing materials
* Librarians/Admins – for managing inventory, issuing books, and monitoring activity
* Institutions – for tracking resource usage, generating reports, and enhancing digital access

This system promotes knowledge access, efficiency, and digital transformation in libraries. The website is built using HTML, CSS, and optionally JavaScript or server-side technologies for dynamic interactions such as user authentication and book tracking

**1. Home Page**

The home page introduces visitors to the library and guides them to essential features. It typically includes:

* A hero section with announcements (e.g., “New Arrivals”, “Library Week Events”)
* A navigation bar linking to Catalog, About, Login, Register, Contact
* Call-to-action buttons such as "Search Catalog" or "Join the Library"
* Testimonials or highlights about services (e.g., e-book availability, reading rooms)
* Introduction to the library’s mission

**Importance:**  
It establishes credibility, encourages user engagement, and sets the tone for digital users. Good visuals and clear layout improve trust and navigation.

**Technologies used:**

* **HTML** for structure
* **CSS** for layout and styling
* Optional **JavaScript** for dynamic features like sliders or search interaction

**2. Catalog Page (Portfolio Equivalent)**

This is the core page where users explore library resources. Organized into clear categories such as:

* **Fiction / Non-fiction**
* **Academic Textbooks**
* **Journals / Research Papers**
* **Magazines / Periodicals**
* **Digital Resources / eBooks**

**Features include:**

* Book cover image and metadata (title, author, publication year)
* Availability status (available/on loan)
* Search bar and filters (by genre, author, date, etc.)
* "Add to Borrow List" button for logged-in users
* Link to book preview or digital download (if available)

**Importance:**  
A searchable and filterable catalog improves discoverability and user satisfaction.

**UX Consideration:**

* Sorting options (by popularity, new arrivals)
* Categorized view (tiles or list)
* Responsive design for mobile users

**3. Dashboard/User Account Page (Cart Equivalent)**

This section allows logged-in users to:

* View borrowed books and due dates
* Renew items
* Track borrowing history
* View notifications for overdue items
* Save favorites for future borrowing

**Importance:**  
Empowers users to self-manage and reduces staff workload through automation.

**4. About Us Page**

**Purpose:**  
To provide users with background on the library, its mission, values, and key personnel.

**Content Includes:**

* The library’s history and purpose (e.g., “Serving readers since 2000…”)
* A mission statement: “Fostering knowledge and lifelong learning”
* Staff bios and roles (e.g., Head Librarian, Digital Media Coordinator)
* Photos of library spaces or archives
* Any awards or recognitions

**5. Contact Page**

Allows users to get in touch for support or feedback.

**Features:**

* A contact form with fields: **Name, Email, Subject, Message**
* Google Maps embed to show physical library location
* Display of phone number, email, and operating hours
* Links to social media for updates

**6. Login & Registration Pages**

**Login Form:**

* Fields: Username or Email, Password
* "Forgot Password?" link
* Redirects to dashboard on success

**Registration Form:**

* Fields: Name, Email, Password, Confirm Password
* Role selection (Reader, Librarian, Admin)
* Checkbox for T&C acceptance
* Option to subscribe to event/newsletter alerts

**7. User/Admin Login Form**

This form authenticates library users and administrators against stored credentials and redirects them to their respective dashboards.

**Fields:**

* Username/Email
* Password
* "Remember Me" checkbox
* "Forgot Password?" link

**Security Considerations:**

* Basic input validation
* For production systems: implement hashed password storage, login rate limiting, and two-factor authentication

**Differentiated Access:**

* **Users (Readers)** can:
  + Browse the catalog
  + Borrow and renew books
  + Track due dates and borrowing history
* **Admins (Librarians)** can:
  + Manage book inventory
  + Add/remove/update book records
  + View borrowing analytics
  + Handle overdue alerts and notifications

**Code:**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Home - Library Management</title>

    <link rel="stylesheet" href="styles.css">

    <style>

        /\* General Styling \*/

        body {

            font-family: 'Poppins', sans-serif;

            background: url('C:\Users\DELL\OneDrive\Desktop\LIBARY MANAGEMANT\images (1).jpeg') no-repeat center center fixed;

            background-size: cover;

            margin: 0;

            padding: 0;

            color: rgb(255, 255, 255);

            position: relative;

        }

  /\* Background Overlay \*/

        body::before {

            content: "";

            position: absolute;

            top: 0;

            left: 0;

            width: 100%;

            height: 100%;

            background: linear-gradient(to right, rgba(0, 0, 0, 0.8), rgba(0, 0, 0, 0.5));

            z-index: -1;

        }

  /\* Navigation Bar \*/

        nav {

            background: rgba(0, 0, 0, 0.8);

            padding: 15px;

            text-align: center;

            width: 100%;

            position: fixed;

            top: 0;

            left: 0;

            z-index: 1000;

            box-shadow: 0 4px 10px rgba(0, 0, 0, 0.3);

        }

        nav ul {

            list-style: none;

            padding: 0;

            margin: 0;

            display: flex;

            justify-content: center;

        }

        nav ul li {

            margin: 0 15px;

        }

        nav ul li a {

            color: white;

            text-decoration: none;

            font-size: 18px;

            font-weight: bold;

            padding: 10px 15px;

            position: relative;

            transition: 0.3s ease;

        }

        nav ul li a::after {

            content: "";

            display: block;

            width: 0;

            height: 3px;

            background: #f39c12;

            transition: width 0.3s ease;

            margin: auto;

        }

        nav ul li a:hover::after {

            width: 100%;

        }

/\* Hero Section \*/

        .container {

            text-align: center;

            padding: 120px 20px;

            position: relative;

            z-index: 1;

        }

        .hero-section {

            max-width: 900px;

            margin: 100px auto;

            background: rgba(0, 0, 0, 0.7);

            padding: 30px;

            border-radius: 15px;

            box-shadow: 0 0 20px rgba(0, 0, 0, 0.4);

            animation: fadeIn 1.2s ease-in-out;

        }

        .hero-section h1 {

            font-size: 40px;

            animation: slideIn 1.5s ease-in-out;

        }

        .hero-section p {

            font-size: 18px;

            margin: 20px 0;

        }

        @keyframes fadeIn {

            from { opacity: 0; transform: scale(0.9); }

            to { opacity: 1; transform: scale(1); }

        }

        @keyframes slideIn {

            from { transform: translateY(-30px); opacity: 0; }

            to { transform: translateY(0); opacity: 1; }

        }

        .btn {

            padding: 12px 20px;

            background: #f39c12;

            color: black;

            font-size: 18px;

            text-decoration: none;

            border-radius: 8px;

            transition: 0.3s ease;

            display: inline-block;

        }

        .btn:hover {

            background: #e67e22;

            transform: scale(1.05);

        }

        /\* Feature Section \*/

        .features {

            margin-top: 50px;

            display: flex;

            justify-content: center;

            gap: 30px;

            flex-wrap: wrap;

        }

        .feature-box {

            width: 250px;

            padding: 20px;

            background: rgba(255, 255, 255, 0.9);

            color: black;

            border-radius: 10px;

            text-align: center;

            box-shadow: 0 0 15px rgba(0, 0, 0, 0.3);

            transition: 0.3s ease;

        }

        .feature-box:hover {

            transform: scale(1.05);

        }

        .feature-box h3 {

            margin-bottom: 10px;

        }

        /\* Footer \*/

        .footer {

            background: #222;

            color: white;

            text-align: center;

            padding: 20px;

            margin-top: 50px;

        }

    </style>

</head>

<body>

    <nav>

        <ul>

            <li><a href="home.html" class="active">Home</a></li>

            <li><a href="books.html">Books</a></li>

            <li><a href="cart.html">Cart</a></li>

            <li><a href="contact.html">Contact</a></li>

            <li><a href="login.html">Login</a></li>

        </ul>

    </nav>

    <div class="container">

        <div class="hero-section">

            <h1>📖 Welcome to Our Library</h1>

            <p>Explore a vast collection of books, manage your library, and enhance your reading experience.</p>

            <a href="books.html" class="btn">📚 Browse Books</a>

        </div>

        <div class="features">

            <div class="feature-box">

                <h3>📚 Huge Collection</h3>

                <p>Thousands of books available in various genres.</p>

            </div>

            <div class="feature-box">

                <h3>🔍 Easy Search</h3>

                <p>Find your favorite books quickly and efficiently.</p>

            </div>

            <div class="feature-box">

                <h3>🛒 Shopping Cart</h3>

                <p>Add books to your cart and borrow easily.</p>

            </div>

        </div>

    </div>

<footer class="footer">

        &copy; 2025 Library Management | Follow us on 📘 Facebook | 🐦 Twitter | 📸 Instagram

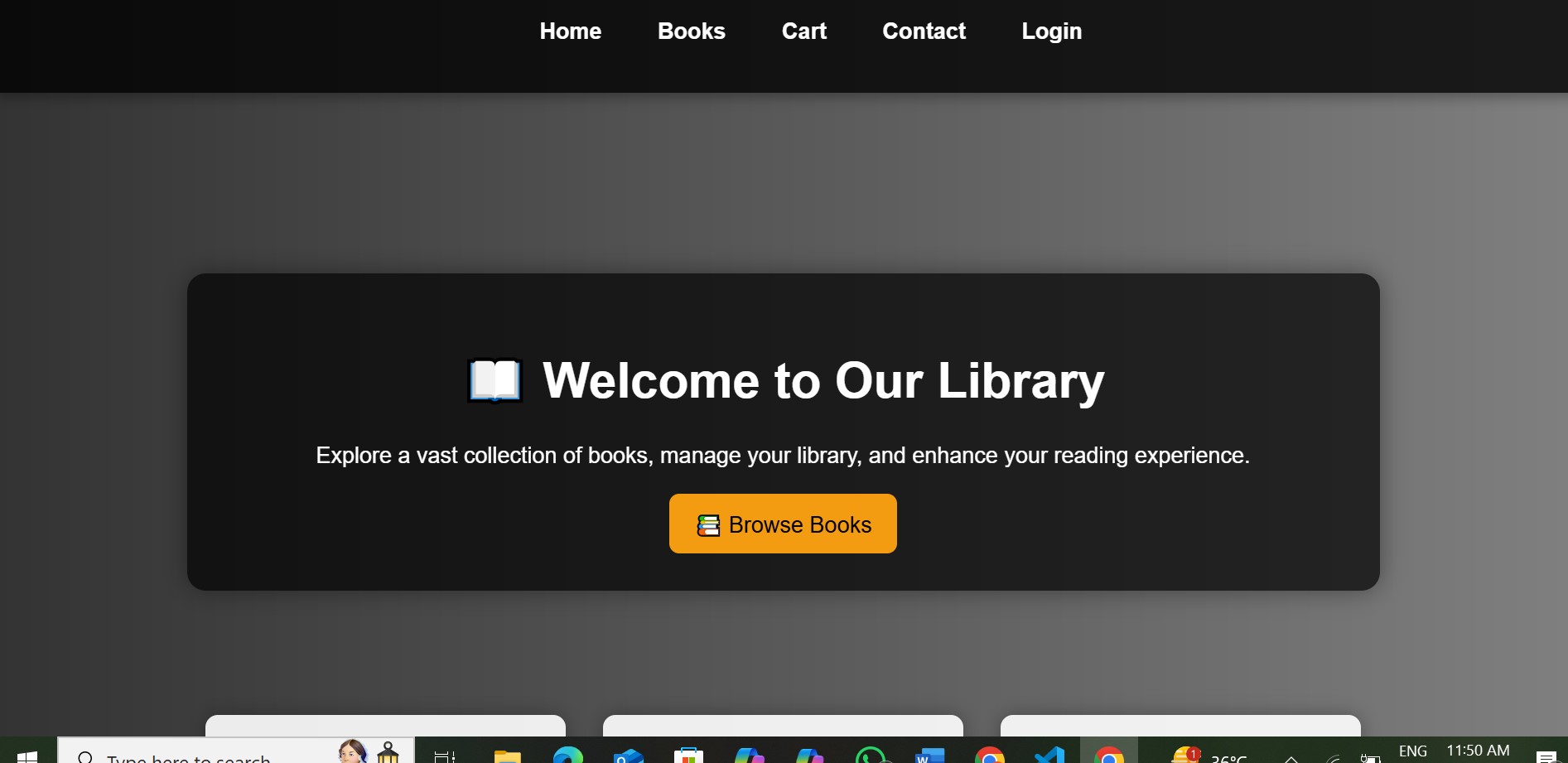
    </footer>

</body>

</html>

**Output:**

A. Index/Home page output:



**Code:**

B. menu/product page:

code:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Products - Library Management</title>

    <link rel="stylesheet" href="styles.css">

    <style>

        body {

            font-family: Arial, sans-serif;

            background: url('library-bg.jpg') no-repeat center center fixed;

            background-size: cover;

            margin: 0;

            padding: 0;

        }

        nav {

            background: rgba(0, 0, 0, 0.8);

            padding: 15px;

            text-align: center;

            width: 100%;

            position: fixed;

            top: 0;

            left: 0;

            z-index: 1000;

        }

        nav ul {

            list-style: none;

            padding: 0;

            margin: 0;

            display: flex;

            justify-content: center;

        }

        nav ul li {

            margin: 0 15px;

        }

        nav ul li a {

            color: white;

            text-decoration: none;

            font-size: 18px;

            font-weight: bold;

            padding: 10px 15px;

            background: #27ae60;

            border-radius: 5px;

            transition: background 0.3s;

        }

        nav ul li a:hover {

            background: #219150;

        }

        .container {

            max-width: 1000px;

            margin: 100px auto;

            padding: 20px;

            background: rgba(255, 255, 255, 0.9);

            border-radius: 10px;

            box-shadow: 0 0 10px rgba(0, 0, 0, 0.2);

            text-align: center;

        }

        .categories {

            display: grid;

            grid-template-columns: repeat(3, 1fr);

            gap: 20px;

            padding: 20px;

        }

        .category-item {

            background: white;

            padding: 15px;

            border-radius: 10px;

            box-shadow: 0 0 5px rgba(0, 0, 0, 0.2);

            text-align: center;

        }

        .category-item img {

            width: 100%;

            height: 200px;

            object-fit: cover;

            border-radius: 5px;

        }

        .category-item h3 {

            margin: 10px 0;

        }

        .category-item p {

            font-size: 14px;

            color: #555;

        }

        .view-books {

            background: #27ae60;

            color: white;

            padding: 10px;

            border: none;

            border-radius: 5px;

            cursor: pointer;

            font-size: 16px;

            margin-top: 10px;

            text-decoration: none;

            display: inline-block;

        }

        .view-books:hover {

            background: #219150;

        }

        .cart-counter {

            background: red;

            color: white;

            border-radius: 50%;

            padding: 5px 10px;

            font-size: 14px;

            position: absolute;

            top: 10px;

            right: 10px;

        }

    </style>

</head>

<body>

    <nav>

        <ul>

            <li><a href="home.html">Home</a></li>

            <li><a href="products.html" class="active">Products</a></li>

            <li><a href="books.html">Books</a></li>

            <li><a href="cart.html">Cart <span id="cart-count" class="cart-counter">0</span></a></li>

            <li><a href="contact.html">Contact</a></li>

            <li><a href="login.html">Login</a></li>

        </ul>

    </nav>

    <div class="container">

        <h1>Browse Our Book Categories</h1>

        <p>Find books from different genres and categories. Click to explore more.</p>

        <div class="categories">

            <div class="category-item">

                <img src="C:\Users\DELL\OneDrive\Desktop\LIBARY MANAGEMANT\images (1).jpeg" alt="Fiction Books">

                <h3>Fiction</h3>

                <p>Explore imaginative stories from best-selling authors.</p>

                <a href="books.html" class="view-books">View Books</a>

            </div>

            <div class="category-item">

                <img src="C:\Users\DELL\OneDrive\Desktop\LIBARY MANAGEMANT\download (3).jpeg" alt="Non-Fiction Books">

                <h3>Non-Fiction</h3>

                <p>Gain knowledge and insights from real-life stories.</p>

                <a href="books.html" class="view-books">View Books</a>

            </div>

            <div class="category-item">

                <img src="C:\Users\DELL\OneDrive\Desktop\LIBARY MANAGEMANT\download (2).jpeg" alt="Comic Books">

                <h3>Comics & Manga</h3>

                <p>Enjoy exciting stories through illustrations.</p>

                <a href="books.html" class="view-books">View Books</a>

            </div>

            <div class="category-item">

                <img src="C:\Users\DELL\OneDrive\Desktop\LIBARY MANAGEMANT\download (1).jpeg" alt="Science Books">

                <h3>Science & Technology</h3>

                <p>Stay updated with the latest in science and tech.</p>

                <a href="books.html" class="view-books">View Books</a>

            </div>

            <div class="category-item">

                <img src="C:\Users\DELL\OneDrive\Desktop\LIBARY MANAGEMANT\download.jpeg" alt="History Books">

                <h3>History</h3>

                <p>Learn about past events and historical figures.</p>

                <a href="books.html" class="view-books">View Books</a>

            </div>

            <div class="category-item">

                <img src="C:\Users\DELL\OneDrive\Desktop\LIBARY MANAGEMANT\download (4).jpeg" alt="Self-Help Books">

                <h3>Self-Help & Motivation</h3>

                <p>Improve yourself with guidance from experts.</p>

                <a href="books.html" class="view-books">View Books</a>

            </div>

        </div>

    </div>

   <script>

        let cartCount = 0;

        document.getElementById("cart-count").textContent = cartCount;

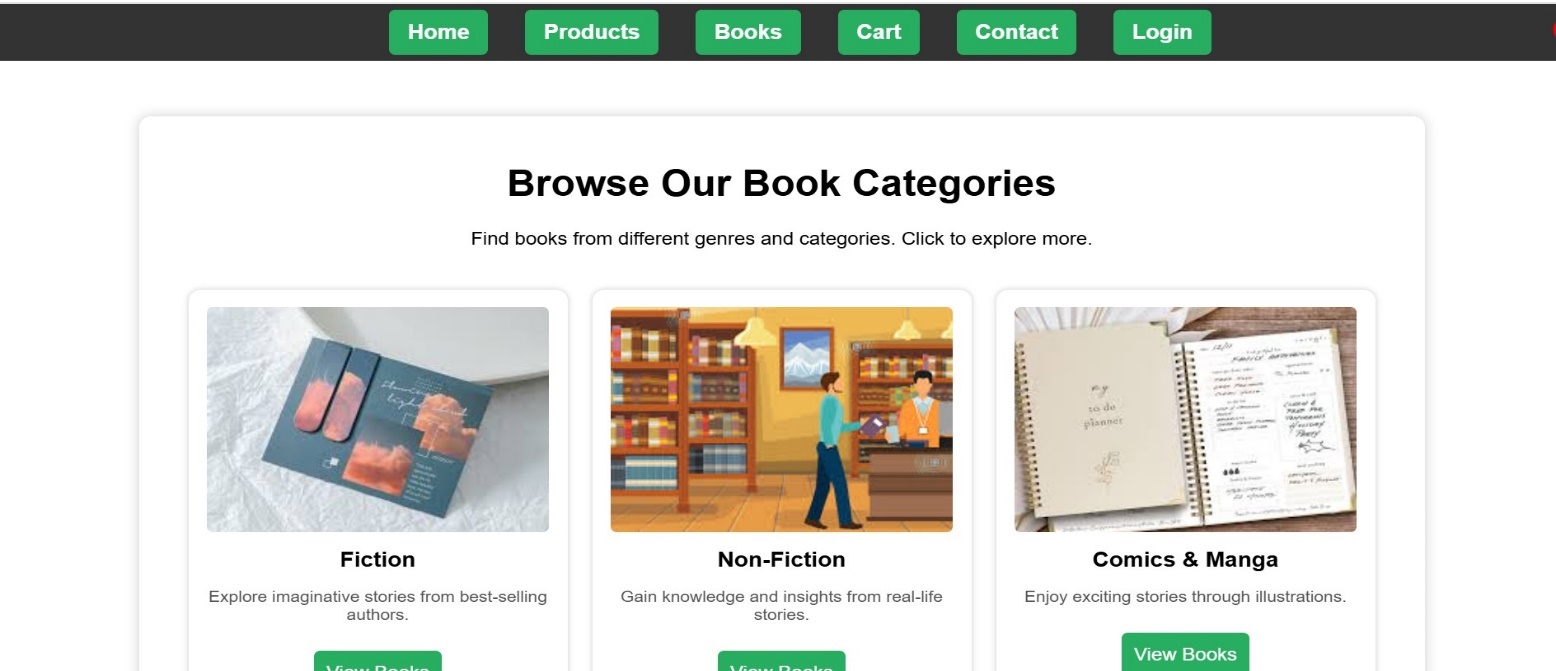
    </script>

</body>

</html>

**Output:**

B. menu/product page output:



**Code:**

C. cart page:

code:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Shopping Cart - Library Management</title>

    <link rel="stylesheet" href="styles.css">

    <style>

        body {

            font-family: Arial, sans-serif;

            background: url('background.jpg') no-repeat center center fixed;

            background-size: cover;

            margin: 0;

            padding: 0;

        }

        nav {

            background: rgba(0, 0, 0, 0.8);

            padding: 15px;

            text-align: center;

            width: 100%;

            position: fixed;

            top: 0;

            left: 0;

        }

        nav ul {

            list-style: none;

            padding: 0;

            margin: 0;

            display: flex;

            justify-content: center;

        }

        nav ul li {

            margin: 0 15px;

        }

        nav ul li a {

            color: white;

            text-decoration: none;

            font-size: 18px;

            font-weight: bold;

            padding: 10px 15px;

            background: #27ae60;

            border-radius: 5px;

            transition: background 0.3s;

        }

        nav ul li a:hover {

            background: #219150;

        }

        .nav-link.active {

            background: #f39c12;

            color: black;

        }

        .cart-count {

            background: red;

            color: white;

            font-size: 14px;

            padding: 3px 7px;

            border-radius: 50%;

            position: absolute;

            top: 10px;

            right: 10px;

            display: none;

        }

        .container {

            max-width: 900px;

            margin: 100px auto;

            padding: 20px;

            background: rgba(255, 255, 255, 0.9);

            border-radius: 10px;

            box-shadow: 0 0 10px rgba(0, 0, 0, 0.2);

        }

        .cart-item {

            display: flex;

            justify-content: space-between;

            padding: 10px;

            border-bottom: 1px solid #ccc;

        }

        .cart-item button {

            background: red;

            color: white;

            border: none;

            padding: 5px;

            cursor: pointer;

            border-radius: 5px;

        }

        .cart-item button:hover {

            background: darkred;

        }

        .cart-total {

            font-size: 18px;

            font-weight: bold;

            margin-top: 20px;

        }

    </style>

</head>

<body>

    <nav>

        <ul>

            <li><a href="home.html" class="nav-link">Home</a></li>

            <li><a href="products.html" class="nav-link">Books</a></li>

            <li><a href="cart.html" class="nav-link active">Cart <span id="cart-count" class="cart-count">0</span></a></li>

            <li><a href="contact.html" class="nav-link">Contact</a></li>

        </ul>

    </nav>

  <div class="container">

        <h1>Your Cart</h1>

        <div id="cart-items"></div>

        <p class="cart-total">Total Books: <span id="total-books">0</span></p>

    </div>

<script>

        function updateCartCount() {

            let cart = JSON.parse(localStorage.getItem("cart")) || [];

            let cartCount = document.getElementById("cart-count");

            cartCount.textContent = cart.length;

            cartCount.style.display = cart.length > 0 ? "inline-block" : "none";

            document.getElementById("total-books").textContent = cart.length;

        }

        function displayCart() {

            let cart = JSON.parse(localStorage.getItem("cart")) || [];

            let cartItemsContainer = document.getElementById("cart-items");

            cartItemsContainer.innerHTML = "";

            if (cart.length === 0) {

                cartItemsContainer.innerHTML = "<p>Your cart is empty.</p>";

                return;

            }

  cart.forEach((item, index) => {

                let div = document.createElement("div");

                div.classList.add("cart-item");

                div.innerHTML = `

                    <span>${item.name} - $${item.price.toFixed(2)}</span>

                    <button onclick="removeFromCart(${index})">Remove</button>

               cartItemsContainer.appendChild(div);

            });

updateCartCount();

        }

function removeFromCart(index) {

            let cart = JSON.parse(localStorage.getItem("cart")) || [];

            cart.splice(index, 1);

            localStorage.setItem("cart", JSON.stringify(cart));

            displayCart();

        }

    document.addEventListener("DOMContentLoaded", () => {

            updateCartCount();

            displayCart();

        });

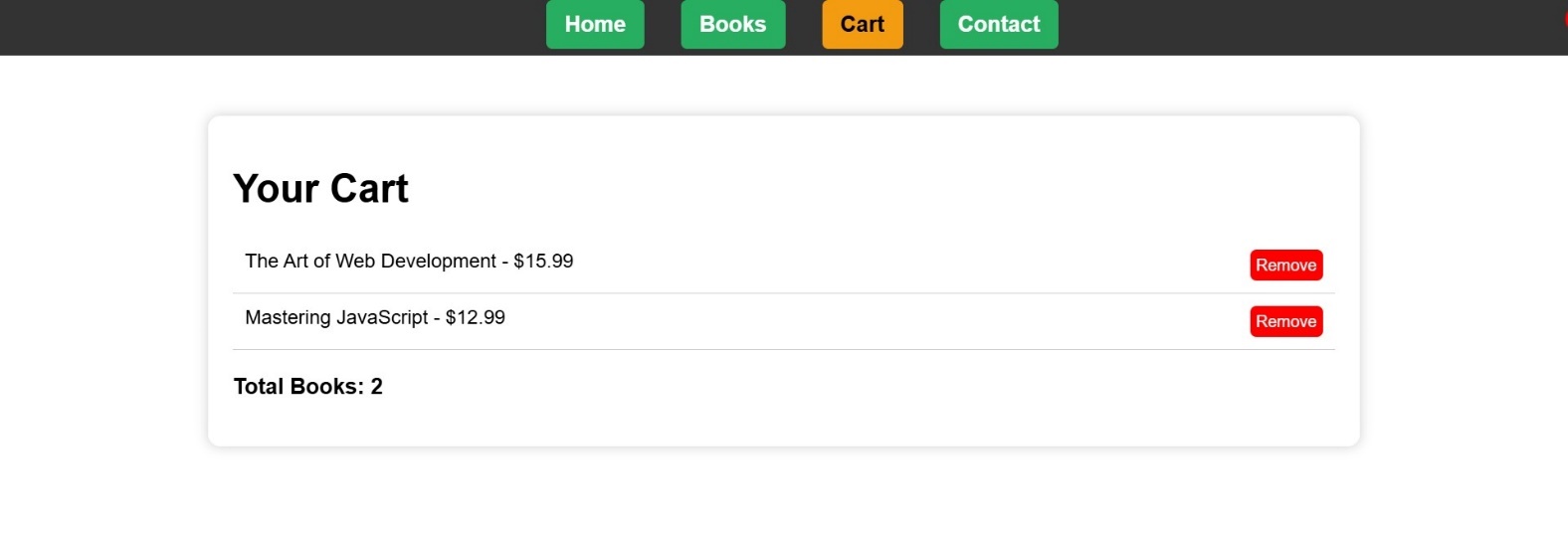
    </script>

</body>

</html>

**Output:**

C. cart page  output:



**Code:**

D. about us page:

code:

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8" />

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

  <title>About Us | Library Management System</title>

  <style>

    \* {

      margin: 0;

      padding: 0;

      box-sizing: border-box;

      font-family: 'Segoe UI', sans-serif;

    }

  body {

      line-height: 1.6;

      background-color: #f5f5f5;

      color: #333;

    }

  header {

      background-color: #6F4F37;

      color: #fff;

      padding: 20px;

      display: flex;

      justify-content: space-between;

      align-items: center;

    }

header .logo {

      font-size: 24px;

      font-weight: bold;

    }

  nav a {

      color: #fff;

      text-decoration: none;

      margin-left: 15px;

      font-weight: 500;

    }

  nav a.active {

      border-bottom: 2px solid #fff;

    }

.about-hero {

      background-color: #D8CAB8;

      padding: 60px 20px;

      text-align: center;

    }

.about-hero h1 {

      font-size: 36px;

      margin-bottom: 10px;

    }

    .about-hero p {

      font-size: 18px;

      max-width: 700px;

      margin: auto;

    }

    .about-content {

      padding: 40px 20px;

      max-width: 900px;

      margin: auto;

    }

    .about-content h2 {

      color: #6F4F37;

      margin-bottom: 10px;

    }

    .about-content p {

      margin-bottom: 20px;

      text-align: justify;

    }

    .team-section {

      padding: 40px 20px;

      background-color: #fff;

      max-width: 1000px;

      margin: auto;

    }

    .team-section h2 {

      text-align: center;

      margin-bottom: 30px;

    }

    .team-grid {

      display: flex;

      flex-wrap: wrap;

      justify-content: space-around;

    }

    .team-member {

      width: 200px;

      text-align: center;

      margin: 20px;

    }

    .team-member img {

      width: 100%;

      border-radius: 10px;

      height: 200px;

      object-fit: cover;

    }

    footer {

      background-color: #6F4F37;

      color: #fff;

      text-align: center;

      padding: 20px;

      margin-top: 40px;

    }

    footer .socials a {

      color: #fff;

      text-decoration: none;

      margin: 0 10px;

    }

  </style>

</head>

<body>

  <header>

    <div class="logo">📚 MyLibrary</div>

    <nav>

      <a href="index.html">Home</a>

      <a href="catalog.html">Catalog</a>

      <a href="about.html" class="active">About</a>

      <a href="contact.html">Contact</a>

      <a href="login.html">Login</a>

    </nav>

  </header>

  <section class="about-hero">

    <h1>About Our Library</h1>

    <p>Empowering minds and communities through books, technology, and lifelong learning opportunities.</p>

  </section>

  <section class="about-content">

    <h2>Our Mission</h2>

    <p>

      The Library Management System was developed to simplify how readers interact with the library. From borrowing books to accessing research journals and managing digital records, we aim to deliver a seamless library experience for students, educators, and lifelong learners.

    </p>

    <p>

      Our system promotes digital literacy, environmental sustainability, and ease of access to a vast catalog of resources. Whether you're preparing for an exam, doing research, or exploring your favorite genre, our library is here to support your journey.

    </p>

    <h2>Our Vision</h2>

    <p>

      We envision a future where every student, regardless of background, can access quality information. Our platform is designed to reduce library bottlenecks, encourage self-learning, and support educational institutions with reliable data management tools.

    </p>

  </section>

  <section class="team-section">

    <h2>Meet the Team</h2>

    <div class="team-grid">

      <div class="team-member">

        <img src="https://via.placeholder.com/200x200?text=Riya" alt="Riya Sharma">

        <h3>Riya Sharma</h3>

        <p>Head Librarian</p>

      </div>

      <div class="team-member">

        <img src="https://via.placeholder.com/200x200?text=Arjun" alt="Arjun Mehta">

        <h3>Arjun Mehta</h3>

        <p>System Developer</p>

      </div>

      <div class="team-member">

        <img src="https://via.placeholder.com/200x200?text=Sneha" alt="Sneha Kapoor">

        <h3>Sneha Kapoor</h3>

        <p>User Coordinator</p>

      </div>

    </div>

  </section>

  <footer>

    <p>&copy; 2025 MyLibrary. All rights reserved.</p>

    <div class="socials">

      <a href="#">Facebook</a> | <a href="#">Instagram</a> | <a href="#">Twitter</a>

    </div>

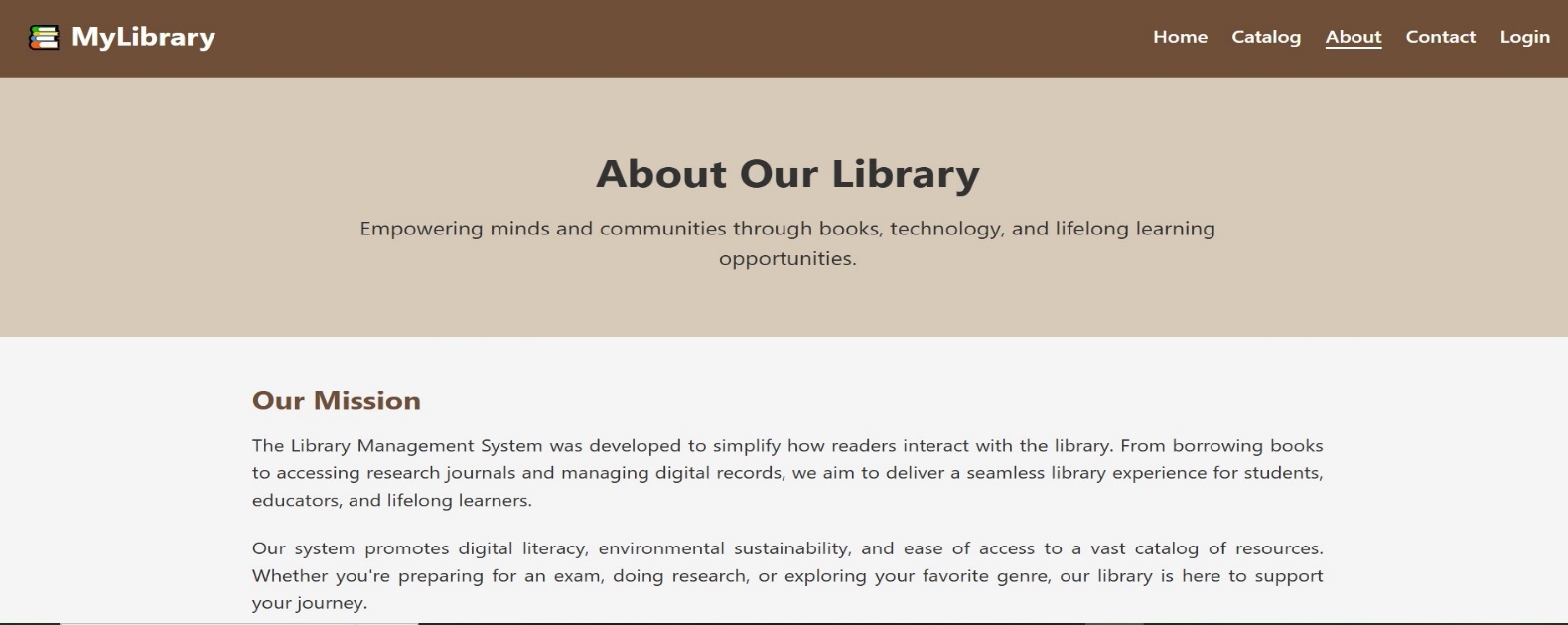
  </footer>

</body>

</html>

**Output:**

D. about us page  output:



**Code:**

E. contact us page:

code:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Contact Us - Library Management</title>

    <link rel="stylesheet" href="styles.css">

    <style>

        body {

            font-family: Arial, sans-serif;

            background: url('background.jpg') no-repeat center center fixed;

            background-size: cover;

            margin: 0;

            padding: 0;

        }

        nav {

            background: rgba(0, 0, 0, 0.8);

            padding: 15px;

            text-align: center;

            width: 100%;

            position: fixed;

            top: 0;

            left: 0;

            z-index: 1000;

        }

        nav ul {

            list-style: none;

            padding: 0;

            margin: 0;

            display: flex;

            justify-content: center;

        }

        nav ul li {

            margin: 0 15px;

        }

        nav ul li a {

            color: white;

            text-decoration: none;

            font-size: 18px;

            font-weight: bold;

            padding: 10px 15px;

            background: #27ae60;

            border-radius: 5px;

            transition: background 0.3s;

        }

        nav ul li a:hover {

            background: #219150;

        }

        .nav-link.active {

            background: #f39c12;

            color: black;

        }

        .container {

            max-width: 600px;

            margin: 100px auto;

            padding: 20px;

            background: rgba(255, 255, 255, 0.9);

            border-radius: 10px;

            box-shadow: 0 0 10px rgba(0, 0, 0, 0.2);

        }

        h1 {

            text-align: center;

        }

        label {

            font-weight: bold;

        }

        input, textarea {

            width: 100%;

            padding: 10px;

            margin: 5px 0 15px;

            border: 1px solid #ccc;

            border-radius: 5px;

        }

        button {

            width: 100%;

            padding: 10px;

            background: #27ae60;

            color: white;

            border: none;

            border-radius: 5px;

            font-size: 16px;

            cursor: pointer;

        }

        button:hover {

            background: #219150;

        }

    </style>

</head>

<body>

    <nav>

        <ul>

            <li><a href="home.html">Home</a></li>

            <li><a href="books.html">Books</a></li>

            <li><a href="cart.html">Cart</a></li>

            <li><a href="contact.html" class="active">Contact</a></li>

        </ul>

    </nav>

    <div class="container">

        <h1>Contact Us</h1>

        <p>If you have any questions, feel free to reach out to us.</p>

        <form action="#" method="post">

            <label for="name">Name:</label>

            <input type="text" id="name" name="name" required>

            <label for="email">Email:</label>

            <input type="email" id="email" name="email" required>

            <label for="message">Message:</label>

            <textarea id="message" name="message" rows="5" required></textarea>

            <button type="submit">Send Message</button>

        </form>

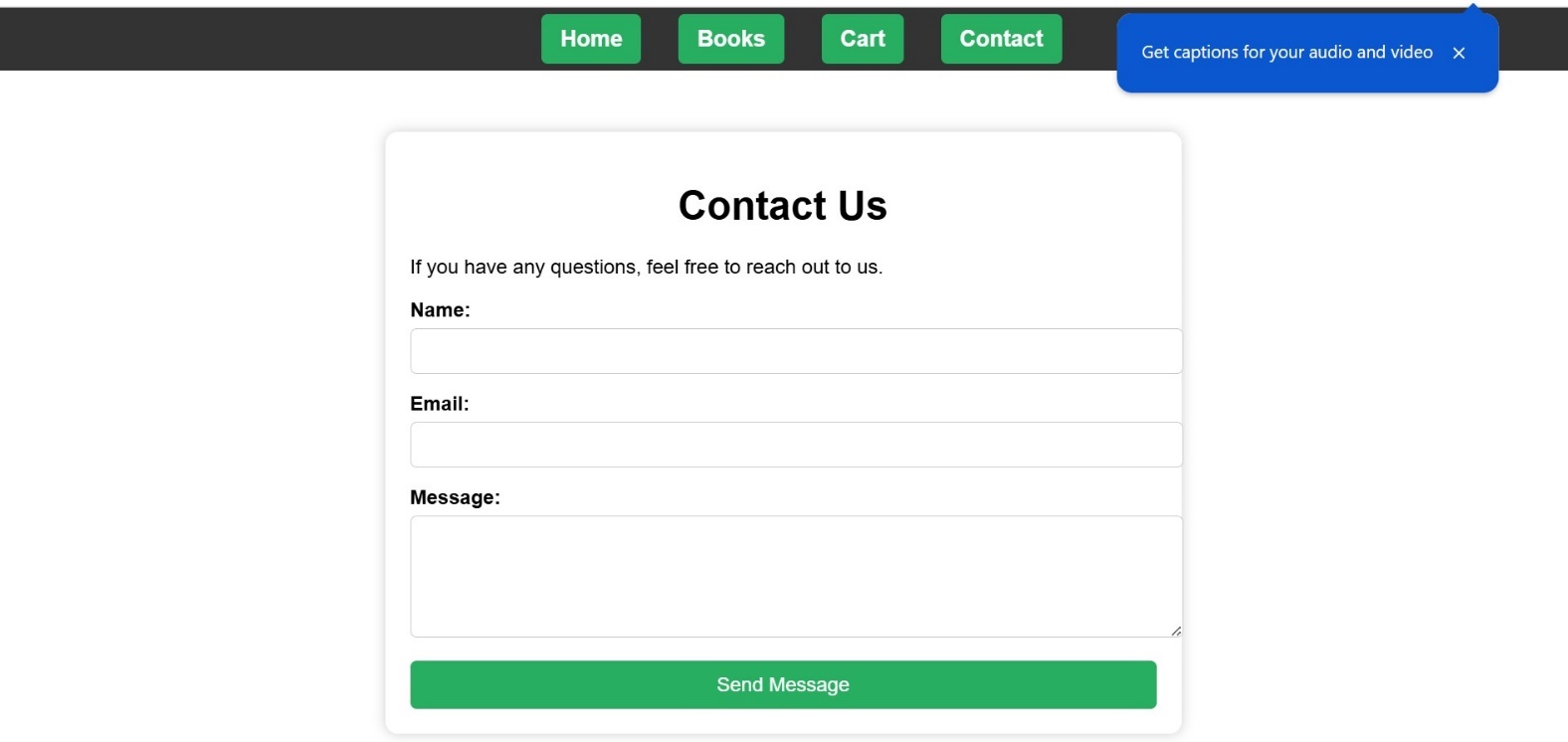
    </div>

</body>

</html>

**Output:**

E. contact us page  output:



**Code:**

F. registration page:

code:

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8" />

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

  <title>Library Registration | MyLibrary</title>

  <style>

    \* {

      margin: 0;

      padding: 0;

      box-sizing: border-box;

      font-family: 'Segoe UI', sans-serif;

    }

    body {

      background-color: #f5f5f5;

      display: flex;

      flex-direction: column;

      align-items: center;

      justify-content: center;

      height: 100vh;

    }

    .form-container {

      background-color: #fff;

      padding: 30px 40px;

      border-radius: 10px;

      box-shadow: 0 0 15px rgba(0,0,0,0.1);

      width: 100%;

      max-width: 400px;

    }

    .form-container h2 {

      text-align: center;

      color: #6F4F37;

      margin-bottom: 20px;

    }

    .form-group {

      margin-bottom: 15px;

    }

    label {

      display: block;

      margin-bottom: 6px;

      color: #333;

    }

    input, select {

      width: 100%;

      padding: 10px;

      border: 1px solid #ccc;

      border-radius: 6px;

    }

    input[type="checkbox"] {

      width: auto;

      margin-right: 8px;

    }

    .terms {

      display: flex;

      align-items: center;

      font-size: 14px;

      margin-top: 10px;

    }

    button {

      width: 100%;

      padding: 10px;

      background-color: #6DBF3A;

      color: #fff;

      border: none;

      border-radius: 6px;

      font-size: 16px;

      cursor: pointer;

      margin-top: 15px;

    }

    button:hover {

      background-color: #5aa732;

    }

    .login-link {

      text-align: center;

      margin-top: 15px;

      font-size: 14px;

    }

    .login-link a {

      color: #6F4F37;

      text-decoration: none;

    }

    .login-link a:hover {

      text-decoration: underline;

    }

  </style>

</head>

<body>

  <div class="form-container">

    <h2>Create Library Account</h2>

    <form action="#" method="post">

      <div class="form-group">

        <label for="name">Full Name:</label>

        <input type="text" id="name" name="name" placeholder="Enter your full name" required />

      </div>

      <div class="form-group">

        <label for="email">Email Address:</label>

        <input type="email" id="email" name="email" placeholder="Enter your email" required />

      </div>

      <div class="form-group">

        <label for="role">Select Role:</label>

        <select id="role" name="role" required>

          <option value="">-- Select --</option>

          <option value="student">Student</option>

          <option value="faculty">Faculty</option>

          <option value="admin">Librarian/Admin</option>

        </select>

      </div>

      <div class="form-group">

        <label for="password">Create Password:</label>

        <input type="password" id="password" name="password" placeholder="Enter password" required />

      </div>

      <div class="terms">

        <input type="checkbox" id="terms" required />

        <label for="terms">I accept the terms & conditions</label>

      </div>

      <button type="submit">Register</button>

    </form>

    <div class="login-link">

      Already have an account? <a href="login.html">Login here</a>

    </div>

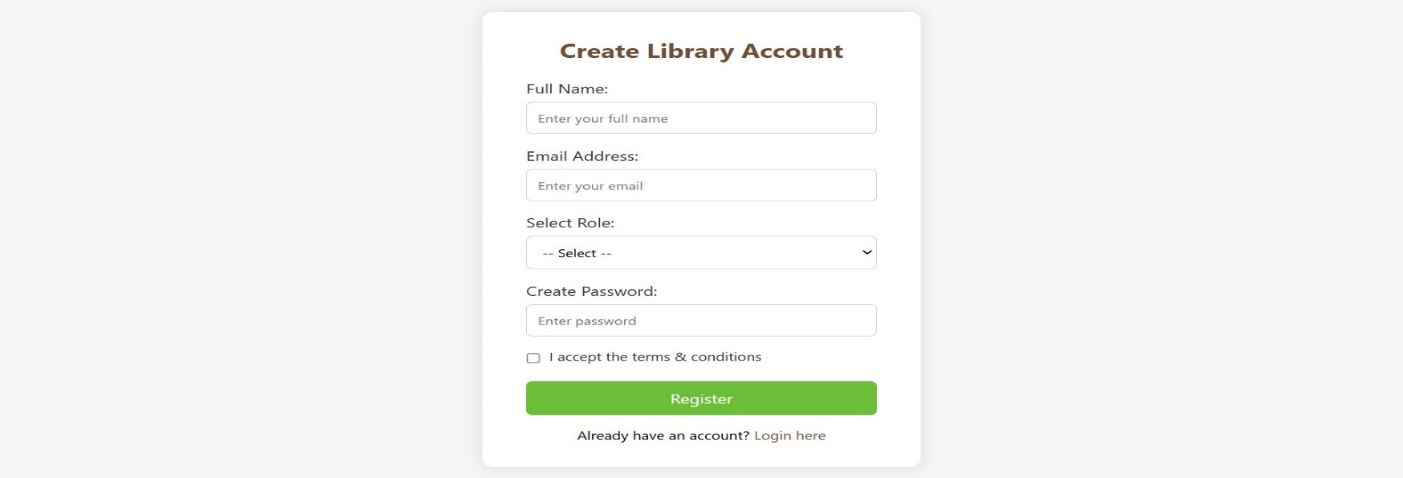
  </div>

</body>

</html>

**Output:**

F. registration page  output:



**Code:**

G. login page:

code:

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8" />

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

  <title>Login - Library Management</title>

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

  <style>

    body {

      font-family: Arial, sans-serif;

      background: url('images/library-bg.jpg') no-repeat center center fixed;

      background-size: cover;

      margin: 0;

      padding: 0;

      color: white;

    }

    nav {

      background: rgba(0, 0, 0, 0.8);

      padding: 15px;

      text-align: center;

      width: 100%;

      position: fixed;

      top: 0;

      left: 0;

      z-index: 1000;

    }

    nav ul {

      list-style: none;

      padding: 0;

      margin: 0;

      display: flex;

      justify-content: center;

    }

    nav ul li {

      margin: 0 15px;

    }

    nav ul li a {

      color: white;

      text-decoration: none;

      font-size: 18px;

      font-weight: bold;

      padding: 10px 15px;

      background: #27ae60;

      border-radius: 5px;

      transition: background 0.3s;

    }

    nav ul li a:hover {

      background: #219150;

    }

    .login-container {

      margin: 120px auto;

      max-width: 400px;

      background: rgba(0, 0, 0, 0.7);

      padding: 30px;

      border-radius: 10px;

      box-shadow: 0 0 10px rgba(0, 0, 0, 0.5);

    }

    h2 {

      text-align: center;

      margin-bottom: 20px;

    }

    input[type="text"],

    input[type="password"] {

      width: 100%;

      padding: 12px;

      margin: 10px 0;

      border: none;

      border-radius: 5px;

      box-sizing: border-box;

    }

    .login-btn {

      width: 100%;

      padding: 12px;

      background: #f39c12;

      color: black;

      font-size: 16px;

      border: none;

      border-radius: 5px;

      cursor: pointer;

    }

    .login-btn:hover {

      background: #e67e22;

    }

  </style>

</head>

<body>

  <nav>

    <ul>

      <li><a href="home.html">Home</a></li>

      <li><a href="books.html">Books</a></li>

      <li><a href="cart.html">Cart</a></li>

      <li><a href="contact.html">Contact</a></li>

      <li><a href="login.html" class="active">Login</a></li>

    </ul>

  </nav>

  <div class="login-container">

    <h2>Login</h2>

    <form onsubmit="handleLogin(event)">

      <input type="text" placeholder="Username" required />

      <input type="password" placeholder="Password" required />

      <button type="submit" class="login-btn">Login</button>

    </form>

  </div>

  <script>

    function handleLogin(event) {

      event.preventDefault(); // Prevents form from reloading the page

      // You can add authentication logic here if needed

      window.location.href = "home.html"; // Redirect to home

    }

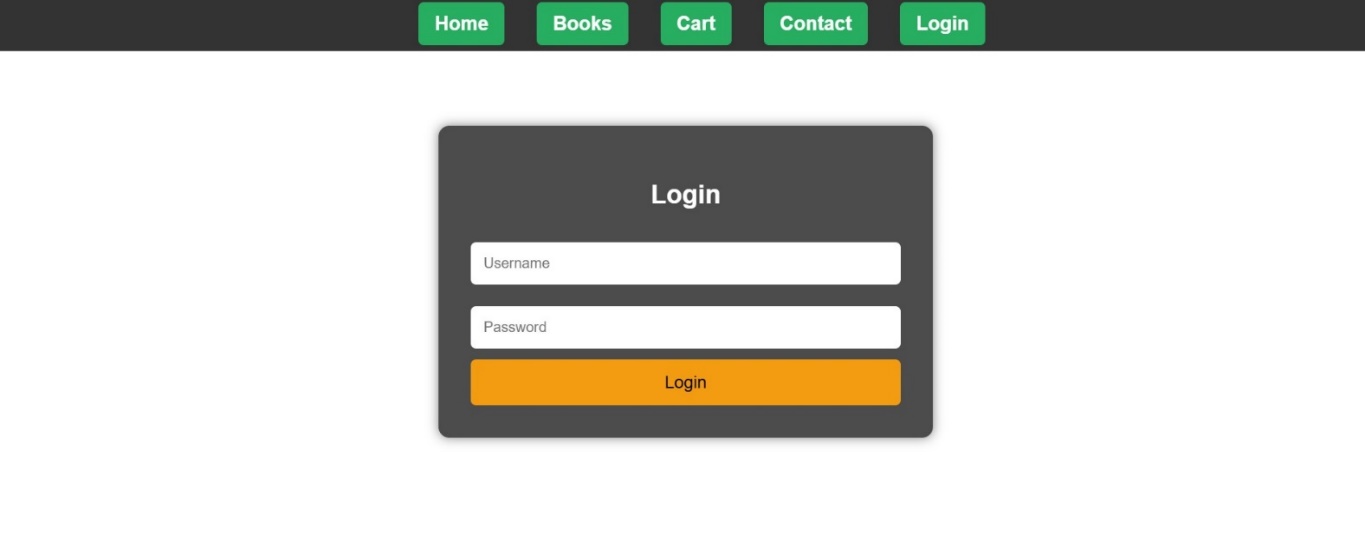
  </script>

</body>

</html>

**Output:**

G. login page  output:



**Conclusion**

The **Library Management System website** combines efficient digital resource access with a mission to promote education and lifelong learning. The use of structured web design, user-friendly forms, and clear categorization of books and academic materials ensures a seamless experience for both readers and administrators. With further backend integration, it can evolve into a fully functional platform supporting digital literacy, sustainable resource management, and enhanced community engagement in academic environments.

**Experiment No.3**

**Problem Statement:**

Enhance the layout of the **Library Management System** website using **CSS Grid** for the **home page**.  
Use CSS Grid to layout the **catalog (books/resources)** in a structured format and style the **categories** with appropriate headings, spacing, separators, images, descriptions, and availability status.

**Theory:**

CSS Theory for Enhancing the Layout of a Library Management Website using CSS Grid

**Introduction to CSS Grid**

**CSS Grid Layout** is a powerful two-dimensional layout system that enables precise placement of items in rows and columns. Unlike Flexbox (which works in one dimension at a time), CSS Grid is ideal for complex and responsive web interfaces—perfect for library systems where resource categorization and layout matter.

Using CSS Grid allows for:

* Clean and consistent book layouts
* Organized sections (such as announcements, new arrivals, and services)
* Structured design for responsive library catalogs

**Why CSS Grid for this Website?**

In a Library Management System, layout and categorization of resources are key to helping users find books, journals, and digital assets efficiently.

**CSS Grid is used to:**

* Display books in responsive rows and columns
* Define areas like “New Arrivals”, “Popular Reads”, or “E-Journals”
* Maintain consistent spacing and alignment of covers, titles, and metadata
* Adapt layouts seamlessly to mobile, tablet, and desktop screens

**1. Home Page Layout with CSS Grid**

The home page is structured using CSS Grid into logical content blocks:

* A full-width navigation header
* A hero section with welcome banner or announcements
* A three-column highlights section (e.g., Digital Library, Reading Rooms, Membership Info)
* A testimonial or news section in row format
* A footer with contact details and useful links

**Grid Benefits on Home Page:**

* Visually defined content zones
* Clean alignment of promotional or informational content
* Easier maintenance and responsive layout without complex media queries

**2. Menu/Product Page Layout Using CSS Grid**

**This page displays all books and resources grouped by categories such as:**

* Fiction
* Non-fiction
* Academic
* Magazines
* E-books

Each resource appears in a grid card layout, showing key details like:

* Book cover image
* Title and author
* Genre or subject
* Availability (e.g., “Available”, “Issued”)
* Action buttons like “Borrow” or “Reserve”

Each product-card inside this grid will have:

* A product image
* A title
* A short description
* Price (highlighted)
* "Add to Cart" button

**Additional Styling Concepts:**

* **Category Headings**: Styled with larger fonts, color backgrounds, or underlines to differentiate sections.
* **Separators**: Thin horizontal lines or borders can visually divide different product categories.
* **Hover Effects**: CSS transitions can enhance interactivity by highlighting cards or changing button styles on hover.
* **Responsive Design**: CSS Grid’s auto-fit and minmax() features allow the grid to adapt automatically to screen size, removing the need for complex media queries.

**Mobile Responsiveness with CSS Grid**

One of CSS Grid’s biggest strengths is its **responsive adaptability**. The grid-template-columns property with auto-fit ensures that items stack or spread out based on available screen space.

**Benefits for mobile users:**

* Grid automatically collapses to 1 or 2 columns
* Touch-friendly layout
* Ensures a smooth browsing experience

Code:

/\* Reset and base styles \*/

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

font-family: 'Segoe UI', sans-serif;

}

body {

background-color: #f5f5f5;

color: #333;

line-height: 1.6;

}

/\* Header Navigation \*/

header {

background-color: #6F4F37;

color: white;

padding: 20px;

display: flex;

justify-content: space-between;

align-items: center;

}

header .logo {

font-size: 24px;

font-weight: bold;

}

nav a {

color: white;

text-decoration: none;

margin-left: 20px;

font-weight: 500;

}

nav a:hover {

text-decoration: underline;

}

/\* Hero Section \*/

.hero {

background-color: #D8CAB8;

padding: 60px 20px;

text-align: center;

}

.hero h1 {

font-size: 36px;

margin-bottom: 15px;

}

.hero p {

font-size: 18px;

max-width: 700px;

margin: 0 auto;

}

/\* Grid Feature Section \*/

.features {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));

gap: 20px;

padding: 40px 20px;

background-color: #ffffff;

}

.feature {

background-color: #FAF3E0;

padding: 20px;

border-radius: 10px;

box-shadow: 0 0 10px rgba(0,0,0,0.05);

text-align: center;

transition: transform 0.3s ease;

}

.feature:hover {

transform: scale(1.03);

}

.feature h3 {

color: #6F4F37;

margin-bottom: 10px;

}

.feature p {

font-size: 15px;

}

/\* Testimonials Section \*/

.testimonials {

background-color: #f0f0f0;

padding: 40px 20px;

}

.testimonials h2 {

text-align: center;

margin-bottom: 30px;

color: #6F4F37;

}

.testimonial-grid {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));

gap: 20px;

}

.testimonial {

background-color: white;

padding: 20px;

border-left: 5px solid #6DBF3A;

border-radius: 6px;

box-shadow: 0 0 8px rgba(0,0,0,0.05);

}

.testimonial p {

font-style: italic;

margin-bottom: 10px;

}

.testimonial span {

font-weight: bold;

display: block;

text-align: right;

}

/\* Footer \*/

footer {

background-color: #6F4F37;

color: white;

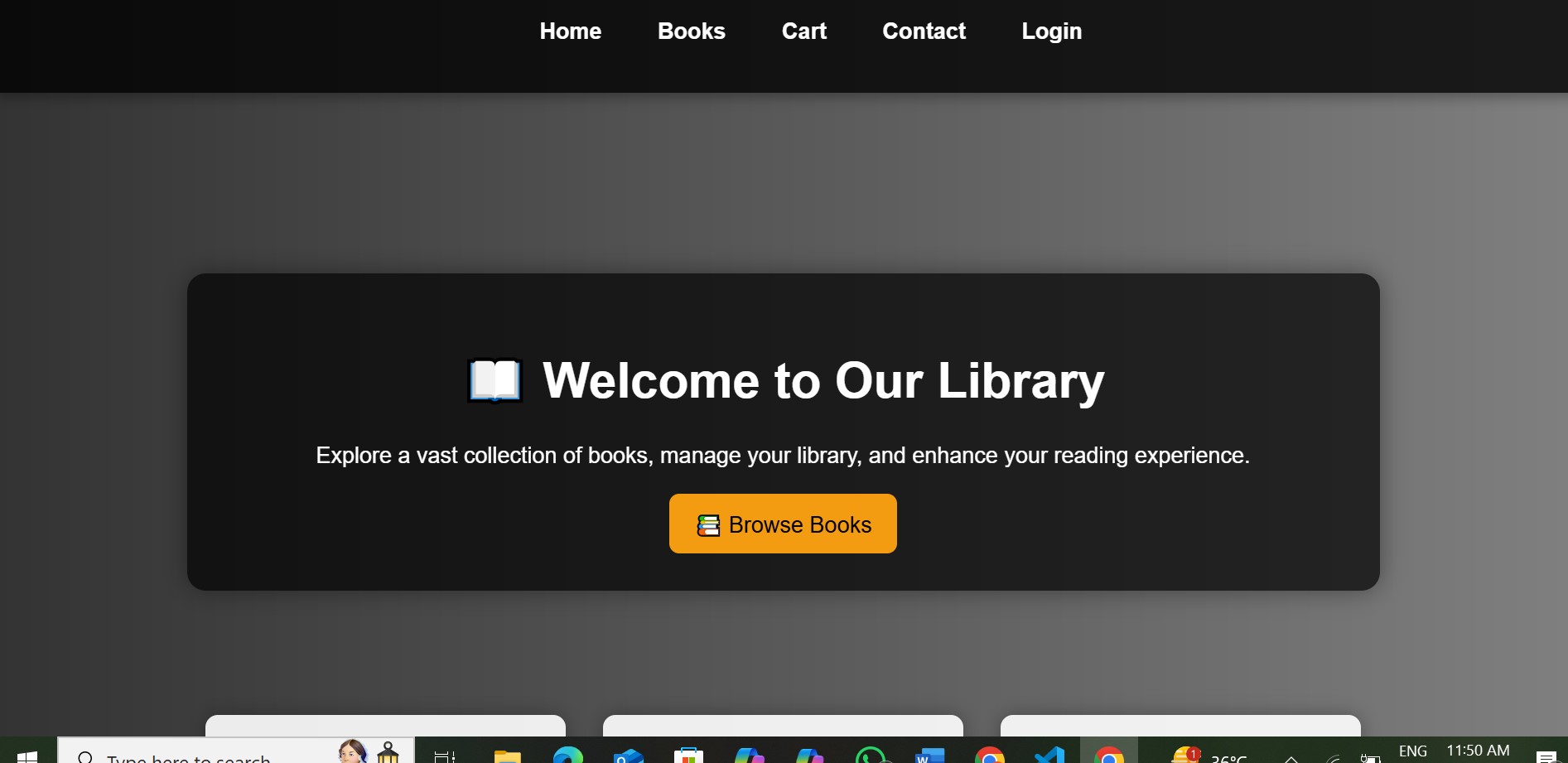
text-align: center;

padding: 20px;

margin-top: 40px;

}

Output



**Conclusion**

CSS Grid allows the Library Management System to present content clearly, responsively, and professionally. Whether it's the home page or the catalog page, Grid ensures every section is organized, scalable, and easy to navigate—improving usability for students, researchers, and admins alike.

**Experiment No.4**

**Problem Statement:**

**CSS Theory:** Enhancing and Styling Key Pages in a Library Management System Website1.

**Why CSS Styling Matters in Library Websites**

When users visit your digital library, their first impression is based on design and layout. A clean, organized, and visually appealing interface builds trust and encourages interaction—whether users are browsing the catalog, logging in, or reserving a book.

Well-applied CSS styling using margins, paddings, spacing, input field designs, and color schemes helps:

* Make content easier to read and scan
* Provide a sense of structure and logical flow
* Improve accessibility and user experience (UX)
* Encourage user actions like logging in, submitting contact forms, or managing borrowed books

**1. Book Cart / Borrowed Items Page**

This page allows users to review books they’ve selected to borrow. It must be clean, organized, and user-focused.

**Key Styling Techniques:**

* Add padding around each listed book item for visual separation
* Use margins to space out the book title, due date, return button, and renewal options
* Style input fields (e.g., renewal dates or quantity of items) with soft borders and large clickable areas
* Highlight due dates or overdue statuses with color indicators (e.g., red for overdue)
* Use bold font and shaded boxes for “Total Items” or “Expected Return Date” summaries

**Result:** A structured, accessible layout that simplifies borrowing management

2 **About Us Page**

This page explains the mission, background, and values of the library, helping users build a connection with the institution.

**Key Styling Techniques:**

* Use adequate line height, justified alignment, and padding for long text readability
* Add white space between key sections like "Our Mission", "History", and "Meet Our Staff"
* Use soft background colors or separator lines for clarity
* Style staff images with rounded corners and equal spacing
* Highlight quotes or core values in callout boxes or grid cards

**Result:** A professional and inviting layout that communicates credibility and purpose

**3. Contact Page**

The contact page should provide easy ways to reach out to library staff for queries, support, or feedback.

**Key Styling Techniques:**

* Style form fields uniformly with consistent width, padding, and border-radius
* Apply margin-bottom between form inputs for spacing
* Use focus states (e.g., border highlight or background change) to improve usability
* Style submit button with a distinct color and hover effect
* Center the form container with balanced margins and paddings

**Result:** A modern, accessible form interface that improves communication and response rates

**4. Admin/User Registration Form**

This form allows students, faculty, or admins to register on the system. It must be secure, clean, and user-friendly.

**Key Styling Techniques:**

* Group inputs logically (e.g., Name, Email, Password, Role)
* Use labels and placeholders for clarity
* Apply consistent spacing and alignment for all fields
* Add a light background with rounded corners and subtle shadows around the form
* Include hover styles for buttons and basic inline validation (e.g., "password must be 8+ characters")

**Result:** A trustworthy and intuitive form that encourages complete registration.

**5. Admin/User Login Form**

Login forms must be fast, focused, and informative—especially in an institutional environment.

**Key Styling Techniques:**

* Center the login box both vertically and horizontally
* Add ample padding inside the form container for clarity
* Use clear focus states and error messages (e.g., red for "invalid credentials")
* Maintain visual hierarchy (larger font for “Login”, smaller for "Forgot Password?")
* Style buttons with hover transitions and align links clearly

**Result:** A polished, accessible login interface that ensures confident, secure access to the system

**Code:**

cart page:

code:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Shopping Cart - Library Management</title>

    <link rel="stylesheet" href="styles.css">

    <style>

        body {

            font-family: Arial, sans-serif;

            background: url('background.jpg') no-repeat center center fixed;

            background-size: cover;

            margin: 0;

            padding: 0;

        }

        nav {

            background: rgba(0, 0, 0, 0.8);

            padding: 15px;

            text-align: center;

            width: 100%;

            position: fixed;

            top: 0;

            left: 0;

        }

        nav ul {

            list-style: none;

            padding: 0;

            margin: 0;

            display: flex;

            justify-content: center;

        }

        nav ul li {

            margin: 0 15px;

        }

        nav ul li a {

            color: white;

            text-decoration: none;

            font-size: 18px;

            font-weight: bold;

            padding: 10px 15px;

            background: #27ae60;

            border-radius: 5px;

            transition: background 0.3s;

        }

        nav ul li a:hover {

            background: #219150;

        }

        .nav-link.active {

            background: #f39c12;

            color: black;

        }

        .cart-count {

            background: red;

            color: white;

            font-size: 14px;

            padding: 3px 7px;

            border-radius: 50%;

            position: absolute;

            top: 10px;

            right: 10px;

            display: none;

        }

        .container {

            max-width: 900px;

            margin: 100px auto;

            padding: 20px;

            background: rgba(255, 255, 255, 0.9);

            border-radius: 10px;

            box-shadow: 0 0 10px rgba(0, 0, 0, 0.2);

        }

        .cart-item {

            display: flex;

            justify-content: space-between;

            padding: 10px;

            border-bottom: 1px solid #ccc;

        }

        .cart-item button {

            background: red;

            color: white;

            border: none;

            padding: 5px;

            cursor: pointer;

            border-radius: 5px;

        }

        .cart-item button:hover {

            background: darkred;

        }

        .cart-total {

            font-size: 18px;

            font-weight: bold;

            margin-top: 20px;

        }

    </style>

</head>

<body>

    <nav>

        <ul>

            <li><a href="home.html" class="nav-link">Home</a></li>

            <li><a href="products.html" class="nav-link">Books</a></li>

            <li><a href="cart.html" class="nav-link active">Cart <span id="cart-count" class="cart-count">0</span></a></li>

            <li><a href="contact.html" class="nav-link">Contact</a></li>

        </ul>

    </nav>

  <div class="container">

        <h1>Your Cart</h1>

        <div id="cart-items"></div>

        <p class="cart-total">Total Books: <span id="total-books">0</span></p>

    </div>

<script>

        function updateCartCount() {

            let cart = JSON.parse(localStorage.getItem("cart")) || [];

            let cartCount = document.getElementById("cart-count");

            cartCount.textContent = cart.length;

            cartCount.style.display = cart.length > 0 ? "inline-block" : "none";

            document.getElementById("total-books").textContent = cart.length;

        }

function displayCart() {

            let cart = JSON.parse(localStorage.getItem("cart")) || [];

            let cartItemsContainer = document.getElementById("cart-items");

            cartItemsContainer.innerHTML = "";

            if (cart.length === 0) {

                cartItemsContainer.innerHTML = "<p>Your cart is empty.</p>";

                return;

            }

cart.forEach((item, index) => {

                let div = document.createElement("div");

                div.classList.add("cart-item");

                div.innerHTML = `

                    <span>${item.name} - $${item.price.toFixed(2)}</span>

                    <button onclick="removeFromCart(${index})">Remove</button>

                `;

                cartItemsContainer.appendChild(div);

            });

            updateCartCount();

        }

  function removeFromCart(index) {

            let cart = JSON.parse(localStorage.getItem("cart")) || [];

            cart.splice(index, 1);

            localStorage.setItem("cart", JSON.stringify(cart));

            displayCart();

        }

  document.addEventListener("DOMContentLoaded", () => {

            updateCartCount();

            displayCart();

        });

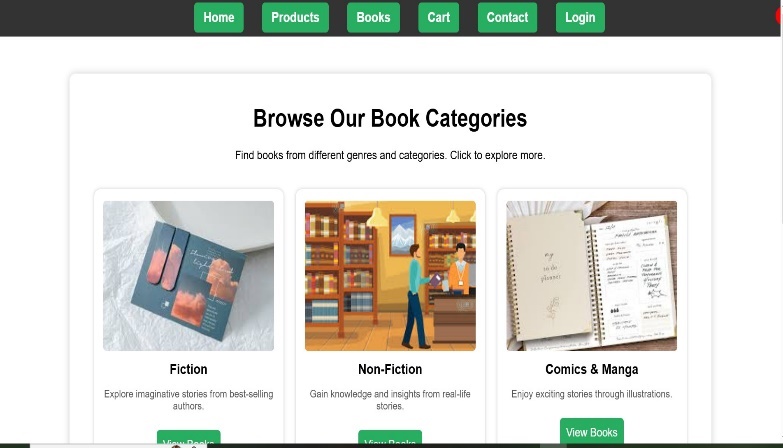
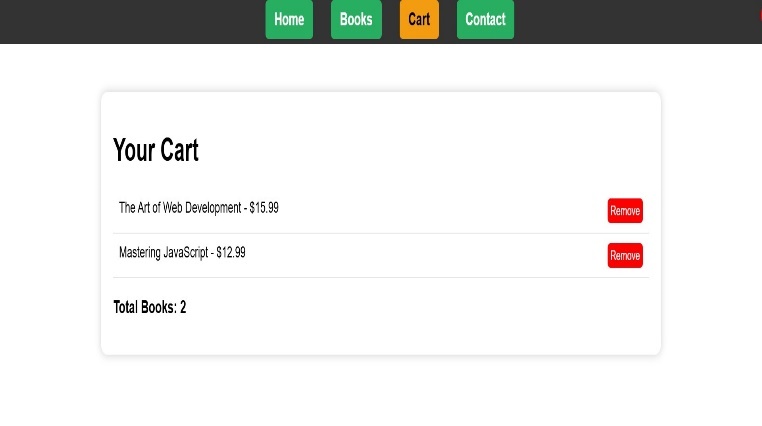
    </script>

</body>

</html>

**Output:**

cart page  output:



**Code:**

registration page:

code:

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8" />

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

  <title>Library Registration | MyLibrary</title>

  <style>

    \* {

      margin: 0;

      padding: 0;

      box-sizing: border-box;

      font-family: 'Segoe UI', sans-serif;

    }

body {

      background-color: #f5f5f5;

      display: flex;

      flex-direction: column;

      align-items: center;

      justify-content: center;

      height: 100vh;

    }

.form-container {

      background-color: #fff;

      padding: 30px 40px;

      border-radius: 10px;

      box-shadow: 0 0 15px rgba(0,0,0,0.1);

      width: 100%;

      max-width: 400px;

    }

.form-container h2 {

      text-align: center;

      color: #6F4F37;

      margin-bottom: 20px;

    }

.form-group {

      margin-bottom: 15px;

    }

  label {

      display: block;

      margin-bottom: 6px;

      color: #333;

    }

input, select {

      width: 100%;

      padding: 10px;

      border: 1px solid #ccc;

      border-radius: 6px;

    }

  input[type="checkbox"] {

      width: auto;

      margin-right: 8px;

    }

.terms {

      display: flex;

      align-items: center;

      font-size: 14px;

      margin-top: 10px;

    }

button {

      width: 100%;

      padding: 10px;

      background-color: #6DBF3A;

      color: #fff;

      border: none;

      border-radius: 6px;

      font-size: 16px;

      cursor: pointer;

      margin-top: 15px;

    }

button:hover {

      background-color: #5aa732;

    }

.login-link {

      text-align: center;

      margin-top: 15px;

      font-size: 14px;

    }

  .login-link a {

      color: #6F4F37;

      text-decoration: none;

    }

.login-link a:hover {

      text-decoration: underline;

    }

  </style>

</head>

<body>

  <div class="form-container">

    <h2>Create Library Account</h2>

    <form action="#" method="post">

      <div class="form-group">

        <label for="name">Full Name:</label>

        <input type="text" id="name" name="name" placeholder="Enter your full name" required />

      </div>

<div class="form-group">

        <label for="email">Email Address:</label>

        <input type="email" id="email" name="email" placeholder="Enter your email" required />

      </div>

<div class="form-group">

        <label for="role">Select Role:</label>

        <select id="role" name="role" required>

          <option value="">-- Select --</option>

          <option value="student">Student</option>

          <option value="faculty">Faculty</option>

          <option value="admin">Librarian/Admin</option>

        </select>

      </div>

  <div class="form-group">

        <label for="password">Create Password:</label>

        <input type="password" id="password" name="password" placeholder="Enter password" required />

      </div>

  <div class="terms">

        <input type="checkbox" id="terms" required />

        <label for="terms">I accept the terms & conditions</label>

      </div>

<button type="submit">Register</button>

    </form>

  <div class="login-link">

      Already have an account? <a href="login.html">Login here</a>

    </div>

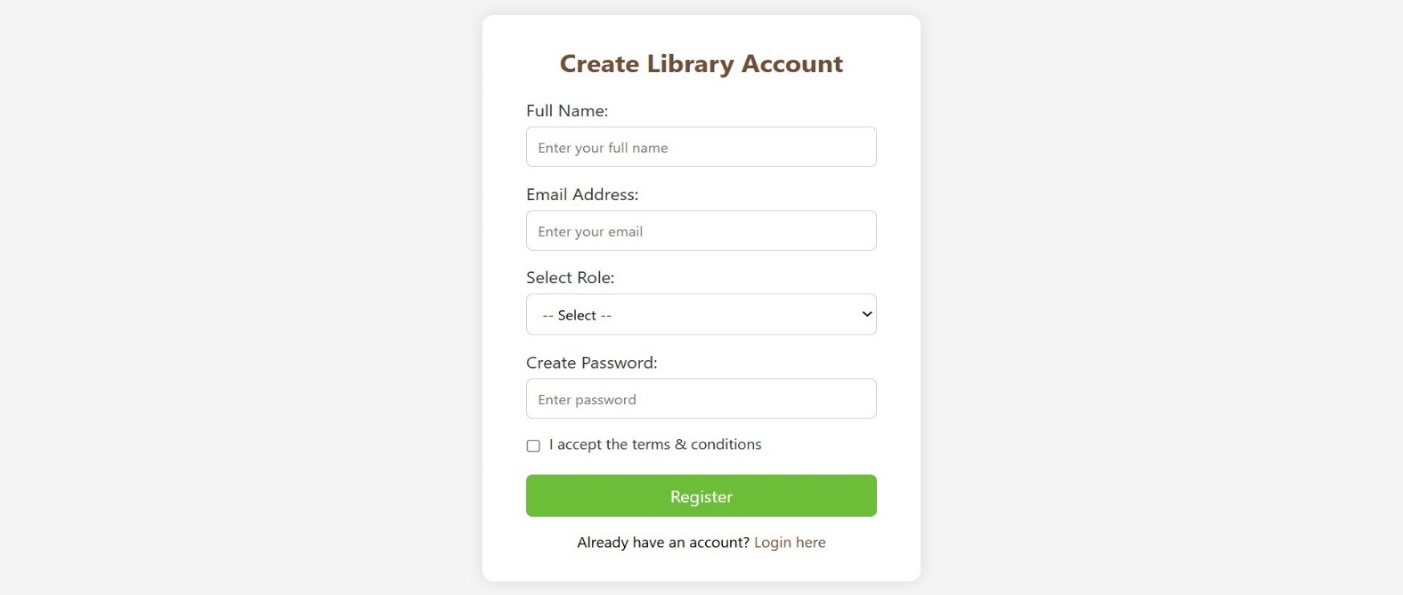
  </div>

</body>

</html>

**Output:**

registration page  output:



**Code:**

login page:

code:

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8" />

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

  <title>Login - Library Management</title>

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

  <style>

    body {

      font-family: Arial, sans-serif;

      background: url('images/library-bg.jpg') no-repeat center center fixed;

      background-size: cover;

      margin: 0;

      padding: 0;

      color: white;

    }

    nav {

      background: rgba(0, 0, 0, 0.8);

      padding: 15px;

      text-align: center;

      width: 100%;

      position: fixed;

      top: 0;

      left: 0;

      z-index: 1000;

    }

    nav ul {

      list-style: none;

      padding: 0;

      margin: 0;

      display: flex;

      justify-content: center;

    }

    nav ul li {

      margin: 0 15px;

    }

    nav ul li a {

      color: white;

      text-decoration: none;

      font-size: 18px;

      font-weight: bold;

      padding: 10px 15px;

      background: #27ae60;

      border-radius: 5px;

      transition: background 0.3s;

    }

    nav ul li a:hover {

      background: #219150;

    }

    .login-container {

      margin: 120px auto;

      max-width: 400px;

      background: rgba(0, 0, 0, 0.7);

      padding: 30px;

      border-radius: 10px;

      box-shadow: 0 0 10px rgba(0, 0, 0, 0.5);

    }

    h2 {

      text-align: center;

      margin-bottom: 20px;

    }

    input[type="text"],

    input[type="password"] {

      width: 100%;

      padding: 12px;

      margin: 10px 0;

      border: none;

      border-radius: 5px;

      box-sizing: border-box;

    }

    .login-btn {

      width: 100%;

      padding: 12px;

      background: #f39c12;

      color: black;

      font-size: 16px;

      border: none;

      border-radius: 5px;

      cursor: pointer;

    }

    .login-btn:hover {

      background: #e67e22;

    }

  </style>

</head>

<body>

  <nav>

    <ul>

      <li><a href="home.html">Home</a></li>

      <li><a href="books.html">Books</a></li>

      <li><a href="cart.html">Cart</a></li>

      <li><a href="contact.html">Contact</a></li>

      <li><a href="login.html" class="active">Login</a></li>

    </ul>

  </nav>

  <div class="login-container">

    <h2>Login</h2>

    <form onsubmit="handleLogin(event)">

      <input type="text" placeholder="Username" required />

      <input type="password" placeholder="Password" required />

      <button type="submit" class="login-btn">Login</button>

    </form>

  </div>

  <script>

    function handleLogin(event) {

      event.preventDefault(); // Prevents form from reloading the page

      // You can add authentication logic here if needed

      window.location.href = "home.html"; // Redirect to home

    }

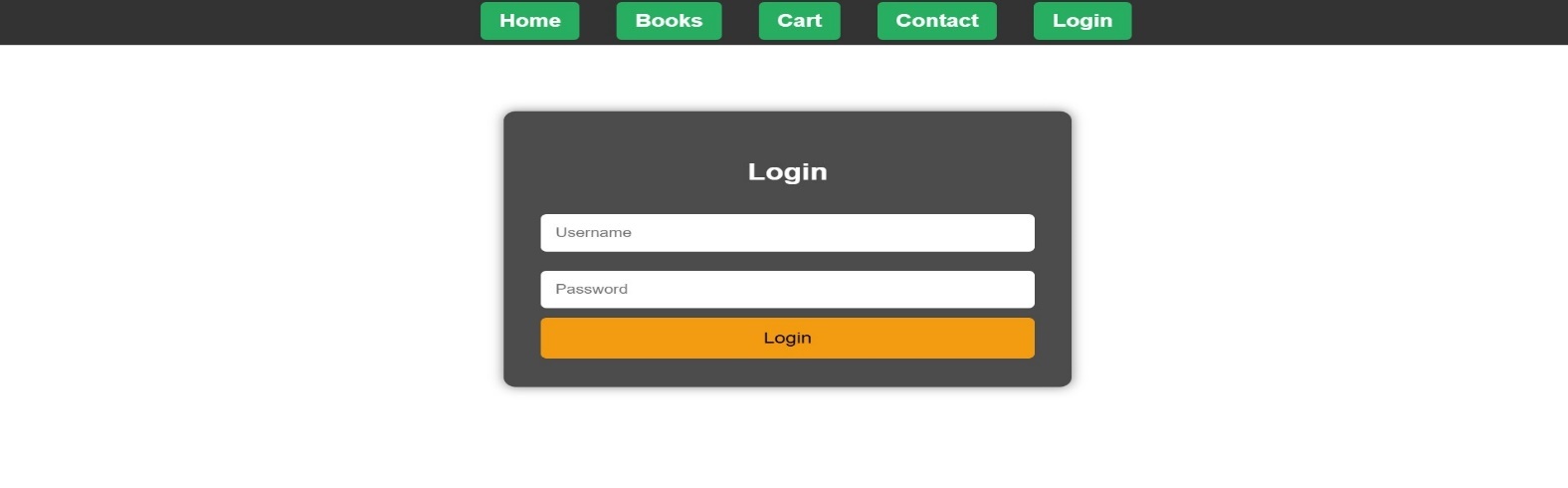
  </script>

</body>

</html>

**Output:**

login page  output:



**Conclusion**

The visual and functional success of any digital library platform, especially one like your Library Management System, relies heavily on how well the pages are styled using CSS. Applying appropriate margins, paddings, spacing, and input field enhancements ensures:

* Better user experience (UX) for students, researchers, and staff
* Improved readability and accessibility of catalogs and forms
* A more polished, professional appearance that reflects institutional quality
* Higher engagement, trust, and usability across devices

Each page — whether it’s the catalog, about page, contact form, or login/registration — plays a vital role in the user's digital journey. Proper CSS styling not only improves functionality and comfort but also reinforces the library’s identity, professionalism, and commitment to accessibility.

In modern web design, CSS is not just about visuals — it's a tool for delivering seamless, user-centered experiences, guiding readers, learners, and educators through the platform one pixel at a time.

**Experiment No.5**

**Problem Statement:**

JavaScript Theory: User Registration, Login, Validation, and Cart Functionality

**Introduction**

In modern web development, client-side scripting using JavaScript is essential for building interactive, responsive, and user-friendly applications. For a Library Management System, implementing features such as user registration, login, form validation, and borrowed books management is critical for ensuring smooth user engagement and personalized library services.

1. **User Registration and Login Forms**

These forms are fundamental for identifying users (students, faculty, or admins) and enabling access to personalized features like borrowing history, reserved books, or digital resources.

Registration Form

The registration form allows new users to sign up by entering personal details such as:

* Full Name
* Email Address
* Password & Confirm Password
* Optional: Phone Number, Role (Student, Faculty, Admin)

JavaScript responsibilities in registration:

* Ensuring no required fields are left blank
* Validating email format using regular expressions
* Checking password complexity (minimum length, uppercase, numbers, symbols)
* Ensuring password and confirm password fields match
* Providing real-time, user-friendly error messages and hints

**Login Form**

The login form enables existing users to access their library accounts securely using email and password.

JavaScript responsibilities in login:

* Ensuring email and password fields are not empty
* Verifying the email format is valid
* Matching login credentials with stored records (in local storage or backend)
* Redirecting users to their user dashboard, book catalog, or borrowed items page after successful login

2. **JavaScript Form Validations**

Form validation is essential for maintaining data integrity, reducing server load, and improving user experience by providing immediate feedback.

Common validation tasks include:

* Verifying all mandatory fields are filled
* Using regex to check valid email format
* Validating password strength (e.g., 8+ characters, 1 number, 1 symbol)
* Confirming both password fields match
* Displaying inline or pop-up error messages for incorrect or incomplete inputs

**Why Client-Side Validation Matters:**

While server-side validation is crucial for security, JavaScript-based validation allows users to correct input errors instantly, without the need for a page reload or round trip to the server.

3. **Borrowed Books / Cart Functionality**

In a Library Management System, the "cart" is equivalent to a list of books a user selects to borrow or reserve. This functionality improves usability by allowing users to manage selections before confirming their borrowing.

Key JavaScript implementations for borrowed books functionality:

* Adding selected books to a virtual borrow list dynamically
* Updating quantity for multi-copy reservations (if allowed)
* Removing books from the borrow list
* Calculating expected return dates or due date warnings
* Storing the borrow list in localStorage or sessionStorage for persistence between sessions
* Dynamically displaying borrow list items using DOM manipulation

JavaScript structures the borrow list as an array of objects, each representing a book with properties such as title, author, ISBN, status, and return date. This makes it efficient to render, update, and manage user interactions with borrowed materials.

**Code:**

F. registration page:

code:

<!DOCTYPE html>

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8" />

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

  <title>Library Registration | MyLibrary</title>

  <style>

    \* {

      margin: 0;

      padding: 0;

      box-sizing: border-box;

      font-family: 'Segoe UI', sans-serif;

    }

body {

      background-color: #f5f5f5;

      display: flex;

      flex-direction: column;

      align-items: center;

      justify-content: center;

      height: 100vh;

    }

form-container {

      background-color: #fff;

      padding: 30px 40px;

      border-radius: 10px;

      box-shadow: 0 0 15px rgba(0,0,0,0.1);

      width: 100%;

      max-width: 400px;

    }

form-container h2 {

      text-align: center;

      color: #6F4F37;

      margin-bottom: 20px;

    }

form-group {

      margin-bottom: 15px;

    }

label {

      display: block;

      margin-bottom: 6px;

      color: #333;

    }

input, select {

      width: 100%;

      padding: 10px;

      border: 1px solid #ccc;

      border-radius: 6px;

    }

    input[type="checkbox"] {

      width: auto;

      margin-right: 8px;

    }terms {

      display: flex;

      align-items: center;

      font-size: 14px;

      margin-top: 10px;

    }

  button {

      width: 100%;

      padding: 10px;

      background-color: #6DBF3A;

      color: #fff;

      border: none;

      border-radius: 6px;

      font-size: 16px;

      cursor: pointer;

      margin-top: 15px;

    }

button:hover {

      background-color: #5aa732;

    }  .login-link {

      text-align: center;

      margin-top: 15px;

      font-size: 14px;

    } .login-link a {

      color: #6F4F37;

      text-decoration: none;

    } .login-link a:hover {

      text-decoration: underline;

    }

  </style>

</head>

<body>

  <div class="form-container">

    <h2>Create Library Account</h2>

    <form action="#" method="post">

      <div class="form-group">

        <label for="name">Full Name:</label>

        <input type="text" id="name" name="name" placeholder="Enter your full name" required />

      </div>

  <div class="form-group">

        <label for="email">Email Address:</label>

        <input type="email" id="email" name="email" placeholder="Enter your email" required />

      </div>

<div class="form-group">

        <label for="role">Select Role:</label>

        <select id="role" name="role" required>

          <option value="">-- Select --</option>

          <option value="student">Student</option>

          <option value="faculty">Faculty</option>

          <option value="admin">Librarian/Admin</option>

        </select>

      </div>

<div class="form-group">

        <label for="password">Create Password:</label>

        <input type="password" id="password" name="password" placeholder="Enter password" required />

      </div>

  <div class="terms">

        <input type="checkbox" id="terms" required />

        <label for="terms">I accept the terms & conditions</label>

      </div>

<button type="submit">Register</button>

    </form>

  <div class="login-link">

      Already have an account? <a href="login.html">Login here</a>

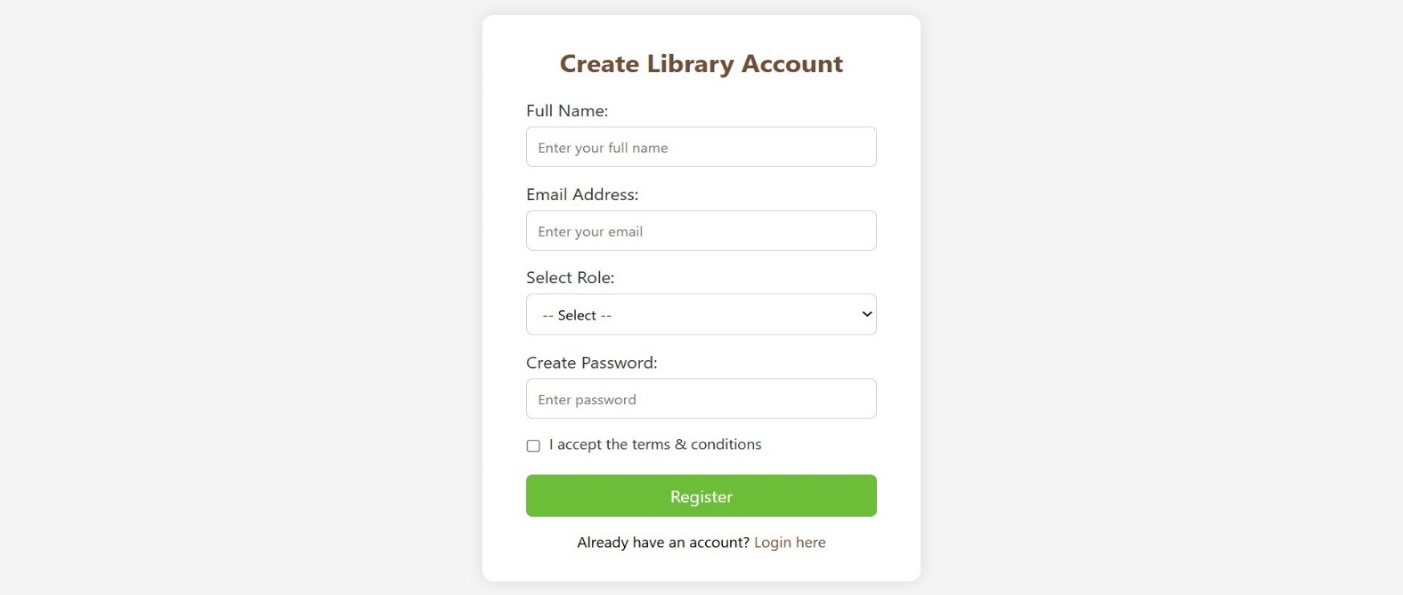
    </div>

  </div>

</body>

</html>

**Output:**

F.registrationpage output: 

**Code:**

G. login page:

code:

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8" />

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

  <title>Login - Library Management</title>

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

  <style>

    body {

      font-family: Arial, sans-serif;

      background: url('images/library-bg.jpg') no-repeat center center fixed;

      background-size: cover;

      margin: 0;

      padding: 0;

      color: white;

    }

    nav {

      background: rgba(0, 0, 0, 0.8);

      padding: 15px;

      text-align: center;

      width: 100%;

      position: fixed;

      top: 0;

      left: 0;

      z-index: 1000;

    }

    nav ul {

      list-style: none;

      padding: 0;

      margin: 0;

      display: flex;

      justify-content: center;

    }

    nav ul li {

      margin: 0 15px;

    }

    nav ul li a {

      color: white;

      text-decoration: none;

      font-size: 18px;

      font-weight: bold;

      padding: 10px 15px;

      background: #27ae60;

      border-radius: 5px;

      transition: background 0.3s;

    } nav ul li a:hover {

      background: #219150;

    }login-container {

      margin: 120px auto;

      max-width: 400px;

      background: rgba(0, 0, 0, 0.7);

      padding: 30px;

      border-radius: 10px;

      box-shadow: 0 0 10px rgba(0, 0, 0, 0.5);

    } h2 {

      text-align: center;

      margin-bottom: 20px;

    }input[type="text"],

    input[type="password"] {

      width: 100%;

      padding: 12px;

      margin: 10px 0;

      border: none;

      border-radius: 5px;

      box-sizing: border-box;

    } .login-btn {

      width: 100%;

      padding: 12px;

      background: #f39c12;

      color: black;

      font-size: 16px;

      border: none;

      border-radius: 5px;

      cursor: pointer;

    } .login-btn:hover {

      background: #e67e22;

    }

  </style>

</head>

<body>

  <nav>

    <ul>

      <li><a href="home.html">Home</a></li>

      <li><a href="books.html">Books</a></li>

      <li><a href="cart.html">Cart</a></li>

      <li><a href="contact.html">Contact</a></li>

      <li><a href="login.html" class="active">Login</a></li>

    </ul>

  </nav>

<div class="login-container">

    <h2>Login</h2>

    <form onsubmit="handleLogin(event)">

      <input type="text" placeholder="Username" required />

      <input type="password" placeholder="Password" required />

      <button type="submit" class="login-btn">Login</button>

    </form>

  </div>

<script>

    function handleLogin(event) {

      event.preventDefault(); // Prevents form from reloading the page

      // You can add authentication logic here if needed

      window.location.href = "home.html"; // Redirect to home

    }

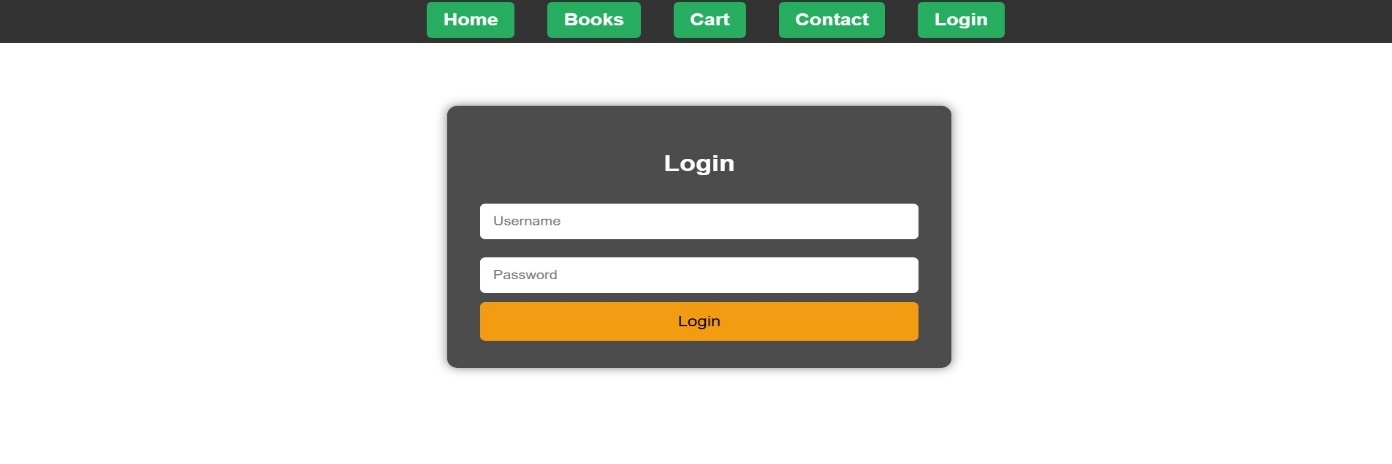
  </script>

</body>

</html>

**Output:**

G. login page  output:



**Conclusion:**

Implementing registration, login, validation, and borrowed books management features using JavaScript is fundamental for any user-centered library system. These features not only enhance usability but also ensure smooth interactions, efficient data handling, and a seamless user journey within the digital library environment.

For a Library Management System, these JavaScript functionalities form the foundation for user authentication and interactive borrowing experiences. Users can securely register and log in, receive immediate validation feedback when filling out forms, and manage their selected books or reservations efficiently. This setup also lays the groundwork for future enhancements such as borrowing history, due date reminders, personalized recommendations, and digital resource access.

JavaScript, therefore, plays a critical role in transforming a static library portal into a dynamic, responsive, and fully functional digital platform—empowering students, faculty, and library administrators alike

**Experiment No.6**

**Problem Statement:**

JavaScript Theory: Persistent Login and Borrowed Book Tracking using Web Storage API

**Introduction**

In modern web applications, delivering a seamless user experience depends on maintaining user session states and persistent data across page reloads or browser restarts. JavaScript’s Web Storage API—comprising localStorage and sessionStorage—provides a lightweight, client-side solution for managing such data effectively.

For a Library Management System, these features greatly enhance usability by enabling persistent login sessions and retaining a user's borrowed or reserved book list, even after they navigate away, refresh the page, or temporarily close the browser.

1. **Persistent Login using localStorage/sessionStorage**

A secure login system allows students, faculty, and library administrators to access their personalized accounts. JavaScript can store authentication state client-side to minimize repetitive login actions.

Storage Options:

* localStorage: Persists data indefinitely unless manually cleared; survives browser closure.
* sessionStorage: Stores data only during the active session (tab/window).

**Implementation Steps:**

* **On successful login, JavaScript saves**:
  + userEmail: to identify the logged-in user.
  + isLoggedIn: boolean flag to confirm session state.
* **On page load or revisit:**
  + JavaScript checks these flags.
  + If present: redirect to the dashboard, catalog, or borrowed books page.
  + If absent: show the login page.
* **On logout:**
  + JavaScript clears stored session values to ensure secure exit.

**Benefits:**

* Reduces login repetition for frequent users.
* Improves user convenience during prolonged study sessions.
* Ideal for prototypes and SPAs where full backend auth isn’t available.

2. **Borrowed Books Tracking using localStorage**

In library systems, users often add books to a “borrowed” or “to-reserve” list before final submission. This list behaves much like a shopping cart in e-commerce, and localStorage can persist it across sessions.

Implementation Steps:

* Each time a user selects, removes, or updates a book:
  + JavaScript serializes the book list (array of book objects) to JSON.
  + Saves it in localStorage.
* On page load:
  + JavaScript checks for existing borrow list.
  + If found: parses and displays it on the UI dynamically.
* The list remains active unless:
  + User logs out.
  + Borrowing is confirmed.
  + Manual list reset is performed.

**Benefits:**

* Prevents data loss from accidental tab closure or refresh.
* Enables users to continue book selection later.
* Facilitates “Reserve Now, Pick Up Later” functionality—even without requiring login.

Use Cases Beyond the Syllabus (Advanced Learning)

These concepts are often beyond the basic academic curriculum but are critical in real-world applications:

* Managing session state with Web Storage API.
* JSON parsing/stringifying for storing complex JavaScript objects.
* Creating stateful SPAs (Single Page Applications) or MPAs without backend support.
* Valuable for real-world capstone projects and portfolio-level applications.

**Summary**

By utilizing JavaScript’s Web Storage API, a Library Management System can:

* Maintain persistent user login sessions.
* Store and retrieve a user’s borrowed book list.
* Provide a seamless, offline-capable experience.
* Simulate a full-fledged borrowing system without backend dependence.

This implementation greatly enhances usability, reliability, and the overall digital experience of a modern library interface.

**Code:**

A. Home page:

code:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Home - Library Management</title>

    <link rel="stylesheet" href="styles.css">

    <style>

        /\* General Styling \*/

        body {

            font-family: 'Poppins', sans-serif;

            background: url('C:\Users\DELL\OneDrive\Desktop\LIBARY MANAGEMANT\images (1).jpeg') no-repeat center center fixed;

            background-size: cover;

            margin: 0;

            padding: 0;

            color: rgb(255, 255, 255);

            position: relative;

        } /\* Background Overlay \*/

        body::before {

            content: "";

            position: absolute;

            top: 0;

            left: 0;

            width: 100%;

            height: 100%;

            background: linear-gradient(to right, rgba(0, 0, 0, 0.8), rgba(0, 0, 0, 0.5));

            z-index: -1;

        } /\* Navigation Bar \*/

        nav {

            background: rgba(0, 0, 0, 0.8);

            padding: 15px;

            text-align: center;

            width: 100%;

            position: fixed;

            top: 0;

            left: 0;

            z-index: 1000;

            box-shadow: 0 4px 10px rgba(0, 0, 0, 0.3);

        }  nav ul {

            list-style: none;

            padding: 0;

            margin: 0;

            display: flex;

            justify-content: center;

        } nav ul li {

            margin: 0 15px;

        }

        nav ul li a {

            color: white;

            text-decoration: none;

            font-size: 18px;

            font-weight: bold;

            padding: 10px 15px;

            position: relative;

            transition: 0.3s ease;

        }

        nav ul li a::after {

            content: "";

            display: block;

            width: 0;

            height: 3px;

            background: #f39c12;

            transition: width 0.3s ease;

            margin: auto;

        }

        nav ul li a:hover::after {

            width: 100%;

        }  /\* Hero Section \*/

        .container {

            text-align: center;

            padding: 120px 20px;

            position: relative;

            z-index: 1;

        }

        .hero-section {

            max-width: 900px;

            margin: 100px auto;

            background: rgba(0, 0, 0, 0.7);

            padding: 30px;

            border-radius: 15px;

            box-shadow: 0 0 20px rgba(0, 0, 0, 0.4);

            animation: fadeIn 1.2s ease-in-out;

        }

        .hero-section h1 {

            font-size: 40px;

            animation: slideIn 1.5s ease-in-out;

        }

        .hero-section p {

            font-size: 18px;

            margin: 20px 0;

        }

        @keyframes fadeIn {

            from { opacity: 0; transform: scale(0.9); }

            to { opacity: 1; transform: scale(1); }

        }

        @keyframes slideIn {

            from { transform: translateY(-30px); opacity: 0; }

            to { transform: translateY(0); opacity: 1; }

        } .btn {

            padding: 12px 20px;

            background: #f39c12;

            color: black;

            font-size: 18px;

            text-decoration: none;

            border-radius: 8px;

            transition: 0.3s ease;

            display: inline-block;

        }

        .btn:hover {

            background: #e67e22;

            transform: scale(1.05);

        }  /\* Feature Section \*/

        .features {

            margin-top: 50px;

            display: flex;

            justify-content: center;

            gap: 30px;

            flex-wrap: wrap;

        }

        .feature-box {

            width: 250px;

            padding: 20px;

            background: rgba(255, 255, 255, 0.9);

            color: black;

            border-radius: 10px;

            text-align: center;

            box-shadow: 0 0 15px rgba(0, 0, 0, 0.3);

            transition: 0.3s ease;

        }

        .feature-box:hover {

            transform: scale(1.05);

        }

        .feature-box h3 {

            margin-bottom: 10px;

        }

  /\* Footer \*/

        .footer {

            background: #222;

            color: white;

            text-align: center;

            padding: 20px;

            margin-top: 50px;

        }

    </style>

</head>

<body>

    <nav>

        <ul>

            <li><a href="home.html" class="active">Home</a></li>

            <li><a href="books.html">Books</a></li>

            <li><a href="cart.html">Cart</a></li>

            <li><a href="contact.html">Contact</a></li>

            <li><a href="login.html">Login</a></li>

        </ul>

    </nav>

    <div class="container">

        <div class="hero-section">

            <h1>📖 Welcome to Our Library</h1>

            <p>Explore a vast collection of books, manage your library, and enhance your reading experience.</p>

            <a href="books.html" class="btn">📚 Browse Books</a>

        </div>

<div class="features">

            <div class="feature-box">

                <h3>📚 Huge Collection</h3>

                <p>Thousands of books available in various genres.</p>

            </div>

            <div class="feature-box">

                <h3>🔍 Easy Search</h3>

                <p>Find your favorite books quickly and efficiently.</p>

            </div>

            <div class="feature-box">

                <h3>🛒 Shopping Cart</h3>

                <p>Add books to your cart and borrow easily.</p>

            </div>

        </div>

    </div>

  <footer class="footer">

        &copy; 2025 Library Management | Follow us on 📘 Facebook | 🐦 Twitter | 📸 Instagram

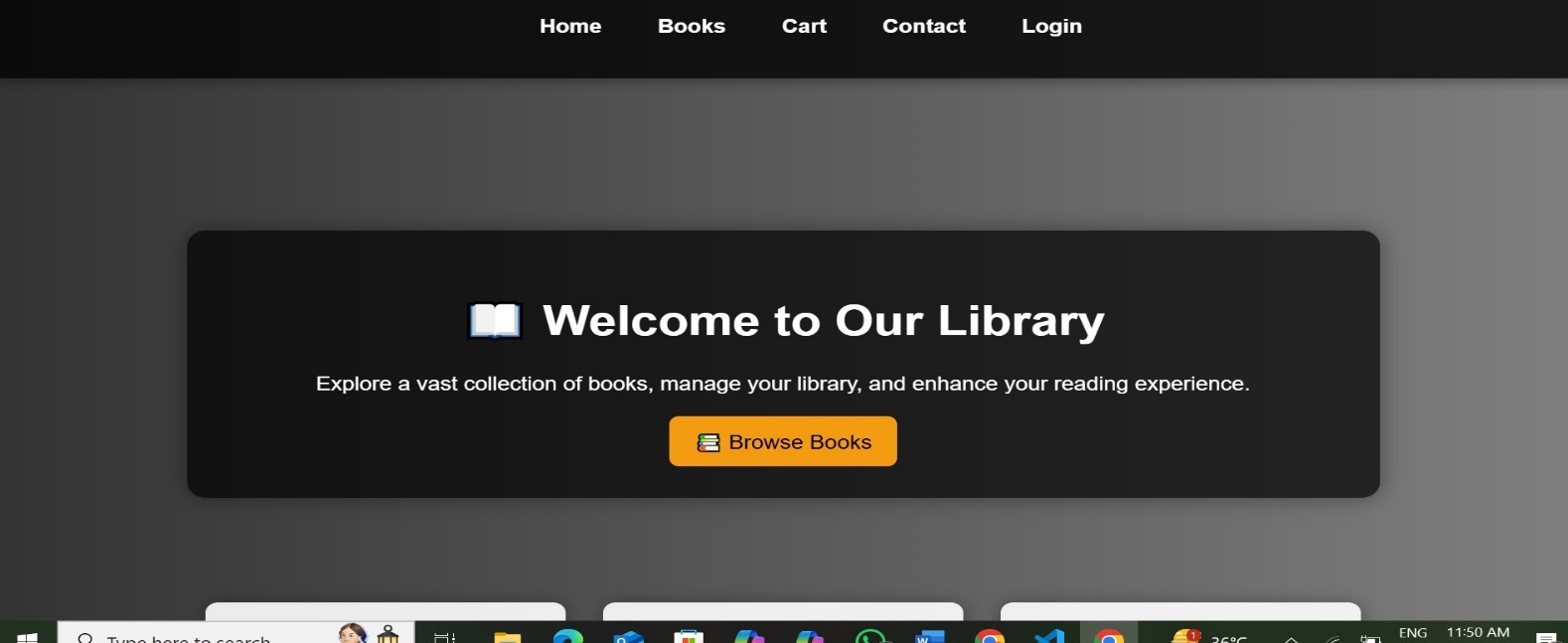
    </footer>

</body>

</html>

**Output:**

1. Index/Home page output:



**Conclusion**

Using JavaScript in combination with the Web Storage API (localStorage/sessionStorage) significantly enhances user experience and functionality in modern web development. For a Library Management System, implementing persistent login and borrowed book tracking ensures users can interact with the system smoothly and without interruption.

By storing authentication states and borrowed book data locally:

* Users remain logged in across multiple sessions
* Borrowed or reserved book lists persist even after page reloads or browser restarts
* The system feels more responsive, personalized, and user-friendly

**Experiment no.7**

**Problem Statement:**

**Develop a PHP Script to Handle User Registration for the Library Management System Website**

**User registration** is a fundamental feature of digital systems like a **Library Management System**, where students, faculty, and staff need personalized access to borrow books, view their activity, or manage digital resources. PHP is a popular server-side language used to handle form submissions, validate user input, interact with a database (like MySQL), and ensure secure data processing.

**The PHP Registration Script Handles the Following:**

1. Collects user input (name, email, password, etc.) from an HTML form
2. Validates the form fields to ensure accuracy and completeness
3. Hashes the password using password\_hash() for security
4. Interacts with a MySQL database to store the user’s information
5. Handles errors gracefully and informs the user
6. Provides confirmation or redirects the user to the login page after successful registration

**B. Implement Error Handling to Notify Users of Any Issues During Registration**

The PHP script checks for common issues such as:

* Empty form fields
* Invalid email format
* Password and confirm password mismatch
* Duplicate email addresses already registered

If any of these conditions occur, the script:

* Stores error messages in an array
* Displays the errors below the form in red text
* Prevents the form from submitting data to the database until all validations are passed

This approach ensures a secure and user-friendly experience, improving form usability and reducing server-side errors.

**C. Provide Feedback to the User Upon Successful Registration**

Upon successful registration:

* The user receives a clear success message (e.g., “Registration successful! Click here to login.”)
* Optionally, the script can **redirect the user to the login page** automatically using header("Location: login.php");

This feedback confirms that the user's data has been securely stored and encourages them to proceed to the next step—logging into their account.

CODE:-

<?php

// Start session (optional, for further use like flash messages)

session\_start();

// Database connection

$servername = "localhost";

$username = "root";

$password = "";

$dbname = "library\_db"; // Use your actual DB name

$conn = new mysqli($servername, $username, $password, $dbname);

// Check DB connection

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

// Initialize error array

$errors = [];

// Form submitted

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

// Fetch and sanitize inputs

$name = trim($\_POST['name']);

$email = trim($\_POST['email']);

$password = $\_POST['password'];

$confirm = $\_POST['confirm\_password'];

// Validate inputs

if (empty($name) || empty($email) || empty($password) || empty($confirm)) {

$errors[] = "All fields are required.";

}

if (!filter\_var($email, FILTER\_VALIDATE\_EMAIL)) {

$errors[] = "Invalid email format.";

} if ($password !== $confirm) {

$errors[] = "Passwords do not match.";

}

// If no errors, process registration

if (empty($errors)) {

// Check if email already exists

$check = $conn->prepare("SELECT id FROM users WHERE email = ?");

$check->bind\_param("s", $email);

$check->execute();

$check->store\_result();

if ($check->num\_rows > 0) {

$errors[] = "This email is already registered.";

} else {

// Hash password and insert

$hashed\_password = password\_hash($password, PASSWORD\_DEFAULT);

$stmt = $conn->prepare("INSERT INTO users (name, email, password) VALUES (?, ?, ?)");

$stmt->bind\_param("sss", $name, $email, $hashed\_password);

if ($stmt->execute()) {

echo "<p>Registration successful! <a href='login.php'>Click here to login</a>.</p>";

} else {

$errors[] = "Registration failed. Please try again.";

}

$stmt->close();

}

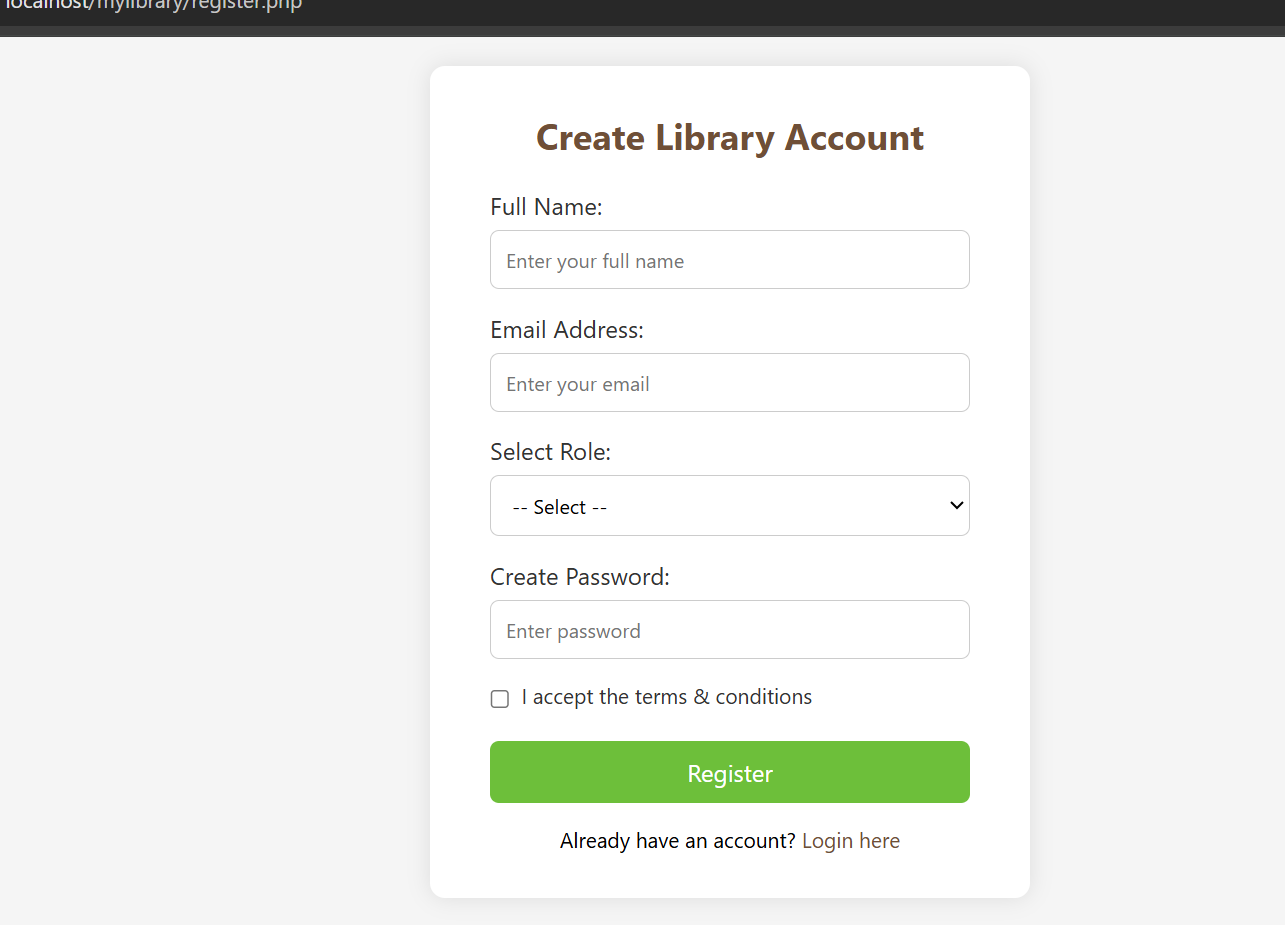
$check->close();

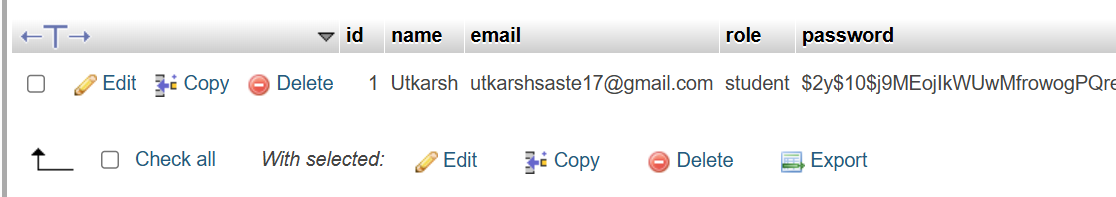
}

}

$conn->close();

?>





**Conclusion:**

Implementing user registration with PHP provides the backbone of user management in a Library Management System. By securely collecting, validating, and storing user data, you enable personalized experiences such as accessing a user dashboard, borrowing books, reserving items, and viewing borrowing history.

This system:

* Promotes user trust by securing sensitive information like passwords
* Ensures data integrity through robust server-side validation
* Enhances user experience by offering real-time feedback and clear error messages during the registration process

Overall, this functionality is essential for building a secure, responsive, and professional digital library platform.

**Experiment 8**

**Problem Statement:**

**Develop a PHP Script to Handle User Login for the Library Management System Website**

User registration is a vital component in web applications, especially in systems like a Library Management System. It enables users such as students, faculty, and administrators to securely access their personalized dashboard, borrow books, and manage their accounts.

**PHP is a robust server-side language widely used for:**

* Handling form submissions
* Validating user inputs
* Interacting with databases (e.g., MySQL)
* Securing sensitive data like passwords

In this Library Management System, the registration form captures user information including full name, email address, and password. Once submitted, PHP validates and stores the input into the database securely after hashing the password.

**B. Implement Error Handling to Notify Users of Any Issues During Registration**

The script checks for common validation issues like:

* Missing fields
* Invalid email formats
* Password mismatch
* Duplicate email registration

**If any of these issues are found:**

* Appropriate error messages are stored and displayed
* The form does not submit to the database
* Users receive immediate feedback to correct errors

**C. Provide Feedback to the User Upon Successful Registration**

Once the registration is successful:

* The script confirms with a success message (e.g., “Registration successful! Login here”)
* Optionally, it can redirect the user to the login page automatically using header("Location: login.php")
* This ensures a clear and smooth user experience

**Core Elements of the PHP Registration Script**

|  |  |
| --- | --- |
| Component | Description |
| Form Handling | Collects data using $\_POST |
| Validation | Ensures required fields are filled, and passwords match |
| Password Hashing | Uses password\_hash() to securely encrypt passwords |
| Database Interaction | Stores the data in a MySQL database using prepared statements |
| Error Handling | Displays user-friendly messages for any form issues or duplication |
| User Feedback | Provides success confirmation or redirection after registration |

CODE:-

<?php

session\_start();

include 'db\_connect.php';

$email = $password = "";

$errors = [];

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

$email = trim($\_POST["email"]);

$password = trim($\_POST["password"]);

// Basic validation

if (empty($email) || !filter\_var($email, FILTER\_VALIDATE\_EMAIL)) {

$errors[] = "Please enter a valid email address.";

}

if (empty($password)) {

$errors[] = "Please enter your password.";

}

// Proceed only if no validation errors

if (empty($errors)) {

$stmt = $conn->prepare("SELECT id, name, email, password FROM users WHERE email = ?");

$stmt->bind\_param("s", $email);

$stmt->execute();

$result = $stmt->get\_result();

if ($result && $result->num\_rows === 1) {

$user = $result->fetch\_assoc();

if (password\_verify($password, $user['password'])) {

// Correct login

$\_SESSION["user\_id"] = $user['id'];

$\_SESSION["user\_name"] = $user['name'];

$\_SESSION["user\_email"] = $user['email'];

echo "<p>Welcome, <strong>" . htmlspecialchars($user['name']) . "</strong>! Redirecting to home...</p>";

header("refresh:2;url=home.php"); // redirect after 2 seconds

exit();

} else {

$errors[] = "Incorrect password.";

}

} else {

$errors[] = "No account found with that email.";

}

$stmt->close();

}

$conn->close();

}

// Display errors if any

foreach ($errors as $error) {

echo "<p style='color:red;'>$error</p>";

}

?>  
Login form:-

<form action="login.php" method="POST">

<h2>Login</h2>

<label>Email:</label><br>

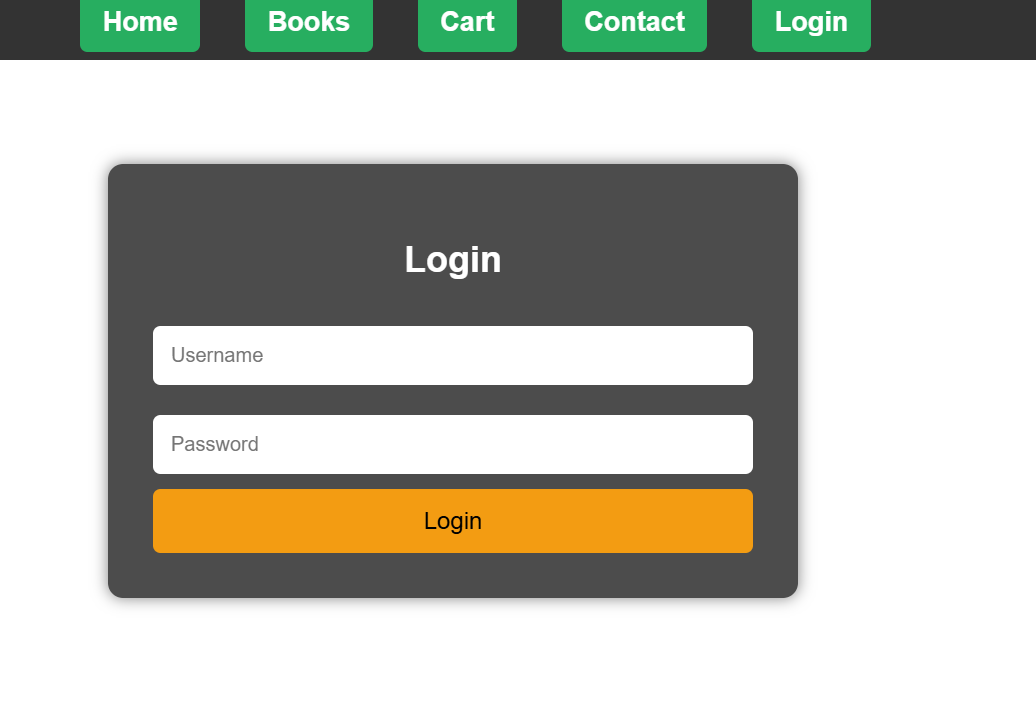
<input type="email" name="email" required><br><br>

<label>Password:</label><br>

<input type="password" name="password" required><br><br>

<input type="submit" value="Login">

</form>  
 output:-

Dashboard:-

<?php

session\_start();

if (!isset($\_SESSION["user\_id"])) {

echo "Access denied. Please <a href='login.html'>login</a>.";

exit();

}

echo "<h2>Welcome back, " . htmlspecialchars($\_SESSION["user\_name"]) . "!</h2>";

echo "<p>You are logged in with email: " . htmlspecialchars($\_SESSION["user\_email"]) . "</p>";

echo "<a href='logout.php'>Logout</a>";

?>

<?php

session\_start();

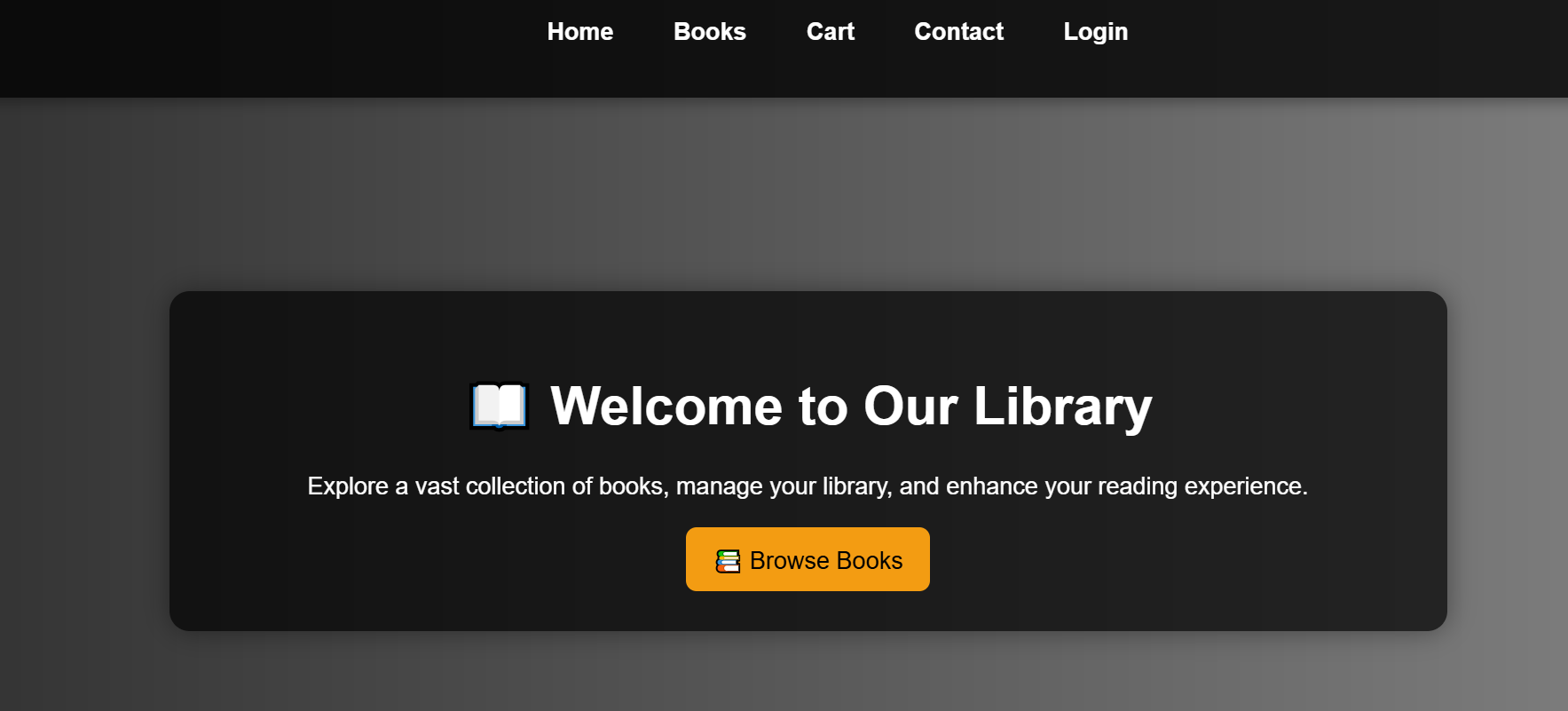
session\_destroy();

header("Location: login.html");

exit();

?>

**Output:-**

****

**Conclusion:**

Implementing a login system with **PHP** ensures a secure and user-friendly experience for your **Library Management System** website. By validating input, securely verifying credentials, and managing PHP sessions:

* You allow users (students, faculty, and staff) to access personalized features such as borrowing history, book reservations, and account settings.
* You prevent unauthorized access to restricted pages and user data.
* You enhance user trust and engagement through a smooth authentication process.

This login system:

* Supports secure authentication using **hashed passwords**
* Provides **real-time feedback** for incorrect login credentials
* Ensures **session persistence** and protects sensitive pages using **login checks and redirects**

Overall, it lays the foundation for secure and efficient digital library operations

**Experiment 9**

**Problem Statement:**

A. Develop a PHP script that allows users to manage their borrowed books cart for a Library Management System. The script should allow users to add books to their cart, view their current borrowed list, and remove books if needed.

B. Develop a PHP script to manage the borrowed books cart for a Library Management System using MySQL. This script should allow users to add books to their borrowing list, view their borrowed cart contents, and remove books from the cart. The cart data should be stored in the MySQL database to allow persistence across sessions.

**Theory: PHP Borrowed Book Cart System**

A borrowed book cart is a key feature of any Library Management System. It acts as a temporary space where users can select and manage the books they wish to borrow. In the context of a library, where some titles may be limited to one or two copies, this functionality becomes essential for efficient resource management.

**Two Types of Borrowed Cart Management Systems in PHP:**

A. Session-Based Borrowing Cart (Without MySQL)

This approach uses PHP sessions to temporarily store borrowing cart data in memory while the user is interacting with the library system. It’s ideal for quick prototypes or small libraries.

**Key Characteristics:**

* Cart data is stored in $\_SESSION
* Data persists only during the browser session
* No need to log in to use the borrow cart
* Borrowed list is cleared once the session ends

**Operations Supported:**

* Add to Cart: Add books by storing Book ID, Title, Author, and Quantity in the session
* View Cart: Display current borrowed items stored in the session
* Remove from Cart: Unset item using Book ID or session index

**Advantages:**

* Simple to implement
* No database complexity

**Limitations:**

* Borrowing list is not persistent after session ends
* Not scalable for systems requiring login or history tracking

**B. Database-Based Borrowing Cart (With MySQL)**

This is a professional and scalable method where borrowed book data is stored in a MySQL database. This makes borrowing records persistent and user-specific, which is essential for a complete library system.

**Key Characteristics:**

* Each user has a unique borrow cart linked via user ID
* Borrowed books are stored in borrow\_cart or borrow\_items tables
* Requires login/session tracking for each user

**Operations Supported:**

* Add to Cart: Insert or update records in the borrow cart database
* View Cart: Query the database for all borrowed books for a user
* Remove from Cart: Delete book from borrow list by item ID or borrow ID

**Advantages:**

* Persistent and user-specific borrowing list
* Accessible across sessions and devices
* Supports audit trails, return tracking, and analytics

**Limitations:**

* Requires database setup and user authentication
* Slightly more complex error handling

CODE:-

CREATE TABLE cart\_items (

id INT AUTO\_INCREMENT PRIMARY KEY,

user\_id INT NOT NULL,

product\_id INT NOT NULL,

product\_name VARCHAR(255),

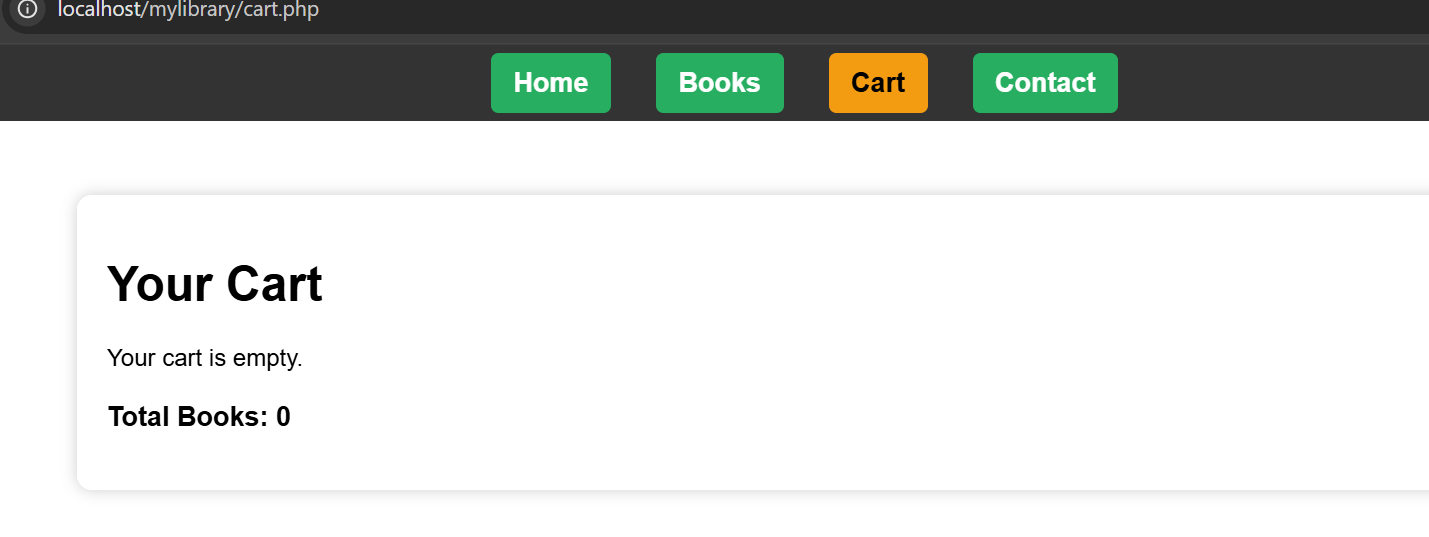
quantity INT DEFAULT 1,

price DECIMAL(10, 2),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

**Output:-**

****

**Conclusion**

A borrowed book cart system, whether session-based or database-driven, is essential for enhancing the user experience and improving service efficiency in your Library Management System.

When using PHP:

* **Session-based carts** offer quick and simple functionality, suitable for guest or short-term users.
* **MySQL-backed carts** provide robust, persistent storage across sessions and devices—ideal for registered users and production-level systems.

For a fully functional and scalable digital library, the **MySQL-based borrowed cart** is highly recommended, as it:

* Improves user experience with persistent borrowed lists
* Enables personalization, borrowing history, and data analytics
* Supports consistent book tracking and reduces conflicts over limited-copy resources

**Experiment 10**

**Problem Statement:**

A. Develop a PHP script to handle the checkout process for users who are ready to complete their purchase. The script should process the cart data and provide feedback to the user upon successful or failed checkout.  
B. Develop a PHP script that processes the checkout process for users who are ready to complete their purchase, integrating the MySQL database for handling user and order information. The script should validate user input, process the cart data, and provide feedback upon successful or failed checkout.

**Theory: PHP Checkout Process**

The checkout process is the final and most crucial step in any **Library Management System**. It converts the selected books into an official borrowing transaction, capturing necessary details such as student ID, issue date, and return due date, and records them into the database for tracking and fines management.

In a library system where some books may be limited in availability, a robust and accurate checkout process ensures that inventory is correctly updated, availability is maintained, and accountability is established.

**Two Approaches to Book Checkout in PHP**

**A. Session-Based Checkout (Without Database Transaction Management)**

In this basic approach:

* All selected book data is temporarily stored in the session ($\_SESSION['cart']).
* On checkout, a simple confirmation message is displayed.
* Ideal for demonstration or prototype systems.

**Workflow:**

1. Retrieve selected books from $\_SESSION.
2. Validate input fields (student ID, name, etc.).
3. Display success or error message.
4. Clear cart after issuing books.

**Advantages:**

* Easy and fast to implement.
* No need for database setup.

**Limitations:**

* Data is not stored permanently.
* Not suitable for real-world systems.
* No transaction records or borrowing history maintained.

**B. MySQL-Based Book Checkout System**

This advanced and production-ready method:

* Stores issued book data in a MySQL database.
* Enables tracking, overdue monitoring, and fine calculation.
* Links issued books to logged-in students.

**Workflow:**

1. Validate that a student is logged in or session is active.
2. Retrieve selected books from session or borrowing\_table.
3. Validate checkout input (student ID, issue date, due date).
4. Insert data into transaction\_table or borrow\_history.
5. Display confirmation or failure message.
6. Clear session cart or update book status.

**Advantages:**

* Persistent and secure.
* Maintains student borrowing history.
* Enables overdue and fine calculation.
* Ensures real-time book availability tracking.

Code:-

MYSQL Code

CREATE TABLE orders (

id INT AUTO\_INCREMENT PRIMARY KEY,

user\_id INT,

customer\_name VARCHAR(255),

customer\_email VARCHAR(255),

customer\_address TEXT,

total DECIMAL(10, 2),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE order\_items (

id INT AUTO\_INCREMENT PRIMARY KEY,

order\_id INT,

product\_id INT,

product\_name VARCHAR(255),

quantity INT,

price DECIMAL(10, 2)

);

Checkout session:-

<?php

session\_start();

if ($\_SERVER['REQUEST\_METHOD'] === 'POST') {

if (!isset($\_SESSION['cart']) || empty($\_SESSION['cart'])) {

echo "Your cart is empty!";

exit;

}

$name = $\_POST['name'] ?? '';

$email = $\_POST['email'] ?? '';

$address = $\_POST['address'] ?? '';

if (empty($name) || empty($email) || empty($address)) {

echo "Please fill in all required fields.";

exit;

}

echo "<h2>Order Summary</h2>";

$total = 0;

foreach ($\_SESSION['cart'] as $item) {

echo "{$item['name']} - Qty: {$item['quantity']} - ₹{$item['price']} <br>";

$total += $item['quantity'] \* $item['price'];

}

echo "<p>Total: ₹$total</p>";

echo "<p>Thank you, $name! Your order has been placed.</p>";

// Clear the cart

unset($\_SESSION['cart']);

} else {

echo "Invalid request method.";

}

?>

MySQL-Based PHP Checkout Script:-

<?php

session\_start();

$conn = new mysqli('localhost', 'root', '', 'gaming\_store');

if ($conn->connect\_error) {

die("Database connection failed: " . $conn->connect\_error);

}

if ($\_SERVER['REQUEST\_METHOD'] === 'POST') {

if (!isset($\_SESSION['cart']) || empty($\_SESSION['cart'])) {

echo "Your cart is empty.";

exit;

}

$name = $\_POST['name'] ?? '';

$email = $\_POST['email'] ?? '';

$address = $\_POST['address'] ?? '';

$user\_id = $\_SESSION['user\_id'] ?? 0;

if (empty($name) || empty($email) || empty($address)) {

echo "All fields are required.";

exit;

}

$total = 0;

foreach ($\_SESSION['cart'] as $item) {

$total += $item['quantity'] \* $item['price'];

}

$stmt = $conn->prepare("INSERT INTO orders (user\_id, customer\_name, customer\_email, customer\_address, total) VALUES (?, ?, ?, ?, ?)");

$stmt->bind\_param("isssd", $user\_id, $name, $email, $address, $total);

if ($stmt->execute()) {

$order\_id = $stmt->insert\_id;

$itemStmt = $conn->prepare("INSERT INTO order\_items (order\_id, product\_id, product\_name, quantity, price) VALUES (?, ?, ?, ?, ?)");

foreach ($\_SESSION['cart'] as $item) {

$itemStmt->bind\_param("iisid", $order\_id, $item['id'], $item['name'], $item['quantity'], $item['price']);

$itemStmt->execute();

}

echo "<h2>Checkout Successful</h2>";

echo "Thank you, <strong>$name</strong>. Your order ID is <strong>$order\_id</strong>.<br>Total: ₹$total";

unset($\_SESSION['cart']);

} else {

echo "Checkout failed. Please try again.";

}

$stmt->close();

$conn->close();

} else {

echo "Invalid request.";

}

?>

Output:

**Conclusion**

The book checkout process is the most critical component of a Library Management System it translates user intent into action by recording official borrowing transactions.

Use Case Importance for Library Management:

* Library systems often have limited copies of books, making accurate, real-time tracking essential.
* Persistent storage with MySQL ensures no data loss and allows long-term tracking of issued books.
* A session-based approach is useful during early development stages or in guest testing environments.

Session-Based Book Checkout Summary:

**Simple and quick to implement:**

* Suitable for demonstrations or prototypes.
* Not recommended for long-term use as data is lost after the session ends.
* No historical records are maintained.

**MySQL-Based Book Checkout Summary:**

* Scalable and production-ready.
* Stores borrowing history, which can be used for reporting, fines, or analytics.
* Enables features such as per-user book limits, fine tracking, due date alerts, and book return management.
* Ideal for a complete, user-specific library system.