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

Mr. Paras Vijay Waghmare

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**	
	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**	
	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 **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** **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**	
	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 Book Search and Shopping, 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 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. Product/Service page

d. Testimonial/review page

e. Support page

f. Starter blog posts

11. Create and collect design elements

12. These design elements define your brand personality and help customers 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 second-hand gaming console store website by planning its layout, content, and visual elements, ensuring it meets user needs and effectively represents the brand.

<u>Theory</u>:

Project Design and Plan for Book Search and Shopping

1. Brief Information about the Project

The project is to create a **user-friendly and visually appealing website** for an **online bookstore** specializing in both new and second-hand books. The site is designed to attract book lovers by showcasing a curated collection of books across various genres, providing a seamless shopping experience, and highlighting key features such as **customer reviews**, **bestseller sections**, and an integrated **contact/support platform**.

2. Goals and Deliverables

Goals

- Develop an engaging and functional website for a book search and shopping.
- Showcase the store's story, products, customer reviews, and contact details.
- Enable users to register, log in, and personalize their experience.
- Create a responsive website that works across all devices.

Deliverables

- Website Pages:
  - Home Page
  - About Page
  - Products/Services Page
  - Testimonials Page
  - Contact Page
  - Login Page
  - Registration Page
  - Starter blog posts or placeholder for future blogs (optional).
- Core Features:
  - Header and footer with consistent navigation.
  - Functional login and registration system.
  - Responsive design adaptable to mobile, tablet, and desktop.
  - Professional design with appropriate use of colors, fonts, and images.

3. Finalize the modules of the project

The second-hand gaming console store website will have a modular structure that ensures easy navigation, usability, and maintenance. Each module corresponds to a distinct functionality or page, helping in modular development and integration. Below is a detailed description of the finalized modules:

Website Modules

1. Home Page Module

- Description:
  The main page of the website welcomes users and highlights essential features. It sets the tone for the user experience.
- Features:
  - Hero section with the tagline and call-to-action buttons (e.g., "Purchase Now" or "Explore Menu").
  - Overview of featured products or promotions.
  - Navigation catalog linking to all website sections (e.g., About, Products, Testimonials, Contact, Login).
  - Footer with contact details, social links, and other information.

2. About Page Module

- Description:
  Offers visitors a glimpse of the second-hand gaming console store's story, mission, and values.
- Features:
  - Introduction to the second-hand gaming console store's history and portable consolesm.
  - Showcase the brand's principles like quality, sustainability, and customer service.
  - Engaging visuals to reflect the second-hand gaming console store's vibe.

3. Products/Services Page Module

- Description:
  Displays the store's product offerings in a user-friendly way.
- Features:
  - Categorized catalog (e.g., Console, Snacks, Handhelds).
  - Images and details for each item, including price and description.
  - Option for filtering or searching products (future enhancement).

4. Testimonials Page Module

- Description:
  Shares positive customer reviews and builds trust with new visitors.
- Features:
  - Slider or grid layout showcasing testimonials.
  - Include a field or section for customers to submit their reviews (optional).

5. Contact Page Module

- Description:
  Enables visitors to get in touch with the second-hand gaming console store.
- Features:
  - A form for user inquiries (fields: Name, Email, Subject, Message).

- Embedded map for the physical store location.
- Display contact details like phone number and working hours.

6. Login Page Module

- Description:
  Provides authentication functionality for returning users.
- Features:
  - Login form with fields for Email and Password.
  - "Forgot Password?" link.
  - Redirection to the registration page for new users.

7. Registration Page Module

- Description:
  Allows new users to sign up for an account.
- Features:
  - Registration form with fields for Name, Email, and Password creation.
  - Terms and conditions acceptance checkbox.
  - Submit button to create an account.

8. Footer Module

- Description:
  A common footer displayed across all pages.
- Features:
  - Links to Privacy Policy, Terms of Service, and social media pages.
  - Address and basic contact info.

9.Target Audience – Bookstore & Online Shopping Website

- **The website for the online bookstore is designed to cater to a broad spectrum of visitors, each with specific reading interests and shopping behaviors. Understanding the audience ensures that the design, content, and features effectively serve their needs. Below is a breakdown of the target audience:**
-
- **a. Book Enthusiasts**
- **Characteristics:**
- **Regular readers and collectors of books across genres.**
- **Interested in new releases, author-signed copies, and limited editions.**
- **Needs:**
- **Detailed book descriptions with high-quality cover images, author bios, and pricing.**
- **Access to exclusive discounts, loyalty rewards, or pre-order options for new titles.**

- ---
- **b. Professionals and Remote Workers**
- **Characteristics:**
- **Seek productivity and self-development books or comfortable reading materials for breaks.**
- **May prefer to order books online or reserve in-store pickup.**
- **Needs:**
- **A responsive support page with store hours, shipping timelines, and FAQs.**
- **Online ordering with flexible delivery or pickup options.**
- ---
- **c. Students**
- **Characteristics:**
- **Typically younger audience searching for affordable textbooks, study materials, and casual reads.**
- **Needs:**
- **Student discounts or bundled book offers.**
- **Easy-to-use catalog and filters to quickly find academic or genre-based books.**
- ---
- **d. Health & Wellness Readers**
- **Characteristics:**
- **Focused on content related to mental health, fitness, nutrition, and mindfulness.**
- **Needs:**
- **Well-organized categories like "Self-Help", "Wellness", "Diet & Fitness".**
- **Highlights of bestsellers and editor's picks in health genres.**
- ---
- **e. Tourists and Gift Shoppers**
- **Characteristics:**
- **Looking for unique local books, souvenirs, or literary gifts.**
- **Needs:**
- **Map integration to find local branches or partner stores.**
- **Clear section for bestsellers, gift cards, and regional/rare book collections.**
- ---
- **f. New Visitors**
- **Characteristics:**
- **First-time users unfamiliar with the bookstore's brand or offerings.**

- **Needs:**
- **A compelling "About Us" page that introduces the mission, values, and story of the store.**
- **A clean, trustworthy interface that encourages browsing and builds credibility.**
-

- **g. Online Shoppers**
- **Characteristics:**
- **Prefer browsing and purchasing books online, from physical novels to e-books and audiobooks.**
- **Needs:**
- **Seamless e-commerce experience with category filters, search functionality, and user reviews.**
- **Secure login and registration system that supports saved carts, wishlists, and past purchases.**

Website Features Mapped to Audience Needs:

Audience Segment	Key Features Needed
Console Enthusiasts	Menu page with detailed descriptions and Books.
Professionals/Remote Workers	Online purchaseing and pickup, clear navigation to amenities, contact page with location and hours.
Students	Discounts, loyalty programs, or group offers listed prominently.
Health-Conscious Customers	Categorized catalog with nutritional facts and Book filters.
Tourists/Travelers	Geolocation features, unique content promoting local specialties.
New Users	Intuitive UI/UX design with clear site navigation and testimonials to build credibility.
Online Storepers	Secure login and product pages with clear categories for gaming consoles, equipment, or console subscriptions.

Why Understanding the Audience is Important

- Helps in creating engaging and relevant content tailored to users' preferences.
- Enhances the user experience (UX) by addressing specific pain points and ensuring seamless navigation.
- Builds brand trust and attracts loyal customers who resonate with the second-hand gaming console store's story and mission.

- Leads to targeted marketing campaigns, such as student promotions, subscription offers for enthusiasts, or health-focused messaging.

5. Describe pain points & the ideal experience (On the basis of existing systems)

1. Identifying Pain Points of Existing Systems

a. Pain Point: Poor Navigation and Cluttered Interface

- Issue: Many second-hand gaming console store websites have complicated or cluttered designs that make it hard for users to find what they are looking for.
- Impact: Users often leave the site due to frustration or lack of usability.

b. Pain Point: Limited Online Purchaseing Functionality

- Issue: Existing systems often do not provide easy-to-use online purchaseing features, resulting in lower conversion rates and fewer sales.
- Impact: Loss of potential customers who prefer the convenience of online purchases.

c. Pain Point: Lack of Mobile Optimization

- Issue: Non-responsive designs lead to poor mobile user experience.
- Impact: Customers using smartphones face issues navigating the site, viewing products, or purchaseing items.

d. Pain Point: Insufficient Product Information

- Issue: Customers do not get enough details about catalog items, including ingredients, dietary considerations, and prices.
- Impact: Potential customers abandon their search due to incomplete information.

e. Pain Point: Weak Engagement Strategies

- Issue: Existing websites lack features like loyalty programs, student discounts, or engaging content like blogs.
- Impact: Missed opportunities for creating brand loyalty and retaining customers.

f. Pain Point: Inefficient Contact and Location Details

- Issue: Many websites fail to prominently display contact and location information, making it difficult for customers to find or connect with the second-hand gaming console store.
- Impact: Customers waste time searching and may opt for competitors insportable consolesd.

g. Pain Point: No Personalization Options

- Issue: The lack of personalized user experiences or features like accounts, favorite purchases, or personalized recommendations.
- Impact: Users feel the service is impersonal, leading to decreased satisfaction.

2. Crafting the Ideal Experience

To address these pain points, the website design and functionality should create a user-friendly, visually appealing, and highly interactive experience.

a. Intuitive Navigation and Clean Design

- Use a clear and consistent layout with a sticky navigation bar.
- Include links to all key pages (Home, About, Menu, Testimonials, Contact, Login/Sign Up).

b. Seamless Online Purchaseing

- Implement a robust e-commerce system allowing customers to browse products, add items to a cart, and complete purchases effortlessly.
- Provide features like "Purchase Now" buttons on the homepage and catalog pages.

c. Mobile-Responsive Design

- Design with a mobile-first approach, ensuring compatibility across devices.
- Use flexible grids, touch-friendly elements, and optimized performance for fast loading times.

d. Comprehensive Product Information

- Include high-quality images, item descriptions, ingredients, prices, and allergy/dietary labels (e.g., gluten-free or vegan).
- Create filters for health-conscious customers, like "Low Calorie" or "Vegan Options."

e. Customer Engagement Features

- Introduce loyalty programs with a points system visible after login.
- Offer a blog with content like console gaming tips, health benefits, or store news.
- Highlight customer reviews and testimonials on a dedicated page.

f. Easy Access to Contact and Location

- Include a contact page with a simple inquiry form, phone number, and email.
- Display an embedded map for the store’s physical location on the homepage or contact page.

g. Personalization

- Allow users to create accounts for saving their favorite items or past purchases.
- Use a welcome message with the customer’s name after login.
- Send personalized offers via email for registered users.

3. The Ideal User Journey

Step 1: Visiting the Website

- Users arrive at a welcoming homepage with clear navigation to different sections.

Step 2: Browsing the Menu

- Users navigate to the products/services page, view clear catalogs, and filter items based on preferences.

Step 3: Placing an Purchase

- Users can seamlessly add items to their cart and complete a purchase with minimal clicks.

Step 4: Finding Location or Contacting Support

- Users easily locate contact and location details for in-store visits or inquiries.

Step 5: Engaging with Content

- Users read blogs or testimonials for a deeper connection with the brand.

Step 6: Creating Loyalty

- Registered users receive personalized promotions or gain points through purchases.

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."

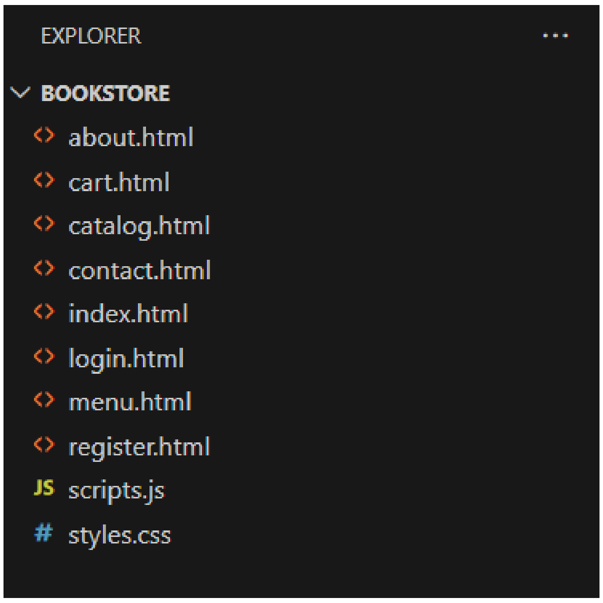
5. Expected Impact of Visual Direction

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

```
book-store-website/
├── css/
│   └── styles.css
├── html/
│   ├── index.html
│   ├── books.html
│   ├── about.html
│   ├── contact.html
│   ├── cart.html
│   ├── register.html
│   └── login.html
├── js/
│   └── scripts.js
├── php/
```

```
|   ├── register.php
|   ├── login.php
|   ├── cart.php
|   ├── checkout.php
├── images/
├── database/
|   └── bookstore.sql
```



```
EXPLORER ...
∨ BOOKSTORE
  about.html
  cart.html
  catalog.html
  contact.html
  index.html
  login.html
  menu.html
  register.html
  scripts.js
  styles.css
```

console store:

1. Home Page

Purpose:

- **Welcome visitors.**
- **Highlight the bookstore's offerings and features. Content Plan:**
- **Header:**
  - **Logo on the left.**
  - **Navigation catalog: Home, About, Books, Testimonials, Contact.**
  - **Login/Sign-Up button on the top right.**
- **Hero Section:**
  - **A high-quality banner image (bookshelves, books on display).**
  - **Tagline: "Discover Your Next Great Read!"**
  - **CTA button: "Browse Our Collection".**

- **Introduction Section:**
    - **Brief about the bookstore (one or two sentences).**
    - **CTA: "Learn More About Us" linking to the About page.**
- **Special Offers Section:**
    - **Carousel or grid of featured books or current offers.**
    - **Text: "Seasonal Sale: Up to 30% off Bestsellers!"**
- **Footer:**
    - **Quick links, social media links, contact information.**

2. About Page

Purpose:

- **Share the story, vision, and people behind the bookstore. Content Plan:**
- **Header: (same as the home page).**
- **About Us Section:**
    - **A short introduction to the bookstore's history (e.g., when and why it was founded).**
    - **Emphasis on values such as passion for books, community, and knowledge.**
- **Meet the Team Section:**
    - **Photos and short bios of the bookstore's team members.**
- **Special Features Section:**
    - **"Why Choose Us?"**
    - **Highlight unique selling points (curated book selection, local author spotlights, etc.).**
- **Footer: (same as the home page).**

3. Books Page

Purpose:

- **Showcase the catalog and books available. Content Plan:**
- **Header: (same as the home page).**
- **Category Sections:**
    - **Categories like Fiction, Non-fiction, New Arrivals, Bestsellers, and more.**
    - **Each category features product images, names, descriptions, and prices.**
- **Highlight Section:**

- **“Top Picks” or “Customer Favorites.”**

- **CTA Section:**
    - **Button: “Buy Now” linking to the Login or Registration page.**
- **Footer: (same as the home page).**

4. Testimonials Page

Purpose:

- **Build trust by showcasing feedback from happy customers. Content Plan:**
- **Header: (same as the home page).**
- **Customer Feedback Section:**
    - **Quotes or testimonials from existing customers.**
    - **Option to display reviews from external sources like Google or Yelp.**
    - **Use star ratings for visual appeal.**
- **Submit a Testimonial:**
    - **Simple form to allow visitors to submit reviews.**
- **Footer: (same as the home page).**

5. Contact Page

Purpose:

- **Allow customers to reach out easily for inquiries, reservations, or feedback. Content Plan:**
- **Header: (same as the home page).**
- **Contact Form:**
    - **Name, Email, Subject, Message.**
    - **Submit button with form validation.**
- **Location Section:**
    - **Embedded Google Map showing the bookstore’s location.**
- **Operating Hours Section:**
    - **Business hours listed clearly.**
- **Footer: (same as the home page).**

6. Login Page

Purpose:

- **Enable existing users to log in to their accounts. Content Plan:**
- **Form:**
  - **Email and Password fields.**
  - **Submit button.**
- **Forgot Password Link:**
  - **Redirects to password recovery.**
- **CTA:**
  - **Link to the Registration page: "Don't have an account? Sign Up Now!"**

7. Registration Page

Purpose:

- **Allow New users to register for an account. Content Plan:**
- **Form Fields:**
  - **Full Name, Email, Password and Confirm Password.**
- **Form Validation:**
  - **Password requirements.**
- **Submit Button:**
  - **Validates data and submits.**
- **Footer: (same as the home page).**

Content, Images & Layout Ideas for Bookstore Website

1. **Home Page:**
   - **Layout Ideas:**
     - **Header Section: Fixed navigation bar with logo on the left and catalog items centered.**
     - **Hero Section: Full-width background image featuring books, with a tagline overlay.**
     - **Featured Section: Cards or tiles showcasing featured books or seasonal offers.**
   - **Content Ideas:**
     - **Welcome message highlighting the bookstore's uniqueness.**
     - **Announcements for specials or promotions.**
   - **Image Ideas:**
     - **A banner image with bookshelves or a cozy reading space.**

    - **Close-up shots of books or book stacks.**

2. **About Page:**
   - **Layout Ideas:**
     - **Story Section: Timeline or column layout detailing the history of the bookstore.**
     - **Team Section: Grid layout with team member bios and photos.**
   - **Content Ideas:**
     - **Mission statement, history, and values.**
     - **Focus on community engagement and passion for books.**
   - **Image Ideas:**
     - **Pictures of the founders and team members.**
     - **Community events, bookstore interior shots.**
3. **Books Page:**
   - **Layout Ideas:**
     - **Category Sections: Display books in different categories (fiction, non-fiction, new releases).**
     - **Highlight Section: Show bestselling or seasonal books.**
   - **Content Ideas:**
     - **Detailed descriptions of books, including author info.**
     - **Add promotions for specific genres.**
   - **Image Ideas:**
     - **High-quality images of book covers, reading spots, and cozy corner settings.**
4. **Testimonials Page:**
   - **Layout Ideas:**
     - **Use a carousel or grid layout for customer reviews.**
     - **Display text along with star ratings.**
   - **Content Ideas:**
     - **Reviews highlighting the bookstore's atmosphere, book quality, and customer service.**
   - **Image Ideas:**
     - **Happy customers browsing or reading in the store.**
5. **Contact Page:**
   - **Layout Ideas:**

    - **Simple contact form with essential fields.**
    - **Embed Google Maps for easy navigation.**
  - **Content Ideas:**
    - **A friendly invitation to reach out with inquiries or book requests.**
  - **Image Ideas:**
    - **Storefront photo or illustrations for contact options.**

Visual Design Ideas for All Pages:

- **Colors:**
  - **Soft, welcoming tones: warm greys, creams, soft blues.**
  - **Highlight color: teal, mustard, or gold for buttons.**
- **Fonts & Typography:**
  - **Elegant serif fonts for headers (e.g., Playfair Display).**
  - **Clean sans-serif for body text (e.g., Lato or Open Sans).**
- **Logo:**
  - **A book or stack of books as part of the logo.**
- **Images:**
  - **Use high-quality, realistic visuals with natural lighting, featuring books, reading spaces, or cozy book corners.**

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 second-hand gaming console store's story, mission, and values.
- Brief history of the business: "Founded in 2025 with a passion for gaming and sustainability..."

Meet the Team:

- Grid layout featuring team members with their names, photos, and brief bios, focusing on technicians, managers, and key staff.

Our Promise:

- Information about how the consoles are sourced, refurbished, and quality-checked.
- Bullet points or icons showcasing eco-friendly practices like console recycling, local sourcing, etc.

Location Section:

- List of store locations with Google Maps integration.
- Hours of operation.

3. Product/Service Page

Console Categories:

- **Categories:** Showcase different types of consoles, such as "Hot Deals," "Accessories," and "Premium Consoles."
- Each product should include an image, a short description (features, conditions), and price.
- "Add to Cart" or "Buy Now" button linking to the purchasing system.

Popular Items & Limited-Time Specials:

- Carousel or featured box showcasing limited-time offers or special deals.

Purchase Online:

- Provide options for mobile or desktop purchasing platforms. Include details on delivery and pick-up options.

Footer (same as Home Page):

- Quick links, social media icons, store locations.

4. Testimonials/Review Page

Customer Reviews:

- A carousel or grid of reviews, each showing a star rating, testimonial, and customer name.
- A “Submit Your Review” button for customers to submit feedback.

Featured Reviews:

- Pull reviews from platforms like Yelp, Facebook, and Google for additional credibility.

Reviewing Process Section:

- A brief explanation of how reviews are managed and shared.

5. Contact Page

Contact Form:

- Name, email, and message fields for inquiries.
- “Submit” button.

Social Media & Address Section:

- Social media icons linking to the store’s Facebook, Instagram, Twitter.
- Full address, phone number, and email.

Interactive Map:

- Google Maps integration to guide customers to the store location.

Support Information:

- Contact details for customer support and FAQs.

6. Starter Blog Posts

Blog Categories:

- **Console Knowledge:** Articles like “How to Choose the Right Second-Hand Console” or “What to Look for in Refurbished Gaming Consoles.”
- **Behind the Scenes:** Features on console refurbishing processes, employee spotlights, or console culture.

- **Sustainability Efforts:** Articles on how the store contributes to sustainability by promoting second-hand gaming.
- **Community Engagement:** Stories about the store's involvement in gaming events or charity support.

7. Login & Registration Pages

Login Page:

- Username/email and password fields.
- "Forgot password?" link.

Registration Page:

- Fields to sign up: Name, email, password.
- Option to subscribe to a newsletter or loyalty program.

8. Overall Website Structure Map

- **Home Page:** Introductory page with links to featured products, testimonials, and social media.
- **About Page:** Overview of the store, team, and values.
- **Product/Service Page:** Showcase products with purchase options.
- **Testimonials/Review Page:** User feedback and submission form.
- **Contact Page:** Contact form and location details.
- **Blog Section:** Articles on gaming, sustainability, and community.
- **Login/Registration Page:** For user accounts and updates.

9. Design Elements

Colors:

- **Primary Colors:** Console Brown (#6F4F37), Beige (#D8CAB8), Cream (#F1E0C6).
- **Accent Colors:** Espresso Black (#2B1B1D), Rich Green (#6DBF3A).

Fonts and Typography:

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

Logo:

- A simple logo with a visual element related to gaming consoles (e.g., controllers, gaming icons).

Imagery and Photos:

- High-quality images of gaming consoles, product close-ups, and the store ambiance.

- Lifestyle shots showing customers enjoying games in the store.

Interactive Elements:

- **Buttons:** Rich green or espresso black for CTA buttons.
- **Icons:** Simple, clean icons representing various site sections like catalog, locations, and store.

Psychological Impact: These colors communicate warmth, comfort, and natural, high-quality ingredients—making it a space people want to return to. Green accents will also create a fresh, eco-friendly atmosphere.

2. Fonts and Typography

The typography should convey a professional yet cozy feel, matching your brand's personality.

- Heading Font:
  - Playfair Display (serif) or Lora: These elegant fonts have an old-fashioned charm, which works well for headings and subheadings on the homepage and catalog pages. It represents traditional console culture with a modern twist.
- Body Font:
  - Open Sans or Roboto (sans-serif): Clean, modern, and highly readable. The body text needs to be easy on the eyes since customers will spend time reading product descriptions or information about the second-hand gaming console store. This font should be used for paragraphs, blogs, and catalog text.
- Font Weights:
  - Use bold or semi-bold weight for headings to create visual hierarchy, and regular font weights for text to ensure ease of reading.

Impact: The mix of serif and sans-serif fonts maintains a balance between tradition and modernity, perfect for a second-hand gaming console store with a warm, upscale yet modern experience.

3. Logo

Your logo represents the visual identity of your second-hand gaming console store and sets the tone for your brand's story.

- Logo Design: The logo should be simple but memorable, combining visual elements that represent console. Consider using stylized console devices, console gaming consoles, or sportable consolesm swirls. These visuals should clearly associate the logo with the essence of the store.
- Color Palette for the Logo: Use the primary colors like console brown and cappuccino beige, along with a touch of espresso black for contrast. If your second-hand gaming console store values organic ingredients, incorporating a bit of green could reinforce the sustainability aspect.

Logo Usage:

- The logo should be placed prominently at the top of each page in the header.
- Ensure its scalability for use on print material, social media, packaging, and within the header of your website.

Impact: The logo serves as the face of your brand, instantly giving customers a sense of the quality and warmth they can expect when visiting your physical or digital space.

4. Imagery and Photos

Imagery on the second-hand gaming console store website has the ability to build a stronger emotional connection by showcasing the console experience.

- Product Photography: High-quality images of console consoles, pastries, and desserts should dominate the site. Think close-ups of frothy cappuccinos, gaming consoles being ground, sportable consolesming devices of console, or beautiful PlayStation art.
    - For the catalog page, show clean, professional shots of the products with descriptions.
    - For the about page, images of the interior of your second-hand gaming console store, people enjoying their console, or portable consolesm photos add authenticity and a sense of community.
- Ambiance Photography: Show the cozy second-hand gaming console store setting with soft lighting, wooden tables, and greenery. These images should showcase the atmosphere visitors will experience in person. Consider using candid photos of customers enjoying console together or a technician preparing a console.
- Lifestyle Photography: In addition to product-specific images, showing a lifestyle—people studying, working, or socializing in your store can be powerful. This reinforces the idea that your store is a place to gather, relax, and socialize.

Impact: High-quality, authentic photos will create a warm, welcoming product presentation, making the site feel as inviting as the physical store itself. They offer a visual sense of what it feels like to enjoy a device of console in your space.

5. Interactive Elements and Buttons

To make sure that the design is functional, interactive elements must be seamlessly integrated, improving the overall experience while keeping in line with your brand's identity.

- Navigation Buttons: Ensure that buttons like “Purchase Now,” “Book a Table,” and “Join Our Newsletter” are easy to see. Use accent colors like rich green or espresso black to make CTAs stand out without being overwhelming.
    - Use hover effects to indicate interactiveness (such as a light shadow or background color change).
- Icons: To enhance the user experience, icons should represent different site sections (like a console device for the catalog, a pin for locations, or a heart for the storeping cart). Simple, clean icons that match your brand colors will guide the customer through the site intuitively.
    - Use lightbulb iconography for new ideas or specials.

<u>Conclusion</u>:

The **Bookstore Management Website** project focuses on developing a functional, visually appealing, and user-friendly platform for a bookstore. The website includes essential pages such as Home, About, Products, Testimonials, Contact, and Login/Registration. The design emphasizes intuitive navigation and engaging content, with harmonious colors, typography, and images that reinforce the bookstore's brand identity. By addressing user needs and structuring the content effectively, the project enhances the online presence of the bookstore and provides a seamless shopping experience for customers.

Experiment No.2

Problem Statement:

- Create a detailed home page for the coffee shop website.
- Create a detailed menu/product page for the coffee shop website, listing all available items categorized appropriately.
- Create a cart page that allows customers to review and manage the items they wish to purchase before proceeding to checkout.
- Create an about us page that provides detailed information about the coffee shop's history, mission, and team.
- Create a contact page that allows customers to easily get in touch with the coffee shop through a form.
- Design and implement admin/user registration form for the coffee shop website.
- Design and implement admin/user login form for the coffee shop website.

Objective:

To create a Book Search and Shopping webpage using HTML.

Theory:

Introduction

In today's digital economy, e-commerce platforms are essential for buying and selling products efficiently. This project focuses on creating a responsive and functional website for a **second-hand gaming console store**. The platform caters to gamers looking for affordable alternatives to brand-new devices, promoting **sustainability** and **cost-effectiveness**.

The website integrates front-end and back-end components to deliver a seamless user experience. Features like **product listings, user authentication, cart management**, and **contact forms** are implemented using HTML, CSS, and optionally JavaScript or server-side scripting in later phases.

1. **HomePage**
   The home page serves as the landing page and provides a snapshot of the store's offerings. It typically includes:

- **A hero section with promotions or bestsellers**
- **A navigation bar for easy access to other sections**
- **Call-to-action buttons ("Shop Now", "Explore", etc.)**
- **Customer testimonials or featured books**

Importance:
It establishes first impressions and helps in brand positioning. An intuitive layout with appealing visuals increases engagement and reduces bounce rate.

Technologiesused:
HTML for structure, CSS for layout and visuals, optional animations using CSS or JavaScript to add interactivity.

2. **Product/MenuPage**
   This page is crucial as it displays the entire product inventory. Items are grouped into categories such as:

- **Fiction (e.g., Romance, Thriller, Mystery)**
- **Non-Fiction (e.g., Biography, Self-Help, History)**
- **Bestsellers**
- **New Arrivals**

Features include:

- **Book image**
- **Title and author**
- **Genre**
- **Price**
- **Add to Cart button**

Importance:
A well-structured catalog improves product discoverability and allows users to compare and select the most suitable options.

UXzConsideration:
Product filters (by genre, author, price range) improve usability and conversion rates.

---

3. **CartPage**
   The cart system is a core part of the e-commerce flow. It displays:

- **All added books with quantity and subtotal**
- **Options to update or remove items**
- **Final checkout button**

Real-worldrelevance:
Gives users control over their purchases and supports decision-making before payment.

Optional enhancements:

- **Cart persistence using localStorage**
- **Live price updates when quantity is changed**

---

4. **AboutUsPage**
   This section gives the business a personal touch. It may include:

- **History of the bookstore**
- **Vision and mission**
- **Founder's message**
- **Team photos and bios**

Purpose:
Builds trust and authenticity with potential buyers, especially in an online bookstore where customer loyalty and satisfaction are key.

---

5. **ContactPage**
   A contact form is essential for customer support and inquiry handling. The form includes:

- **Name**
- **Email**
- **Subject**
- **Message**

Additional elements:

- **Phone number and address**
- **Map location using Google Maps embed**
- **Social media links**

UXFactor:
Quick and easy communication increases customer satisfaction and helps resolve concerns related to orders or returns.

---

6. **User/AdminRegistrationForm**
   This page allows new users and admins to create an account. It collects:

- **Full name**
- **Email or phone**
- **Password and confirmation**
- **User type (dropdown or radio buttons)**

Functionality:

- **Form validation (password match, email format)**
- **Secure data storage (in real deployment, through backend/database)**

Whymatters:
Allows personalized experiences, loyalty features, and secure access for admins to manage the platform.

---

7. **User/AdminLoginForm**
   This form validates users or admins 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**
- **In production: hashing passwords, rate limiting, two-factor authentication**

Differentiated Access:

- **Users can shop, view order history**
- **Admins can manage inventory, view analytics, and process orders**

Technological Stack Overview (Future Enhancement)

While this version is made using HTML/CSS, it can later be extended with:

- **JavaScript for dynamic features (live cart updates, animations)**
- **PHP/Node.js for server-side logic**
- **MySQL/MongoDB for database storage**
- **Session management and authentication for secure login systems**

Sustainability Impact

The bookstore promotes eco-conscious consumerism by reselling gently used books and supporting sustainability practices. It reduces waste and encourages reading.

- **Reselling pre-owned books at affordable prices**
- **Offering discounts for book exchanges**
- **Educating users on sustainable reading habits**

Code:

A. Home page:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Online Bookstore</title>
  <link rel="stylesheet" href="style.css">
  <style>
    body {
      margin: 0;
      font-family: 'Segoe UI', sans-serif;
      background: url('images/bg-books.jpg') no-repeat center center fixed;
      background-size: cover;
      color: #fff;
    }

    header {
      background-color: rgba(0, 0, 0, 0.7);
      padding: 20px;
      text-align: center;
    }

    header h1 {
      margin: 0;
      font-size: 2.5rem;
    }

    nav {
      background-color: rgba(0, 0, 0, 0.6);
      padding: 10px;
    text-align: center;
}

nav a {
    color: #fff;
    text-decoration: none;
    margin: 0 15px;
    font-weight: bold;
}

nav a:hover {
    text-decoration: underline;
}

.hero {
    padding: 50px 20px;
    text-align: center;
}

.hero h2 {
    font-size: 2rem;
}

.book-list {
    display: flex;
    flex-wrap: wrap;
    justify-content: center;
    padding: 40px 20px;
    gap: 20px;
    background-color: rgba(255, 255, 255, 0.1);
```

```
}

.book-card {
    background-color: #fff;
    color: #000;
    border-radius: 8px;
    box-shadow: 0 4px 8px rgba(0,0,0,0.3);
    padding: 20px;
    width: 220px;
    text-align: center;
}

.book-card img {
    width: 100%;
    height: 300px;
    object-fit: cover;
    border-radius: 5px;
}

.book-card h4 {
    margin: 10px 0 5px;
}

.book-card p {
    margin: 5px 0;
}

.add-to-cart {
    margin-top: 10px;
    padding: 8px 16px;
```

```
            background-color: #007bff;
            color: #fff;
            border: none;
            border-radius: 4px;
            cursor: pointer;
        }

        .add-to-cart:hover {
            background-color: #0056b3;
        }

        footer {
            background-color: rgba(0, 0, 0, 0.7);
            text-align: center;
            padding: 15px;
            position: fixed;
            bottom: 0;
            width: 100%;
        }

    </style>
</head>
<body>
    <header>
        <h1>Welcome to Our Bookstore</h1>
    </header>

    <nav>
        <a href="index.html">Home</a>
        <a href="about.html">About</a>
        <a href="products.html">Books</a>

        <a href="testimonial.html">Testimonials</a>

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

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

        <a href="register.html">Register</a>

    </nav>



    <section class="hero">

        <h2>Discover Great Books at Unbeatable Prices</h2>

        <p>Browse our collection of bestsellers, classics, and new releases.</p>

    </section>

    <h2 style="text-align: center; padding-top: 30px; font-size: 2rem; color: #0b0b0b;">📚 Featured Books</h2>



      <div class="book-list">

        <div class="book-card">

            <img src="https://tse1.explicit.bing.net/th/id/OIP.1rNUR9cNB-u7bRMe-B7W8QHaLL?cb=iwp2&rs=1&pid=ImgDetMain" alt="The Alchemist">

            <h4>The Alchemist</h4>

            <p>by Paulo Coelho</p>

            <p>$12.99</p>

            <button class="add-to-cart">Add to Cart</button>

        </div>

        <div class="book-card">

            <img src="https://mir-s3-cdn-cf.behance.net/project_modules/1400/b468d093312907.5e6139cf2ab03.png" alt="1984">

            <h4>1984</h4>

            <p>by George Orwell</p>

            <p>$10.49</p>

            <button class="add-to-cart">Add to Cart</button>

        </div>

        <div class="book-card">
```

```
            <img src="https://m.media-amazon.com/images/I/81q77Q39nEL.jpg" alt="Harry Potter">
            <h4>Harry Potter & the Sorcerer's Stone</h4>
            <p>by J.K. Rowling</p>
            <p>$14.99</p>
            <button class="add-to-cart">Add to Cart</button>
        </div>
    </div>

    <footer>
        &copy; 2025 Bookstore. All rights reserved.
    </footer>
</body>
</html>
```

Css Code-

```
/* === BASE RESET === */
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}

body {
  font-family: 'Segoe UI', sans-serif;
  background-color: #f9f9f9;
  color: #333;
  line-height: 1.6;
  padding: 20px;
}

/* === SHARED ELEMENTS === */
```

```
.container {
  max-width: 900px;
  margin: auto;
  padding: 20px;
  background: #fff;
  border-radius: 10px;
  box-shadow: 0 4px 12px rgba(0,0,0,0.1);
}

.section-title {
  font-size: 2rem;
  margin-bottom: 20px;
  border-bottom: 2px solid #1abc9c;
  padding-bottom: 5px;
  color: #2c3e50;
}

/* === 1. CART PAGE === */
.cart-item {
  display: flex;
  justify-content: space-between;
  align-items: center;
  padding: 15px 0;
  border-bottom: 1px solid #eee;
}

.cart-item input[type="number"] {
  width: 60px;
  padding: 6px;
  border-radius: 4px;
  border: 1px solid #ccc;
  margin: 0 10px;
}

.cart-item button {
  background: #e74c3c;
  color: #fff;
  padding: 8px 12px;
  border: none;
  border-radius: 4px;
  cursor: pointer;
}

.cart-total {
  font-weight: bold;
  background: #ecf0f1;
  padding: 15px;
  margin-top: 20px;
  text-align: right;
}

/* === 2. ABOUT US PAGE === */
.about-section {
  margin-bottom: 40px;
}

.about-section p {
  text-align: justify;
  padding: 10px 0;
}
```

```
.about-section img {
  width: 100%;
  max-width: 200px;
  border-radius: 10px;
  margin-top: 10px;
}

.values-box {
  background: #eafaf1;
  border-left: 4px solid #1abc9c;
  padding: 15px;
  margin-top: 20px;
}

/* === 3. CONTACT PAGE === */
.contact-form input,
.contact-form textarea {
  width: 100%;
  padding: 10px;
  margin-bottom: 15px;
  border: 1px solid #ccc;
  border-radius: 6px;
}

.contact-form input:focus,
.contact-form textarea:focus {
  border-color: #1abc9c;
  outline: none;
}
```

```
.contact-form button {
  padding: 10px 20px;
  background-color: #1abc9c;
  color: white;
  border: none;
  border-radius: 6px;
  cursor: pointer;
  transition: background 0.3s;
}

.contact-form button:hover {
  background-color: #16a085;
}

/* === 4. REGISTRATION FORM === */
.form-group {
  margin-bottom: 15px;
}

label {
  display: block;
  margin-bottom: 5px;
  font-weight: bold;
}

input[type="text"],
input[type="email"],
input[type="password"] {
  width: 100%;
  padding: 10px;

  border: 1px solid #ccc;

  border-radius: 6px;

}


.form-section {

  background-color: #f1fdfb;

  padding: 25px;

  border-radius: 10px;

  box-shadow: 0 0 10px rgba(0,0,0,0.08);

}


/* === 5. LOGIN FORM === */

.login-box {

  max-width: 400px;

  margin: 50px auto;

  padding: 30px;

  border-radius: 10px;

  background: #fff;

  box-shadow: 0 5px 15px rgba(0,0,0,0.1);

}


.login-box h2 {

  text-align: center;

  margin-bottom: 20px;

}


.login-box input {

  width: 100%;

  padding: 10px;
  margin-bottom: 15px;
  border: 1px solid #ddd;
  border-radius: 6px;
}

.login-box button {
  width: 100%;
  padding: 10px;
  background: #1abc9c;
  color: #fff;
  border: none;
  border-radius: 6px;
  cursor: pointer;
  font-weight: bold;
}

.login-box button:hover {
  background-color: #16a085;
}

.message {
  font-size: 0.9rem;
  padding: 5px 0;
}

.message.error {
  color: #e74c3c;
}

.message.success {
  color: #2ecc71;
}
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}

body {
  font-family: 'Roboto', sans-serif;
  background-color: #f4f4f4;
  color: #333;
}

header {
  background-color: #1e293b;
  color: #fff;
  padding: 20px 0;
  text-align: center;
}

nav {
  background-color: #334155;
  display: flex;
  justify-content: center;
  gap: 20px;
  padding: 10px 0;
}

nav a {
  color: #fff;
  text-decoration: none;
  font-weight: 500;
}

nav a:hover {
  color: #facc15;
}

.hero {
  background: url('images/bg-books.jpg') no-repeat center center/cover;
  color: white;
  text-align: center;
  padding: 80px 20px;
}

.hero-content {
  background: rgba(0, 0, 0, 0.6);
  padding: 30px;
  border-radius: 10px;
  display: inline-block;
}

.section-title {
  text-align: center;
  margin: 40px 0 10px;
}

.book-list {
  display: flex;
  flex-wrap: wrap;
  justify-content: center;
  gap: 30px;
  padding: 20px 40px;
}

.book-card {
  background: white;
  border-radius: 10px;
  overflow: hidden;
  box-shadow: 0 5px 15px rgba(0,0,0,0.15);
  width: 230px;
  transition: transform 0.3s;
}

.book-card:hover {
  transform: translateY(-5px);
}

.book-card img {
  width: 100%;
  height: 300px;
  object-fit: cover;
}

.book-card h4, .book-card p {
  padding: 10px 15px;
  margin: 0;
}
```

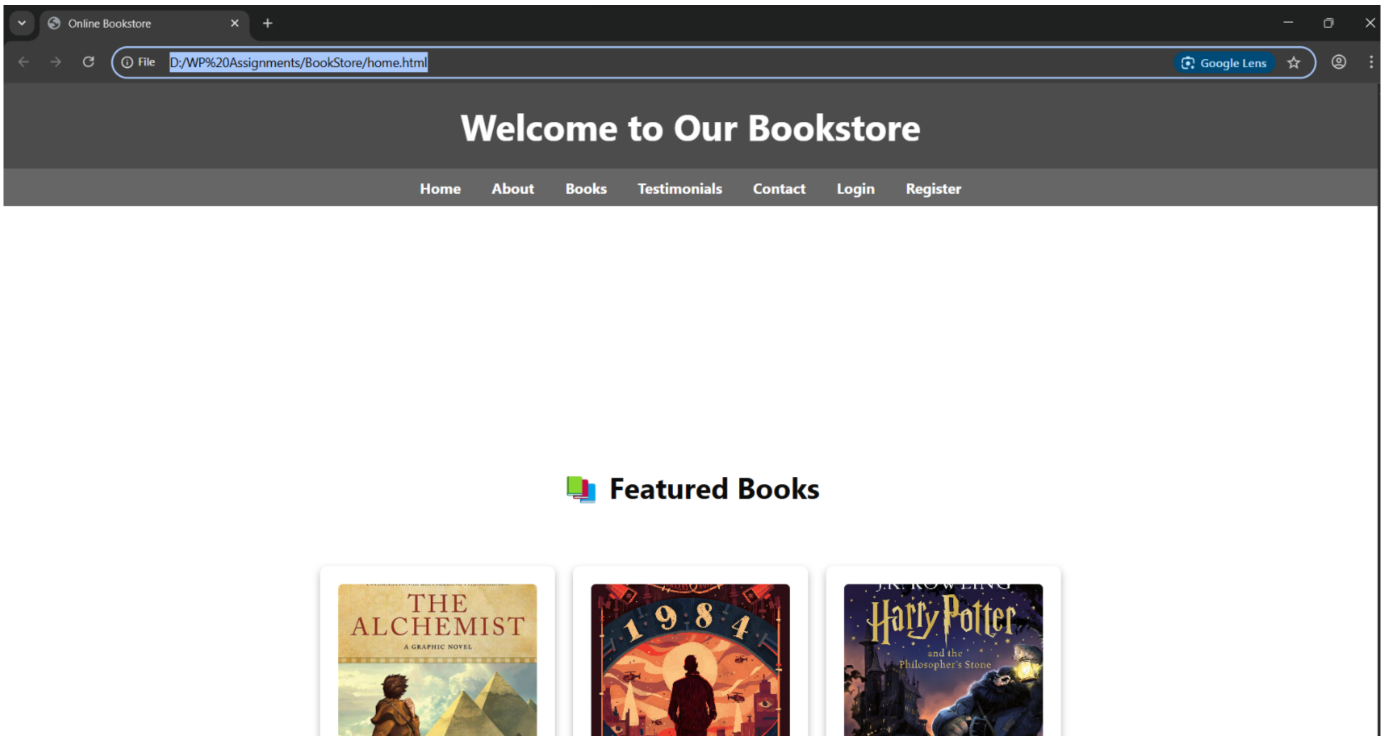
```
.add-to-cart {
  display: block;
  width: 90%;
  margin: 10px auto 15px;
  padding: 10px;
  background-color: #2563eb;
  color: white;
  border: none;
  border-radius: 6px;
  cursor: pointer;
  font-weight: bold;
}

.add-to-cart:hover {
  background-color: #1d4ed8;
}

.loading-text {
  text-align: center;
  padding: 40px;
  font-size: 1.2rem;
  color: #666;
}

footer {
  background-color: #1e293b;
  color: #fff;
  text-align: center;
  padding: 15px;
  margin-top: 40px;
```

```
}


Script code:-
const API_URL = 'books.json'; // Replace with actual API or data file


document.addEventListener('DOMContentLoaded', () => {
  fetch(API_URL)
    .then(response => response.json())
    .then(data => renderBooks(data))
    .catch(error => {
      document.getElementById('bookList').innerHTML = `
        <p class="loading-text">⚠️ Failed to load books. Please try again later.</p>`;
      console.error("Error fetching books:", error);
    });
});


function renderBooks(books) {
  const bookList = document.getElementById('bookList');
  bookList.innerHTML = '';


  books.forEach(book => {
    const card = document.createElement('div');
    card.className = 'book-card';
    card.innerHTML = `
      <img src="${book.image}" alt="${book.title}">
      <h4>${book.title}</h4>
      <p>by ${book.author}</p>
      <p>${book.price}</p>
      <button class="add-to-cart">Add to Cart</button>
    `;
    bookList.appendChild(card);
  });

  attachCartListeners();
}

function attachCartListeners() {
  const cart = JSON.parse(localStorage.getItem('cart')) || [];

  document.querySelectorAll('.add-to-cart').forEach(button => {
    button.addEventListener('click', () => {
      const card = button.closest('.book-card');
      const title = card.querySelector('h4').textContent;
      const author = card.querySelector('p').textContent;
      const price = card.querySelectorAll('p')[1].textContent;
      const image = card.querySelector('img').src;

      const book = { title, author, price, image };
      cart.push(book);
      localStorage.setItem('cart', JSON.stringify(cart));

      button.textContent = "✓ Added";
      button.disabled = true;
      setTimeout(() => {
        button.textContent = "Add to Cart";
        button.disabled = false;
      }, 1500);
    });
  });
```

```
}
```

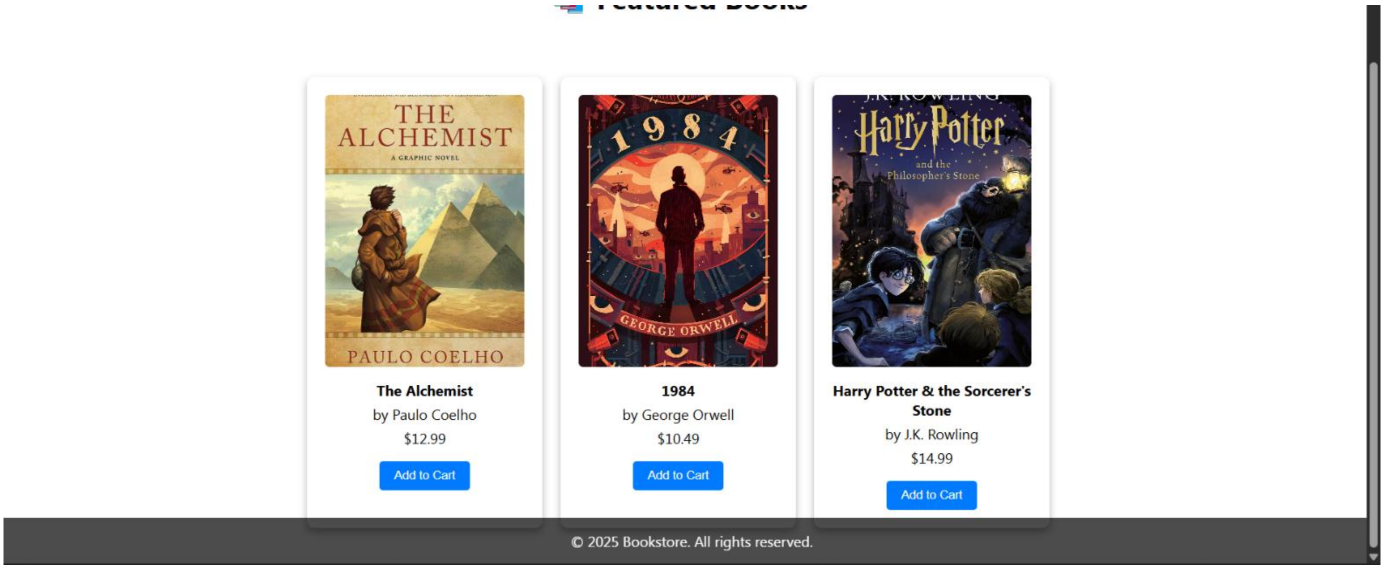
Output:

A. Index/Home page output:



Welcome to Our Bookstore

📚 Featured Books



The Alchemist
by Paulo Coelho
$12.99

1984
by George Orwell
$10.49

Harry Potter & the Sorcerer's Stone
by J.K. Rowling
$14.99

© 2025 Bookstore. All rights reserved.

Code:

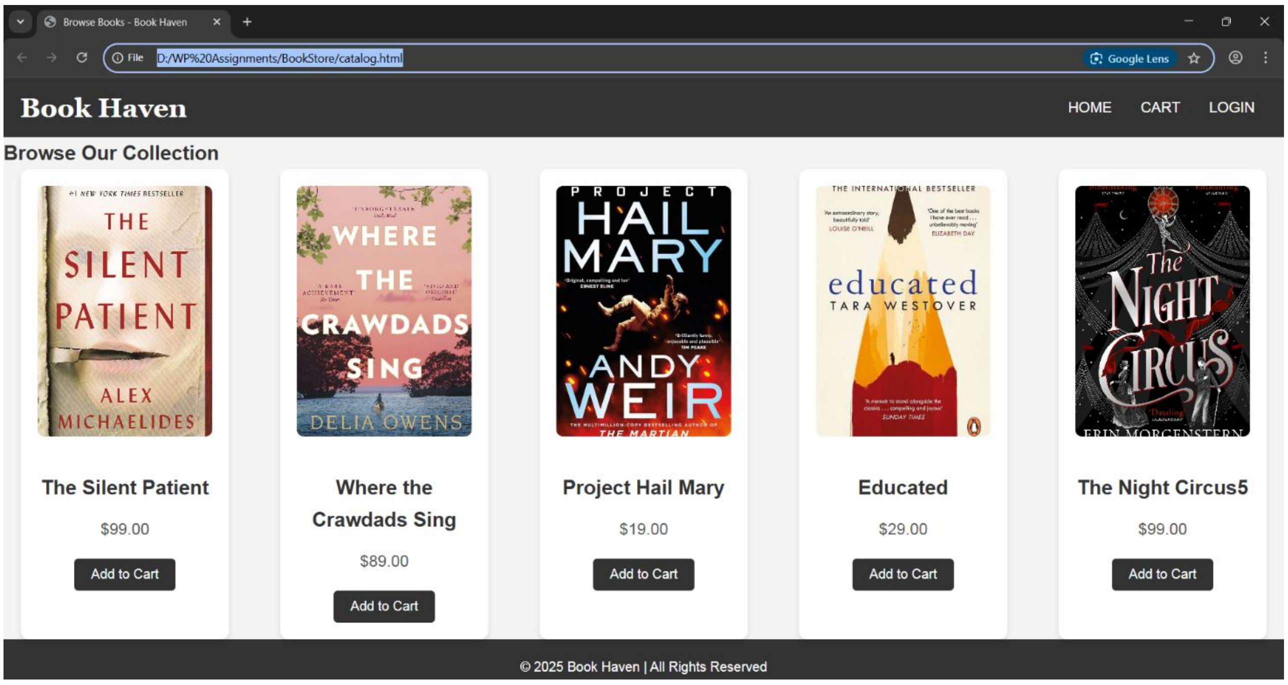
B. menu/product page:

code:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Browse Books - Book Haven</title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <header>
    <div class="logo">
      <h1>Book Haven</h1>
    </div>
    <nav>
      <a href="index.html">Home</a>
      <a href="cart.html">Cart</a>
      <a href="login.html">Login</a>
    </nav>
  </header>

  <main>
    <h2>Browse Our Collection</h2>
    <div class="book-list">
      <!-- Book 1 -->
      <div class="book-card">
        <img src="images/book1.jpg" alt="Book Title 1">
        <h4>Book Title 1</h4>
        <p>₹1,499.00</p>
        <button class="add-to-cart">Add to Cart</button>
    </div>
    <!-- Book 2 -->
    <div class="book-card">
        <img src="images/book2.jpg" alt="Book Title 2">
        <h4>Book Title 2</h4>
        <p>₹899.00</p>
        <button class="add-to-cart">Add to Cart</button>
    </div>
    <!-- Book 3 -->
    <div class="book-card">
        <img src="images/book3.jpg" alt="Book Title 3">
        <h4>Book Title 3</h4>
        <p>₹1,199.00</p>
        <button class="add-to-cart">Add to Cart</button>
    </div>
    <!-- Book 4 -->
    <div class="book-card">
        <img src="images/book4.jpg" alt="Book Title 4">
        <h4>Book Title 4</h4>
        <p>₹1,299.00</p>
        <button class="add-to-cart">Add to Cart</button>
    </div>
    <!-- Book 5 -->
    <div class="book-card">
        <img src="images/book5.jpg" alt="Book Title 5">
        <h4>Book Title 5</h4>
        <p>₹999.00</p>
        <button class="add-to-cart">Add to Cart</button>
    </div>
</div>
```

```
    </main>


    <footer>
        <p>&copy; 2025 Book Haven | All Rights Reserved</p>
    </footer>


    <script src="scripts.js"></script>
</body>
</html>
```

Output:


Browse Books - Book Haven
D:/WP%20Assignments/BookStore/catalog.html
Google Lens
HOME
CART
LOGIN

Book Haven

Browse Our Collection

The Silent Patient
$99.00
Add to Cart

Where the Crawdads Sing
$89.00
Add to Cart

Project Hail Mary
$19.00
Add to Cart

Educated
$29.00
Add to Cart

The Night Circus5
$99.00
Add to Cart

© 2025 Book Haven | All Rights Reserved

Code:

C. cart page:

code:

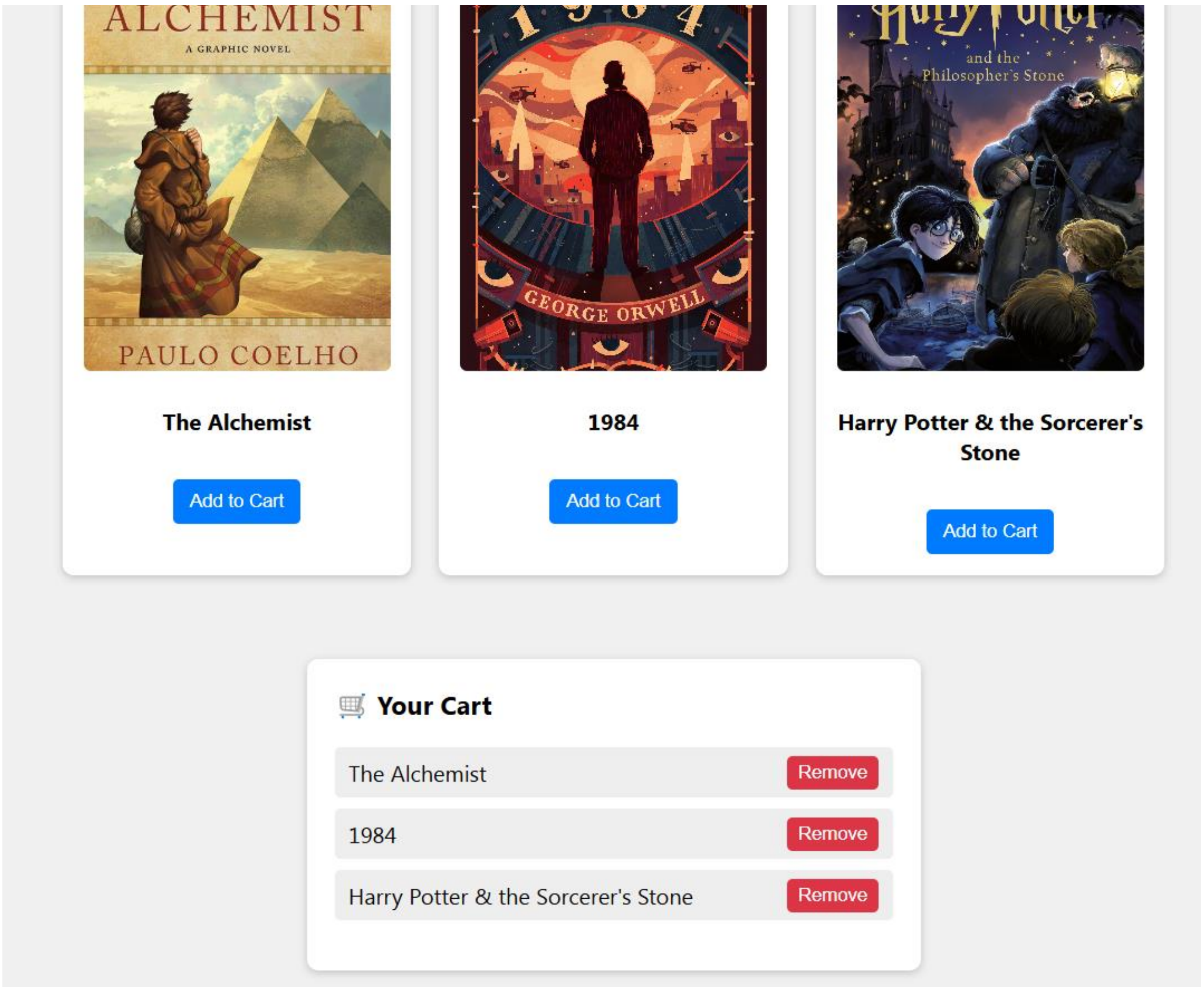
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Your Cart</title>
    <link rel="stylesheet" href="styles.css">
```

```
</head>
<body>
    <header>
        <div class="logo">
            <h1>Book Haven</h1>
        </div>
        <nav>
            <a href="index.html">Home</a>
            <a href="about.html">About Us</a>
            <a href="menu.html">Shop</a>
            <a href="contact.html">Contact</a>
            <a href="cart.html">Cart</a>
            <a href="login.html">Login</a>
        </nav>
    </header>

    <section id="cart-items">
        <!-- Cart items will be dynamically loaded here -->
    </section>

    <section class="checkout">
        <button class="checkout-btn">Checkout</button>
    </section>

    <footer>
        <p>&copy; 2025 Book Haven. All Rights Reserved.</p>
    </footer>

    <script src="scripts.js"></script>
</body>
```

```
</html>
```

Output:

C. cart



The Alchemist

1984

Harry Potter & the Sorcerer's Stone

Your Cart

The Alchemist Remove

1984 Remove

Harry Potter & the Sorcerer's Stone Remove

Code:

D. about us page:

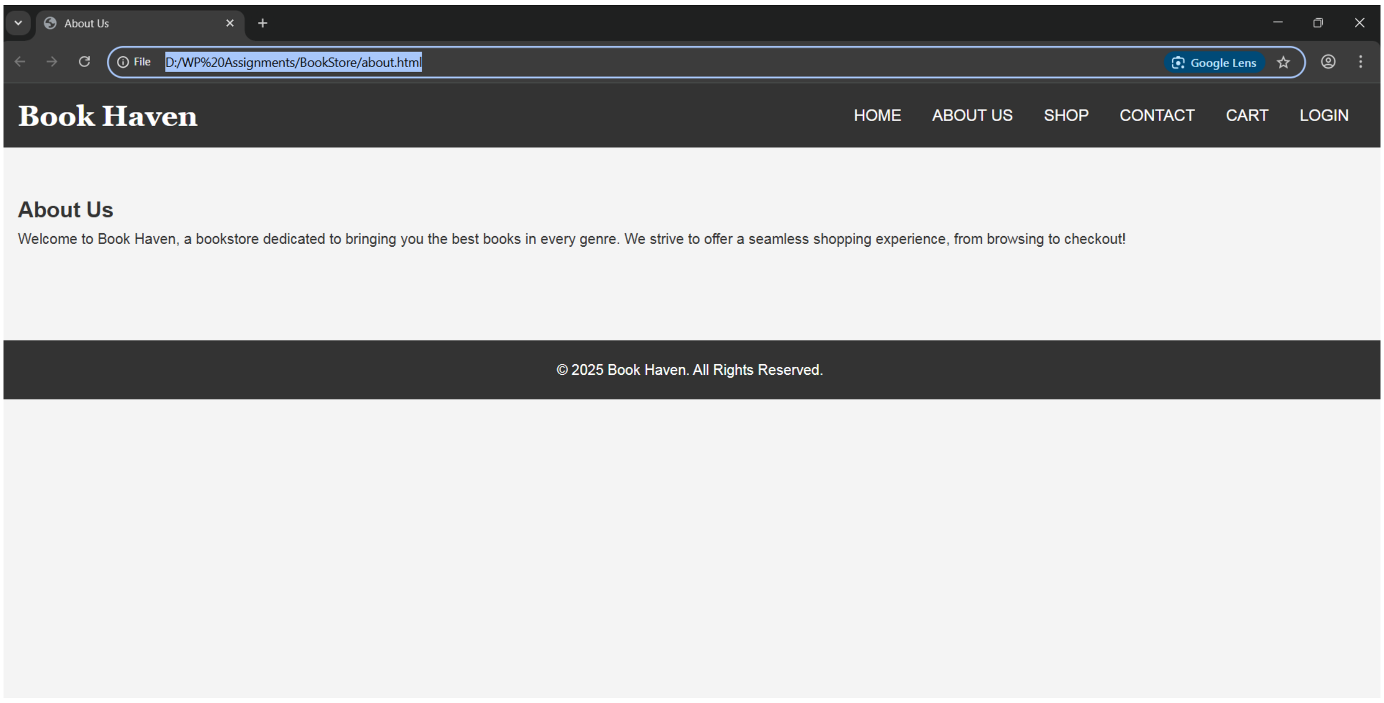
code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>About Us</title>
    <link rel="stylesheet" href="styles.css">
```

```
</head>

<body>

    <header>

        <div class="logo">

            <h1>Book Haven</h1>

        </div>

        <nav>

            <a href="index.html">Home</a>

            <a href="about.html">About Us</a>

            <a href="menu.html">Shop</a>

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

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

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

        </nav>

    </header>


    <section class="about">

        <h2>About Us</h2>

        <p>Welcome to Book Haven, a bookstore dedicated to bringing you the best books in every genre. We strive to
offer a seamless shopping experience, from browsing to checkout!</p>

    </section>


    <footer>

        <p>&copy; 2025 Book Haven. All Rights Reserved.</p>

    </footer>


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

</body>

</html>
```

Output:

D. about us page output:



About Us

D:/WP%20Assignments/BookStore/about.html

Book Haven

HOME ABOUT US SHOP CONTACT CART LOGIN

About Us

Welcome to Book Haven, a bookstore dedicated to bringing you the best books in every genre. We strive to offer a seamless shopping experience, from browsing to checkout!

© 2025 Book Haven. All Rights Reserved.

<u>Code</u>:

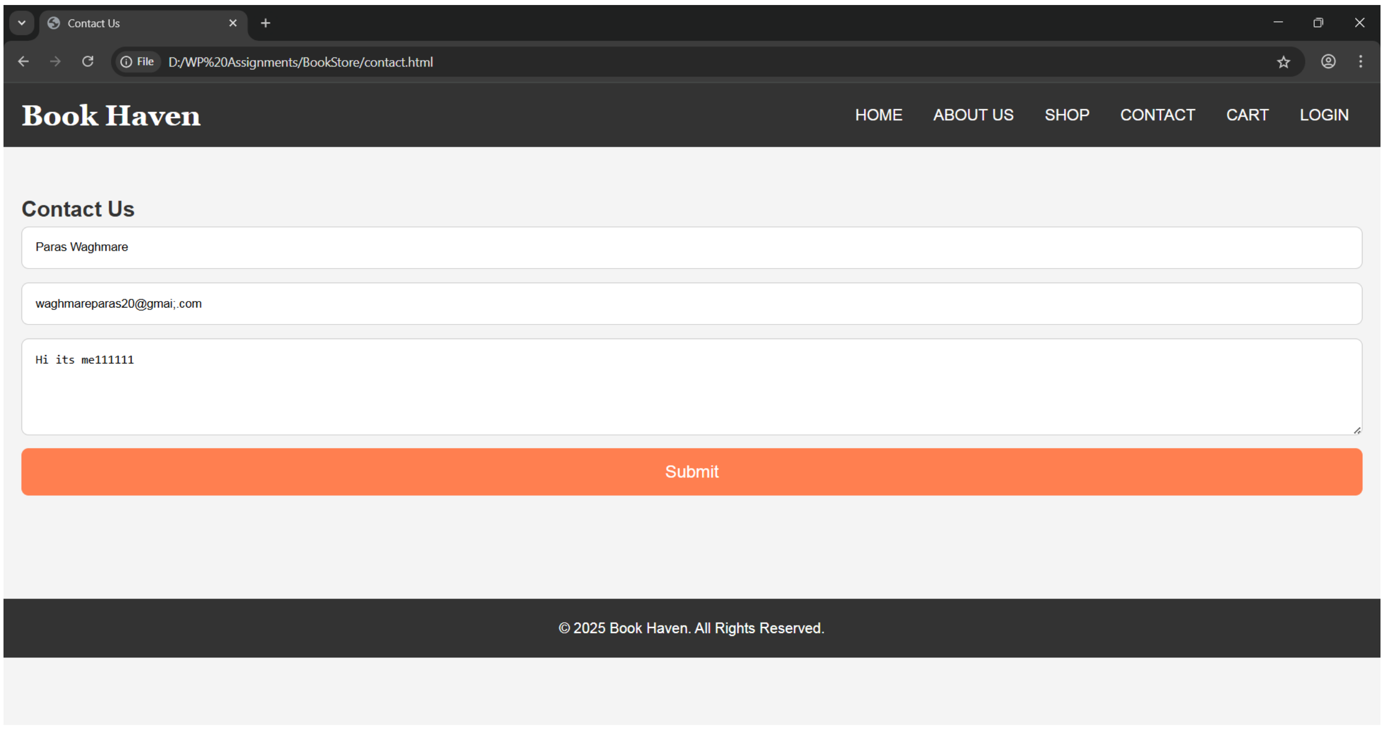
E. contact us page:

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</title>
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <header>
        <div class="logo">
            <h1>Book Haven</h1>
        </div>
        <nav>
            <a href="index.html">Home</a>
            <a href="about.html">About Us</a>
            <a href="menu.html">Shop</a>
            <a href="contact.html">Contact</a>
            <a href="cart.html">Cart</a>
            <a href="login.html">Login</a>
        </nav>
    </header>

    <section class="contact">
        <h2>Contact Us</h2>
        <form action="#">
            <input type="text" name="name" placeholder="Your Name" required>
            <input type="email" name="email" placeholder="Your Email" required>
            <textarea name="message" rows="5" placeholder="Your Message" required></textarea>
            <button type="submit">Submit</button>
        </form>
    </section>

    <footer>
        <p>&copy; 2025 Book Haven. All Rights Reserved.</p>
    </footer>

    <script src="scripts.js"></script>
</body>
</html>
```

Output:

E. contact us page output:


Contact Us

D:/WP%20Assignments/BookStore/contact.html

Book Haven

HOME ABOUT US SHOP CONTACT CART LOGIN

Contact Us

Paras Waghmare

waghmareparas20@gmai;.com

Hi its me111111

Submit

© 2025 Book Haven. All Rights Reserved.

<u>Code</u>:

F. registration page:

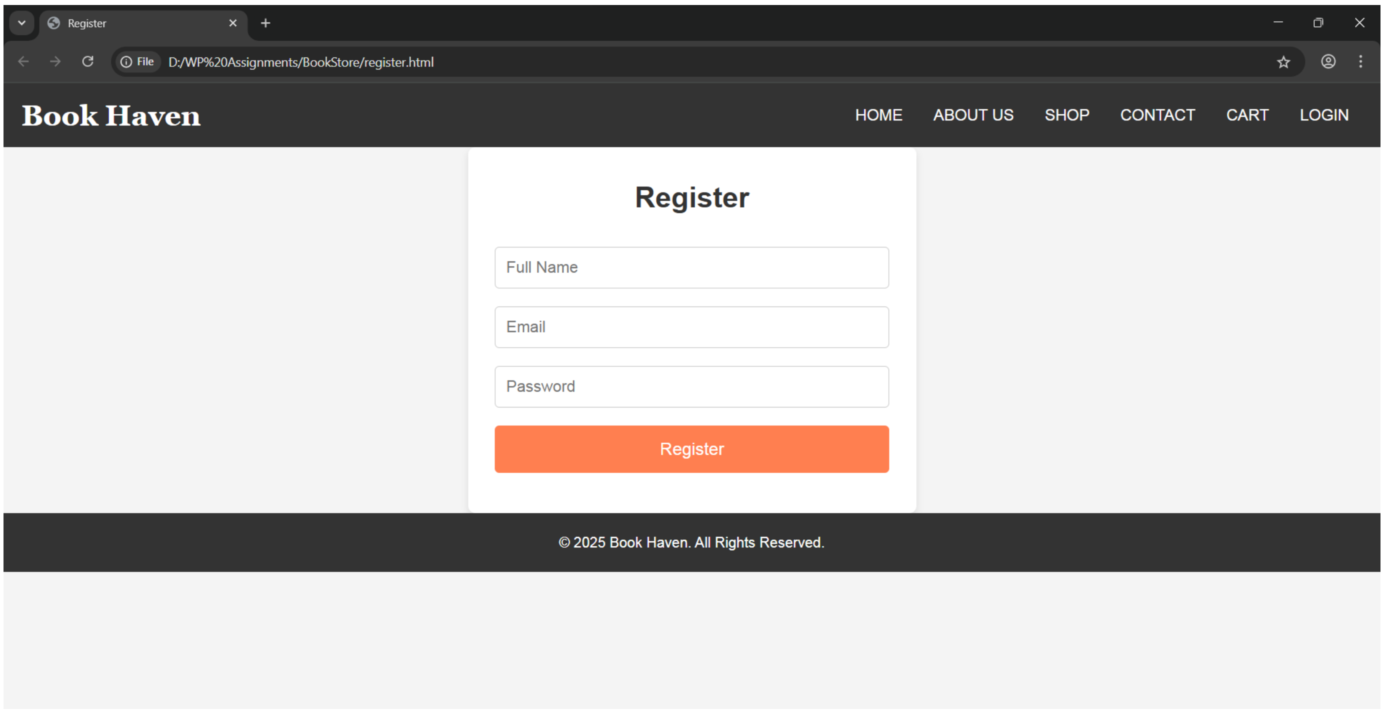
code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Register</title>
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <header>
        <div class="logo">
            <h1>Book Haven</h1>
        </div>
        <nav>
```

```
            <a href="index.html">Home</a>
            <a href="about.html">About Us</a>
            <a href="menu.html">Shop</a>
            <a href="contact.html">Contact</a>
            <a href="cart.html">Cart</a>
            <a href="login.html">Login</a>
        </nav>
    </header>

    <section class="register">
        <h2>Register</h2>
        <form id="register-form">
            <input type="text" name="name" placeholder="Full Name" required>
            <input type="email" name="email" placeholder="Email" required>
            <input type="password" name="password" placeholder="Password" required>
            <button type="submit">Register</button>
        </form>
    </section>

    <footer>
        <p>&copy; 2025 Book Haven. All Rights Reserved.</p>
    </footer>

    <script src="scripts.js"></script>
</body>
</html>
```

Output:

F. registration page output:



Register

D:/WP%20Assignments/BookStore/register.html

Book Haven

HOME ABOUT US SHOP CONTACT CART LOGIN

Register

Full Name

Email

Password

Register

© 2025 Book Haven. All Rights Reserved.

<u>Code</u>:

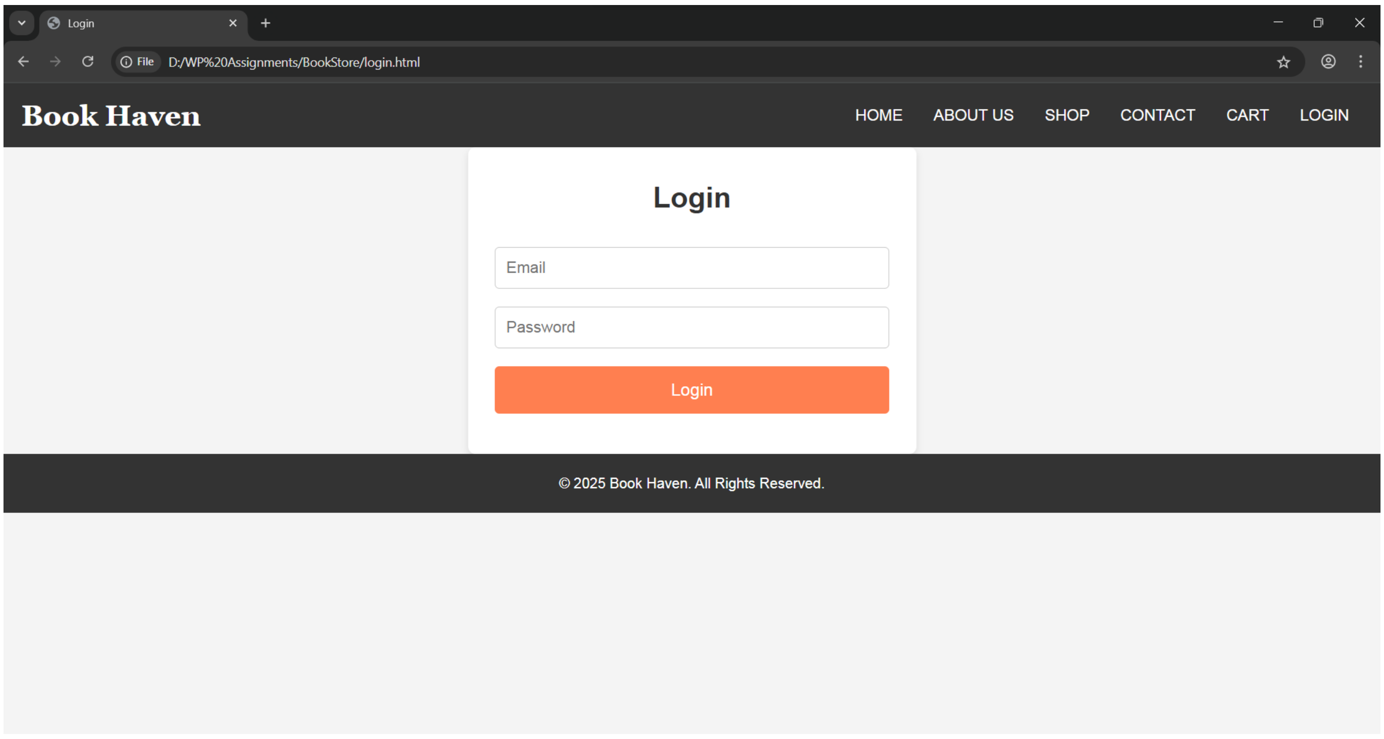
G. login page:

code:

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Login</title>

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

</head>

<body>

    <header>

        <div class="logo">

            <h1>Book Haven</h1>

        </div>

        <nav>

            <a href="index.html">Home</a>
            <a href="about.html">About Us</a>

            <a href="menu.html">Shop</a>

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

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

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

        </nav>

    </header>


    <section class="login">

        <h2>Login</h2>

        <form id="login-form">

            <input type="email" name="email" placeholder="Email" required>

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

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

        </form>

    </section>


    <footer>

        <p>&copy; 2025 Book Haven. All Rights Reserved.</p>

    </footer>


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

</body>

</html>
```

Output:

G. login page output:



Login

Email

Password

Login

© 2025 Book Haven. All Rights Reserved.

Conclusion

The second-hand gaming console store website combines practical e-commerce features with a sustainability-driven mission. The use of structured web design, user-friendly forms, and clear product categorization ensures a seamless experience for both shoppers and admins. With further backend integration, it can evolve into a fully operational online business supporting green technology use and community engagement.

<u>Experiment No.3</u>

Problem Statement: In today's digital era, readers frequently encounter challenges when searching for and purchasing books online due to fragmented platforms, limited filtering options, and inconsistent access to detailed information about books. Many existing websites are not user-friendly and often lack features that support efficient book discovery and smooth purchasing experiences. Furthermore, independent bookstores and self-published authors face difficulties in reaching a wider audience through accessible and streamlined web solutions. There is a clear need for a web-based system that allows users to search for books, explore relevant details, and complete purchases with ease. The proposed Book Search and Shopping web application aims to address this gap by offering a simple, user-centered platform that integrates intelligent search, comprehensive book information, shopping cart functionality, and a secure checkout process—ultimately enhancing the overall experience for both readers and book sellers.

Enhance the layout of the coffee shop website using CSS Grid for the home page.

Use CSS Grid to layout the menu/product items in a structured and style the menu categories with appropriate headings, spacing, separators, images, descriptions, and prices.

Theory:

CSS Theory for Enhancing the Layout of a Book Search and Shopping Website using CSS Grid CSS Grid Layout is a powerful two-dimensional layout system in CSS that allows developers to design web pages with more precision and flexibility. For a book search and shopping website, an intuitive and responsive user interface plays a critical role in improving the browsing and purchasing experience. CSS Grid provides a structured way to align content in rows and columns, making it ideal for organizing books, search results, navigation panels, and product listings.

By using CSS Grid, developers can define a grid container and place child elements into specified grid areas. This eliminates the need for complex float or positioning hacks and enhances both maintainability and scalability of the layout. For instance, on a homepage or search result page, books can be displayed in a neatly aligned grid of cards, each containing a book cover, title, author, price, and an "Add to Cart" button. The layout remains consistent regardless of screen size, with media queries allowing the grid to adapt for mobile, tablet, or desktop views.

Moreover, CSS Grid supports features such as grid-template-areas, auto-placement, and flexible sizing (fr units), enabling the layout to prioritize space for key sections like the book gallery, filters, navigation bar, and shopping cart. This structured approach improves the visual hierarchy and guides the user's attention effectively.

In summary, CSS Grid enhances the layout design of a book search and shopping website by providing clean, adaptable, and user-friendly structures. It allows for a consistent and professional appearance across devices, ensuring that users can easily navigate, search for books, and make purchases without friction.

Introduction to CSS Grid

- **CSS Grid Layout is a powerful two-dimensional layout system ideal for web interfaces. Unlike Flexbox (which handles layout in a single dimension), CSS Grid allows control over both rows and columns, making it perfect for creating responsive layouts like those used in online bookstores.**
- **Using CSS Grid, designers can create clean, well-structured, and consistent page layouts — especially useful for:**
- **Homepage sections like featured books, bestsellers, or offers**
- **Category or genre-based listings (e.g., fiction, non-fiction, academic)**
- **Structured book cards in the cart or gallery view**

-

Why CSS Grid for this Bookstore Website?

- **For a book shopping website, how books are presented greatly impacts user engagement. Visitors want to browse genres, compare prices, and add books to their cart easily.**
- **CSS Grid is used to:**
- **Arrange book cards in neat, multi-column layouts (2x2, 3x3, etc.)**
- **Organize sections like "Featured Books", "New Arrivals", or "Editor's Picks"**
- **Maintain consistent alignment of book covers, titles, and prices**
- **Ensure the layout adjusts smoothly across mobile, tablet, and desktop screens**
-

---

1. Home Page Layout with CSS Grid

- **The homepage of the bookstore is structured using CSS Grid to define key areas:**
- **A full-width navigation header**
- **A large banner or hero section**
- **A three-column highlight section for bestsellers or popular genres**
- **A testimonial row from readers or authors**
- **A footer with newsletter signup and social links**
- **Grid Benefits on the Home Page:**
- **Easy control over large content areas**
- **Consistent alignment of banners, text blocks, and call-to-action buttons**
- **Layout remains clean and adaptable without complex media queries**
-

---

2. Book/Product Listing Page Using CSS Grid

- **This page displays books by categories like:**
- **Fiction**
- **Non-Fiction**
- **Academic**
- **Children's Books**
- **Self-Help / Motivation**
- **Each book is shown in a card format, laid out using CSS Grid to balance readability and visual appeal.**
- **Key Grid Features on Product Page:**
- **Uniform card sizes for a tidy look**
- **Gaps between cards for better spacing**

- **Book cover, title, author, price, and “Add to Cart” button neatly aligned**
- **Supports 2, 3, or 4 columns depending on screen size**
- **Example CSS Grid Layout:**
- **css**
- **CopyEdit**
- **.books-grid {**
- ** display: grid;**
- ** grid-template-columns: repeat(auto-fit, minmax(220px, 1fr));**
- ** grid-gap: 25px;**
- ** padding: 20px;**
- **}**
- **Each .book-card inside this grid contains:**
- **Book cover image**
- **Title and author**
- **Short description or genre**
- **Price tag**

  3.“Add to Cart” button

-

- 🧾 **Additional Styling Concepts**
- **Category Headings: Use bold fonts, color bands, or underlines to visually separate genres**
- **Grid Separators: Thin lines or colored backgrounds help separate each book category**
- **Hover Effects: Cards can slightly zoom or lift on hover, making the site feel more interactive**
- **Responsive Design: auto-fit and minmax() allow the layout to adjust seamlessly to any screen size**
-

---

  4. Mobile Responsiveness with CSS Grid

- **CSS Grid shines on mobile devices thanks to its adaptability. For example:**
- **Grid columns collapse to 1 or 2 per row depending on screen width**
- **Content remains well-spaced and readable**
- **Users can browse and tap book cards easily**
- **Benefits for Mobile Users:**
- **Effortless scrolling and interaction**
- **Cleaner, uncluttered layout**
- **Consistent experience across all devices**

Code:

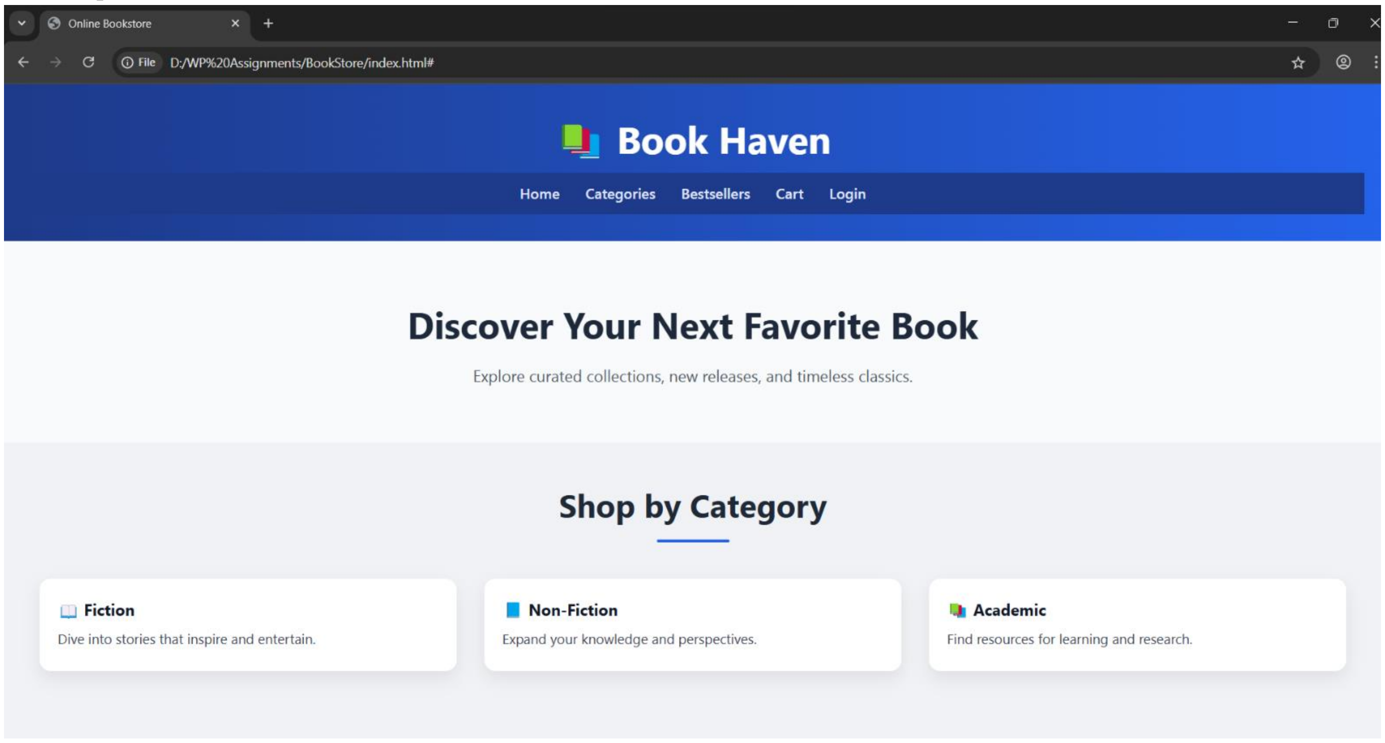
```
<!DOCTYPE html>
<html>
<head>
  <title>Second Hand Consoles</title>
  <style>
    .image-container {
      display: flex;
      justify-content: center;
      gap: 20px;
      padding: 20px;
    }

    .image-container div {
      text-align: center;
      width: 40%;
    }

    .image-container img {
      width: 100%;
      height: auto;
      aspect-ratio: 16 / 9;
      border-radius: 8px;
      border: 3px solid #333;
    }

    * {
      margin: 0;
    padding: 0;

    box-sizing: border-box;

}


body {

    font-family: Arial, sans-serif;

    background-color: #f4f4f4;

}


nav {

    background-color: #333;

    padding: 1rem;

}


nav ul {

    display: flex;

    justify-content: space-between;

    list-style: none;

    align-items: center;

}


.nav-left, .nav-right {

    display: flex;

    gap: 15px;

}


nav ul li a {

    color: white;

    text-decoration: none;

    font-weight: bold;
```

```
        }

        .content {
          background-color: white;
          padding: 20px;
          text-align: center;
          border-radius: 8px;
          margin: 20px;
        }

        footer {
          background-color: #333;
          color: white;
          text-align: center;
          padding: 10px;
        }

        .logout-btn {
          background: none;
          border: none;
          color: white;
          font-weight: bold;
          cursor: pointer;
        }
    </style>
</head>
<body>
    <nav>
        <ul>
          <div class="nav-left">
```

```
            <li><a href="homepage.html">Home</a></li>

            <li><a href="contact_us.html">Contact Us</a></li>

            <li><a href="about_us.html">About Us</a></li>

            <li><a href="product.html">Products</a></li>

        </div>

        <div class="nav-right" id="authLinks">

            <!-- This will be filled by JavaScript -->

        </div>

    </ul>

</nav>


<div class="content">

    <h2>Welcome to Our Website!</h2>

    <h3>We sell premium grade second-hand gaming consoles!</h3>

</div>


<div class="content">

    <h3>Products</h3>

    <p>Check out our latest products!</p>

    <div class="image-container">

        <div>

            <img src="images/ps 4.jpg" alt="PS4">

            <p>Price: $350</p>

        </div>

        <div>

            <img src="images/ps5.jpg" alt="PS5">

            <p>Price: $500</p>

        </div>

    </div>

</div>
```

```
    <footer>
        <p>&copy; 2025 Tsunami. All Rights Reserved.</p>
    </footer>

    <script>
        const authLinks = document.getElementById('authLinks');
        const userDetails = JSON.parse(localStorage.getItem('userDetails'));

        if (userDetails && userDetails.username) {
            authLinks.innerHTML = `
                <li style="color: white;">Welcome, ${userDetails.username}</li>
                <li><button class="logout-btn" onclick="logout()">Logout</button></li>
            `;
        } else {
            authLinks.innerHTML = `
                <li><a href="login.html">Login</a></li>
                <li><a href="registration.html">Registration</a></li>
            `;
        }

        function logout() {
            localStorage.removeItem('userDetails');
            window.location.reload();
        }
    </script>
</body>
</html>
```

Output



Online Bookstore

D:/WP%20Assignments/BookStore/index.html#

Book Haven

Home Categories Bestsellers Cart Login

Discover Your Next Favorite Book

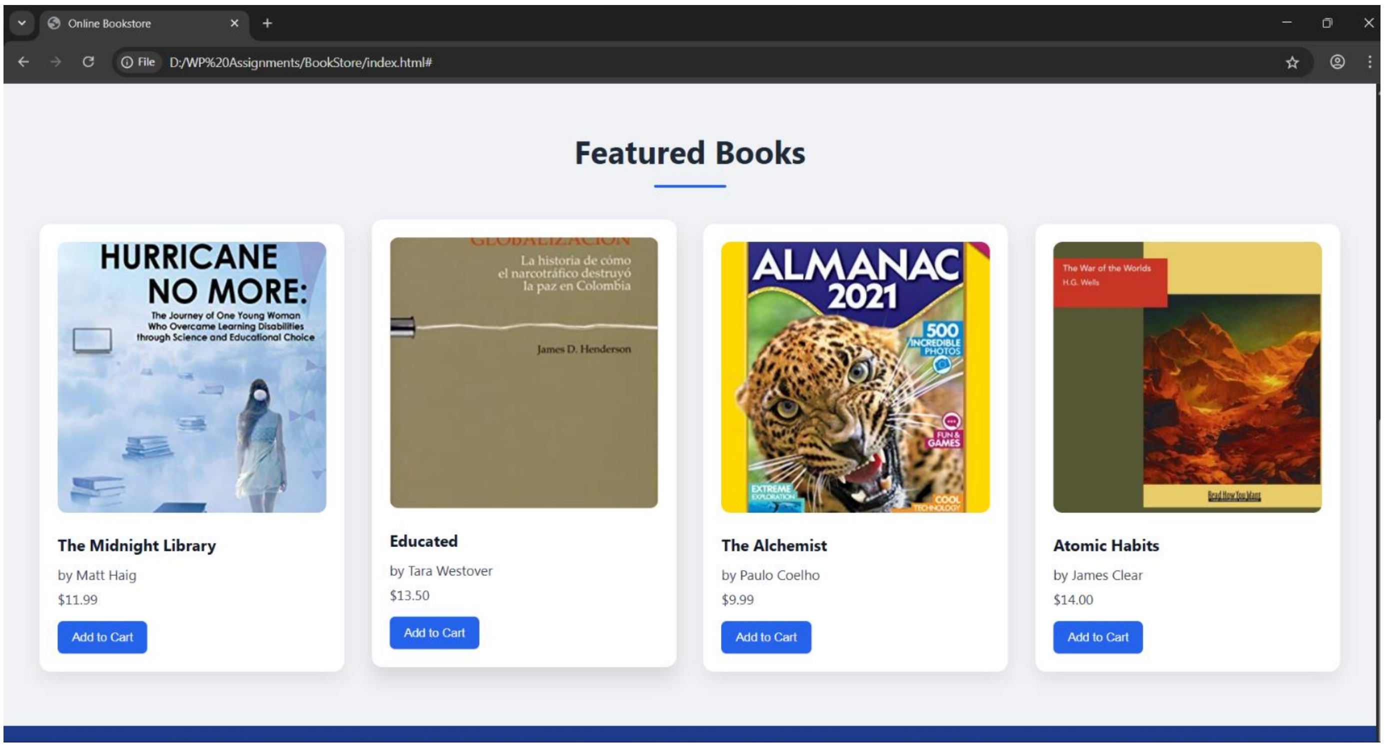
Explore curated collections, new releases, and timeless classics.

Shop by Category

Fiction
Dive into stories that inspire and entertain.

Non-Fiction
Expand your knowledge and perspectives.

Academic
Find resources for learning and research.



Online Bookstore

D:/WP%20Assignments/BookStore/index.html#

Featured Books

The Midnight Library
by Matt Haig
$11.99
Add to Cart

Educated
by Tara Westover
$13.50
Add to Cart

The Alchemist
by Paulo Coelho
$9.99
Add to Cart

Atomic Habits
by James Clear
$14.00
Add to Cart

Conclusion

CSS Grid is a powerful tool for building modern, responsive, and structured websites. In the case of a second-hand gaming console store, **CSS Grid simplifies complex layout structures**, enhances visual clarity, and provides a clean, user-friendly interface.

By using CSS Grid:

- The **home page** becomes visually appealing and sectioned clearly for better navigation.
- The **product menu** is organized and readable, allowing users to quickly explore items.
- The site adapts beautifully across devices without writing dozens of media queries.
- Layout and spacing between elements remain consistent, ensuring a **professional and polished appearance**.

In summary, CSS Grid plays a crucial role in improving the **aesthetic appeal, usability, and responsiveness** of your e-commerce platform—making it both functional and engaging for your users.

Experiment No.4

Problem Statement: In today’s digital era, readers frequently encounter challenges when searching for and purchasing books online due to fragmented platforms, limited filtering options, and inconsistent access to detailed information about books. Many existing websites are not user-friendly and often lack features that support efficient book discovery and smooth purchasing experiences. Furthermore, independent bookstores and self-published authors face difficulties in reaching a wider audience through accessible and streamlined web solutions. There is a clear need for a web-based system that allows users to search for books, explore relevant details, and complete purchases with ease. The proposed Book Search and Shopping web application aims to address this gap by offering a simple, user-centered platform that integrates intelligent search, comprehensive book information, shopping cart functionality, and a secure checkout process—ultimately enhancing the overall experience for both readers and book sellers.

Theory:

CSS Theory: Enhancing and Styling Key Pages in a Book Search and Shopping Cascading Style Sheets (CSS) play a vital role in enhancing the visual appeal, usability, and overall user experience of a book search and shopping website. By applying well-structured CSS rules, developers can ensure that key pages—such as the homepage, book listings, product detail pages, cart, and checkout—are not only visually appealing but also responsive and easy to navigate.

Styling the homepage involves creating a welcoming interface with intuitive navigation. CSS can be used to define consistent typography, padding, spacing, and color schemes that reflect the brand identity. The use of hero banners, featured book sections, and responsive navigation bars can be implemented using CSS Flexbox and Grid for clean alignment and scalability across devices.

In the book listing or search results page, CSS is crucial for organizing content into grids or card layouts. Each book card typically includes an image, title, author name, price, and an “Add to Cart” button. CSS allows developers to maintain consistent sizing, spacing, and hover effects, making the interface interactive and user-friendly. Responsive design techniques using media queries ensure that the layout adjusts seamlessly on mobile and tablet devices.

The product detail page benefits from CSS by emphasizing hierarchy—larger font sizes and bold headings for the book title, stylized price tags, and visually distinct call-to-action buttons like “Buy Now.” Elements like tabs for reviews, descriptions, and author info can be styled for clarity using borders, spacing, and smooth transitions.

The shopping cart and checkout pages require a clean and focused layout. CSS helps group related information—such as item lists, pricing, and payment fields—using forms, tables, or flex containers. Consistent use of color and spacing helps reduce cognitive load, guiding the user step-by-step through the purchasing process.

In addition, CSS animations and transitions can be used to enhance feedback (e.g., when an item is added to the cart) and improve perceived performance by making interactions feel smoother.

In conclusion, CSS is a foundational technology that transforms raw HTML structures into engaging, responsive, and aesthetically pleasing web interfaces. For a book search and shopping platform, effective CSS usage ensures a seamless experience that supports both functional goals (like easy navigation and checkout) and emotional goals (trust, comfort, and enjoyment).

1. Why CSS Styling Matters in E-commerce Websites

When users land on your site, the first thing they notice is how it looks and feels. Clean, well-structured, and visually appealing interfaces significantly improve user trust, navigation, and engagement.

Whether it's a cart, contact form, or registration page, proper styling with CSS margins, paddings, spacing, input design, and color schemes:

- Makes the content easier to read
- Provides a sense of structure and flow
- Enhances accessibility and user experience (UX)
- Encourages actions like completing a purchase, registering, or submitting a form

Page-wise CSS Styling Theory

1. Cart Page

The cart page is where users review their selected products before checking out, so it needs to be clear, clean, and action-oriented.

Key Styling Techniques:

- Add padding around each cart item for separation
- Use margins to space out product name, quantity input, price, and "remove" button
- Style input fields (quantity, update buttons) with soft borders and enough clickable area
- Highlight the total amount with a bold font and distinct background
- Use consistent font sizes and spacing for price breakdown and tax summaries

Result: A structured layout that minimizes confusion and maximizes conversion.

2. About Us Page

This page tells your brand's story, builds credibility, and helps users connect emotionally with your mission.

Key Styling Techniques:

- Use line height, padding, and justified alignment for readability
- Add white space between sections like “Our Story,” “Our Mission,” and “Our Team”
- Use subtle background colors or separator lines for each section
- Style images (e.g., founders/team) with rounded borders and spacing
- Highlight values or mission using boxes, grids, or quote-styling

Result: A professional and inviting presentation that builds trust.

3. Contact Page

Your contact page should make it effortless for users to reach out for support, queries, or feedback.

Key Styling Techniques:

- Style input fields with equal width, padding, and soft border-radius
- Use margin-bottom to separate form fields
- Provide visual feedback on focus (e.g., border color change)
- Add submit button styling for emphasis (hover effects, background color)
- Layout the form centrally with balanced padding on all sides

Result: A visually appealing and accessible form that encourages engagement.

4. Admin/User Registration Form

This form is critical for onboarding new users/admins, and should feel secure and easy to use.

Key Styling Techniques:

- Organize input fields in logical groups (e.g., personal info, password)
- Add labels and placeholders for clarity
- Use consistent input sizes, padding, and spacing
- Style the form card with shadows, rounded borders, and a light background
- Include hover effects for buttons and inline validation messages

Result: An intuitive form that encourages complete and accurate registration.

5. Admin/User Login Form

Login forms should be quick to use, visually balanced, and provide immediate clarity for mistakes.

Key Styling Techniques:

- Center the login form on the page
- Add sufficient padding inside the form container
- Style input fields with enough spacing and highlight on focus
- Use subtle background colors or semi-transparent overlays
- Style error messages in red and success in green
- Provide clear visual hierarchy (larger font for "Login", smaller for "Forgot Password?")

Result: A clean and efficient login interface that builds user confidence.

Code:

cart page:

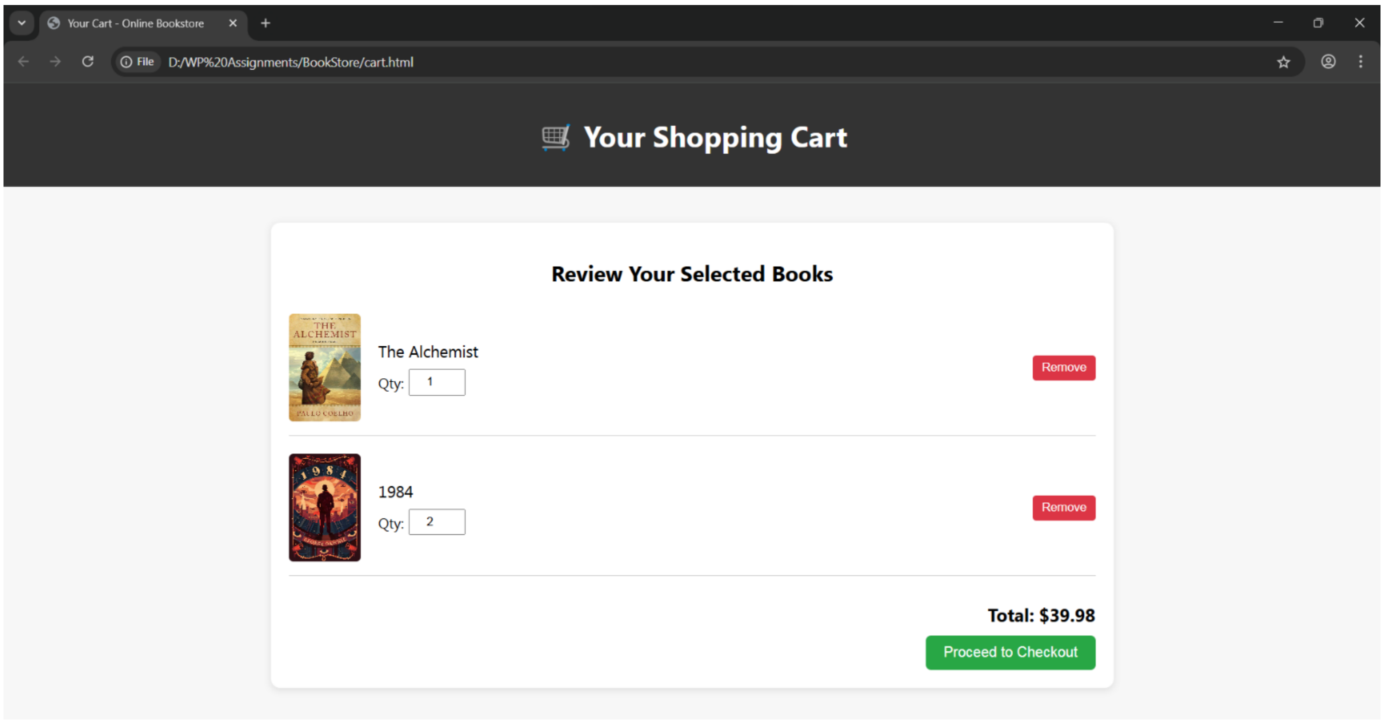
code:

```
<!DOCTYPE html>

<html>

<head>

    <title>Shopping Cart</title>

    <style>

        body {

            font-family: Arial, sans-serif;

            background-color: #f4f4f4;

            text-align: center;

        }

        .cart-container {

            background-color: white;

            padding: 20px;

            margin: 50px auto;

            width: 50%;

            border-radius: 8px;

            box-shadow: 0px 0px 10px rgba(0, 0, 0, 0.1);

        }

        .cart-items {

            list-style: none;

            padding: 0;

        }

        .cart-items li {

            display: flex;

            justify-content: space-between;
            padding: 10px;
            border-bottom: 1px solid #ddd;
        }
        .remove-item {
            background-color: red;
            color: white;
            border: none;
            padding: 5px 10px;
            cursor: pointer;
            border-radius: 5px;
        }
        .remove-item:hover {
            background-color: darkred;
        }
        .checkout-btn, .home-btn {
            display: block;
            margin: 10px auto;
            padding: 10px 15px;
            border: none;
            color: white;
            cursor: pointer;
            border-radius: 5px;
            width: 200px;
        }
        .checkout-btn {
            background-color: #007bff;
        }
        .checkout-btn:hover {
            background-color: #0056b3;
        }
```

```
        .home-btn {
            background-color: #28a745;
        }
        .home-btn:hover {
            background-color: #218838;
        }
    </style>
</head>
<body>
    <div class="cart-container">
        <h2>Your Shopping Cart</h2>
        <ul id="cart-items" class="cart-items"></ul>
        <p id="total-price">Total Price: $0</p>
        <button class="checkout-btn" onclick="checkout()">Proceed to Checkout</button>
        <button class="home-btn" onclick="goHome()">Return to Home</button>
    </div>

    <script>
        let cartItems = JSON.parse(localStorage.getItem('cart')) || [];
        let totalPrice = cartItems.reduce((sum, item) => sum + item.price, 0);

        function renderCart() {
            let cartList = document.getElementById('cart-items');
            let totalPriceElement = document.getElementById('total-price');
            cartList.innerHTML = '';

            cartItems.forEach((item, index) => {
                let li = document.createElement('li');
                li.innerHTML = `${item.name} - $${item.price} <button class="remove-item" onclick="removeItem(${index})">Remove</button>`;
                cartList.appendChild(li);
            });
            totalPriceElement.innerText = `Total Price: $${totalPrice}`;
        }

        function removeItem(index) {
            totalPrice -= cartItems[index].price;
            cartItems.splice(index, 1);
            localStorage.setItem('cart', JSON.stringify(cartItems));
            renderCart();
        }

        function checkout() {
            alert("Proceeding to checkout...");
        }

        function goHome() {
            window.location.href = "homepage.html";  // Redirect to the home page
        }

        renderCart();
    </script>
</body>
</html>
```

<u>Output</u>:

cart page output:


Your Cart - Online Bookstore

D:/WP%20Assignments/BookStore/cart.html

🛒 Your Shopping Cart

Review Your Selected Books

The Alchemist
Qty: 1
Remove

1984
Qty: 2
Remove

Total: $39.98

Proceed to Checkout

<u>Code</u>:

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>Register - Book Haven</title>
  <link rel="stylesheet" href="styles.css" />
  <style>
    body {
      margin: 0;
      font-family: 'Segoe UI', sans-serif;
      background-color: #f4f4f4;
    }

    header, footer {
```

```
  background-color: #333;
  color: white;
  text-align: center;
  padding: 15px;
}

nav a {
  color: white;
  margin: 0 10px;
  text-decoration: none;
}

.register {
  max-width: 600px;
  margin: 50px auto;
  padding: 30px;
  background-color: white;
  border-radius: 8px;
  box-shadow: 0 4px 10px rgba(0,0,0,0.1);
}

.register h2 {
  text-align: center;
  margin-bottom: 30px;
}

form {
  display: flex;
  flex-direction: column;
  gap: 15px;
```

```
}

input, select, textarea {
  padding: 10px;
  font-size: 16px;
  border-radius: 5px;
  border: 1px solid #ccc;
  width: 100%;
}

button {
  padding: 12px;
  font-size: 16px;
  background-color: #007bff;
  color: white;
  border: none;
  border-radius: 6px;
  cursor: pointer;
  transition: background-color 0.3s ease;
}

button:hover {
  background-color: #0056b3;
}

.form-row {
  display: flex;
  gap: 15px;
}
```

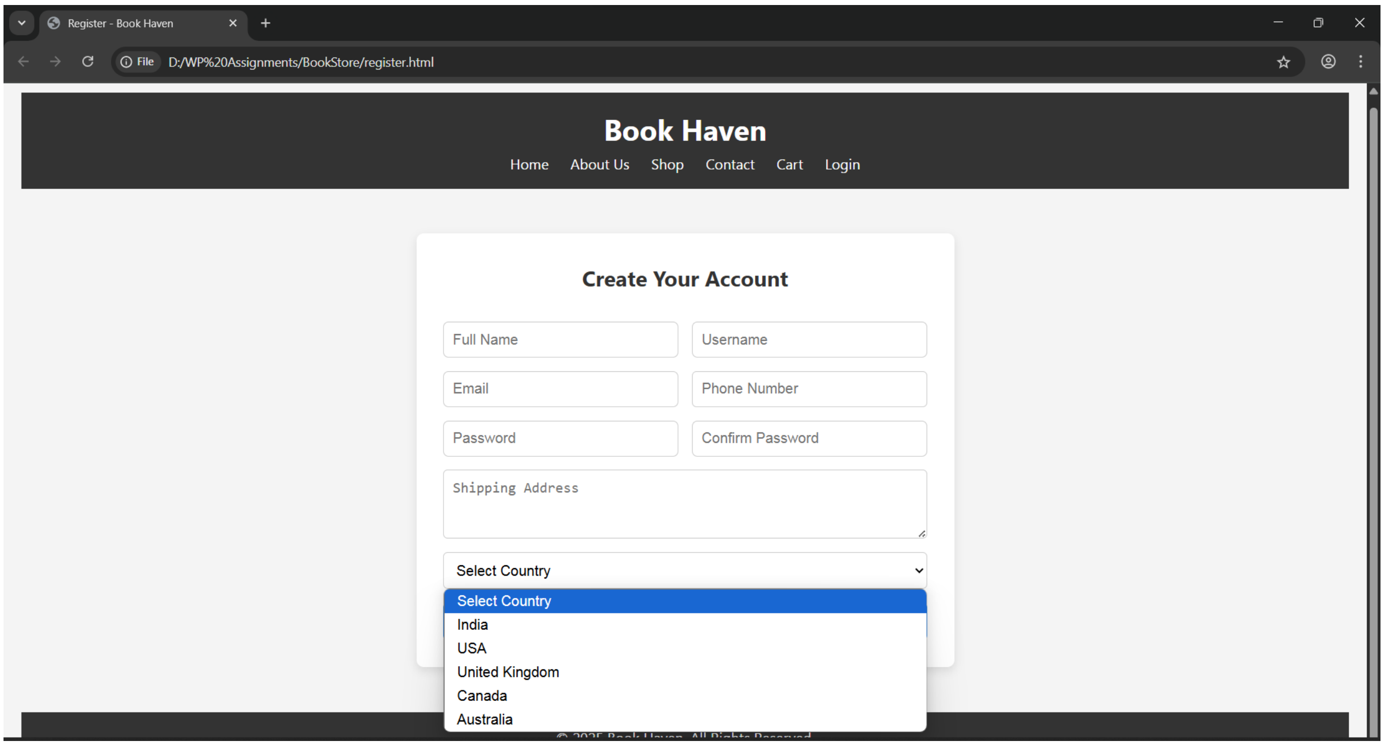
```
    .form-row > div {
      flex: 1;
    }

    textarea {
      resize: vertical;
    }

    footer p {
      margin: 0;
    }
  </style>
</head>
<body>

  <header>
    <div class="logo">
      <h1>Book Haven</h1>
    </div>
    <nav>
      <a href="index.html">Home</a>
      <a href="about.html">About Us</a>
      <a href="menu.html">Shop</a>
      <a href="contact.html">Contact</a>
      <a href="cart.html">Cart</a>
      <a href="login.html">Login</a>
    </nav>
  </header>

  <section class="register">
```

```
<h2>Create Your Account</h2>
<form id="register-form">
  <div class="form-row">
    <div><input type="text" id="name" placeholder="Full Name" required /></div>
    <div><input type="text" id="username" placeholder="Username" required /></div>
  </div>

  <div class="form-row">
    <div><input type="email" id="email" placeholder="Email" required /></div>
    <div><input type="tel" id="phone" placeholder="Phone Number" required /></div>
  </div>

  <div class="form-row">
    <div><input type="password" id="password" placeholder="Password" required /></div>
    <div><input type="password" id="confirm-password" placeholder="Confirm Password" required /></div>
  </div>

  <textarea id="address" rows="3" placeholder="Shipping Address" required></textarea>

  <select id="country" required>
    <option value="">Select Country</option>
    <option value="India">India</option>
    <option value="USA">USA</option>
    <option value="UK">United Kingdom</option>
    <option value="Canada">Canada</option>
    <option value="Australia">Australia</option>
  </select>

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

```
  </section>


  <footer>
    <p>&copy; 2025 Book Haven. All Rights Reserved.</p>
  </footer>


</body>
</html>
```

Output:

registration page output:



Register - Book Haven

D:/WP%20Assignments/BookStore/register.html

Book Haven

Home About Us Shop Contact Cart Login

Create Your Account

Full Name	Username
Email	Phone Number
Password	Confirm Password

Shipping Address

Select Country

- Select Country
- India
- USA
- United Kingdom
- Canada
- Australia

© 2025 Book Haven. All Rights Reserved.

Code:

login page:

code:

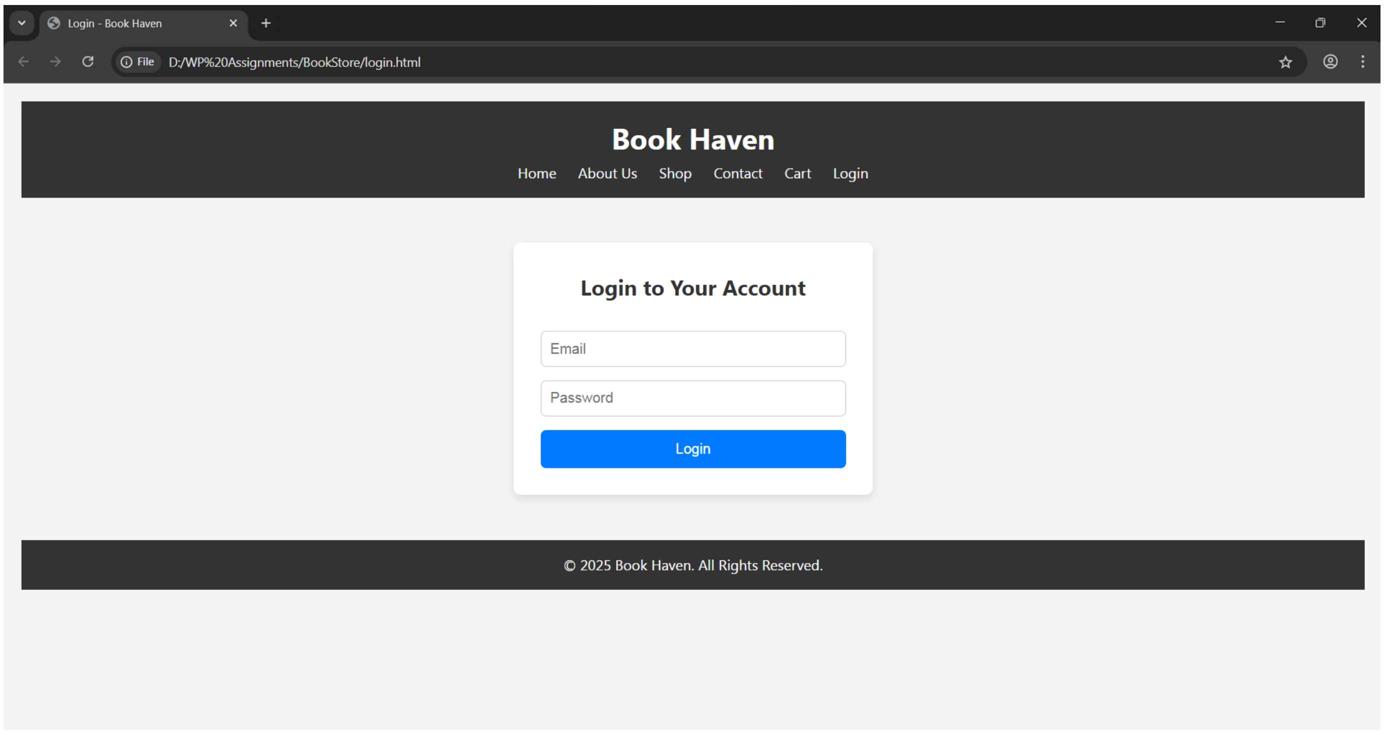
```
<!DOCTYPE html>
<html lang="en">
<head>
```

```
<meta charset="UTF-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0"/>
<title>Login - Book Haven</title>
<link rel="stylesheet" href="styles.css" />
<style>
 body {
  margin: 0;
  font-family: 'Segoe UI', sans-serif;
  background-color: #f4f4f4;
 }

 header, footer {
  background-color: #333;
  color: white;
  text-align: center;
  padding: 15px;
 }

 nav a {
  color: white;
  margin: 0 10px;
  text-decoration: none;
 }

 .login {
  max-width: 400px;
  margin: 50px auto;
  padding: 30px;
  background-color: white;
  border-radius: 8px;
  box-shadow: 0 4px 10px rgba(0,0,0,0.1);
}

.login h2 {
  text-align: center;
  margin-bottom: 30px;
}

form {
  display: flex;
  flex-direction: column;
  gap: 15px;
}

input {
  padding: 10px;
  font-size: 16px;
  border-radius: 5px;
  border: 1px solid #ccc;
  width: 100%;
}

button {
  padding: 12px;
  font-size: 16px;
  background-color: #007bff;
  color: white;
  border: none;
  border-radius: 6px;
  cursor: pointer;
```

```
      transition: background-color 0.3s ease;
    }

    button:hover {
      background-color: #0056b3;
    }

    footer p {
      margin: 0;
    }
  </style>
</head>
<body>

  <header>
    <div class="logo">
      <h1>Book Haven</h1>
    </div>
    <nav>
      <a href="index.html">Home</a>
      <a href="about.html">About Us</a>
      <a href="menu.html">Shop</a>
      <a href="contact.html">Contact</a>
      <a href="cart.html">Cart</a>
      <a href="login.html" class="active">Login</a>
    </nav>
  </header>

  <section class="login">
    <h2>Login to Your Account</h2>
    <form id="login-form">
      <input type="email" name="email" placeholder="Email" required />
      <input type="password" name="password" placeholder="Password" required />
      <button type="submit">Login</button>
    </form>
  </section>

  <footer>
    <p>&copy; 2025 Book Haven. All Rights Reserved.</p>
  </footer>

</body>
</html>
```

<u>Output</u>:

login page output:



Conclusion

The visual and functional success of any e-commerce platform, especially one like your second-hand gaming console website, 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)
- Improved readability and accessibility
- A more polished, professional appearance
- Higher engagement, conversion, and trust

Each page — whether it's the cart, about, contact, or form — serves a critical role in the user's journey. Styling them properly not only improves usability but also communicates quality, attention to detail, and brand identity.

In modern web design, CSS is not just about making things look pretty — it’s about guiding users through a seamless experience, one pixel at a time.

Experiment No.5

Problem Statement: In today's digital era, readers frequently encounter challenges when searching for and purchasing books online due to fragmented platforms, limited filtering options, and inconsistent access to detailed information about books. Many existing websites are not user-friendly and often lack features that support efficient book discovery and smooth purchasing experiences. Furthermore, independent bookstores and self-published authors face difficulties in reaching a wider audience through accessible and streamlined web solutions. There is a clear need for a web-based system that allows users to search for books, explore relevant details, and complete purchases with ease. The proposed Book Search and Shopping web application aims to address this gap by offering a simple, user-cantered platform that integrates intelligent search, comprehensive book information, shopping cart functionality, and a secure checkout process—ultimately enhancing the overall experience for both readers and book sellers**.**

JavaScript Theory: User Registration, Login, Validation, and Cart Functionality JavaScript is a versatile client-side scripting language that plays a crucial role in enhancing interactivity and dynamic behavior in modern web applications. For a book search and shopping website, JavaScript enables core functionalities such as user registration, login authentication, form validation, and real-time shopping cart updates—all essential for delivering a smooth and secure user experience.

1. User Registration and Login

JavaScript is used to handle client-side interactions during the registration and login processes. When a user fills out a registration form, JavaScript can capture the input fields (e.g., name, email, password) and validate them before sending the data to the server. Similarly, during login, JavaScript helps match user-provided credentials with stored data (typically via server or local/session storage in simple apps). Event listeners are attached to the "Submit" buttons to trigger validation or send AJAX requests to backend APIs for secure authentication.

2. Input Validation

Form validation is critical for preventing invalid or insecure data from being submitted. JavaScript allows developers to perform:

- **Syntactic validation**: Ensuring the format of fields like email, password, or phone number using regular expressions.
- **Logical validation**: Ensuring passwords match, required fields are not empty, and terms are accepted. Validation improves both user experience and security by reducing server-side processing of bad data and guiding users to correct errors in real time.

3. Cart Functionality

JavaScript is essential in implementing interactive and persistent shopping cart features:

- **Adding items to the cart**: By clicking an "Add to Cart" button, JavaScript dynamically adds selected books to a cart array stored in memory, localStorage, or sessionStorage.
- **Updating quantities and removing items**: JavaScript allows users to change item quantities or remove books, updating totals and cart content instantly without reloading the page.
- **Calculating totals**: JavaScript loops through the cart items, calculates the subtotal, applies discounts or taxes, and updates the cart UI dynamically.

This functionality ensures a real-time, interactive shopping experience, closely mirroring modern e-commerce platforms.

4. Session Management

While basic implementations may use localStorage or sessionStorage to store user or cart data temporarily, full implementations often use tokens (like JWT) and session IDs to manage logged-in states securely. JavaScript is used to check whether a user is logged in, redirect unauthorized users, or show relevant options (e.g., “Logout” or “Welcome [User]”).

Introduction

In modern web development, client-side scripting using JavaScript is essential for creating interactive, responsive, and user-friendly applications. For an e-commerce website, particularly one focusing on second-hand gaming consoles, implementing registration, login, form validation, and shopping cart functionality is a core requirement to facilitate smooth user engagement and personalized services.

1. User Registration and Login Forms

These forms are critical for establishing user identity and enabling personalized user experiences. JavaScript is used to enhance the responsiveness and usability of these forms before the data is submitted to the server or stored locally in a prototype.

Registration Form

The registration form allows new users to create an account by entering their personal details. This form typically includes fields like full name, email address, password, confirm password, and optionally phone number or address.

Key responsibilities of JavaScript in registration:

- Ensuring that no field is left empty
- Verifying the validity of the email using regular expressions
- Checking that the password meets certain criteria (e.g., minimum length, use of special characters)
- Validating that both password and confirm password fields match
- Providing real-time feedback to the user in case of errors

Login Form

The login form allows returning users to access their accounts using their email and password.

Key responsibilities of JavaScript in login:

- Ensuring that the email and password fields are not empty
- Validating the format of the email address
- Matching the input credentials with previously registered data (locally or via backend)
- Redirecting the user to a dashboard or main page upon successful authentication

2. JavaScript Form Validations

Form validation ensures the accuracy and completeness of user input. It is crucial for data integrity and a better user experience.

Typical validation tasks include:

- Ensuring all mandatory fields are filled
- Validating email address formats using regular expressions
- Verifying password strength (length, characters, etc.)
- Checking that passwords match
- Displaying inline error messages when incorrect input is detected

Client-side validation is often complemented by server-side validation for enhanced security, but using JavaScript provides immediate feedback and reduces unnecessary server requests.

3. Cart Functionality

The shopping cart is an essential component of any e-commerce website. It allows users to review their selections, modify quantities, and proceed to checkout.

Key JavaScript implementations for the cart include:

- Adding selected products to the cart dynamically
- Updating the quantity of items and recalculating totals
- Removing items from the cart
- Storing the cart state in local storage or session storage for persistence
- Rendering cart items in real-time using dynamic DOM manipulation

By maintaining the cart structure as an array of objects in JavaScript, developers can efficiently manage item details, prices, and totals.

<u>Code</u>:

F. registration page:

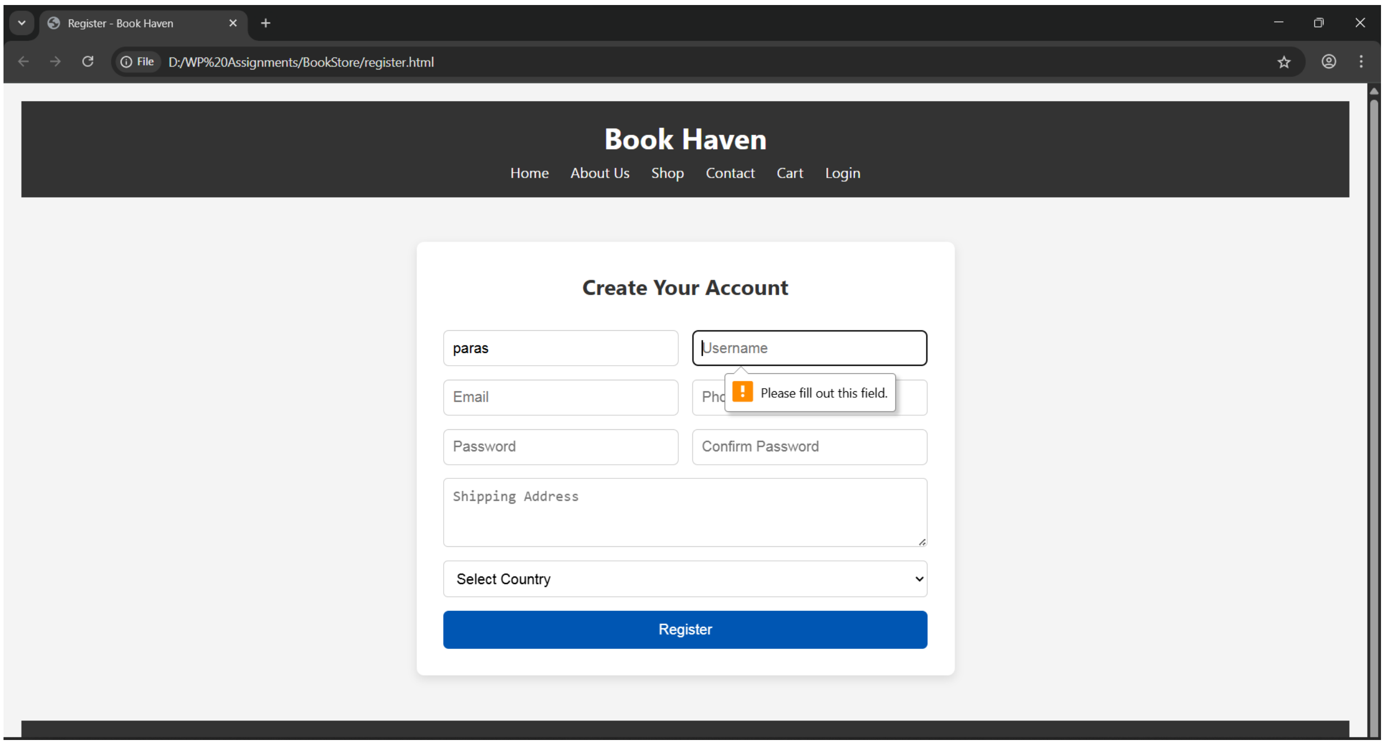
code:

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8" />

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

  <title>Register - Book Haven</title>

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

```
<style>
 body {
  margin: 0;
  font-family: 'Segoe UI', sans-serif;
  background-color: #f4f4f4;
 }

 header, footer {
  background-color: #333;
  color: white;
  text-align: center;
  padding: 15px;
 }

 nav a {
  color: white;
  margin: 0 10px;
  text-decoration: none;
 }

 .register {
  max-width: 600px;
  margin: 50px auto;
  padding: 30px;
  background-color: white;
  border-radius: 8px;
  box-shadow: 0 4px 10px rgba(0,0,0,0.1);
 }

 .register h2 {
  text-align: center;
  margin-bottom: 30px;
}

form {
  display: flex;
  flex-direction: column;
  gap: 15px;
}

input, select, textarea {
  padding: 10px;
  font-size: 16px;
  border-radius: 5px;
  border: 1px solid #ccc;
  width: 100%;
}

button {
  padding: 12px;
  font-size: 16px;
  background-color: #007bff;
  color: white;
  border: none;
  border-radius: 6px;
  cursor: pointer;
  transition: background-color 0.3s ease;
}

button:hover {
      background-color: #0056b3;
    }

    .form-row {
      display: flex;
      gap: 15px;
    }

    .form-row > div {
      flex: 1;
    }

    textarea {
      resize: vertical;
    }

    footer p {
      margin: 0;
    }
  </style>
</head>
<body>

  <header>
    <div class="logo">
      <h1>Book Haven</h1>
    </div>
    <nav>
      <a href="index.html">Home</a>
      <a href="about.html">About Us</a>
    <a href="menu.html">Shop</a>
    <a href="contact.html">Contact</a>
    <a href="cart.html">Cart</a>
    <a href="login.html">Login</a>
  </nav>
</header>

<section class="register">
  <h2>Create Your Account</h2>
  <form id="register-form">
    <div class="form-row">
      <div><input type="text" id="name" placeholder="Full Name" required /></div>
      <div><input type="text" id="username" placeholder="Username" required /></div>
    </div>

    <div class="form-row">
      <div><input type="email" id="email" placeholder="Email" required /></div>
      <div><input type="tel" id="phone" placeholder="Phone Number" required /></div>
    </div>

    <div class="form-row">
      <div><input type="password" id="password" placeholder="Password" required /></div>
      <div><input type="password" id="confirm-password" placeholder="Confirm Password" required /></div>
    </div>

    <textarea id="address" rows="3" placeholder="Shipping Address" required></textarea>

    <select id="country" required>
      <option value="">Select Country</option>
      <option value="India">India</option>
    <option value="USA">USA</option>

    <option value="UK">United Kingdom</option>

    <option value="Canada">Canada</option>

    <option value="Australia">Australia</option>

   </select>


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

  </form>

 </section>


 <footer>

  <p>&copy; 2025 Book Haven. All Rights Reserved.</p>

 </footer>


</body>

</html>
```

<u>Output</u>:

F. registration page output:



Register - Book Haven

D:/WP%20Assignments/BookStore/register.html

Book Haven

Home About Us Shop Contact Cart Login

Create Your Account

paras

Username

Email

Pho

Please fill out this field.

Password

Confirm Password

Shipping Address

Select Country

Register

<u>Code</u>:

G. login page:

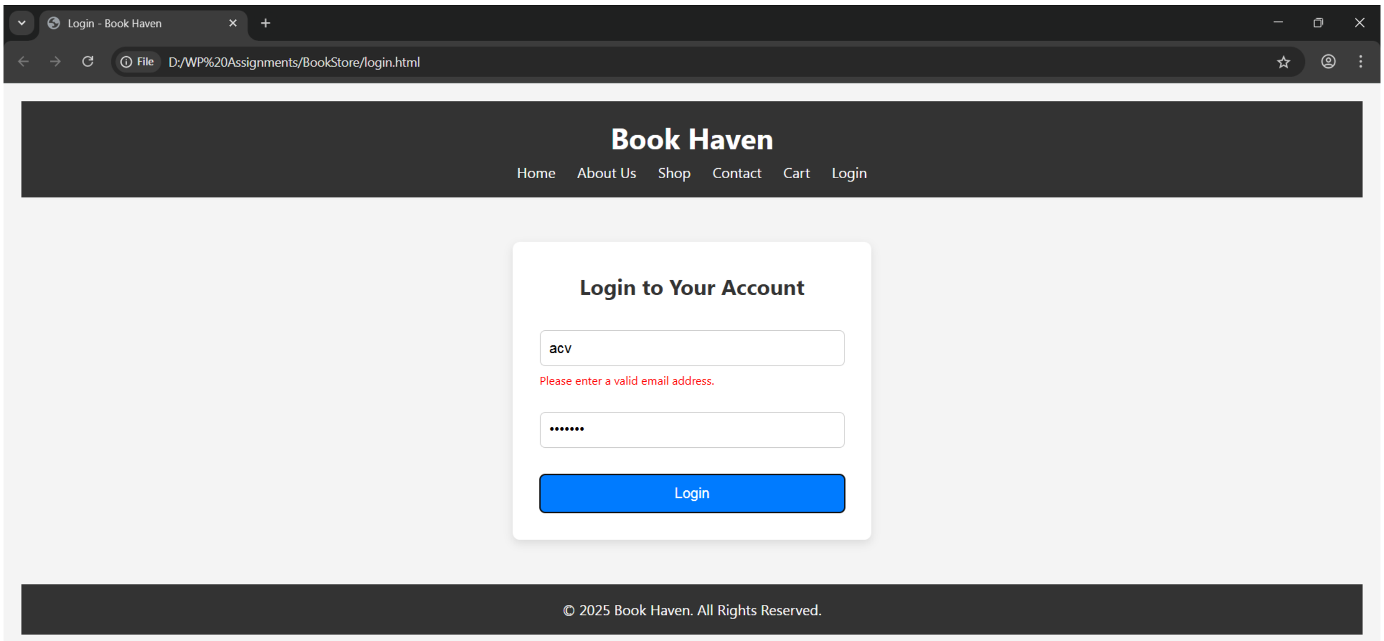
code:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
  <title>Login - Book Haven</title>
  <link rel="stylesheet" href="styles.css" />
  <style>
    body {
      margin: 0;
      font-family: 'Segoe UI', sans-serif;
      background-color: #f4f4f4;
    }

    header, footer {
      background-color: #333;
      color: white;
      text-align: center;
      padding: 15px;
    }

    nav a {
      color: white;
      margin: 0 10px;
      text-decoration: none;
    }

    .login {
      max-width: 400px;
      margin: 50px auto;
      padding: 30px;
      background-color: white;
      border-radius: 8px;
      box-shadow: 0 4px 10px rgba(0,0,0,0.1);
    }

    .login h2 {
      text-align: center;
      margin-bottom: 30px;
    }

    form {
      display: flex;
      flex-direction: column;
      gap: 15px;
    }

    input {
      padding: 10px;
      font-size: 16px;
      border-radius: 5px;
      border: 1px solid #ccc;
      width: 100%;
    }
```

```
    .error {
      color: red;
      font-size: 13px;
      margin-top: -10px;
      margin-bottom: 10px;
    }

    button {
      padding: 12px;
      font-size: 16px;
      background-color: #007bff;
      color: white;
      border: none;
      border-radius: 6px;
      cursor: pointer;
      transition: background-color 0.3s ease;
    }

    button:hover {
      background-color: #0056b3;
    }

    footer p {
      margin: 0;
    }
  </style>
</head>
<body>

  <header>
    <div class="logo">
      <h1>Book Haven</h1>
    </div>
    <nav>
      <a href="index.html">Home</a>
      <a href="about.html">About Us</a>
      <a href="menu.html">Shop</a>
      <a href="contact.html">Contact</a>
      <a href="cart.html">Cart</a>
      <a href="login.html" class="active">Login</a>
    </nav>
  </header>

  <section class="login">
    <h2>Login to Your Account</h2>
    <form id="login-form" novalidate>
      <input type="email" id="email" placeholder="Email" />
      <p class="error" id="emailError"></p>
      <input type="password" id="password" placeholder="Password" />
      <p class="error" id="passwordError"></p>
      <button type="submit">Login</button>
    </form>
  </section>

  <footer>
    <p>&copy; 2025 Book Haven. All Rights Reserved.</p>
  </footer>

  <script>
    const form = document.getElementById("login-form");
  const email = document.getElementById("email");

  const password = document.getElementById("password");

  const emailError = document.getElementById("emailError");

  const passwordError = document.getElementById("passwordError");


  form.addEventListener("submit", function (e) {

    e.preventDefault();

    let valid = true;


    // Email validation

    if (!email.value.trim()) {

      emailError.textContent = "Email is required.";

      valid = false;

    } else if (!/^\S+@\S+\.\S+$/.test(email.value.trim())) {

      emailError.textContent = "Please enter a valid email address.";

      valid = false;

    } else {

      emailError.textContent = "";

    }


    // Password validation

    if (!password.value.trim()) {

      passwordError.textContent = "Password is required.";

      valid = false;

    } else {

      passwordError.textContent = "";

    }


    // If all is valid

    if (valid) {
        alert("Login successful!"); // Replace with actual login handling
        form.reset();
      }
    });
  </script>
</body>
</html>
```

<u>Output</u>:

G. login page output:



Login - Book Haven

D:/WP%20Assignments/BookStore/login.html

Book Haven

Home About Us Shop Contact Cart Login

Login to Your Account

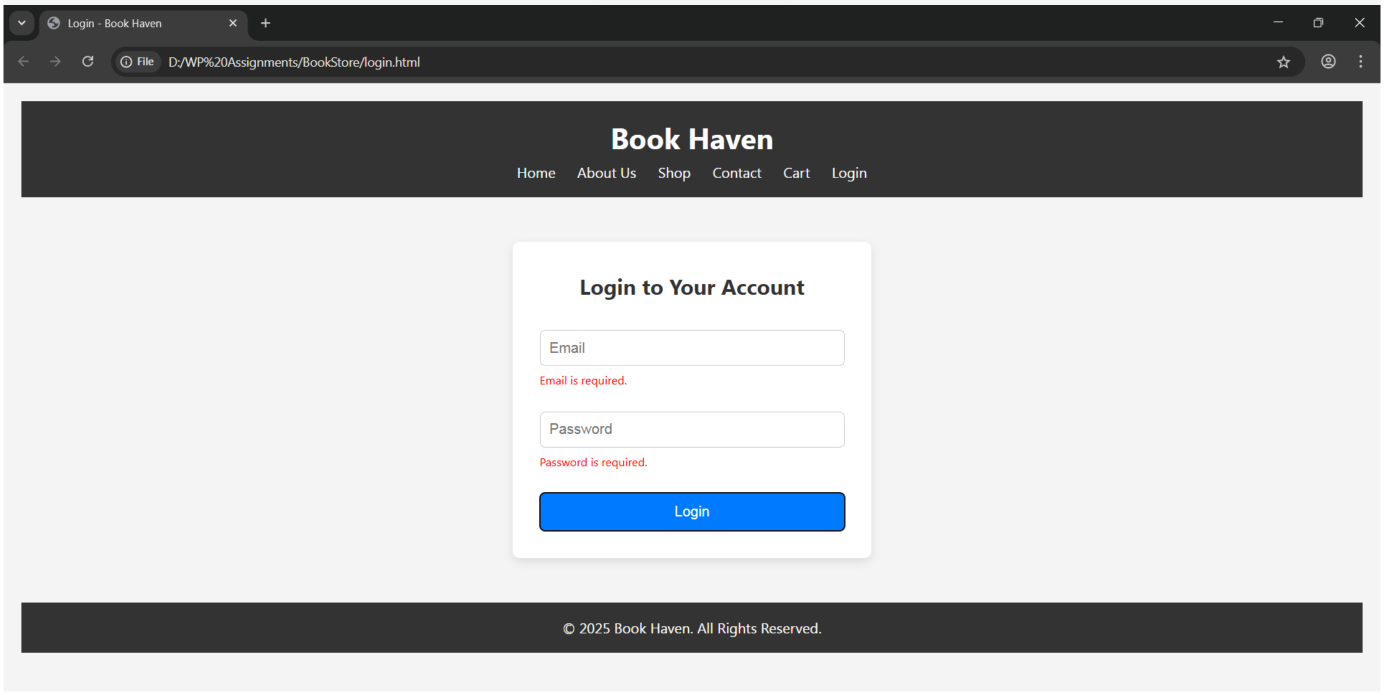
acv

Please enter a valid email address.

•••••••

Login

© 2025 Book Haven. All Rights Reserved.



Login - Book Haven

D:/WP%20Assignments/BookStore/login.html

Book Haven

Home About Us Shop Contact Cart Login

Login to Your Account

Email

Email is required.

Password

Password is required.

Login

© 2025 Book Haven. All Rights Reserved.

Conclusion

Implementing registration, login, validation, and cart features using JavaScript is fundamental for any user-centered e-commerce platform. These features not only enhance usability but also ensure smooth interactions, better data handling, and a seamless user journey.

For a second-hand gaming console website, these JavaScript functionalities provide the foundation for user management and interactive shopping experiences. Users can securely register and log in, receive immediate validation feedback, and manage their cart efficiently. This setup lays the groundwork for integrating advanced features like order history, wish lists, and secure checkout in future iterations.

JavaScript, therefore, plays a critical role in transforming a static product listing site into a dynamic and functional e-commerce platform.

Experiment No.6

Problem Statement: In today's digital era, readers frequently encounter challenges when searching for and purchasing books online due to fragmented platforms, limited filtering options, and inconsistent access to detailed information about books. Many existing websites are not user-friendly and often lack features that support efficient book discovery and smooth purchasing experiences. Furthermore, independent bookstores and self-published authors face difficulties in reaching a wider audience through accessible and streamlined web solutions. There is a clear need for a web-based system that allows users to search for books, explore relevant details, and complete purchases with ease. The proposed Book Search and Shopping web application aims to address this gap by offering a simple, user-centered platform that integrates intelligent search, comprehensive book information, shopping cart functionality, and a secure checkout process—ultimately enhancing the overall experience for both readers and book sellers.

JavaScript Theory:

Persistent Login and Cart Functionality using Web Storage API In modern web applications, maintaining session state across browser reloads and visits is essential for a seamless user experience. JavaScript, along with the **Web Storage API**, provides a powerful solution for achieving **persistent login** and **cart functionality** without requiring a backend database in basic applications. The two key components of the Web Storage API are localStorage and sessionStorage, which allow developers to store key-value pairs in a user's browser.

Persistent Login

Persistent login ensures that users remain logged in even after refreshing or closing the browser. This is achieved by storing authentication data—such as a username or token—in localStorage. Upon every page load, JavaScript checks the storage and automatically restores the session if valid credentials are found.

How it works:

- When a user logs in successfully, JavaScript saves their login data (e.g., user ID or token) in localStorage.
- On subsequent visits, the application checks localStorage to determine if the user is already authenticated.
- If so, it updates the UI to reflect the logged-in state and grants access to protected pages or features.

This avoids the need for the user to log in repeatedly, improving convenience and user retention. For more secure applications, session expiration and token-based validation should be added.

Persistent Cart Functionality

Shopping cart data is typically dynamic and user-specific. By using localStorage, developers can ensure that a user's cart persists across browser sessions—even if they close the tab or refresh the page.

How it works:

- When a user adds a book to the cart, JavaScript updates the cart array and saves it as a serialized JSON object in localStorage.
- On page load, JavaScript retrieves the cart data from localStorage, parses it back into an array or object, and repopulates the cart UI.
- When the user removes items or changes quantities, the cart state is updated in real time and synced with the stored data.

This approach ensures a **consistent shopping experience**, even if the user accidentally navigates away from the page or revisits later.

Advantages of Using Web Storage API

- **Simplicity:** Easy to implement and does not require a backend database for small-scale projects.
- **Performance:** Fast read/write operations as the data is stored locally in the browser.
- **Persistence:** localStorage persists data even after the browser is closed, making it ideal for login and cart features.
- **Security:** Data is stored per origin, meaning websites can only access their own data.

Introduction

In modern web applications, offering a seamless user experience requires maintaining user session states and data across different pages or after a page refresh. JavaScript's Web Storage API—comprising localStorage and sessionStorage—is a lightweight solution to store data on the client side. For a second-hand gaming console website, using these features can significantly improve usability by allowing persistent login sessions and retaining cart data even after page reloads or temporary site exits.

1. Persistent Login using localStorage/sessionStorage

The login system allows users to securely enter their credentials (email and password) to gain access to their accounts. Once validated, their login status and user identifier (like email or user ID) are stored in the browser using either:

- localStorage: Stores data with no expiration time, persisting even after the browser is closed and reopened.
- sessionStorage: Stores data only for the duration of the page session (i.e., until the tab or browser is closed).

Implementation Features:

- After successful login, JavaScript stores:
  - userEmail: to identify the current user
  - isLoggedIn: a boolean flag to indicate the login status
- On subsequent visits or page reloads:
  - JavaScript checks for these flags and either redirects to the dashboard or shows the login screen
- Logout functionality clears the stored values, ending the session

Benefits:

- Eliminates the need to re-login on every visit
- Enhances user convenience and session continuity
- Reduces server load for small-scale or prototype apps

2. Cart Data Management using localStorage

Shopping carts are central to any e-commerce website. Users expect that the items they add remain intact even if they leave or refresh the page. localStorage enables this by preserving the state of the cart.

Implementation Features:

- Every time a user adds, removes, or updates a product in the cart:
    - JavaScript serializes the cart array/object into JSON
    - This data is saved to localStorage
- On page load:
    - JavaScript checks if cart data exists in localStorage
    - If it does, it parses and loads it into the cart view
- The cart remains persistent until explicitly cleared

Benefits:

- Prevents loss of user data on reload or accidental tab closure
- Creates a more seamless and intuitive shopping experience
- Ensures continuity across visits without requiring account creation

Use Cases Beyond the Syllabus (Advanced Learning):

These implementations represent concepts often covered beyond standard academic curricula:

- Managing state with client-side storage
- Working with JSON and JavaScript objects dynamically
- Handling user sessions in single-page or multi-page applications without a backend
- Creating realistic e-commerce simulations or prototypes for portfolio projects

Code:

A. Home page:

code:

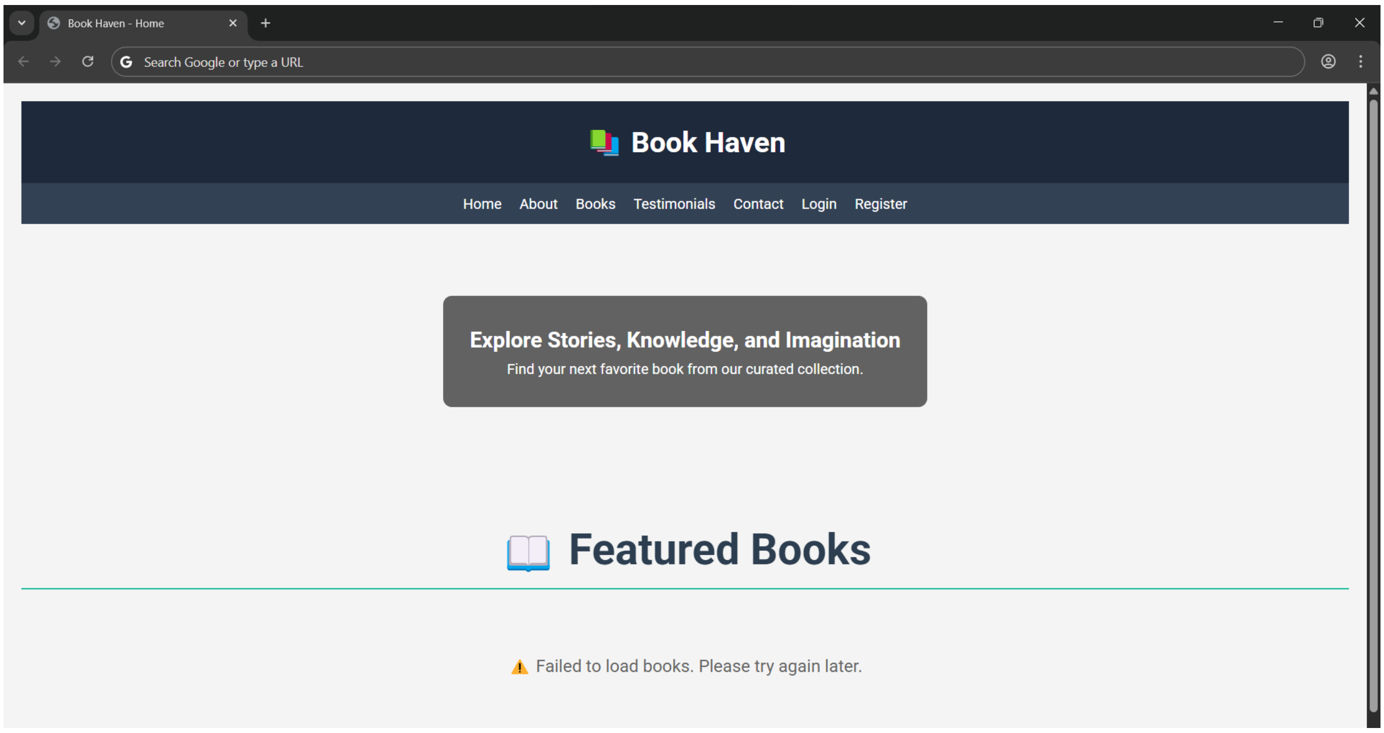
```
<!DOCTYPE html>

<html>

<head>

  <title>Login</title>

  <style>

    body {

      font-family: Arial, sans-serif;
      background-color: #f4f4f4;

      display: flex;

      justify-content: center;

      align-items: center;

      height: 100vh;

    }


    form {

      background-color: white;

      padding: 25px;

      border-radius: 8px;

      box-shadow: 0 0 10px rgba(0,0,0,0.1);

      width: 300px;

    }


    h2 {

      margin-bottom: 15px;

      text-align: center;

    }


    label {

      display: block;

      margin-top: 10px;

      font-weight: bold;

    }


    input.box {

      width: 100%;

      padding: 8px;

      margin-top: 5px;
      border: 1px solid #ccc;
      border-radius: 4px;
    }

    #show-pass {
      margin-top: 10px;
      background: none;
      border: none;
      color: blue;
      cursor: pointer;
      text-decoration: underline;
    }

    #submit-btn {
      margin-top: 15px;
      width: 100%;
      padding: 10px;
      background-color: #333;
      color: white;
      border: none;
      border-radius: 4px;
      cursor: pointer;
      opacity: 0.5;
    }

    #submit-btn:enabled {
      opacity: 1;
    }

    .msg {
```

```
            margin-top: 10px;
            font-weight: bold;
            text-align: center;
        }
    </style>
</head>
<body>
    <form>
        <h2>Login Form</h2>
        <label for="username">User Name</label>
        <input type="text" class="box" placeholder="Enter User name" id="username" name="username">

        <label for="pass">Password</label>
        <input type="password" class="box" placeholder="Enter Password" id="pass" name="pass">

        <button id="show-pass">Show Password</button>

        <input type="submit" id="submit-btn" value="Login" disabled>

        <div class="msg"></div>
    </form>

    <script>
        const submit = document.getElementById('submit-btn');
        const msgElement = document.querySelector('.msg');
        const showPassBtn = document.getElementById('show-pass');
        const usernameInput = document.getElementById('username');
        const passwordInput = document.getElementById('pass');

        const validUser = "OjasUmate";
const validPass = "Ojas@123";

// Enable login button when both fields are filled
usernameInput.addEventListener('input', validateForm);
passwordInput.addEventListener('input', validateForm);

function validateForm() {
    if (usernameInput.value.trim() && passwordInput.value.trim()) {
        submit.disabled = false;
    } else {
        submit.disabled = true;
    }
}

// Toggle password visibility
showPassBtn.addEventListener('click', function (e) {
    e.preventDefault();
    passwordInput.type = passwordInput.type === "password" ? "text" : "password";
    showPassBtn.textContent = passwordInput.type === "password" ? "Show Password" : "Hide Password";
});

// Handle form submission
submit.addEventListener('click', function (e) {
    e.preventDefault();

    let enteredUser = usernameInput.value.trim();
    let enteredPass = passwordInput.value;

    if (enteredUser === validUser && enteredPass === validPass) {
        msgElement.style.color = 'green';
```

```
                msgElement.textContent = 'Successfully logged in';

                localStorage.setItem('userDetails', JSON.stringify({ username: enteredUser }));

                setTimeout(() => {
                    window.location.href = "homepage.html";
                }, 2000);
            } else {
                msgElement.style.color = 'red';
                msgElement.textContent = 'Invalid Username or Password';
            }
        });
    </script>
</body>
</html>
```

Output:

A. Index/Home page output:


Book Haven - Home

Search Google or type a URL

Book Haven

Home About Books Testimonials Contact Login Register

Explore Stories, Knowledge, and Imagination

Find your next favorite book from our curated collection.

Featured Books

Failed to load books. Please try again later.

Conclusion

Using JavaScript in combination with the Web Storage API (localStorage/sessionStorage) significantly enhances user experience and functionality in web development. For a second-hand gaming console website, implementing persistent login and cart functionality ensures that users have a smooth, uninterrupted interaction with the site.

By storing authentication states and cart data locally:

- Users remain logged in across sessions
- Cart items persist across visits
- The website feels more responsive and user-centric

These techniques mimic real-world behavior found in professional e-commerce platforms, making them excellent additions to projects meant for academic distinction or professional portfolios. Ultimately, mastering such features prepares developers to build more dynamic, reliable, and user-friendly web applications.

Experiment no.7

Problem Statement: In today’s digital era, readers frequently encounter challenges when searching for and purchasing books online due to fragmented platforms, limited filtering options, and inconsistent access to detailed information about books. Many existing websites are not user-friendly and often lack features that support efficient book discovery and smooth purchasing experiences. Furthermore, independent bookstores and self-published authors face difficulties in reaching a wider audience through accessible and streamlined web solutions. There is a clear need for a web-based system that allows users to search for books, explore relevant details, and complete purchases with ease. The proposed Book Search and Shopping web application aims to address this gap by offering a simple, user-centered platform that integrates intelligent search, comprehensive book information, shopping cart functionality, and a secure checkout process—ultimately enhancing the overall experience for both readers and book sellers**.**

Theory:

User registration is a fundamental component of any dynamic web application, including an online Coffee Shop website. It allows users to create personalized accounts, save preferences, and perform secure transactions. Using PHP (Hypertext Preprocessor), developers can build server-side scripts that securely process form input, validate data, store user credentials in a database, and provide appropriate feedback to users.

A. Handling User Registration Input with PHP

To begin, a PHP script is created to handle data submitted through an HTML registration form. The form typically includes fields such as:

- **Full Name**
- **Email Address**
- **Password**
- **Confirm Password**

The $_POST superglobal array is used in PHP to capture the submitted data. Upon form submission, the script processes this input and prepares it for validation and storage. Before storing data, it is essential to sanitize inputs using functions like htmlspecialchars() or filter_var() to prevent injection attacks and ensure data integrity.

The password must be hashed using PHP’s password_hash() function before being stored in the database, ensuring that plaintext passwords are never saved.

B. Implementing Error Handling and Validation

Validation ensures that user input meets the required criteria. This includes:

- **Ensuring no field is empty**
- **Verifying email format using filter_var($email, FILTER_VALIDATE_EMAIL)**
- **Checking password length and match with the confirmation field**
- **Checking whether the email is already registered (via a SELECT query in the database)**

If any of these checks fail, the PHP script returns appropriate error messages to the user. These errors are often displayed next to the relevant form fields or collectively at the top of the page using PHP conditional blocks.

Additionally, error handling can be extended to manage database issues (e.g., connection failure or duplicate entries), using try-catch blocks or checking for query success using mysqli_query() or PDO::execute().

C. Providing Feedback on Successful Registration

Once the input is validated and successfully stored in the database, the user should receive confirmation. This can be done in two ways:

1. **Displaying a success message on the same page.**
2. **Redirecting the user to a login page using header("Location: login.php") with an optional success message passed via URL or session.**

This final feedback loop enhances user experience and confirms that their data has been safely and correctly stored. The use of session variables can also enable temporary success messages using $_SESSION['message'].

A. Develop a PHP script to handle user registration for the Coffee Shop website. The script should accept input from users for their name, email address, password, etc. (all required fields for registration).
B. Implement error handling to notify users of any issues during registration, such as validation errors.
C. Provide feedback to the user upon successful registration, either through a confirmation message or a redirect to a login page.

User registration is a fundamental component of web applications, particularly in e-commerce platforms like your second-hand gaming console website. PHP is widely used on the server side to handle form submissions, validate user inputs, interact with databases (like MySQL), and ensure secure data processing.

In this system, the registration form captures user details (name, email, password, etc.). Once submitted, the PHP script validates the inputs and then stores them securely into a database. To maintain security, user passwords are hashed before storage.

Core Elements of the PHP Registration Script:

1. Form Handling: Grabs data using $_POST.
2. Validation: Ensures fields are not empty and email is valid.
3. Password Hashing: Uses password_hash() to securely hash passwords.
4. Database Interaction: Uses MySQLi or PDO to store user data.
5. Error Handling: Displays messages for missing fields or registration failures.
6. User Feedback: Provides confirmation or redirection upon success.

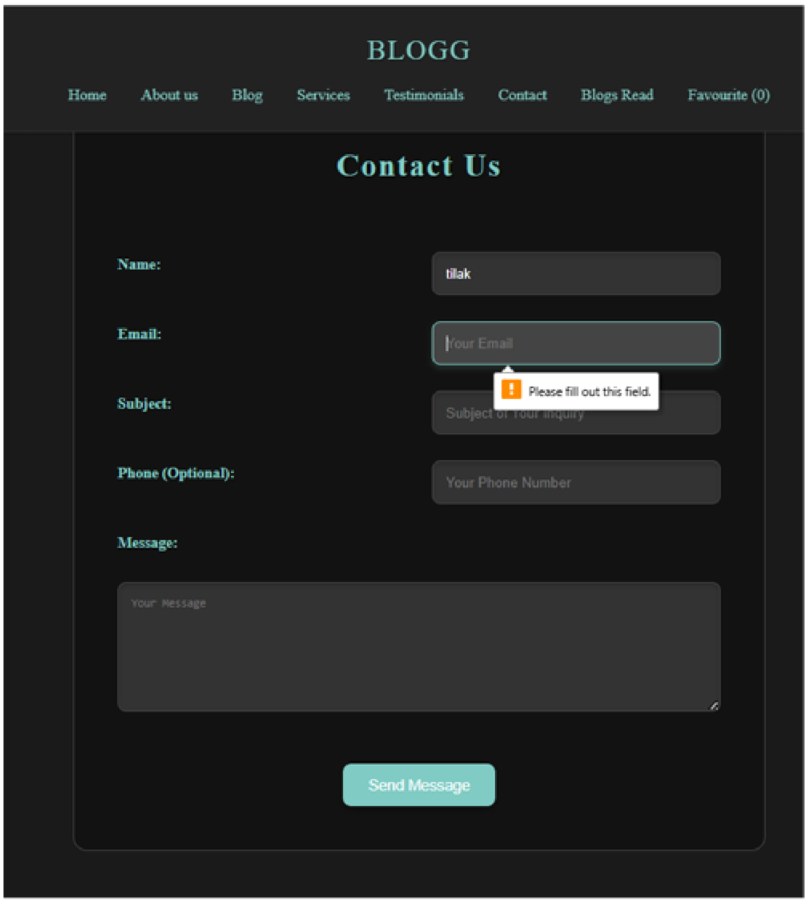
CODE:-

```
<?php
// db_connect.php (include this file wherever needed)
$host = 'localhost';
$user = 'root';
$password = '';
$dbname = 'gaming_store';

$conn = new mysqli($host, $user, $password, $dbname);

if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
?>
```

Registration:-

```
<?php
include 'db_connect.php';

$name = $email = $password = "";
$errors = [];

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // Get input values and sanitize
    $name = trim($_POST["name"]);
    $email = trim($_POST["email"]);
    $password = trim($_POST["password"]);

    // Basic validation
    if (empty($name)) $errors[] = "Name is required.";

    if (empty($email) || !filter_var($email, FILTER_VALIDATE_EMAIL)) $errors[] = "Valid email is required.";

    if (empty($password) || strlen($password) < 6) $errors[] = "Password must be at least 6 characters.";


    // If no errors, proceed to store user

    if (empty($errors)) {

        $hashedPassword = password_hash($password, PASSWORD_BCRYPT);


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

        $stmt->bind_param("sss", $name, $email, $hashedPassword);


        if ($stmt->execute()) {

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

        } else {

            echo "<p>Error: " . $stmt->error . "</p>";

        }


        $stmt->close();

    } else {

        foreach ($errors as $error) {

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

        }

    }


    $conn->close();

}

?>
```

Output:



BLOGG

Contact Us

Name: tilak

Email: Your Email

Please fill out this field.

Subject: Subject of Your Inquiry

Phone (Optional): Your Phone Number

Message: Your Message

Conclusion

Implementing user registration with PHP provides the backbone of user management in your website. By securely collecting, validating, and storing user data, you enable personalized experiences and functionalities such as login, saving favorites, or viewing past orders.

This system:

- Promotes user trust by securing sensitive data like passwords.
- Ensures data integrity through server-side validation.
- Enhances the user experience with real-time feedback and clear error handling.

Experiment 8

Problem Statement: In today's digital era, readers frequently encounter challenges when searching for and purchasing books online due to fragmented platforms, limited filtering options, and inconsistent access to detailed information about books. Many existing websites are not user-friendly and often lack features that support efficient book discovery and smooth purchasing experiences. Furthermore, independent bookstores and self-published authors face difficulties in reaching a wider audience through accessible and streamlined web solutions. There is a clear need for a web-based system that allows users to search for books, explore relevant details, and complete purchases with ease. The proposed Book Search and Shopping web application aims to address this gap by offering a simple, user-centered platform that integrates intelligent search, comprehensive book information, shopping cart functionality, and a secure checkout process—ultimately enhancing the overall experience for both readers and book sellers**.**

Theory: User login is a core feature in dynamic web applications that enables secure access to personalized features. For a Coffee Shop website, a login system allows registered users to manage their profiles, browse personalized offers, and place orders securely. PHP, being a powerful server-side scripting language, facilitates the creation of secure and user-friendly login mechanisms.

A. Developing a PHP Script to Handle User Login

The login system begins with a user-facing HTML form where the user enters:

- **Email Address** or **Username**
- **Password**

Upon submission, PHP captures these values using the $_POST superglobal. The script then checks whether the input fields are filled and sanitizes them using functions like trim() and htmlspecialchars() to prevent injection attacks.

The login PHP script queries the database to retrieve the stored hashed password associated with the entered email or username. The built-in password_verify() function is used to compare the entered password with the stored hash securely.

B. Providing Feedback on Successful Login

Once the credentials are validated, and authentication is successful, PHP provides feedback in one of two ways:

1. **Display a success message**, such as "Login Successful, Welcome!"
2. **Redirect the user** to a dashboard or homepage using:

php

Copy code

```
header("Location: home.php");

exit();
```

This ensures the user knows the login attempt was successful and provides immediate access to personalized features on the site.

C. Implementing Error Handling for Login Failures

Robust error handling is crucial for informing users of unsuccessful login attempts without revealing sensitive information. Common login errors include:

- **Incorrect email/username**
- **Incorrect password**
- **Blank fields**

The PHP script checks these conditions and provides user-friendly messages like:

- "Incorrect email or password."
- "All fields are required."
- "Account not found."

If the database connection fails or other technical errors occur, the script can use try-catch blocks (in PDO) or check connection responses (in MySQLi) to display a generic error message, helping the user understand that something went wrong.

D. Providing Personalized Feedback or Redirection Upon Login

After a successful login, PHP can:

- **Start a session** using session_start()
- Store user information in session variables, such as:

php

Copy code

```
$_SESSION['username'] = $username;
```

This allows the application to display a personalized message like:

php

Copy code

```
echo "Welcome, " . $_SESSION['username'] . "!";
```

Alternatively, the script can **redirect the user to the homepage** with a welcome message displayed there using stored session data.

A. Develop a PHP script to handle user login for the Coffee Shop website. The script should accept input from users for their login credentials. (all required fields for login).
B. Provide feedback to the user upon successful login, either through a confirmation message or a redirect to a welcome page.
C. Implement error handling to notify users of login failures due to incorrect credentials or other errors.

D. Provide feedback to the user upon successful login, either through a welcome user name message or a redirect to a home page.

Theory: PHP Login System

A user login system is a fundamental component of most websites, especially e-commerce platforms. It enables secure access to personalized features like managing carts, tracking orders, or viewing saved products. In PHP, login functionality typically involves:

- Capturing login credentials via a form (email and password).
- Validating inputs.
- Comparing credentials against stored data in a database.
- Starting a session upon successful login.
- Redirecting or displaying a welcome message.
- Showing errors for invalid credentials.

Security Aspects:

- **Password Hashing & Verification**: Passwords are stored as hashes using password_hash() during registration. PHP's password_verify() is used to compare hashes during login.
- **Session Handling**: PHP sessions are used to maintain the user's login state across pages.

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:-

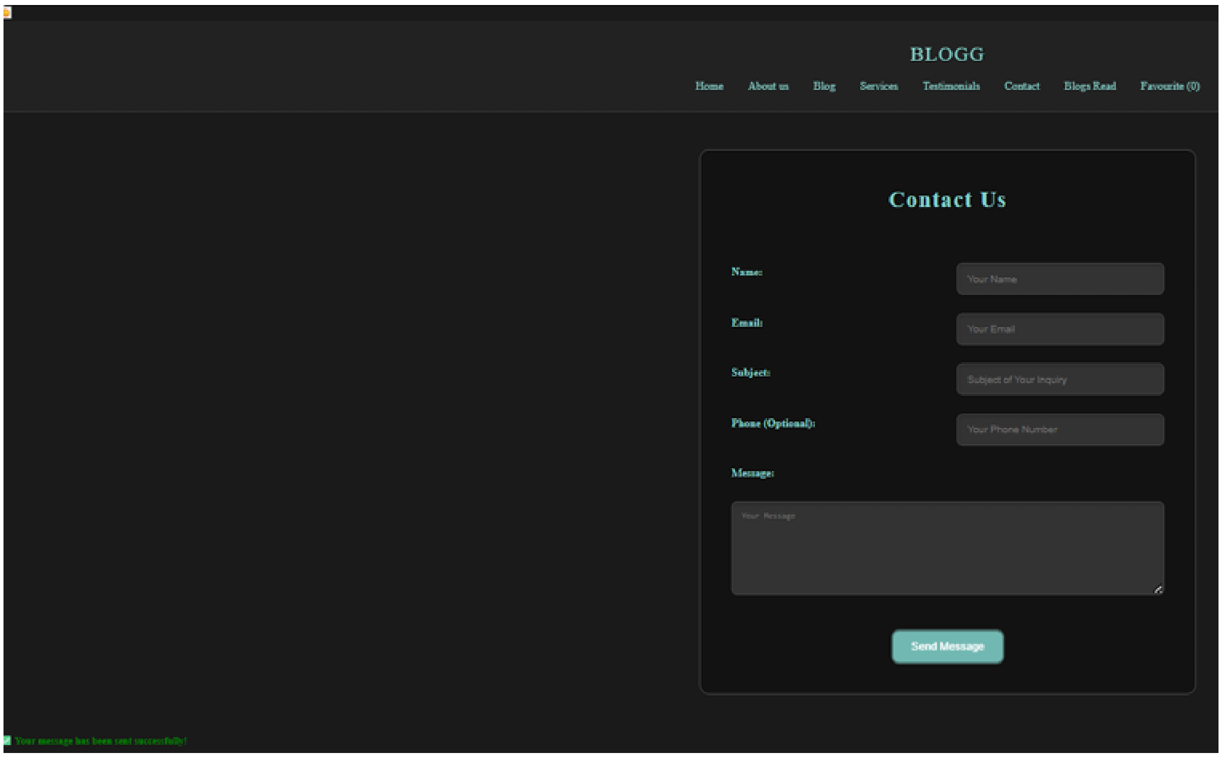
```
<form action="login.php" method="POST">
    <h2>Login</h2>

    <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>
```

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:



✅ Your message has been sent successfully!

Conclusion

Implementing a login system with PHP ensures a secure and user-friendly experience for your second-hand gaming console website. By validating input, securely verifying credentials, and using PHP sessions:

- You allow users to access personalized features.
- You prevent unauthorized access.
- You improve user engagement and trust.

This login system:

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

Experiment 9

Problem Statement: In today's digital era, readers frequently encounter challenges when searching for and purchasing books online due to fragmented platforms, limited filtering options, and inconsistent access to detailed information about books. Many existing websites are not user-friendly and often lack features that support efficient book discovery and smooth purchasing experiences. Furthermore, independent bookstores and self-published authors face difficulties in reaching a wider audience through accessible and streamlined web solutions. There is a clear need for a web-based system that allows users to search for books, explore relevant details, and complete purchases with ease. The proposed Book Search and Shopping web application aims to address this gap by offering a simple, user-centered platform that integrates intelligent search, comprehensive book information, shopping cart functionality, and a secure checkout process—ultimately enhancing the overall experience for both readers and book sellers**.**

Theory: A shopping cart system is a critical component of any e-commerce platform, providing users with the ability to select, review, and manage products before purchase. In the context of a second-hand gaming consoles website, a shopping cart allows users to temporarily store selected consoles, compare options, and prepare for checkout. PHP, as a server-side scripting language, is well-suited to handle such functionalities efficiently, both with and without the use of a backend database like MySQL.

A. Session-Based Shopping Cart Using PHP

For simpler applications, PHP's built-in session management allows for implementing a shopping cart without the need for a database. Each user's cart is stored in the $_SESSION superglobal array.

Advantages:

- **Quick to implement**
- **No need for a database**
- **Cart persists during the session**

Limitations:

- **Cart is lost when the session ends or user logs out**
- **Not scalable or persistent across devices**

B. Database-Persistent Cart Using PHP and MySQL

For robust and persistent cart systems—especially those that must survive user logouts or browser closures—MySQL integration is essential.

Key Components:

1. **MySQL Tables:**

- **users – to identify and associate carts with individual accounts**
- **products – storing console details**
- **cart – storing cart data (user_id, product_id, quantity, added_on)**

2. **Add to Cart:**
   When a user adds a console to the cart, a PHP script inserts or updates a record in the cart table based on whether the item already exists.
3. **View Cart:**
   A SELECT query with JOINs retrieves product details (name, price, etc.) using product_id from the cart table.
4. **Remove Items:**
   PHP executes a DELETE query to remove the selected item from the user's cart in the database.

Benefits of MySQL-Backed Cart:

- **Data persists across user sessions and devices**
- **Suitable for registered users with login systems**
- **Scalable for larger applications**

Security Considerations:

- **Use prepared statements (PDO or MySQLi) to prevent SQL injection.**
- **Validate and sanitize all inputs from users.**
- **Maintain session integrity with session_start() and secure session handling.**

A. Develop a PHP script that allows users to manage their shopping cart for an second hand gaming consoles website. The script should allow users to add items to their cart, view their cart contents, and remove items if needed.
B. Develop a PHP script to manage the shopping cart for an second hand gaming consoles website using MySQL. This script should allow users to add items to their cart, view their cart contents, and remove items from the cart. The cart data should be stored in the MySQL database to allow persistence across sessions

Theory: PHP Shopping Cart System

A shopping cart is a core component of any e-commerce platform. It serves as a temporary storage space where users can collect and manage the items they wish to purchase. In the case of a second-hand gaming consoles website, where products can be unique and availability may be limited to single units, the shopping cart system plays an even more critical role.

Two Types of Cart Management Systems in PHP:

A. Session-Based Shopping Cart (Without MySQL)

This approach uses PHP sessions to temporarily store cart data in memory while the user is browsing. It is useful for fast prototyping and requires no database interaction.

Key Characteristics:

- Cart data is stored in $_SESSION.
- Data persists during the browsing session.
- No need to log in to use the cart.
- Items are lost if the session expires or the browser is closed.

Operations Supported:

- **Add to Cart**: Add items by storing product ID, name, quantity, and price in session.
- **View Cart**: Display the contents stored in session.
- **Remove from Cart**: Unset item by ID or index from the session.

Advantages:

- Simple to implement.
- No database overhead.

Limitations:

- Not persistent after session end.
- Not scalable for logged-in user experiences.

B. Database-Based Shopping Cart (With MySQL)

This is the professional and scalable approach where cart data is stored in a **MySQL database**. It allows cart contents to persist across user sessions, devices, and logins.

Key Characteristics:

- Each user has a unique cart identified by user ID.
- Cart contents are stored in a cart table, and optionally a cart_items table for item details.
- Requires user login or session management.

Operations Supported:

- **Add to Cart**: Insert or update records in the cart_items table.
- **View Cart**: Query database for all cart items belonging to a specific user.
- **Remove from Cart**: Delete an item from the database by item ID or cart ID.

Advantages:

- Cart is persistent and user-specific.
- Works across sessions and devices.

- Enables cart analytics and user behavior tracking.

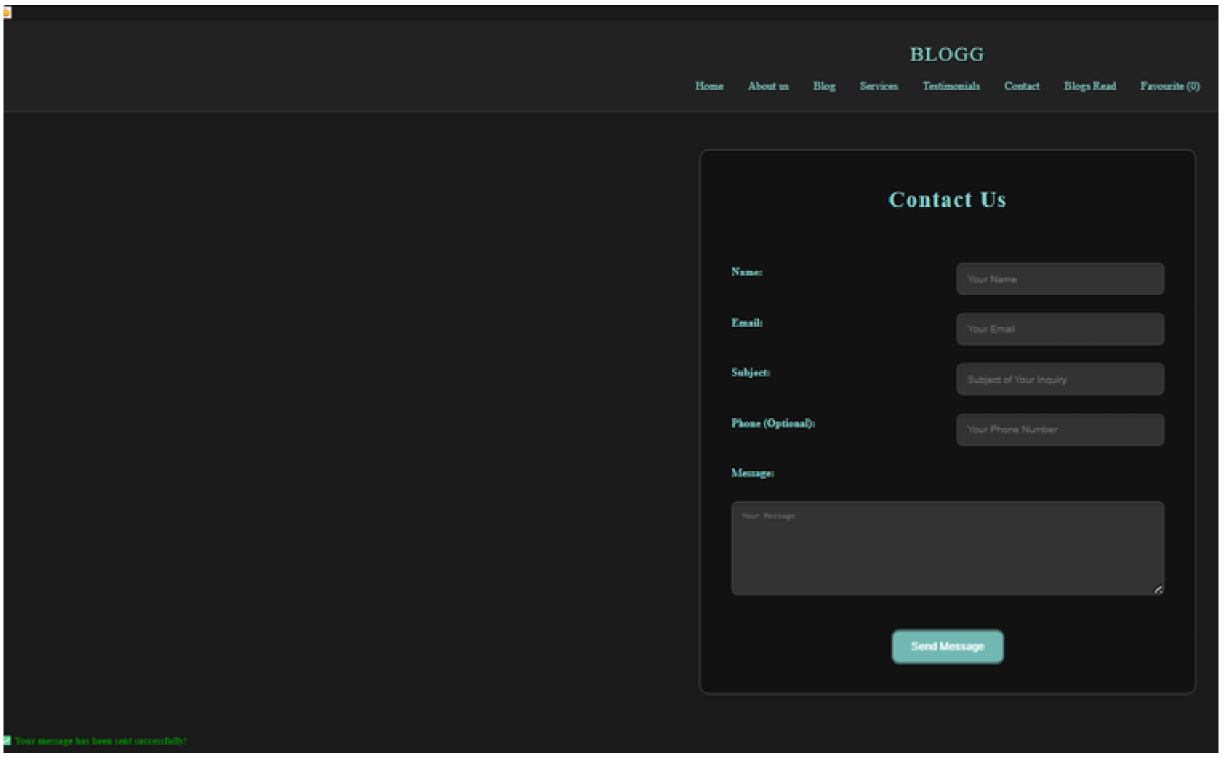
Limitations:

- Requires more setup and error handling.
- Needs secure login system to link cart with user.

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:



✅ Your message has been sent successfully!

Conclusion

A shopping cart system, whether session-based or database-driven, is essential for enhancing the user experience and improving sales on your second-hand gaming console website.

When using PHP:

- **Session-based carts** offer fast and simple cart functionality, ideal for guest users.
- **MySQL-backed carts** provide reliable, persistent storage across sessions and devices—ideal for logged-in users and production-level systems.

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

- Improves user experience with persistent carts.
- Enables personalization and user analytics.
- Supports consistent item tracking (especially when each console unit is unique).

Experiment 10

Problem Statement: In today's digital era, readers frequently encounter challenges when searching for and purchasing books online due to fragmented platforms, limited filtering options, and inconsistent access to detailed information about books. Many existing websites are not user-friendly and often lack features that support efficient book discovery and smooth purchasing experiences. Furthermore, independent bookstores and self-published authors face difficulties in reaching a wider audience through accessible and streamlined web solutions. There is a clear need for a web-based system that allows users to search for books, explore relevant details, and complete purchases with ease. The proposed Book Search and Shopping web application aims to address this gap by offering a simple, user-centered platform that integrates intelligent search, comprehensive book information, shopping cart functionality, and a secure checkout process—ultimately enhancing the overall experience for both readers and book sellers**.**

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 e-commerce platform. It translates the user's cart into an official order, capturing necessary details such as billing, shipping, and payment, then recording it into the database for processing and fulfillment.

On a second-hand gaming console website, where products may be unique or limited, a **robust and accurate checkout system** ensures that stock integrity is maintained and customer satisfaction is upheld.

Two Approaches to Checkout

A. Session-Based Checkout (Without Database Order Management)

In this basic approach:

- All data is stored in the session ($_SESSION['cart']).
- On checkout, a confirmation message is shown.
- Useful for simple or demo applications. **Workflow:**

1. Retrieve cart from $_SESSION.
2. Validate input fields (name, email, address).
3. Show success or error message.
4. Clear cart after checkout.

Advantages:

- Quick to implement.
- Minimal setup required.

Limitations:

- Data not persistent.
- Not scalable or production-ready.
- No order history.

B. MySQL-Based Checkout System

This advanced and scalable approach:

- Stores order details in a MySQL database.
- Supports persistence, analytics, and back-end processing.
- Links orders to logged-in users.

Workflow:

1. Validate user session or login status.
2. Retrieve cart items from session or database.
3. Validate checkout fields (shipping info, contact).
4. Insert data into orders and order_items tables.
5. Display success/failure message.
6. Clear session cart.

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:-

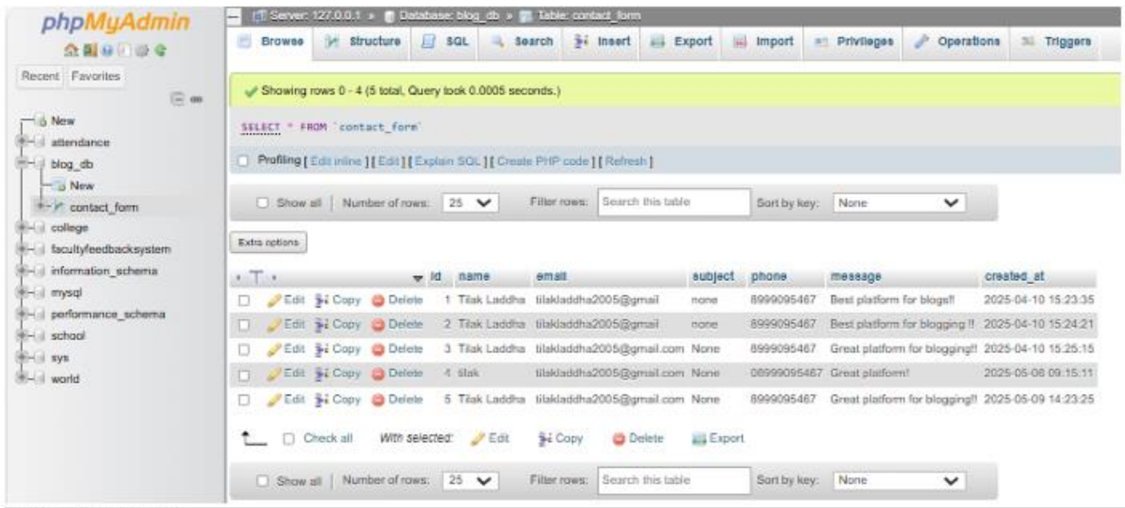
```
<?php
session_start();

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:



Server: 127.0.0.1 » Database: blog_db » Table: contact_form

Showing rows 0 - 4 (5 total, Query took 0.0005 seconds.)

SELECT * FROM `contact_form`

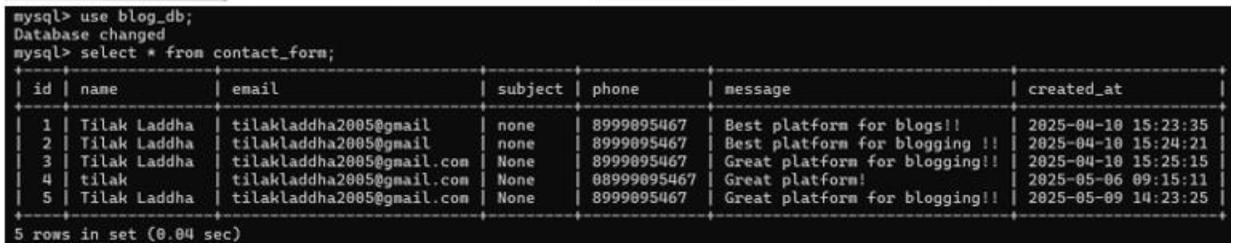
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

	id	name	email	subject	phone	message	created_at
Edit Copy Delete	1	Tilak Laddha	tilakladdha2005@gmail	none	8999095467	Best platform for blogs!!	2025-04-10 15:23:35
Edit Copy Delete	2	Tilak Laddha	tilakladdha2005@gmail	none	8999095467	Best platform for blogging !!	2025-04-10 15:24:21
Edit Copy Delete	3	Tilak Laddha	tilakladdha2005@gmail.com	None	8999095467	Great platform for blogging!!	2025-04-10 15:25:15
Edit Copy Delete	4	tilak	tilakladdha2005@gmail.com	None	08999095467	Great platform!	2025-05-06 09:15:11
Edit Copy Delete	5	Tilak Laddha	tilakladdha2005@gmail.com	None	8999095467	Great platform for blogging!!	2025-05-09 14:23:25

With selected:

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None



```
mysql> use blog_db;
Database changed
mysql> select * from contact_form;
+----+--------------+---------------------------+---------+-------------+-------------------------------+---------------------+
| id | name         | email                     | subject | phone       | message                       | created_at          |
+----+--------------+---------------------------+---------+-------------+-------------------------------+---------------------+
|  1 | Tilak Laddha | tilakladdha2005@gmail     | none    | 8999095467  | Best platform for blogs!!     | 2025-04-10 15:23:35 |
|  2 | Tilak Laddha | tilakladdha2005@gmail     | none    | 8999095467  | Best platform for blogging !! | 2025-04-10 15:24:21 |
|  3 | Tilak Laddha | tilakladdha2005@gmail.com | None    | 8999095467  | Great platform for blogging!! | 2025-04-10 15:25:15 |
|  4 | tilak        | tilakladdha2005@gmail.com | None    | 08999095467 | Great platform!               | 2025-05-06 09:15:11 |
|  5 | Tilak Laddha | tilakladdha2005@gmail.com | None    | 8999095467  | Great platform for blogging!! | 2025-05-09 14:23:25 |
+----+--------------+---------------------------+---------+-------------+-------------------------------+---------------------+
5 rows in set (0.04 sec)
```

Conclusion

The checkout process is the most vital component of an e-commerce platform—it turns intent into action. For your second-hand gaming consoles website:

Use Case Importance:

- **Unique item inventory** means precise, real-time cart tracking is essential.
- **Persistence** through MySQL helps avoid loss of user choices and enables full order management.
- **Session-based approach** is useful in early development or guest checkout situations.

Session-Based Checkout Summary:

- Simple and fast.
- Best suited for demos or early-stage projects.
- Not ideal for multi-session or long-term tracking.

MySQL-Based Checkout Summary:

- Scalable and professional.
- Captures order history.
- Supports user-specific orders, data analytics, and future features like order cancellation or tracking.