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-II

Mr. Suraj Hippargi

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 second-hand gaming console store website, including the home page layout with a header, main content area, and footer.

Prepare a common project website design and plan document for all assignments. Consider 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 **Hotel Menu System** website by planning its layout, content, and visual elements,

ensuring it enhances the dining experience for users, allows easy browsing of the menu, and effectively represents the hotel's brand and hospitality.

Theory:

Project Design and Plan Document for Hotel Menu System Website

1. Brief Information about the Project

The project is to create a user-friendly and visually appealing website for a Hotel Menu System. It aims to enhance the guest experience by providing an interactive menu, promoting signature dishes, and offering features such as customer testimonials and a support/contact page. The website will also support login and registration to personalize user experiences and provide exclusive services such as order history and saved preferences.

2. Goals and Deliverables

Goals

- Develop an engaging and functional website for a Hotel Menu System.
- Showcase the hotel's story, menu offerings, customer reviews, and support 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
    - Menu 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 Hotel Menu System 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 key offerings. It sets the tone for the guest's experience.
- Features:
  - Hero section with the tagline and call-to-action buttons (e.g., "Order Now" or "Explore Menu").
  - Overview of featured dishes or daily specials.
  - Navigation menu linking to all website sections (e.g., About, Menu, 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:
  - Overview of the hotel's dining philosophy and history.
  - Introduction to the chefs and kitchen team.
  - Engaging visuals to reflect the hotel's ambiance and cuisine.

3. Menu Page Module

- Description:
  Displays the hotel's food offerings in an organized and visually appealing manner.
- Features:
  - Categorized sections (e.g., Starters, Main Course, Desserts, Beverages).
  - Dish images, ingredients, prices, and availability.
  - Optional filtering for dietary preferences (e.g., vegan, gluten-free).

4. Contact Page Module

- Description:
  Allows guests to reach out to the hotel for inquiries or support.
- Features:
  - Contact form (Name, Email, Subject, Message).
  - Embedded map for the hotel location.
  - Contact details such as phone number, email, and opening hours.

5. Login Page Module

- Description:
  Enables returning guests to log in for a personalized experience.

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

6. 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.

7.Cart page Module

Description:
The Cart Page allows users to review and manage the items they've added to their cart before proceeding to checkout. It gives them a chance to edit their selections, see the total cost, and proceed with payment.

Features:

- Displays all the items added to the cart with their names, quantities, and prices.
- Option to increase or decrease the quantity of selected items.

4. Define the audience

Target Audience

The Hotel Menu System website is tailored for a diverse audience, each with specific expectations. Understanding these segments ensures the site design, content, and features are aligned with user needs.

a. Hotel Guests:

- Characteristics:
    - In-house or visiting diners looking to explore the menu.
- Needs:
    - Easy access to food options, descriptions, and prices.
    - Option to log in and save favourite dishes.

b**. Event Organizers & Corporate Clients:**

- Characteristics:
    - Booking the hotel for conferences or special occasions.
- Needs:

- Detailed menu options for group catering.
- Contact and support for event coordination.

c. **Tourists and Travelers:**

- Characteristics:
  - Visiting the city and exploring local dining.
- Needs:
  - Menu with regional specialties.

d. Health-Conscious Diners:

- Characteristics:
  - Looking for organic, vegan, gluten-free, or low-calorie options.
- Needs:
  - Menu filters for dietary preferences.
  - Nutrition details on selected dishes.

e. Families:

- Characteristics:
  - Dining with kids or older adults.
- Needs:
  - Kid-friendly menu options.
  - Clear information about allergens or spice levels.

f. Business Professionals:

- Characteristics:
  - Hotel guests or walk-in visitors dining during work breaks.
- Needs:
  - Quick navigation and concise menu view.
  - Support for early reservations and fast service.

g. First-Time Visitors

- Characteristics:
  - Unfamiliar with the hotel and its offerings.
- Needs:
  - A welcoming homepage and informative “About” page.
  - Trust-building design and clear navigation.

Website Features Mapped to Audience Needs:

Audience Segment	Key Features Needed
Hotel Guests	Menu page with detailed descriptions, images, and chef recommendations.
Business Professionals	Quick access to lunch specials, reservation options, and efficient navigation for time-sensitive dining.
Students	Special offers, student discounts, and easy-to-access promotions or combo deals.
Health-Conscious Diners	Categorized menu with nutritional information, filters for vegan/gluten-free/low-calorie dishes.
Tourists/Travelers	Map integration, multilingual support, and highlights of local or regional cuisine.
First-Time Visitors	Simple and attractive UI, engaging "About Us" section, and clear navigation with photos/testimonials.
Online Order Users	Secure login system, responsive menu interface, and a seamless online ordering and customization feature.

Why Understanding the Audience is Important

- Helps in creating engaging and relevant content tailored to guests' dining preferences, dietary requirements, and cultural interests.
- Enhances the user experience (UX) by addressing specific pain points such as easy navigation, clear menu categories, and personalized recommendations.
- Builds brand trust and attracts loyal customers by reflecting the hotel's values, atmosphere, and culinary expertise.
- Leads to targeted marketing campaigns, such as special promotions for tourists, student discounts, health-conscious menu highlights, or exclusive offers for loyal guests.

Here is the **Pain Points & Ideal Experience** section tailored for your **Hotel Menu System** project:

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 hotel menu websites have complicated or cluttered designs, making it difficult for guests to find their preferred dishes, special offers, or promotions.
- **Impact:** Users experience frustration and are likely to leave the site, reducing customer engagement and bookings.

b. Pain Point: Limited Online Ordering Functionality

- **Issue:** Existing systems may lack a user-friendly and intuitive online ordering system, leading to a poor guest experience.
- **Impact:** Guests may opt for other services or experience confusion when trying to place an order, impacting sales and customer satisfaction.

c. Pain Point: Lack of Mobile Optimization

- **Issue:** Non-responsive designs result in poor experiences for guests browsing the menu on their mobile devices.
- **Impact:** Customers using smartphones face navigation challenges, leading to a decrease in conversions and engagement.

d. Pain Point: Insufficient Menu Information

- **Issue:** Customers often find that the menu lacks detailed descriptions, images, or dietary information such as ingredients, allergens, and prices.
- **Impact:** Potential customers abandon their search, as they are not given enough information to make informed decisions.

e. Pain Point: Weak Engagement Strategies

- **Issue:** Many hotel menu systems lack features that engage customers, like loyalty programs, seasonal promotions, or special discounts.
- **Impact:** Missed opportunities for encouraging repeat visits and building customer loyalty.

f. Pain Point: Inefficient Contact and Location Details

- **Issue:** Contact information and location details are either buried in the site or not easily accessible.
- **Impact:** Customers waste time searching for how to reach the hotel, leading to frustration and a potential loss of business.

g. Pain Point: No Personalization Options

- **Issue:** Lack of personalization features, such as saving favorite dishes or offering tailored recommendations based on past orders.
- **Impact:** Customers feel the experience is impersonal, which decreases satisfaction and loyalty.

2. Crafting the Ideal Experience

To address these pain points, the website should be designed to offer a seamless, user-friendly, and visually appealing experience for all users.

a. Intuitive Navigation and Clean Design

- Use a simple, consistent layout with easy access to all major sections (Home, Menu, Promotions, Testimonials, Contact, Login/Sign Up).
- Implement sticky navigation to keep key menu items visible as users scroll.

b. Seamless Online Ordering

- Enable an intuitive ordering system where customers can easily browse dishes, select items, and complete their order with minimal clicks.
- Add "Order Now" buttons on the homepage and menu pages to streamline the process.

c. Mobile-Responsive Design

- Design the website using a mobile-first approach to ensure it's fully responsive across devices (mobile, tablet, desktop).
- Use touch-friendly elements, flexible grids, and optimize loading times to ensure a smooth experience on all devices.

d. Comprehensive Menu Information

- Include high-quality images, item descriptions, pricing, and dietary information such as gluten-free, vegan, or low-calorie options.
- Allow users to filter menu items based on preferences like "Healthy," "Vegetarian," or "Spicy."

e. Customer Engagement Features

- Introduce loyalty programs where guests can earn points or discounts for repeat visits or orders.
- Add a blog section with relevant content, like food trends, chef insights, or seasonal specials.
- Display customer reviews and testimonials to build credibility.

f. Easy Access to Contact and Location

- Include a clear contact page with an inquiry form, phone number, and email.
- Embed an interactive map showing the hotel's location on the homepage or contact page to make it easy for guests to find the restaurant.

g. Personalization

- Allow guests to create accounts for saving favorite dishes, viewing past orders, and receiving personalized recommendations.
- Offer personalized promotions or discounts to registered users based on their past orders or preferences.

3. The Ideal User Journey

Step 1: Visiting the Website

- Users arrive at a welcoming homepage with clear navigation to menu sections, promotions, and contact details.

Step 2: Browsing the Menu

- Users explore the menu with clear categorization, high-quality images, and filters that match their dietary needs or preferences.

Step 3: Placing an Order

- Users can add items to their cart, review selections, and complete the order seamlessly with minimal steps.

Step 4: Finding Location or Contacting Support

- Users easily find the contact form, phone number, and store location, along with details like operating hours and directions.

Step 5: Engaging with Content

- Users explore blog posts, special offers, or customer reviews to learn more about the restaurant and its offerings.

Step 6: Creating Loyalty

- Registered users receive personalized offers, discounts, or loyalty points after placing orders, encouraging repeat business.

Here is the **Visual Direction** tailored for your **Hotel Menu System** project:

6. Set the Visual Direction

1. Visual Design Goals

The visual design of the hotel menu website should reflect its brand, create an inviting environment for customers, and make the menu browsing and ordering process seamless. The visual direction should align with the following principles:

- **Welcoming and Comfortable:** The website should feel inviting and user-friendly, creating a cozy atmosphere similar to the dining experience guests will have at the hotel.
- **Modern and Minimalistic:** Clean layouts, modern design elements, and simple navigation will ensure that the website remains easy to use while still feeling stylish.
- **Brand Representation:** The visual elements, such as colors, fonts, and images, should communicate the hotel's values of quality, comfort, and luxury while appealing to the target audience.

2. Defining the Core Visual Elements

a. Color Palette

The color scheme should reflect a modern, elegant, and warm atmosphere. Soft, natural tones with a pop of contrast can help create an inviting and comfortable feel for the website.

Color	Hex Code	Usage
Warm Taupe	#D8CAB8	Header, footer, buttons, and key highlights.
Soft Cream	#F4F1EC	Background for warmth and contrast.
Elegant Charcoal	#2E2E2E	Text and important accents for legibility.
Ivory White	#FAF9F1	Secondary backgrounds and subtle contrasts.
Deep Green	#3A4E4F	Call-to-action buttons for inviting interaction.

b. Typography

Fonts should be chosen to reflect the upscale, yet approachable nature of the hotel. Simple, clean, and elegant fonts will make the website easy to read while maintaining a sense of sophistication.

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

c. Logos and Branding

A sleek, minimalist logo representing the hotel's atmosphere is essential. The logo should be simple yet distinctive.

- **Logo Design:** Use an elegant motif that represents the essence of the hotel – possibly incorporating elements like a dining plate, fork, or abstract shapes that symbolize hospitality.
- **Logo Colors:** The logo should reflect the website's color palette and be versatile for different placements, including headers and footers.
- **Monochrome Version:** A simplified black or white version of the logo for clean, understated use.

d. Imagery and Icons

High-quality images will provide an inviting and vibrant feel, showcasing the hotel's ambiance, dishes, and dining experience.

- **Photography:**
    - High-quality images of signature dishes, dining experiences, the interior of the hotel, and guests enjoying meals.
    - Showcasing food preparation or in-house chefs adding a personal touch to the brand.
- **Icons:**
    - Use minimalist icons for categories like menu items, contact, location, testimonials, etc.
    - Ensure the icons complement the overall design language and are easy to understand.
- **Hero Images:**
    - Carousel or static banners on the homepage featuring seasonal promotions, featured dishes, or special events.

3. Applying Visual Design to Pages

a. Home Page

- **Banner Area:** Hero image with a text overlay showcasing a featured dish or a seasonal promotion.
- **Color Scheme:** Warm tones for buttons and text, soft cream or ivory for the background to maintain a light and welcoming atmosphere.
- **Typography:** Bold headings, e.g., "Welcome to [Hotel Name]'s Dining Experience."

b. About Page

- **Imagery:** Authentic photos of the hotel staff, kitchen, or dining spaces.
- **Colors:** Soft and warm background tones that align with the overall aesthetic, using minimal text for impactful storytelling.

c. Product/Service Page

- **Product Cards:** Each menu item should have a visually appealing card with an image, description, and price.

- **Hover Effects:** Use gentle hover effects for interactivity, highlighting selected dishes or adding them to the cart.

d. Testimonial Page

- **Customer Photos and Quotes:** Display photos of guests or reviews in clean, card-style layouts.
- **Interactive Slider:** Allow users to scroll through testimonials smoothly to foster trust.

e. Contact Page

- **Contact Form:** Keep the form layout minimal, with clear labels for input fields and a soft background.
- **Embedded Map:** Include a Google map to show the hotel's location along with a green-toned call-to-action button for reservations.

f. Login and Registration Pages

- **Minimal Form Layout:** Simple, clean form design with visible input fields and clear instructions.
- **Button Design:** Call-to-action buttons in warm taupe or deep green to maintain visual consistency and encourage user interaction.

4. Layout and Design Hierarchy

The visual hierarchy is essential to ensure users easily navigate the website. The structure should help guide the visitor through the content in an intuitive manner:

1. **Headers and Banners:** These elements should be prominent and eye-catching to draw users' attention immediately.
2. **Navigation Bar:** A sticky navigation bar that stays at the top of the screen for easy access to all main sections (Menu, Contact, Testimonials, etc.).
3. **Sections and Grids:** Use structured grids to divide content with clear breaks for a clean layout, helping users easily browse through categories.
4. **Call-to-Action Buttons:** Prominent and visually distinct buttons to encourage user actions, such as "Order Now," "Sign Up," and "Learn More."

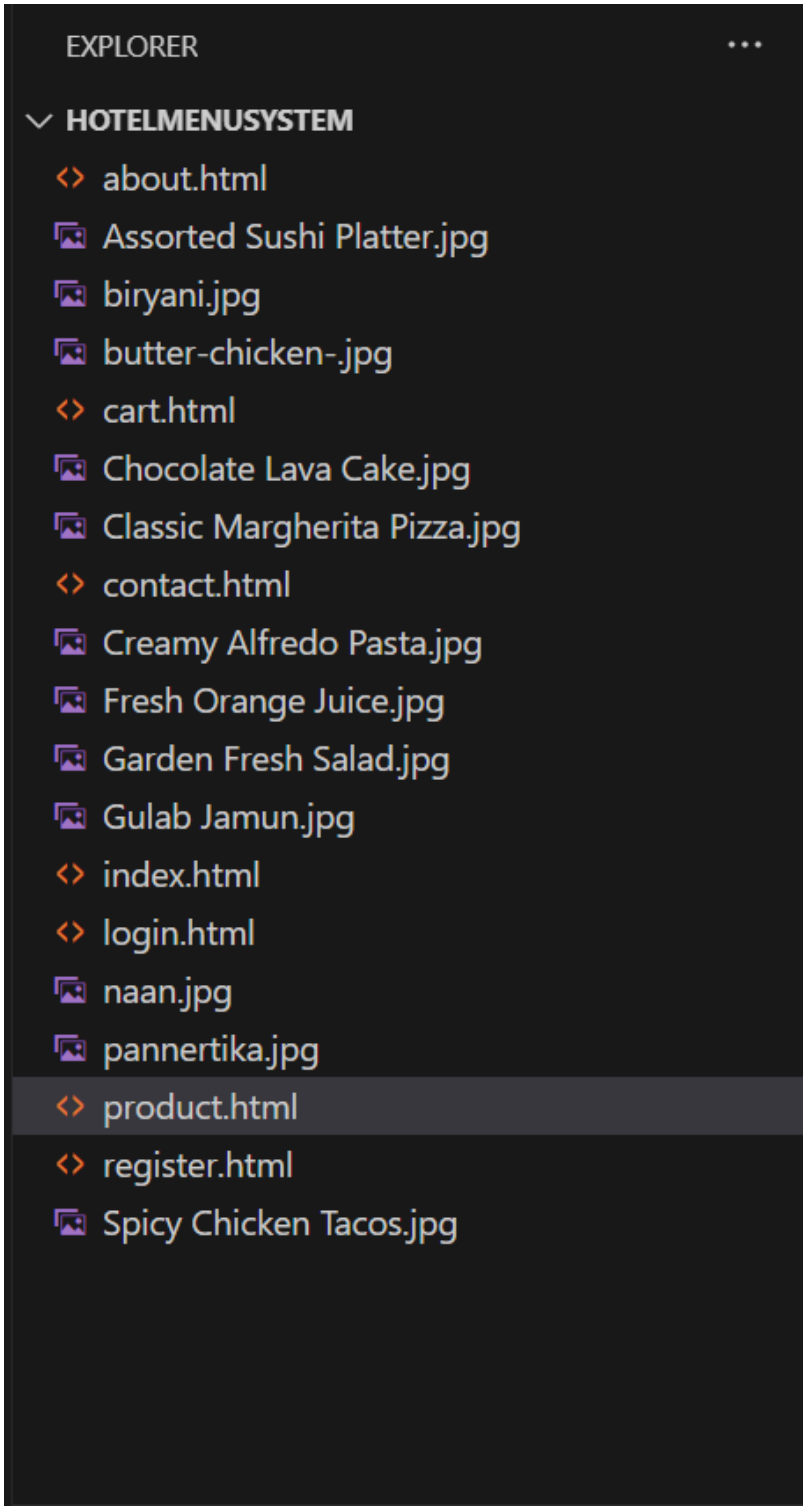
5. Expected Impact of Visual Direction

- **Enhanced Engagement:** A warm and inviting design will create a welcoming atmosphere, encouraging users to stay on the site longer and explore more.
- **Stronger Branding:** Consistency in colors, typography, and imagery will reinforce the hotel's identity and help visitors remember the brand.
- **Better Retention:** A user-friendly and visually appealing layout will make the website easier to navigate, increasing the likelihood that users will return.
- **Higher Conversions:** Effective call-to-action placement and streamlined ordering processes will lead to more successful reservations and orders, increasing conversions and overall sales.

7. Map out the Project structure

console_store_website/

```
│
├── index.html            # Home page
├── about.html            # About page
├── Menu.html          # Products/Services page
├── contact.html          # Support page
├── login.html            # Login page
├── register.html         # Registration page
├── cart.html             # cart page
│
├── assets/
│   ├── css/
│   │   ├── style.css       # Global CSS
│   │   ├── responsive.css   # Media queries for mobile optimization
│   │
│   ├── js/
│   │   ├── main.js         # Interactive scripts
│   │   ├── formValidation.js # Scripts for login and registration validation
│   │
│   ├── images/
│       ├── logo.png        # Website logo
│       ├── homepage_banner.jpg  # Hero banner for home
│       ├── console_catalog/     # Images for catalog items
│       ├── portable consolesm_photos/     # Images for the about page
│       ├── icons/          # Icons for UI elements
│
├── fonts/
│   ├── Poppins/           # Primary font
│   ├── OpenSans/          # Secondary font
│
└── README.md              # Project documentation file
```


EXPLORER
HOTELMENUSYSTEM
about.html
Assorted Sushi Platter.jpg
biryani.jpg
butter-chicken-.jpg
cart.html
Chocolate Lava Cake.jpg
Classic Margherita Pizza.jpg
contact.html
Creamy Alfredo Pasta.jpg
Fresh Orange Juice.jpg
Garden Fresh Salad.jpg
Gulab Jamun.jpg
index.html
login.html
naan.jpg
pannertika.jpg
product.html
register.html
Spicy Chicken Tacos.jpg

8. Plan the content for each page

Page	Purpose	Key Content
Home Page	Welcome visitors and introduce the store's offerings.	Hero image, tagline, featured products, navigation bar, call-to-actions, special highlights.
About Page	Share story, values, and team behind the store.	Store history, team photos, mission, "Why Choose Us?" section.
Products Page	Display available consoles and services.	Product categories, item tiles with images and prices, filters, "Top Picks" section.
Cart Page	Let users review, update, or remove items before checkout.	Item list with quantity & price, subtotal, taxes, total, "Proceed to Checkout" button.
Contact Page	Allow customers to reach out or find the store location.	Contact form, embedded map, business hours, quick inquiry section.
Login Page	Enable users to log into their accounts.	Email/password fields, forgot password link, redirect to registration.
Registration Page	Allow new users to create an account.	Name, email, password fields, confirm password, T&C checkbox, submit button.

9. Add ideas for content, images & layout

1. Home Page

Layout Ideas:

- **Header:**
  - Logo (hotel-themed or food-themed).
  - Navigation: Home, About, Menu, Cart, Contact.
  - Login/Register buttons on the right.
- **Hero Section:**
  - Full-width image of a signature dish or elegant dining setup.
  - Tagline: "Savor the Flavors of Luxury."
  - CTA: "Explore Our Menu"
- **Featured Section:**
  - 3–4 food highlights (e.g., Chef's Specials).
- **Footer:**

- Social media, contact, and quick links.

Content Ideas:

- Welcome note about the hotel dining experience.
- Any seasonal dishes or promotional offers.

Image Ideas:

- Banner image of gourmet dishes or hotel restaurant.
- Close-up shots of menu items.

2. About Page

Layout Ideas:

- **Intro Section:**
  - Short paragraph on hotel history and dining services.
- **Why Choose Us:**
  - Icons + descriptions (e.g., Chef Expertise, Organic Ingredients, Luxurious Ambience).

Content Ideas:

- Hotel's culinary vision.
- Dining philosophy, sourcing, and specialties.

Image Ideas:

- Chefs in action.
- Dining area interiors.
- Staff photos (optional).

3. Menu (Products) Page

Layout Ideas:

- **Categories Tabs:**
  - Indian, Continental, Beverages, Desserts, etc.
- **Menu Cards:**
  - Each with an image, name, description, and price.
  - "Add to Cart" button.

Content Ideas:

- Dish details: spice level, ingredients (if relevant), vegetarian/non-veg tags.

Image Ideas:

- High-res food photography of every category.
- Clean, minimal layout with a focus on visuals.

4. Cart Page

Layout Ideas:

- **Cart Table/List:**
  - Items added from Menu.
  - Columns: Item Name, Quantity, Price, Remove/Edit.
- **Total Summary:**
  - Subtotal, taxes, total.
  - Checkout button.

Content Ideas:

- Confirmation text: “Review your order before checkout.”
- Note field for special instructions.

Image Ideas:

- Simple food icons or background image of a table setup.

5. Contact Page

Layout Ideas:

- **Form Section:**
  - Fields: Name, Email, Subject, Message.
- **Map Section:**
  - Embedded Google Maps for hotel location.
- **Info Block:**
  - Address, phone number, email, operating hours.

Content Ideas:

- Invite for reservations or inquiries.
- Social media/contact links.

Image Ideas:

- Hotel exterior.
- Restaurant/dining ambience.

6. Login Page

Layout Ideas:

- Clean form with fields for Email and Password.
- Link: “Don’t have an account? Sign up.”

Image Ideas:

- Background: Elegant dining photo with a transparent overlay.

7. Registration Page

Layout Ideas:

- Form Fields: Full Name, Email, Password, Confirm Password.
- Checkbox: Accept terms and conditions.

Image Ideas:

- Welcoming image or food graphic for visual warmth.
-

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

a. Home page

b. About page

c. Product page

d. Cart page

e. Contact page

1. Home Page

The home page introduces the hotel menu system and guides visitors to explore dishes, learn about the hotel, and place orders.

◈ Header:

- **Logo:** Elegant logo featuring a plate, spoon, or cloche icon.
- **Navigation Links:** Menu, About, Contact, Cart, Login/Register.
- **Call-to-Action Button:** "Order Now" or "Explore Menu".

◈ Hero Section:

- **Background Image:** High-resolution image of the hotel’s dining area or signature dishes.
- **Text Overlay:** “Experience Gourmet Dining, Right at Your Table.”
- **CTA Button:** “View Menu” or “Order Today.”

◈ About Teaser:

- A short introduction to the hotel and its culinary philosophy.
- "Learn More" link to the About page.

🔹 **Dish Highlights:**

- **Featured Dishes:** Showcasing 3–4 top-selling or chef-recommended dishes.
- Image + short description + "Add to Cart".

🔹 **Footer:**

- Quick links: Menu, Location, FAQs.
- Social media icons (Instagram, Facebook).
- Address + Google Map embed.

2. About Page

Tells the story of the hotel and culinary excellence.

🔹 **Introduction:**

- History and mission: "Serving quality cuisine since 1995..."
- Emphasis on taste, hygiene, and guest experience.

🔹 **Meet the Team:**

- Photos and bios of head chef, kitchen team, and management.

🔹 **Our Promise:**

- Icons + text explaining hygiene, fresh ingredients, fast delivery, vegetarian/non-veg options.

🔹 **Location & Hours:**

- Google Maps integration + working hours for dine-in, takeaway, and delivery.

3. Menu Page (Product/Service)

The core of your system: showcases all dishes with ordering features.

🔹 **Menu Categories:**

- Starters, Main Course, Desserts, Beverages, Combos, Chef Specials.

🔹 **Dish Details:**

- Each item includes:
    - Image, Name, Price
    - Description (e.g., "Butter Chicken – Rich & creamy North Indian classic")
    - "Add to Cart" button

🔹 Specials:

- Carousel or grid for “Today’s Specials” or “Limited-Time Offers”.

🔹 Filters/Sorting:

- Filter by category, dietary preference, or price range.

4. Cart Page

Helps users manage their orders before checkout.

🔹 Features:

- Table of added dishes: Name, quantity, subtotal.
- “Edit Quantity” / “Remove” options.
- Total price calculator.
- “Proceed to Checkout” button.

5. Contact Page

🔹 Contact Form:

- Fields: Name, Email, Message
- “Send Message” button

🔹 Social Media & Address:

- Instagram, Facebook links
- Hotel’s physical address, phone number, support email

🔹 Map Integration:

- Embedded Google Map for directions.

🔹 FAQs:

- Delivery times, payment options, allergen info.

6. Login & Registration Pages

🔹 Login Page:

- Email + Password fields
- “Forgot Password?” link

🔹 Registration Page:

- Full Name, Email, Password, Confirm Password
- Optional: Subscribe to newsletter or SMS updates

• Home Page – Introduction, featured dishes, navigation

• About Page – Story, team, promises, values

• Menu Page – Full categorized menu with “Add to Cart”

• Cart Page – Order summary and checkout

• Contact Page – Form, social links, map, support

• Login/Register Page – Account management

(Optional) Blog Page – Articles on food and hotel dining

Design Elements

1. Color Scheme

Convey warmth, luxury, and appetite:

- **Primary:** Rich Maroon (#800000), Cream (#FDF6E3)
- **Accent:** Gold (#FFD700), Forest Green (#228B22)
- **Backgrounds:** Light beige (#FAF4EC), Deep brown for footers

2. Fonts & Typography

◈ Heading Font:

- **Playfair Display** or **Lora** – Elegant serif fonts for titles (e.g., menu headings, dish names)

◈ Body Font:

- **Roboto** or **Open Sans** – Clean sans-serif for descriptions and readable text

◈ Font Weights:

- Bold for dish names/headlines
- Regular for descriptions, FAQs

3. Logo Design

- Simple icon: Silver plate with steam, chef hat, or covered dish
- Colors: Use primary colors (maroon/gold)
- Scalable: Appears on all pages and suitable for mobile/print

4. Imagery and Photos

◈ Product Photography:

- Close-up shots of dishes with clean background
- Plated food, beverages, desserts in natural lighting

🔹 Ambiance Photos:

- Dining setup, kitchen, chef at work
- Guests enjoying meals

🔹 Lifestyle Shots:

- People dining, smiling staff, delivery in action
- Focus on warmth, satisfaction, and service quality

5. Interactive Elements and Buttons

🔹 CTAs:

- "Order Now," "Explore Menu," "Add to Cart"
- Color: Gold or Green for visibility
- Hover Effects: Shadow or slight color change

🔹 Icons:

- Dish → 🍽️, Cart → 🛒, Location → 📍, Chef → 👨‍🍳
- Use simple icon packs matching site colors

Conclusion

This Hotel Menu System website brings the hotel's dining experience online. From structured navigation and seamless ordering to a visually rich interface, it caters to both customer convenience and brand elegance.

Key takeaways:

- Clear structure with essential pages: Home, About, Menu, Cart, Contact, Login/Register
- Strong brand through consistent design, imagery, and typography
- User-friendly with interactive elements, responsive design, and high-quality visuals

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 Second-Hand Gaming Consoles webpage using HTML.

Theory:

In today's digital economy, online platforms are essential for improving service efficiency in the hospitality industry. This project focuses on developing a responsive and functional website for a **Hotel Menu System**. The platform is designed to allow guests and customers to conveniently browse food and beverage offerings, place orders, and view detailed menu descriptions, all from their devices — enhancing both accessibility and user experience.

The website combines front-end and back-end components to deliver a smooth interaction flow. Core features include **digital menu listings, user authentication, cart/order management**, and a **contact form** for customer support. The front end is implemented using **HTML and CSS**, with optional use of **JavaScript or server-side scripting** for interactivity and dynamic content management in later phases.

1. Home Page

The home page serves as the landing page and provides a snapshot of the hotel's culinary offerings. It typically includes:

- A **hero section** with featured dishes or chef's specials
- A **navigation bar** for seamless access to other sections like Menu, About, and Contact
- **Call-to-action buttons** like "Order Now", "View Menu", etc.
- Customer testimonials or highlighted dishes

Importance:
It creates the first impression and sets the tone of the dining experience. A clean, user-friendly layout with appealing food visuals encourages customer engagement and reduces bounce rate.

Technologies Used:

HTML for structure, CSS for layout and visuals, and optional animations using CSS or JavaScript for interactivity.

2. Menu Page

This is the core page of the Hotel Menu System where all available food and drink items are listed. Items are grouped into categories like:

- **Starters & Soups**
- **Main Course (Indian/Continental/Asian)**
- **Desserts**
- **Beverages**

Features include:

- Dish image
- Name and description (ingredients, portion size)
- Price
- "Add to Cart" button

Importance:
A well-organized digital menu improves discoverability and helps users explore the full dining experience.

UX Consideration:
Filters (by cuisine, vegetarian/non-veg, spice level) enhance usability and customer satisfaction.

3. Cart Page

This page allows users to view and manage their selected dishes before placing an order. It displays:

- All selected items with quantity and subtotal
- Options to update quantity or remove items
- Final checkout/confirm order button

Real-world relevance:
Gives customers flexibility and control over their dining orders before final submission.

Optional Enhancements:

- Cart persistence using localStorage
- Live price updates on quantity changes

4. About Us Page

This section provides background about the hotel and its culinary identity. It may include:

- Brief history or introduction of the hotel
- Mission and values (e.g., "Serving traditional flavors with a modern twist")
- Message from the head chef or manager
- Team photos (chefs, service staff)

Purpose:
Builds trust and connection with customers. It humanizes the hotel and adds credibility to its offerings.

5. Contact Page

A contact form for customer inquiries and feedback. It includes:

- Name
- Email
- Subject
- Message

Additional Elements:

- Hotel address and phone number
- Google Map integration for location
- Social media links (Instagram, Facebook, etc.)

UX Factor:
Provides customers with quick access to support for reservations, feedback, or menu-related questions.

6. User/Admin Registration Page

This allows both customers and hotel staff/admins to register. It includes:

- Full Name
- Email or phone
- Password and confirmation
- User type (Customer or Admin)

Functionality:

- Input validation for data accuracy
- (In full deployment) Secure data storage via back-end

Why it matters:
Enables order tracking for customers and menu/content management for admins.

7. User/Admin Login Page

This page authenticates users and redirects them based on their role.

Fields:

- Username/email
- Password

- “Remember Me” checkbox
- “Forgot Password?” link

Security Considerations:

- Basic input validation
- For future deployment: hashed passwords, 2FA, etc.

Role-Based Access:

- Customers: place orders, view past orders
- Admins: update menu items, manage orders, view sales data

Technological Stack Overview (Future Enhancement)

While the current version uses HTML and CSS, it can be extended using:

- JavaScript for cart interactions and menu filtering
- PHP or Node.js for order processing and back-end operations
- MySQL or MongoDB for order and user data storage
- Secure session handling for login/logout features

Sustainability Impact

The Hotel Menu System encourages paperless ordering, reducing the need for printed menus and bills. Benefits include:

- Reduced paper waste through digital menus
- Efficient resource use with online ordering and inventory tracking
- Supporting sustainable practices by minimizing overheads and waste

Code:

A. Home page

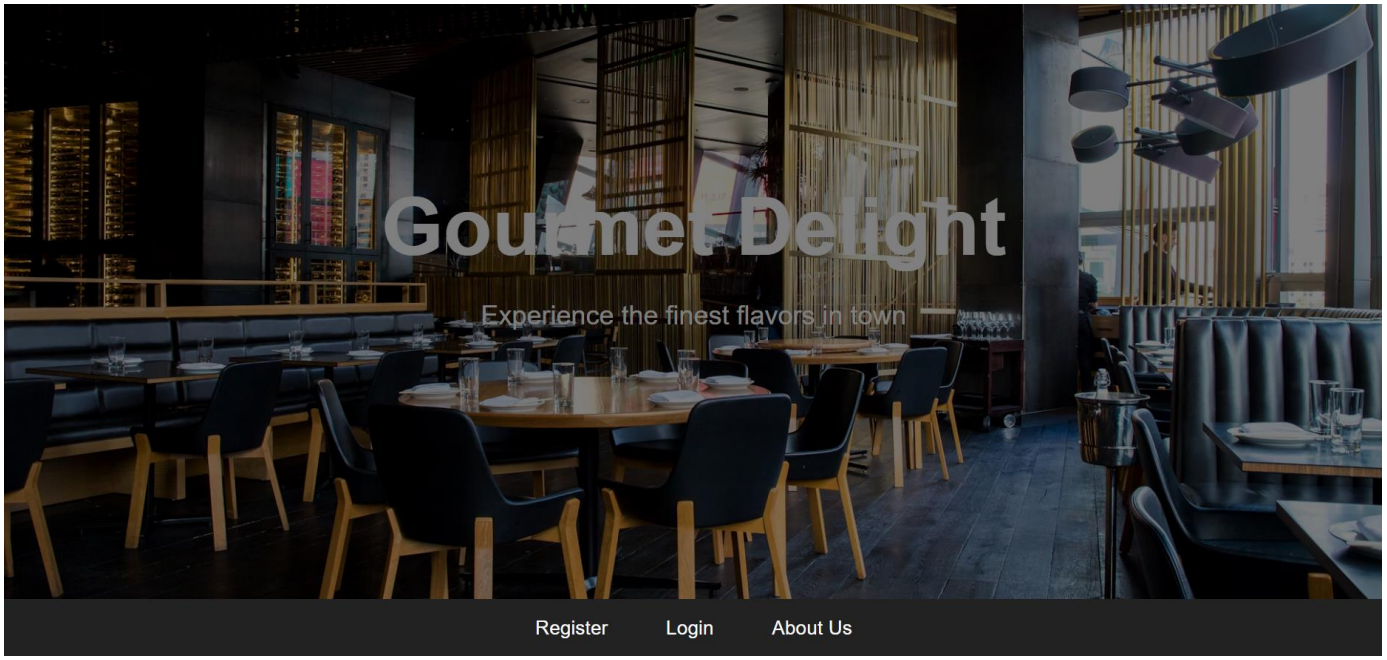
code:

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

```
</head>
<body>

  <!-- Header -->
  <header>
    <div>
      <h1>Gourmet Delight</h1>
      <p>Experience the finest flavors in town</p>
    </div>
  </header>

  <!-- Navigation -->
  <nav>
    <a id="register" href="register.html">Register</a>
    <a id="login" href="login.html">Login</a>
    <a id="about" href="about.html">About Us</a>
    <a id="menu" href="product.html" style="display: none;">Menu</a>
    <span id="welcome" style="display: none;"></span>
    <a id="logout" href="#" style="display: none;">Logout</a>
  </nav>

  <!-- Footer -->
  <footer>
    <p>&copy; 2023 Gourmet Delight. All rights reserved.</p>
  </footer>

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

Output:

A. Index/Home page output:



Code:

B. Product page

code:

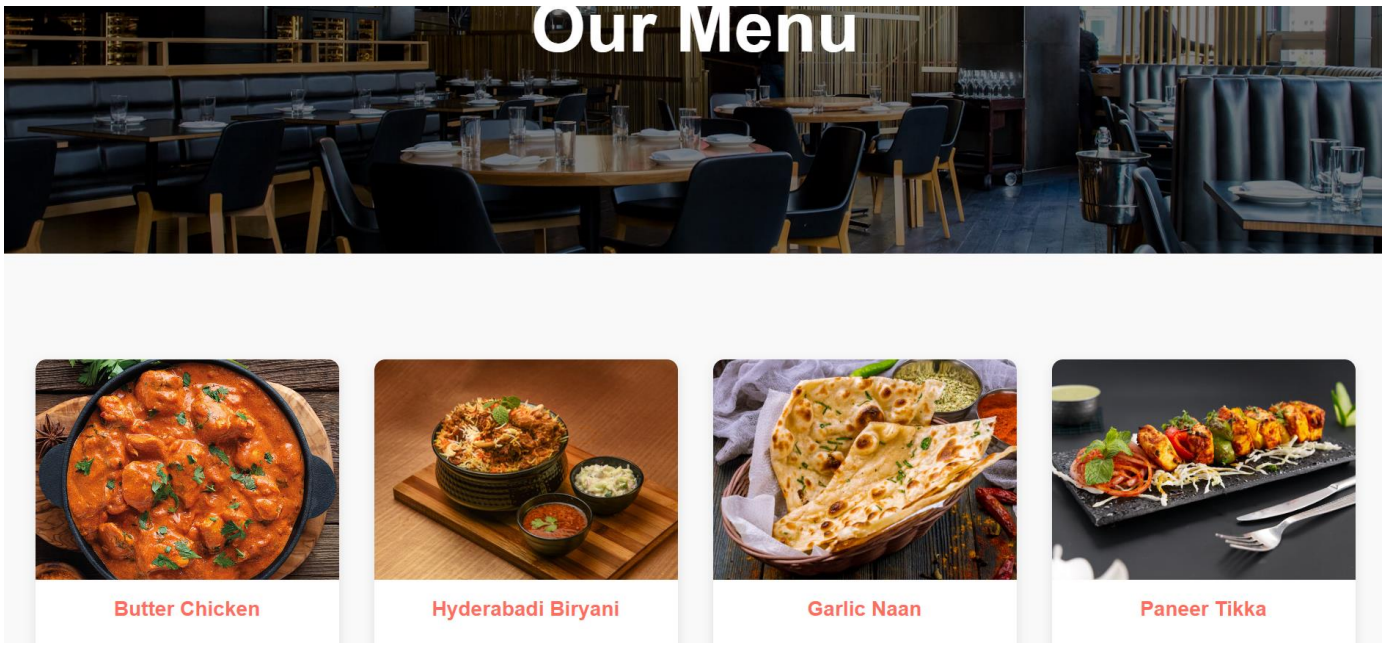
```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
  <title>Menu - Gourmet Delight</title>
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css"/>
  <link rel="stylesheet" href="menu.css">
</head>
<body>

  <!-- Hero Header -->
  <header>
    <h1>Our Menu</h1>
  </header>
```

```
<!-- Home and Cart Buttons -->
<div class="home-button">
  <a href="index.html" aria-label="Go to Home"><i class="fas fa-home"></i></a>
</div>

<div class="cart-container">
  <a href="cart.html" aria-label="View Cart">
    <div class="cart-icon">
      <i class="fas fa-shopping-cart"></i>
      <span class="cart-count" id="cart-count">0</span>
    </div>
  </a>
</div>

<!-- Menu Section -->
<section class="menu-section">
  <div class="menu-grid">

    <!-- Sample Menu Item -->
    <div class="menu-item">
      <img src="images/butter-chicken.jpg" alt="Butter Chicken">
      <h3>Butter Chicken</h3>
      <p>Creamy tomato-based curry with tender chicken and spices.</p>
      <button class="add-to-cart">Add to Cart</button>
    </div>

    <!-- Repeat similar blocks for other dishes -->

  </div>
</section>
```

```
  <!-- Footer -->
  <footer>
    <p>&copy; 2023 Gourmet Delight. All rights reserved.</p>
  </footer>

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

Output:

B. menu/product page output:



Code:

C. cart page

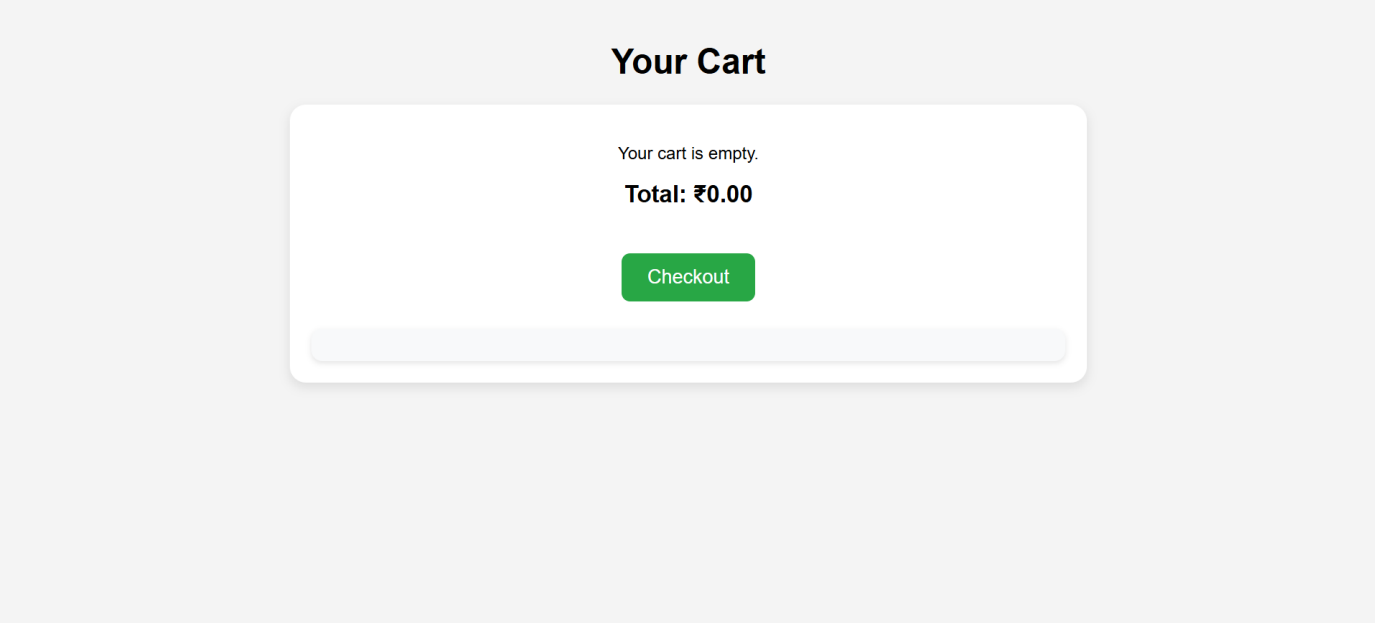
code:

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

```
  <link rel="stylesheet" href="cart.css">
</head>
<body>
  <h1>Your Cart</h1>

  <div class="cart-container">
    <div id="cart-items"></div>
    <p class="total">Total: ₹<span id="total-price">0</span></p>
    <button class="checkout-btn" onclick="checkout()">Checkout</button>
    <div class="bill" id="bill"></div>
  </div>

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

Output:

C. cart page output:



Code:

D. about us page

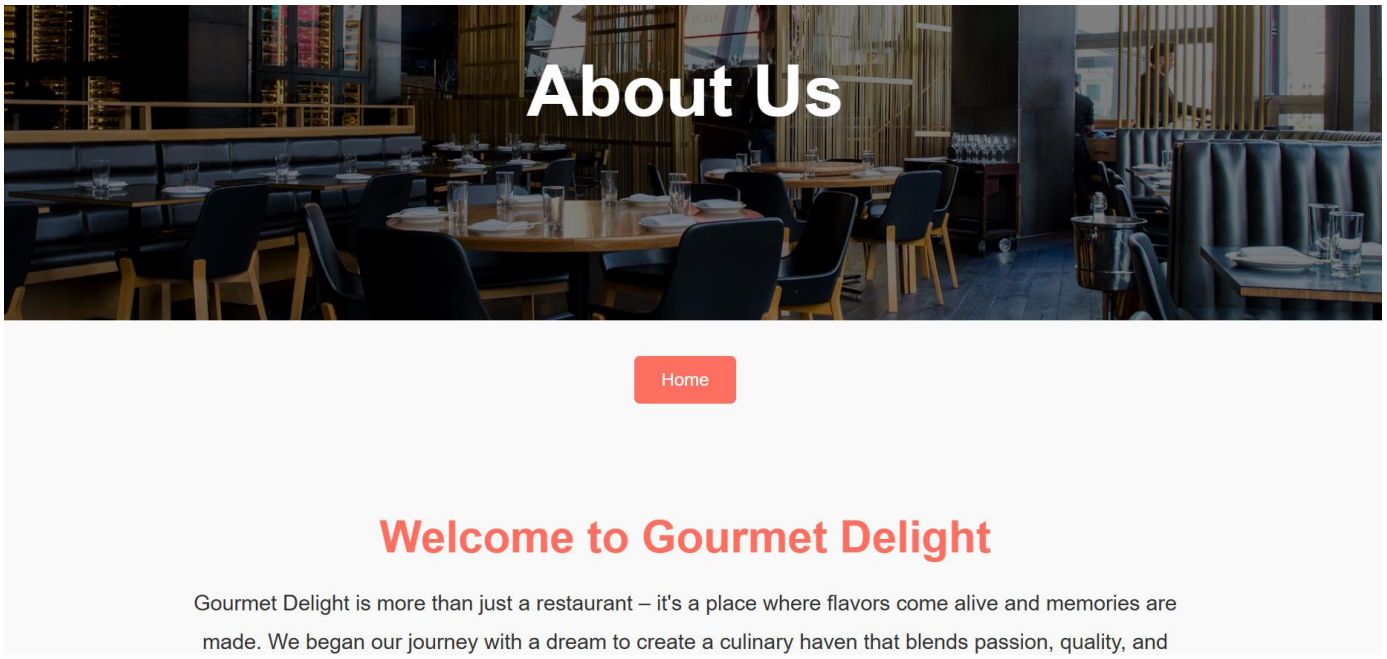
code:

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

    <!-- Hero Header -->
    <header>
        <h1>About Us</h1>
    </header>

    <!-- Home Button -->
    <div style="text-align: center;">
        <a href="index.html" class="home-btn">Home</a>
    </div>

    <!-- About Content -->
    <section class="about-section">
        <h2>Welcome to Gourmet Delight</h2>
        <p>
            Gourmet Delight is more than just a restaurant – it's a place where flavors come alive and memories are made...
            <!-- Keep the rest of your original paragraph content here -->
        </p>
    </section>

    <!-- Footer -->
```

```
    <footer>
        <p>&copy; 2023 Gourmet Delight. All rights reserved.</p>
    </footer>


</body>
</html>
```

Output:

C. cart page output:



Code:

E. contact us page

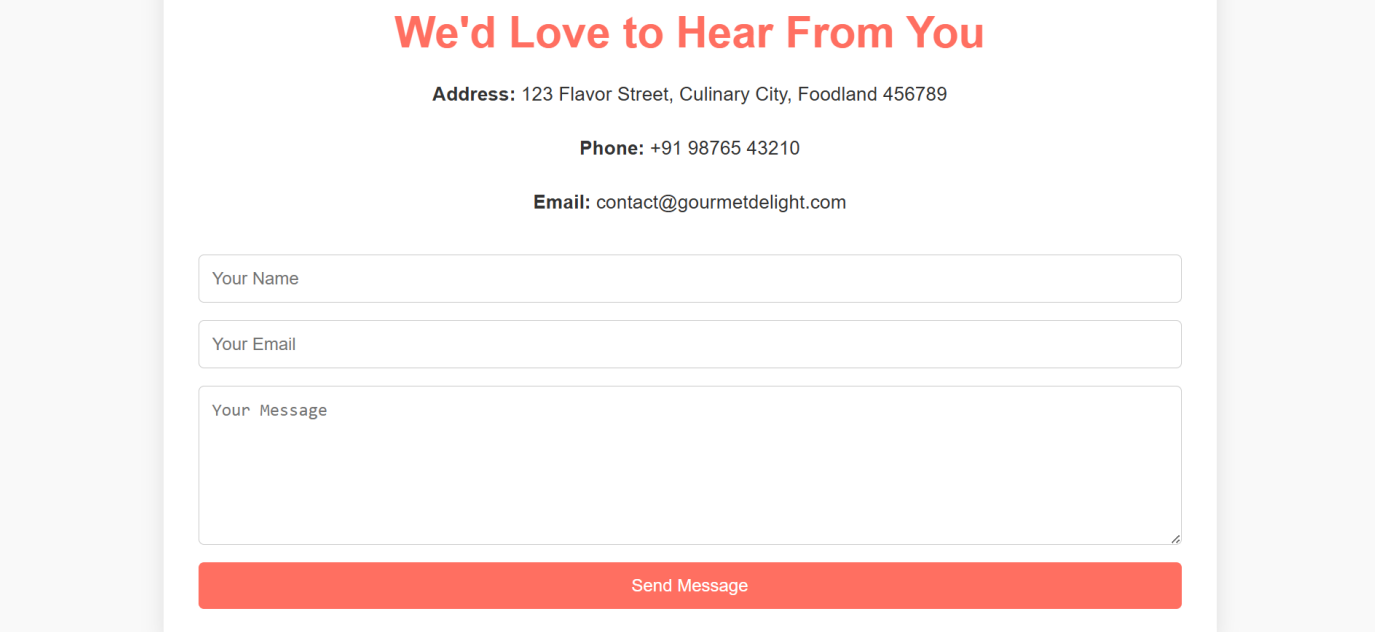
code:

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

```
    <header>
        <h1>Contact Us</h1>
    </header>

    <section class="contact-section">
        <h2>We'd Love to Hear From You</h2>

        <div class="contact-info">
            <p><strong>Address:</strong> 123 Flavor Street, Culinary City, Foodland 456789</p>
            <p><strong>Phone:</strong> +91 98765 43210</p>
            <p><strong>Email:</strong> contact@gourmetdelight.com</p>
        </div>

        <form id="contact-form">
            <input type="text" name="name" placeholder="Your Name" required>
            <input type="email" name="email" placeholder="Your Email" required>
            <textarea name="message" placeholder="Your Message" required></textarea>
            <button type="submit">Send Message</button>
        </form>
    </section>

    <footer>
        <p>&copy; 2023 Gourmet Delight. All rights reserved.</p>
    </footer>

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

Output:

E. contact us page output:



Code:

F. registration page

code:

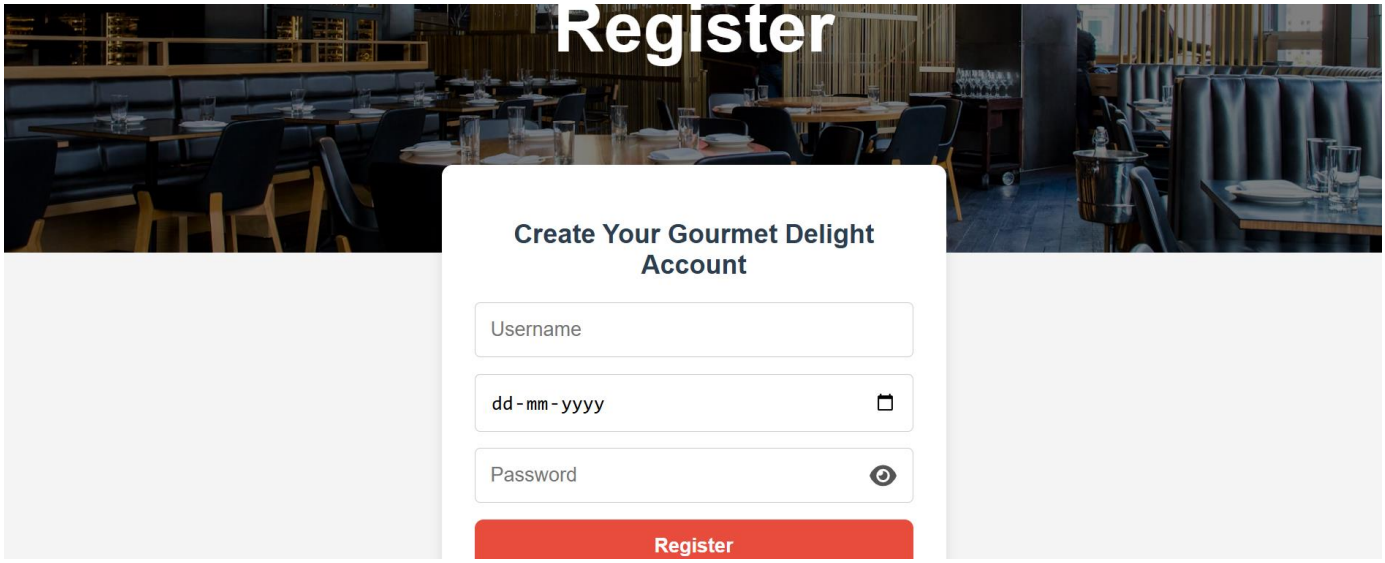
```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
  <title>Gourmet Delight - Register</title>
  <link rel="stylesheet" href="register.css" />
</head>
<body>
  <nav>
    <a href="index.html">Home</a>
    <a href="login.html">Login</a>
  </nav>

  <!-- Hero Header -->
  <header>
```

```
  <h1>Register</h1>
 </header>

 <!-- Register Form -->
 <div class="register-container">
  <h2>Create Your Gourmet Delight Account</h2>
  <form id="register-form">
   <div class="input-container">
    <input type="text" id="username" placeholder="Username" required oninput="validateUsername()">
    <span class="error-text" id="username-error">Username must be at least 6 characters and only contain letters, numbers, or @ # $</span>
   </div>

   <div class="input-container">
    <input type="date" id="dob" required>
   </div>

   <div class="input-container">
    <div class="password-container">
     <input type="password" id="password" placeholder="Password" required oninput="validatePassword()">
     <i class="toggle-password fa fa-eye" onclick="togglePassword()"></i>
    </div>
    <span class="error-text" id="password-error">Password must be at least 6 characters with letters, numbers, and @ # $ % & !</span>
   </div>

   <button type="submit">Register</button>
  </form>
  <a href="login.html" class="back">Already have an account? Login</a>
 </div>
```

```
  <script src="register.js"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.5.1/js/all.min.js"></script>
</body>
</html>
```

Output:

F. registration page output:



G. login page:

code:

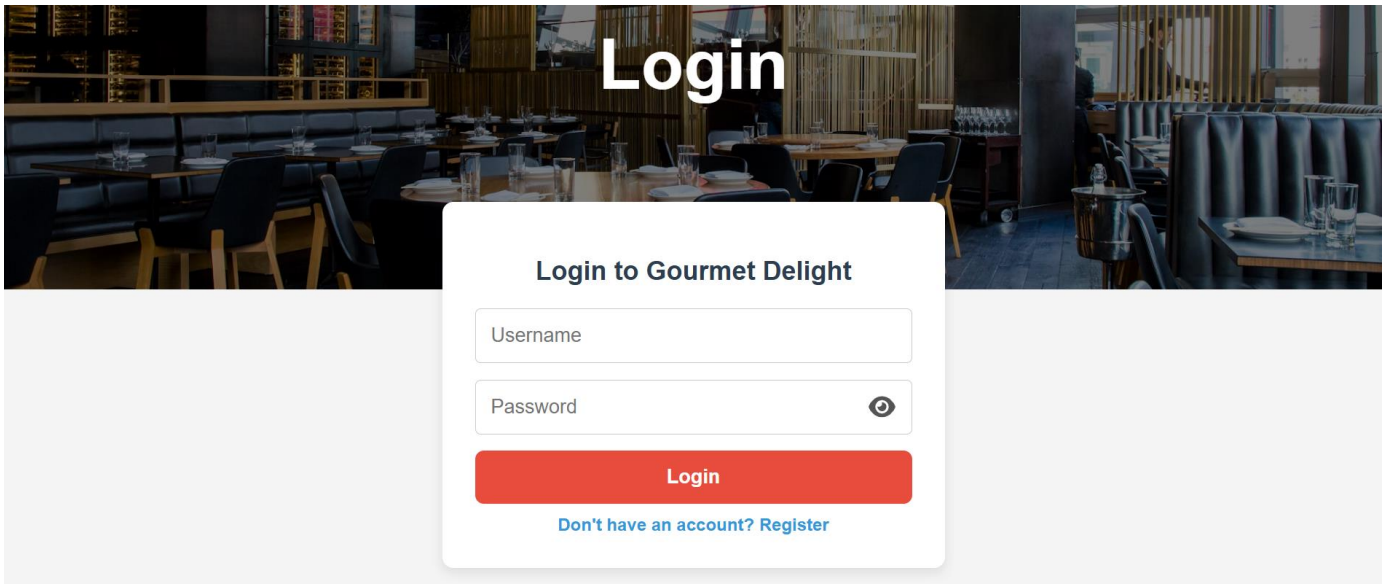
```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Gourmet Delight - Login</title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>

  <!-- Navigation Bar -->
  <nav>
    <a id="login" href="login.html">Login</a>
```

```
  <a id="register" href="register.html">| Register</a>
  <span id="welcome"></span>
 </nav>

 <!-- Hero Header -->
 <header>
  <h1>Login</h1>
 </header>

 <!-- Login Form -->
 <div class="login-container">
  <h2>Login to Gourmet Delight</h2>
  <form onsubmit="login(event)">
   <div class="input-container">
    <input type="text" id="username" placeholder="Username" required oninput="validateUsername()">
    <span class="error-text" id="username-error">Username must be at least 6 characters</span>
   </div>

   <div class="input-container">
    <div class="password-container">
     <input type="password" id="password" placeholder="Password" required oninput="validatePassword()">
     <i class="toggle-password fa fa-eye" onclick="togglePassword()"></i>
    </div>
    <span class="error-text" id="password-error">Password must be at least 8 characters and contain @, #, or $</span>
   </div>

   <button type="submit">Login</button>
  </form>
  <a href="register.html" class="back">Don't have an account? Register</a>
 </div>
```

```
  <!-- Scripts -->
  <script src="script.js"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.5.1/js/all.min.js"></script>
</body>
</html>
```

Output:

G. login page output:



Conclusion

The Hotel Menu System website effectively blends essential digital features with a user-friendly dining experience. By implementing structured web design, intuitive navigation, and clear menu categorization, the platform offers seamless functionality for both customers and admins. It sets a foundation for digital ordering, efficient menu management, and improved customer engagement. With future backend integration, this system can evolve into a fully functional food ordering platform that supports sustainable, paperless operations and enhances the hospitality experience.

Experiment No.3

Problem Statement

Enhance the layout of the shopping platform website using CSS Grid for the homepage. Use CSS Grid to structure blog listings and organize categories with headings, spacing, images, summaries, and reading links.

Theory: CSS Grid for Hotel Menu System

Introduction to CSS Grid

CSS Grid Layout is a powerful two-dimensional layout system that provides flexibility in designing complex web layouts. Unlike Flexbox, which works on a one-dimensional axis (either rows or columns), CSS Grid allows for simultaneous control over both rows and columns, making it highly efficient for structuring content with a defined grid system. This capability is especially beneficial for websites that require a clear, organized layout for displaying information, such as a hotel menu system.

In a hotel menu system, items like appetizers, main courses, desserts, and drinks need to be categorized clearly and displayed in an organized manner. With CSS Grid, it becomes easier to arrange these items into visually appealing grids that ensure both readability and accessibility. By using a grid structure, the menu items can be aligned neatly, with ample space between them, creating a clean and structured presentation.

Why CSS Grid for Hotel Menu System?

1. Organized Layout:
   A hotel menu typically includes various categories of food and beverages. CSS Grid allows for a clean and structured layout, enabling designers to categorize menu items efficiently. For example, appetizers can be placed in one section, while main courses and desserts can be aligned in separate grid rows. This structured approach not only improves the visual appeal but also ensures that the customers can easily navigate through the menu, find what they are looking for, and make informed decisions.
2. Responsiveness:
   One of the most significant benefits of using CSS Grid for a hotel menu system is its responsiveness. The design can easily adapt to different screen sizes, from large desktop monitors to smaller mobile devices. By utilizing features such as auto-fit, minmax(), and media queries, the layout can automatically adjust the number of columns and rows based on the available screen space, ensuring that the menu looks great on all devices without any extra coding.
   - On larger screens, the menu can have multiple columns, displaying items side by side.
   - On smaller screens, the grid can automatically rearrange to display items in a single column or two columns, ensuring a seamless mobile experience.

This mobile-first approach is critical for modern websites, as many users will likely access the menu from their smartphones or tablets.

3. Consistency:
   Consistency is crucial in any design, and CSS Grid provides a consistent layout by aligning items in rows and columns. This eliminates the chaos of misaligned elements and ensures that all content, such as item images, titles, descriptions, and prices, is uniformly distributed across the screen. Using Grid, each section of the menu (e.g., appetizers, main courses, drinks) can have equal space, and the items within those sections will align neatly, creating a professional and appealing design.
4. Flexibility:
   CSS Grid is highly flexible, allowing designers to make adjustments based on their specific needs. For instance, a grid can be adjusted to accommodate different numbers of menu items, or special sections like "Chef's Special" or "Seasonal Dishes" can be highlighted using grid properties. CSS Grid also allows for easy insertion of new items in the menu without disrupting the overall layout, making it easier to update the website as the menu changes over time.

5. Cleaner User Experience:
   CSS Grid promotes a cleaner and more intuitive user experience. When users visit the website, the organized layout helps them find what they are looking for quickly. Using a grid structure ensures that the menu is both readable and aesthetically pleasing. The layout also contributes to faster load times, as items are organized in a way that optimizes rendering speed on both mobile and desktop devices.

Benefits of Using CSS Grid in a Hotel Menu System:

1. Better Control Over Layout:
   CSS Grid gives designers more control over the layout compared to traditional methods, like floats or inline-blocks. It allows for precise control over the placement of items within rows and columns, which is essential for ensuring a consistent and professional-looking design.
2. Easy Scalability:
   A hotel menu is subject to change over time as new dishes are added, or existing items are modified. CSS Grid makes it easy to scale the layout. New items can be added to the grid without disrupting the existing content, ensuring that the layout remains intact. This scalability is particularly beneficial in the context of an online menu system, where updates and changes happen frequently.
3. Clean and Consistent User Experience:
   With CSS Grid, all items on the menu will align properly, making the user experience smooth and pleasant. Whether the user is browsing the menu on a phone or a desktop, they will encounter consistent spacing and alignment, which enhances usability and reduces the cognitive load when navigating the menu.
4. Mobile Optimization:
   Given that a significant portion of users will access the hotel menu via smartphones, having a mobile-optimized design is crucial. CSS Grid facilitates the creation of responsive layouts that adjust seamlessly to various screen sizes. The layout can transform from multiple columns on larger screens to single or double columns on smaller screens, ensuring that the menu is always readable and navigable on any device.
5. Improved Visual Aesthetics:
   The grid system provides an organized and aesthetically pleasing way to present the hotel's menu. By controlling the alignment of items and maintaining consistent spacing, CSS Grid helps achieve a polished, professional look. This improves the overall appearance of the website, making it more appealing to customers and giving it a modern, clean design.

1. Index Page (Home Page)

The **index page** of the Hotel Menu System is typically the first point of contact for users. This page should provide an engaging introduction to the hotel's offerings and create a welcoming experience.

CSS Considerations:

- **Layout**: A clean, structured layout using CSS Grid or Flexbox will help organize the content in a way that feels intuitive. For example, the **hero section** might take up most of the screen space with a large background image, and below it, the menu categories can be displayed in grid or card style.
- **Typography**: Choosing the right fonts is essential for readability and branding. Use CSS to ensure text is legible with appropriate line heights and font sizes. For headings, you might want a bold, large font, while body text can be smaller and simpler.
- **Color Scheme**: CSS allows you to choose a color scheme that aligns with the brand of the hotel or restaurant. Use contrasting colors for CTAs (like buttons) and harmonious tones for background and text to ensure readability.

2. Product Page (Menu Page)

The **product (menu) page** will display all the available food and drink items on the hotel's menu. CSS ensures that these items are neatly arranged and accessible to the user.

CSS Considerations:

- **Grid Layout**: Use CSS Grid to create a clean, responsive layout for the menu items. Each menu item can be placed in individual grid items (cards) with a title, image, and description.
- **Hover Effects**: Adding hover effects to menu items or buttons using CSS helps improve interactivity. For example, when the user hovers over a menu item, the background color can change, or a subtle shadow can appear to provide feedback.

3. Cart Page

The **cart page** shows users the items they have selected and allows them to update quantities or remove items.

CSS Considerations:

- **Table Layout**: Use CSS Grid or Flexbox to display cart items in a tabular form. Each row will represent a menu item, with columns for quantity, price, and total.
- **Responsive Design**: Ensure the cart page is responsive, especially when viewed on mobile devices. The cart layout should adjust to a single column or a smaller grid on small screens.

4. Login and Register Pages

CSS for the **login** and **register** pages should create a clean, simple form layout that is easy for users to navigate.

CSS Considerations:

- **Form Layout**: Use Flexbox or Grid to center forms on the page and space out input fields for an optimal user experience.
- **Form Validation**: CSS can be used to provide visual feedback (e.g., changing border color) when an input field is invalid.

5. About Page

The **about page** provides information about the hotel, its mission, and history. Use CSS to present this information in a visually appealing way.

CSS Considerations:

- **Layout**: Use Grid or Flexbox to create a clean and balanced layout. Sections like "About the Hotel," "Our Mission," and "Meet the Team" should be well-spaced and easy to read.
- **Typography**: Use larger fonts for headings and smaller fonts for body text to create a hierarchy.

Code:

A. Homepage(index)

code:

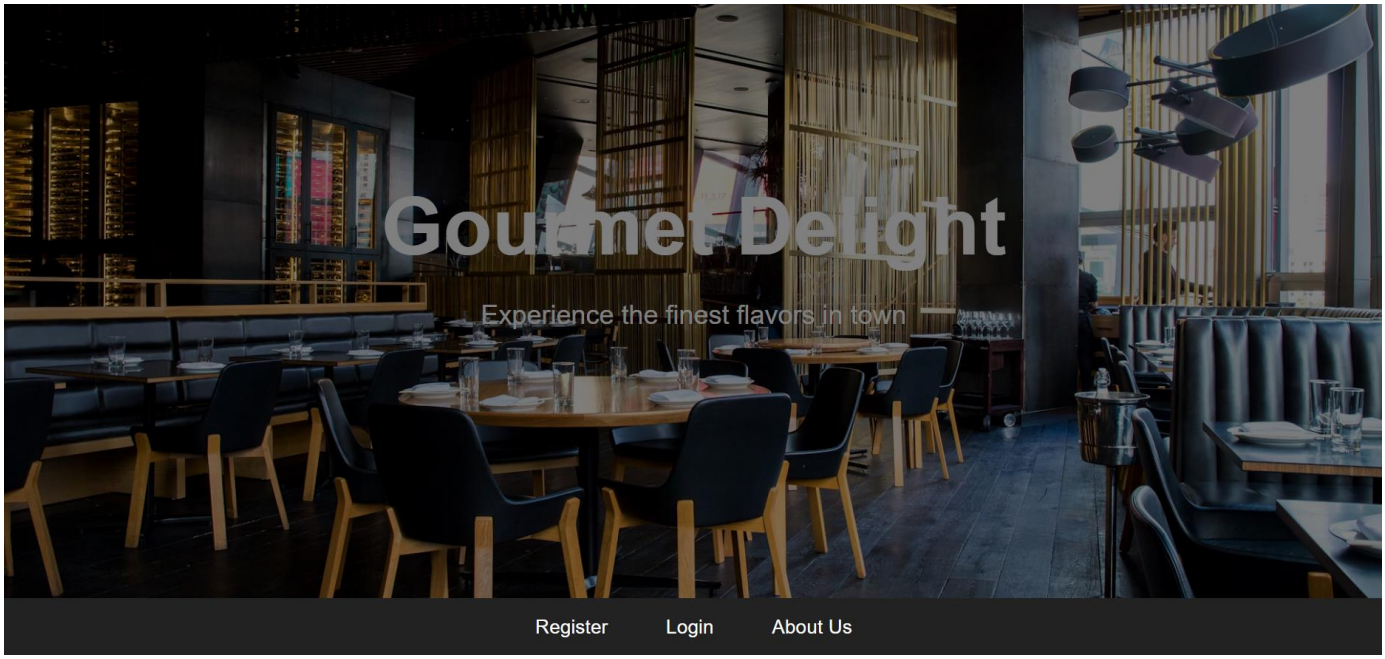
```
body {
  font-family: 'Poppins', sans-serif;
  margin: 0;
  padding: 0;
  background-color: #f9f9f9;
  color: #333;
```

```
}

header {
  background: url('https://images.unsplash.com/photo-1517248135467-4c7edcad34c4?ixlib=rb-1.2.1&auto=format&fit=crop&w=1950&q=80') no-repeat center center/cover;
  height: 100vh;
  display: flex;
  justify-content: center;
  align-items: center;
  color: #fff;
  text-align: center;
  position: relative;
}

header::before {
  content: '';
  position: absolute;
  top: 0;
  left: 0;
  width: 100%;
  height: 100%;
  background: rgba(0, 0, 0, 0.5);
}

header h1 {
  font-size: 4.5rem;
  margin: 0;
  z-index: 1;
```

```
  animation: fadeIn 2s ease-in-out;
}

header p {
  font-size: 1.5rem;
  z-index: 1;
  animation: fadeIn 3s ease-in-out;
}

@keyframes fadeIn {
  from { opacity: 0; }
  to { opacity: 1; }
}

nav {
  background: #222;
  padding: 1rem 0;
  text-align: center;
  position: sticky;
  top: 0;
  z-index: 1000;
}

nav a, #welcome {
  color: #fff;
  text-decoration: none;
  margin: 0 1.5rem;
  font-size: 1.1rem;
```

```
  transition: color 0.3s ease;
}

nav a:hover, #logout:hover {
  color: #ff6f61;
}

#logout {
  cursor: pointer;
}

footer {
  background: #222;
  color: #fff;
  text-align: center;
  padding: 2rem 0;
  margin-top: 4rem;
}

footer p {
  margin: 0;
}
```

Output:



B. Product page(menu)

code:

```
body {
  font-family: 'Poppins', sans-serif;
  margin: 0;
  padding: 0;
  background-color: #f9f9f9;
  color: #333;
}


header {
  background: url('https://images.unsplash.com/photo-1517248135467-4c7edcad34c4?ixlib=rb-1.2.1&auto=format&fit=crop&w=1950&q=80') no-repeat center center/cover;
  height: 70vh;
  display: flex;
  justify-content: center;
  align-items: center;
  color: #fff;
```

```
  text-align: center;
  position: relative;
}

header::before {
  content: '';
  position: absolute;
  top: 0;
  left: 0;
  width: 100%;
  height: 100%;
  background: rgba(0, 0, 0, 0.5);
}

header h1 {
  font-size: 4rem;
  z-index: 1;
  animation: fadeIn 2s ease-in-out;
  margin: 0;
}

@keyframes fadeIn {
  from { opacity: 0; }
  to { opacity: 1; }
}

.home-button {
  position: absolute;
```

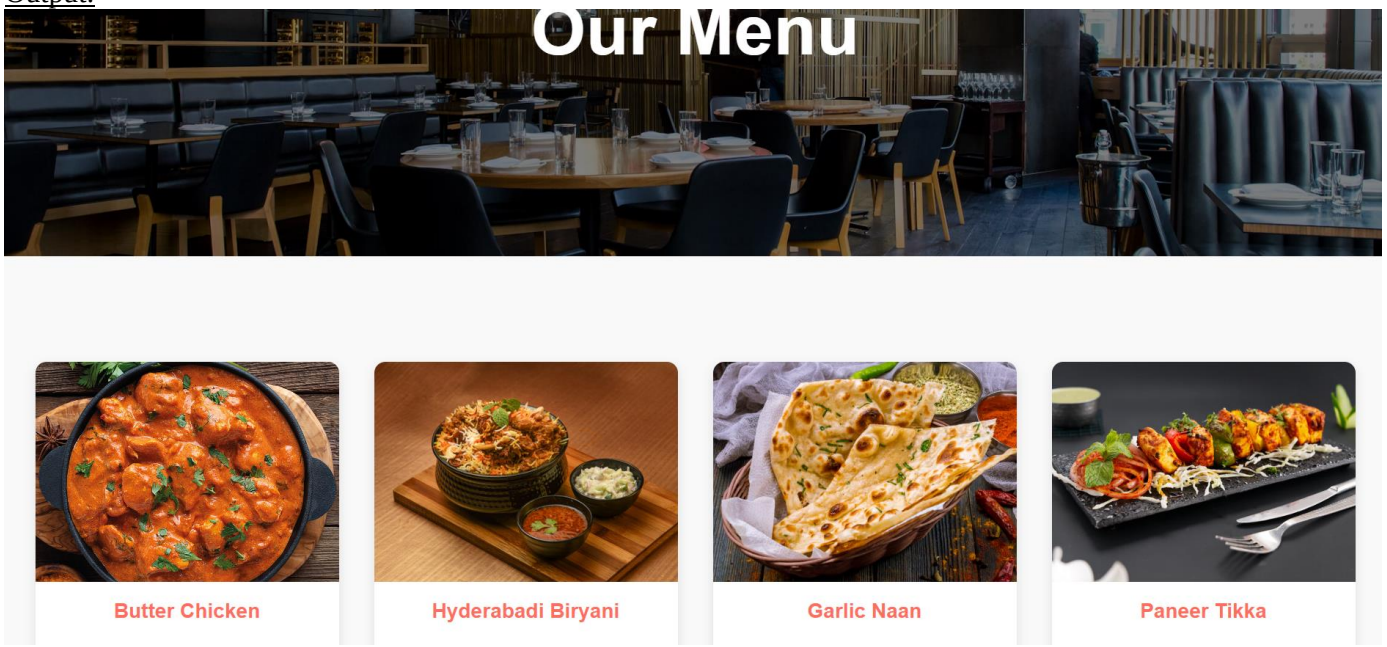
```
  top: 20px;
  left: 20px;
}

.home-button a {
  background: white;
  color: black;
  padding: 10px 15px;
  border-radius: 5px;
  text-decoration: none;
}

.cart-container {
  position: absolute;
  top: 20px;
  right: 20px;
}

.cart-icon {
  background: white;
  padding: 10px;
  border-radius: 50%;
  position: relative;
}

.cart-icon i {
  color: black;
}
```

```
.cart-count {
  position: absolute;
  top: -8px;
  right: -8px;
  background: black;
  color: white;
  border-radius: 50%;
  padding: 2px 6px;
  font-size: 12px;
}

.menu-section {
  padding: 4rem 2rem;
  max-width: 1200px;
  margin: auto;
  text-align: center;
}

.menu-grid {
  display: grid;
  grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));
  gap: 2rem;
  margin-top: 2rem;
}

.menu-item {
  background: #fff;
```

```
  border-radius: 10px;
  box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
  overflow: hidden;
  transition: transform 0.3s ease;
  display: flex;
  flex-direction: column;
  justify-content: space-between;
}

.menu-item:hover {
  transform: translateY(-5px);
}

.menu-item img {
  width: 100%;
  height: 200px;
  object-fit: cover;
}

.menu-item h3 {
  color: #ff6f61;
  margin: 1rem 0 0.5rem;
}

.menu-item p {
  padding: 0 1rem;
  font-size: 0.95rem;
}
```

```
.add-to-cart {
  background: grey;
  color: black;
  border: none;
  padding: 10px 15px;
  margin: 1rem auto;
  border-radius: 5px;
  cursor: pointer;
  font-size: 0.9rem;
  transition: background 0.3s ease;
}

.add-to-cart:hover {
  background: #333;
}

footer {
  background: #222;
  color: #fff;
  text-align: center;
  padding: 2rem 0;
  margin-top: 4rem;
}

footer p {
  margin: 0;
}
```

Output:



C. Cart page

code:

```
body {
  font-family: 'Poppins', sans-serif;
  text-align: center;
  background-color: #f4f4f4;
  margin: 0;
  padding: 20px;
}
.cart-container {
  max-width: 700px;
  margin: auto;
  background: white;
  padding: 20px;
  border-radius: 15px;
  box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
}
```

```
.cart-item {
  display: flex;
  justify-content: space-between;
  align-items: center;
  padding: 15px;
  border-bottom: 1px solid #ddd;
  font-size: 18px;
}
.remove-btn {
  background: #ff4d4d;
  color: white;
  border: none;
  padding: 7px 12px;
  border-radius: 7px;
  cursor: pointer;
  transition: background 0.3s;
}
.remove-btn:hover {
  background: #cc0000;
}
.total {
  font-size: 22px;
  font-weight: bold;
  margin-top: 15px;
}
.checkout-btn {
  margin-top: 20px;
  padding: 12px 24px;
```

```
  background: #28a745;
  color: white;
  font-size: 18px;
  border: none;
  border-radius: 8px;
  cursor: pointer;
  transition: background 0.3s;
}
.checkout-btn:hover {
  background: #218838;
}
.bill {
  margin-top: 25px;
  text-align: left;
  padding: 15px;
  border-radius: 10px;
  background: #f8f9fa;
  box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
}
.bill-item {
  display: flex;
  justify-content: space-between;
  font-size: 18px;
  padding: 5px 0;
}
.product-link {
  text-decoration: none;
  color: #007bff;
```

font-weight: bold;

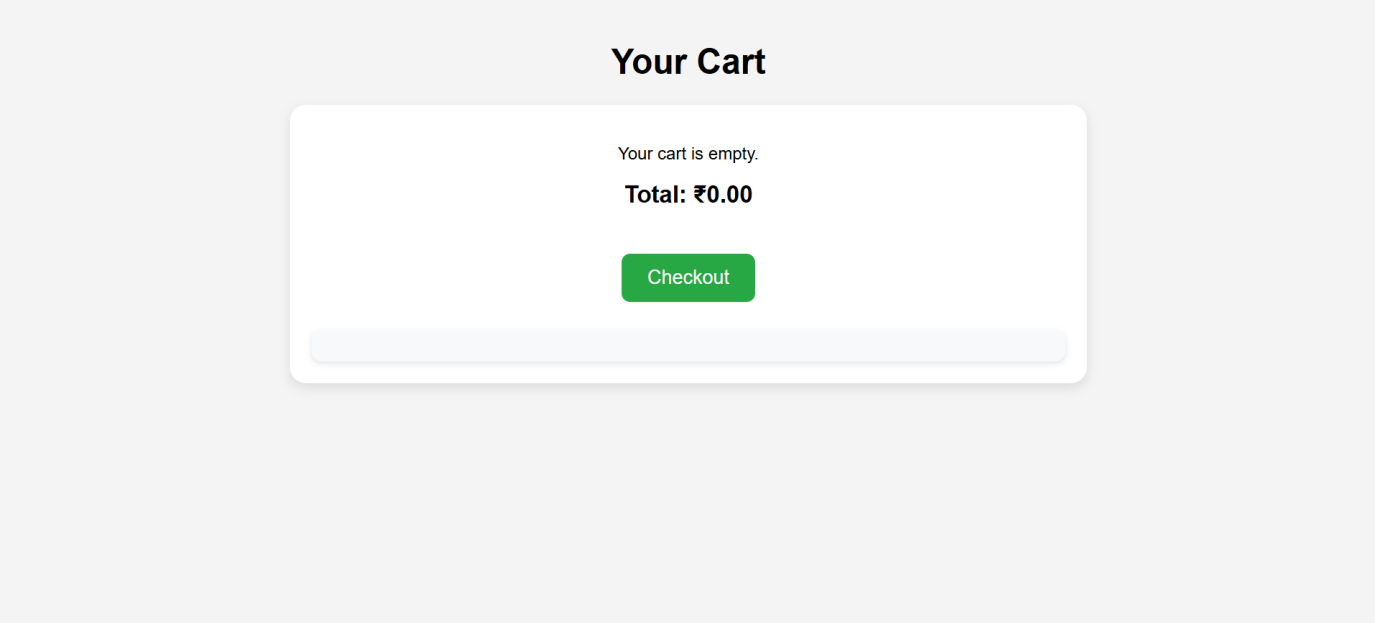
}

.product-link:hover {

text-decoration: underline;

}

Output:



D, About page

code:

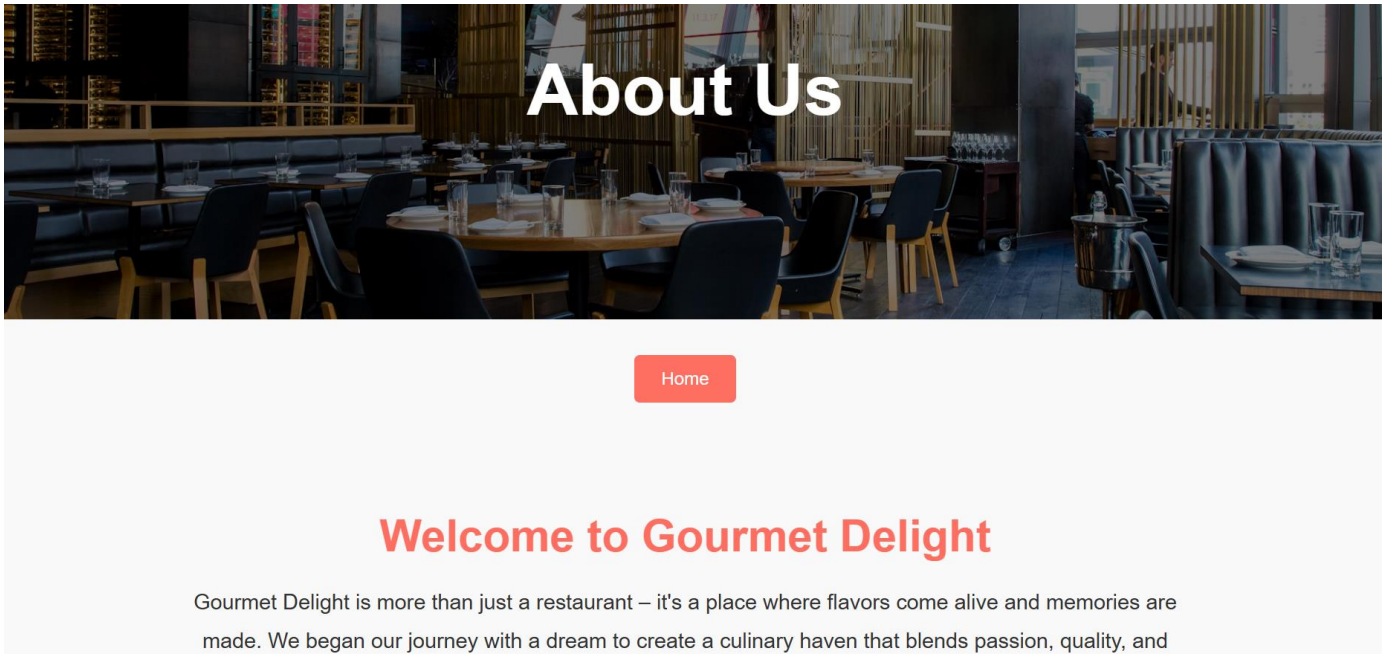
```
body {
    font-family: 'Poppins', sans-serif;
    margin: 0;
    padding: 0;
    background-color: #f9f9f9;
    color: #333;
}

/* Hero Header */
header {
```

```
    background: url('https://images.unsplash.com/photo-1517248135467-4c7edcad34c4?ixlib=rb-1.2.1&auto=format&fit=crop&w=1950&q=80') no-repeat center center/cover;
    height: 70vh;
    display: flex;
    justify-content: center;
    align-items: center;
    color: #fff;
    text-align: center;
    position: relative;
}

header::before {
    content: '';
    position: absolute;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    background: rgba(0, 0, 0, 0.5);
}

header h1 {
    font-size: 4rem;
    z-index: 1;
    animation: fadeIn 2s ease-in-out;
    margin: 0;
}
```

```
@keyframes fadeIn {
    from { opacity: 0; }
    to { opacity: 1; }
}

.home-btn {
    display: inline-block;
    margin: 2rem auto 0;
    background: #ff6f61;
    color: #fff;
    padding: 0.75rem 1.5rem;
    border: none;
    text-decoration: none;
    font-size: 1rem;
    border-radius: 5px;
    transition: background 0.3s ease;
}

.home-btn:hover {
    background: #e85a4f;
}

.about-section {
    padding: 4rem 2rem;
    max-width: 900px;
    margin: auto;
    text-align: center;
}
```

```
.about-section h2 {
    font-size: 2.5rem;
    margin-bottom: 1rem;
    color: #ff6f61;
}

.about-section p {
    font-size: 1.2rem;
    line-height: 1.8;
}

footer {
    background: #222;
    color: #fff;
    text-align: center;
    padding: 2rem 0;
    margin-top: 4rem;
}

footer p {
    margin: 0;
}
```

Output:



E. Contact Us Page

code:

```
body {
    font-family: 'Poppins', sans-serif;
    margin: 0;
    padding: 0;
    background-color: #f9f9f9;
    color: #333;
}


header {
    background: url('https://images.unsplash.com/photo-1504674900247-0877df9cc836?ixlib=rb-1.2.1&auto=format&fit=crop&w=1950&q=80') no-repeat center center/cover;
    height: 70vh;
    display: flex;
    justify-content: center;
    align-items: center;
    color: #fff;
```

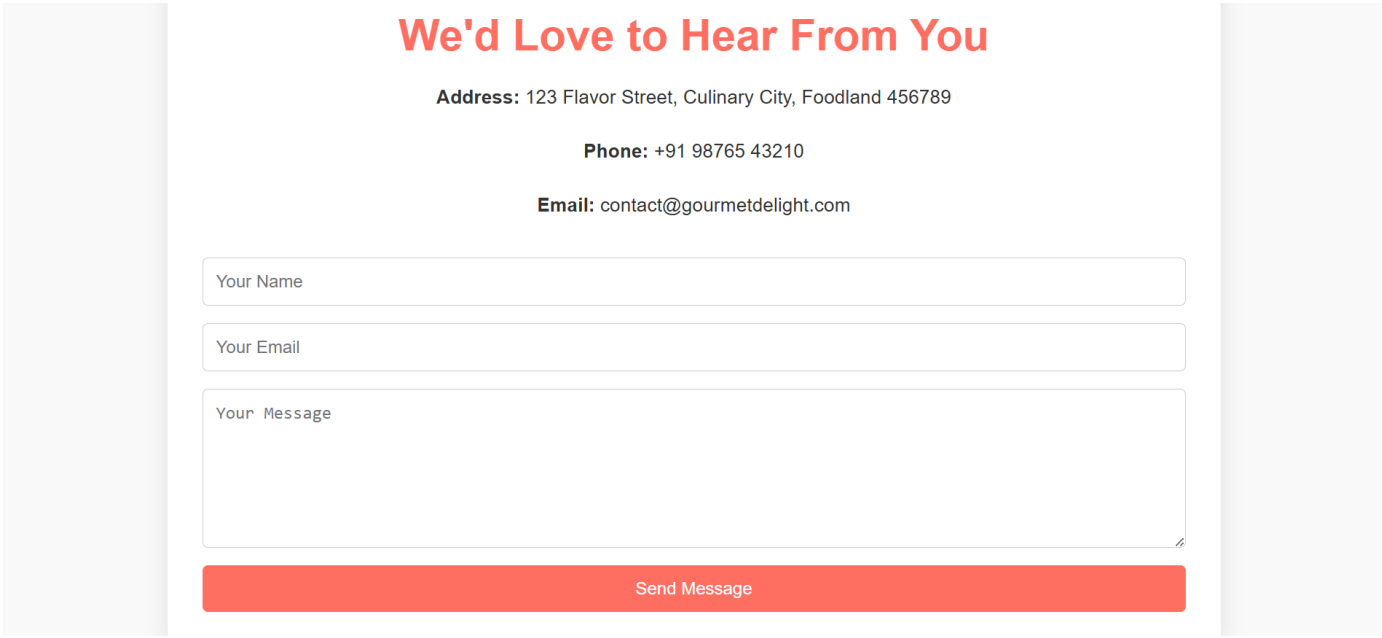
```
    text-align: center;
    position: relative;
}

header::before {
    content: '';
    position: absolute;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    background: rgba(0, 0, 0, 0.5);
}

header h1 {
    font-size: 4rem;
    z-index: 1;
    animation: fadeIn 2s ease-in-out;
    margin: 0;
}

@keyframes fadeIn {
    from { opacity: 0; }
    to { opacity: 1; }
}

.contact-section {
    max-width: 900px;
```

```
    margin: 4rem auto;
    padding: 2rem;
    background: #fff;
    border-radius: 10px;
    box-shadow: 0 0 20px rgba(0,0,0,0.1);
}

.contact-section h2 {
    font-size: 2.5rem;
    color: #ff6f61;
    margin-bottom: 1rem;
    text-align: center;
}

.contact-info {
    margin-bottom: 2rem;
    text-align: center;
}

.contact-info p {
    font-size: 1.1rem;
    line-height: 1.8;
}

form {
    display: flex;
    flex-direction: column;
    gap: 1rem;
```

```
}

input, textarea {
    padding: 0.75rem;
    font-size: 1rem;
    border: 1px solid #ccc;
    border-radius: 5px;
}

textarea {
    resize: vertical;
    min-height: 120px;
}

button {
    padding: 0.75rem;
    font-size: 1rem;
    background-color: #ff6f61;
    color: white;
    border: none;
    border-radius: 5px;
    cursor: pointer;
    transition: background 0.3s ease;
}

button:hover {
    background-color: #e85a4f;
}
```

```
footer {
    background: #222;
    color: #fff;
    text-align: center;
    padding: 2rem 0;
    margin-top: 4rem;
}

footer p {
    margin: 0;
}
```

Output:



F. Register Page:

code:

```
body {
```

```
  font-family: 'Poppins', sans-serif;
  background-color: #f4f4f4;
  margin: 0;
}

nav {
  background: #2c3e50;
  position: fixed;
  top: 0;
  width: 100%;
  display: flex;
  justify-content: center;
  padding: 15px;
  z-index: 1000;
}

nav a {
  color: white;
  text-decoration: none;
  padding: 12px 20px;
  font-size: 18px;
  transition: 0.3s;
}

nav a:hover {
  background: #e74c3c;
  border-radius: 5px;
}
```

```
header {
  background: url('https://images.unsplash.com/photo-1517248135467-4c7edcad34c4?ixlib=rb-1.2.1&auto=format&fit=crop&w=1950&q=80') no-repeat center center/cover;
  height: 70vh;
  display: flex;
  justify-content: center;
  align-items: center;
  color: #fff;
  text-align: center;
  position: relative;
  margin-top: 60px;
}

header::before {
  content: '';
  position: absolute;
  top: 0;
  left: 0;
  width: 100%;
  height: 100%;
  background: rgba(0, 0, 0, 0.5);
}

header h1 {
  font-size: 4rem;
  z-index: 1;
  animation: fadeIn 2s ease-in-out;
```

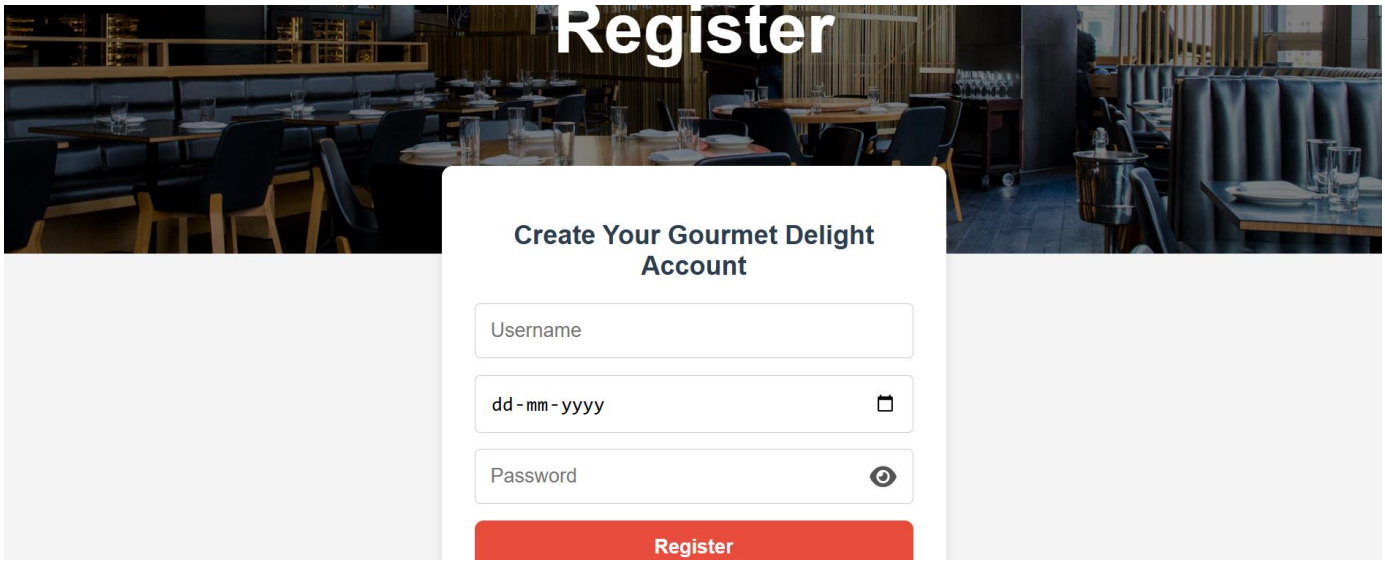
```
  margin: 0;
}

@keyframes fadeIn {
  from { opacity: 0; }
  to { opacity: 1; }
}

.register-container {
  background: white;
  padding: 30px;
  border-radius: 10px;
  color: #333;
  width: 400px;
  text-align: center;
  box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
  margin: -80px auto 40px;
  position: relative;
  z-index: 10;
}

.register-container h2 {
  color: #2c3e50;
  margin-bottom: 20px;
}

.input-container {
  position: relative;
```

```
  text-align: left;
  margin-bottom: 15px;
}

input {
  width: 100%;
  padding: 14px;
  border-radius: 5px;
  border: 1px solid #ccc;
  font-size: 18px;
  display: block;
  transition: border 0.3s;
  box-sizing: border-box;
}

input.correct {
  border: 2px solid green;
}

input.incorrect {
  border: 2px solid red;
}

.password-container {
  position: relative;
  width: 100%;
}
```

```
.toggle-password {
  position: absolute;
  right: 15px;
  top: 50%;
  transform: translateY(-50%);
  cursor: pointer;
  font-size: 22px;
  color: #555;
}

.error-text {
  font-size: 14px;
  color: red;
  margin-top: 5px;
  display: none;
}

button {
  width: 100%;
  padding: 14px;
  background: #e74c3c;
  color: white;
  border: none;
  border-radius: 8px;
  cursor: pointer;
  font-size: 18px;
  font-weight: bold;
  transition: 0.3s;
```

```
  box-sizing: border-box;
}

button:hover {
  background: #c0392b;
  transform: scale(1.05);
}

.back {
  color: #3498db;
  text-decoration: none;
  display: block;
  margin-top: 10px;
  font-weight: bold;
}
.back:hover {
  text-decoration: underline;
}
```

Output:



G. Login page

code:

```
body {
  font-family: 'Poppins', sans-serif;
  background-color: #f4f4f4;
  margin: 0;
}

nav {
  background: #2c3e50;
  position: fixed;
  top: 0;
  width: 100%;
  display: flex;
  justify-content: center;
  align-items: center;
  padding: 15px;
  z-index: 1000;
  gap: 15px;
}

nav a {
  color: white;
  text-decoration: none;
  padding: 12px 20px;
  font-size: 18px;
  transition: 0.3s;
```

```
}

nav a:hover {
  background: #e74c3c;
  border-radius: 5px;
}

#welcome {
  color: white;
  font-weight: bold;
  font-size: 18px;
}

header {
  background: url('https://images.unsplash.com/photo-1517248135467-4c7edcad34c4?ixlib=rb-1.2.1&auto=format&fit=crop&w=1950&q=80') no-repeat center center/cover;
  height: 70vh;
  display: flex;
  justify-content: center;
  align-items: center;
  color: #fff;
  text-align: center;
  position: relative;
  margin-top: 60px;
}

header::before {
  content: '';
```

```
  position: absolute;
  top: 0;
  left: 0;
  width: 100%;
  height: 100%;
  background: rgba(0, 0, 0, 0.5);
}

header h1 {
  font-size: 4rem;
  z-index: 1;
  animation: fadeIn 2s ease-in-out;
  margin: 0;
}

@keyframes fadeIn {
  from { opacity: 0; }
  to { opacity: 1; }
}

.login-container {
  background: white;
  padding: 30px;
  border-radius: 10px;
  color: #333;
  width: 400px;
  text-align: center;
  box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
```

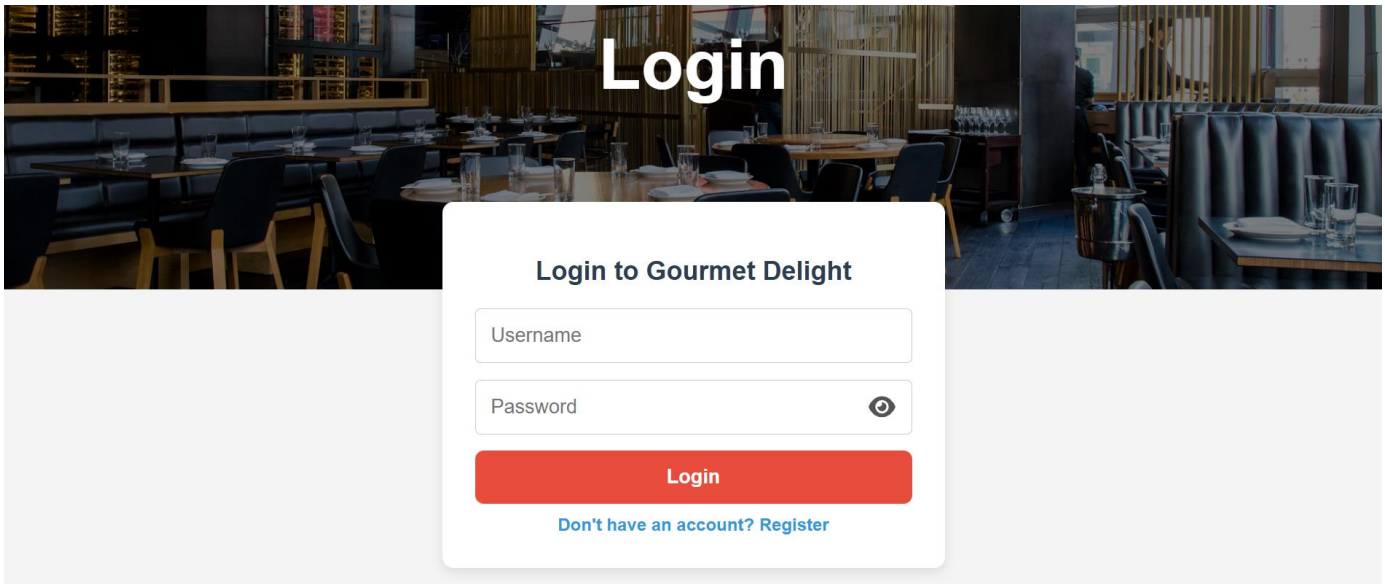
```
  margin: -80px auto 40px;
  position: relative;
  z-index: 10;
}

.login-container h2 {
  color: #2c3e50;
  margin-bottom: 20px;
}

.input-container {
  position: relative;
  text-align: left;
  margin-bottom: 15px;
}

input {
  width: 100%;
  padding: 14px;
  border-radius: 5px;
  border: 1px solid #ccc;
  font-size: 18px;
  display: block;
  transition: border 0.3s;
  box-sizing: border-box;
}

input.correct {
```

```
  border: 2px solid green;
}

input.incorrect {
  border: 2px solid red;
}

.password-container {
  position: relative;
  width: 100%;
}

.toggle-password {
  position: absolute;
  right: 15px;
  top: 50%;
  transform: translateY(-50%);
  cursor: pointer;
  font-size: 22px;
  color: #555;
}

.error-text {
  font-size: 14px;
  color: red;
  margin-top: 5px;
  display: none;
}
```

```
button {
  width: 100%;
  padding: 14px;
  background: #e74c3c;
  color: white;
  border: none;
  border-radius: 8px;
  cursor: pointer;
  font-size: 18px;
  font-weight: bold;
  transition: 0.3s;
  box-sizing: border-box;
}

button:hover {
  background: #c0392b;
  transform: scale(1.05);
}

.back {
  color: #3498db;
  text-decoration: none;
  display: block;
  margin-top: 10px;
  font-weight: bold;
}
```

```
.back:hover {

  text-decoration: underline;

}
```

Output:



Conclusion

Incorporating CSS Grid into the design of a hotel menu system allows for a more structured, flexible, and responsive layout. It enhances the overall user experience by ensuring that the menu is easy to navigate, looks great across all devices, and can be easily updated as the menu evolves. By using CSS Grid, hotel websites can deliver a high-quality, professional-looking digital menu that meets the needs of today's diverse, tech-savvy customers.

Experiment No.4

CSS Theory: Enhancing and Styling Key Pages in the Hotel Menu System Website

1. Why CSS Styling Matters in the Hotel Menu System

CSS plays a crucial role in the design and usability of a Hotel Menu System Website. A well-designed menu page ensures that visitors have a seamless and enjoyable experience while navigating through food options, making orders, and interacting with the platform. CSS helps:

- Create a visually appealing layout: A clean design makes the menu easy to navigate, enhancing user experience.
- Organize content effectively: Grouping similar dishes together with adequate spacing ensures that the menu is easy to browse.
- Ensure responsiveness: A mobile-friendly design ensures that users can view the menu on any device, from mobile phones to desktops.
- Build trust: A well-structured, aesthetically pleasing page makes users feel confident in their choice of restaurant.

Page-wise CSS Styling Theory

1. Add to Cart / Saved Items Page (Cart Equivalent)

The Add to Cart page allows users to review their selected dishes before making the final purchase. Proper styling ensures the page is easy to read and navigate.

Key Styling Techniques:

- Spacing and Margins: Adding adequate spacing around each cart item ensures that the content doesn't feel cramped and is easier to scan.
- Card-Style Layout: Each item in the cart can be displayed in a card-style layout with subtle shadows, making the page feel more organized.
- Product Name and Features: Make the product name stand out by using bold fonts. Use padding to provide space between the text and the item’s border.
- Hover Effects for Buttons: Apply hover effects on buttons such as "Remove" to make the page feel interactive and dynamic.

Result: A visually organized, user-friendly cart that encourages easy decision-making with clear separation of product information and action buttons.

2. About Us Page

The About Us page tells the story of the hotel, its mission, and its team. This page builds connection and trust with users by showcasing the hotel’s values and the people behind it.

Key Styling Techniques:

- Increased Line Height: Enhance readability by using higher line heights for paragraphs. This prevents the text from feeling too dense.

- Grid or Flexbox for Team Layout: Use a grid or flexbox layout to arrange team members' photos side by side, ensuring a clean and symmetrical look.
- Rounded Photos with Hover Effects: Round the team members' photos and add hover effects, such as a subtle zoom or display of name and position.
- Soft Boxes for Quotes: Style quotes or important values in soft, colored sections or boxes, adding visual appeal and helping the text stand out.

Result: A friendly and professional layout that builds trust with users, highlighting the hotel's values and commitment.

3. Product (Menu) Page

The Product (Menu) page displays all available dishes and drinks offered by the hotel. Proper styling helps organize these items and makes it easy for users to browse through the menu.

Key Styling Techniques:

- Grid Layout for Menu Items: Use CSS Grid to display menu items in a responsive grid format. This ensures the menu adapts to various screen sizes, from mobile devices to desktops.
- Card-Style Design: Each dish should be in a card-style layout, making it easy to distinguish between different dishes. This can include images of the dish, its name, a short description, and price.
- Hover Effects for Interactive Feedback: Add hover effects for each dish to make the website feel interactive. For example, highlighting the dish name or changing the background color when the user hovers over a dish.

Result: A clean, visually structured, and interactive product page that encourages users to browse and select dishes with ease.

4. User Registration Page

The Registration page is where users can create an account. CSS styling plays a significant role in making the form visually appealing and easy to use.

Key Styling Techniques:

- Form Layout with Padding: Group input fields into clear sections (such as "Personal Info" and "Login Credentials"). Use padding around the fields to make the form feel less cramped.
- Background Colors and Shadows: Add a subtle background color and box-shadow to the form container to give it depth and ensure it stands out on the page.
- Real-Time Validation: Display validation messages for incorrect or missing information (such as password strength or email format). Use green for successful input and red for errors.

Result: A user-friendly and intuitive registration form that feels trustworthy and easy to complete.

5. User Login Page

The Login page is the entry point for users to access their account. Styling this page effectively helps increase user confidence and accessibility.

Key Styling Techniques:

- Centered Form Layout: Position the login form centrally on the screen, ensuring it's the focal point. Add adequate padding between input fields.
- Error Message Styling: Style error messages clearly (using red for errors and green for success messages) to make them easily noticeable.
- Minimalist Design: Use a clean and minimalist design with simple fonts and input boxes to keep the page easy to read and use.
- "Show Password" Toggle: Include a password visibility toggle button to enhance user experience.

Result: A clear, effective login page that is both functional and visually appealing.

Bonus: Styling Tips Across All Pages

- CSS Variables for Consistency: Use variables for common values like primary colors (--primary-color, --bg-color) to ensure consistency across pages.
- Typography: Choose clean and readable fonts such as Open Sans or Roboto for body text, and use more distinct fonts for headings to create a visual hierarchy.
- Mobile-First Design: Use media queries to adjust layouts for smaller screens, ensuring the site is fully responsive and mobile-friendly.
- Dark Mode: Implement light/dark mode themes in CSS to cater to user preferences, improving accessibility and enhancing the user experience.

Cart page

code:

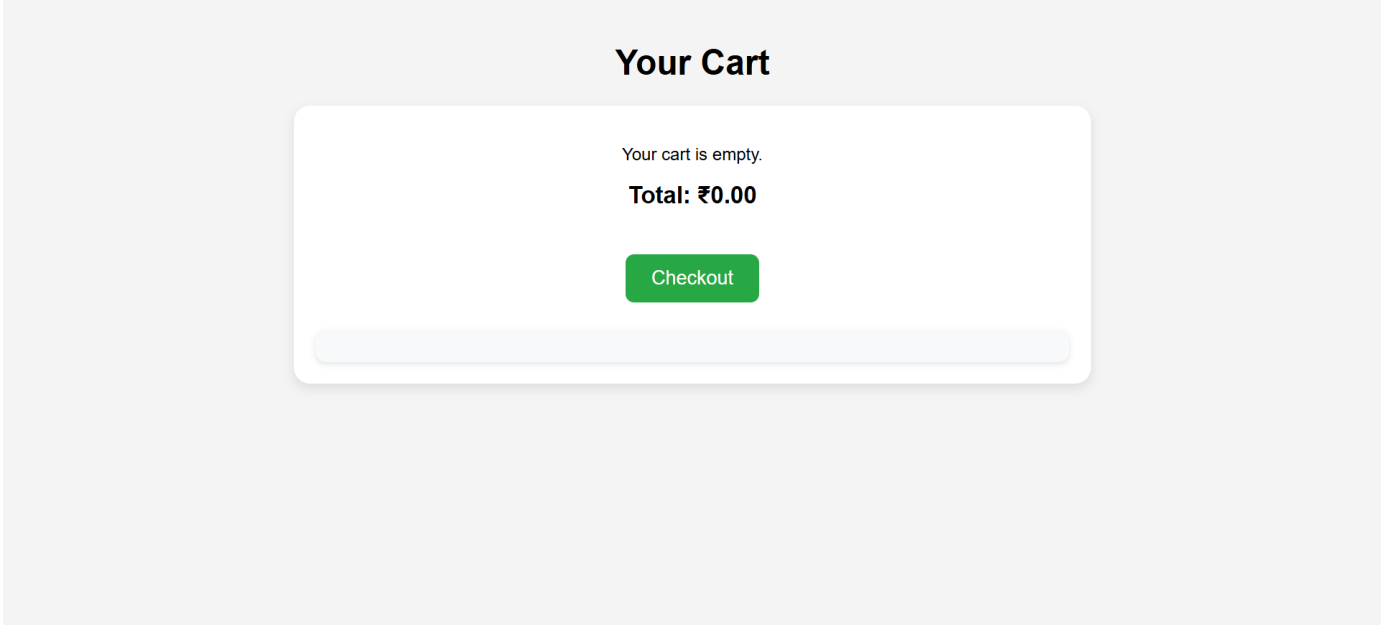
```
body {
  font-family: 'Poppins', sans-serif;
  text-align: center;
  background-color: #f4f4f4;
  margin: 0;
  padding: 20px;
}
.cart-container {
  max-width: 700px;
  margin: auto;
  background: white;
```

```
  padding: 20px;
  border-radius: 15px;
  box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
}
.cart-item {
  display: flex;
  justify-content: space-between;
  align-items: center;
  padding: 15px;
  border-bottom: 1px solid #ddd;
  font-size: 18px;
}
.remove-btn {
  background: #ff4d4d;
  color: white;
  border: none;
  padding: 7px 12px;
  border-radius: 7px;
  cursor: pointer;
  transition: background 0.3s;
}
.remove-btn:hover {
  background: #cc0000;
}
.total {
  font-size: 22px;
  font-weight: bold;
  margin-top: 15px;
```

```
}
.checkout-btn {
  margin-top: 20px;
  padding: 12px 24px;
  background: #28a745;
  color: white;
  font-size: 18px;
  border: none;
  border-radius: 8px;
  cursor: pointer;
  transition: background 0.3s;
}
.checkout-btn:hover {
  background: #218838;
}
.bill {
  margin-top: 25px;
  text-align: left;
  padding: 15px;
  border-radius: 10px;
  background: #f8f9fa;
  box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
}
.bill-item {
  display: flex;
  justify-content: space-between;
  font-size: 18px;
  padding: 5px 0;
```

```
}
.product-link {
  text-decoration: none;
  color: #007bff;
  font-weight: bold;
}
.product-link:hover {
  text-decoration: underline;
}
```

Output:



Login page

code:

```
body {
  font-family: 'Poppins', sans-serif;
  background-color: #f4f4f4;
  margin: 0;
}
```

```
nav {
  background: #2c3e50;
  position: fixed;
  top: 0;
  width: 100%;
  display: flex;
  justify-content: center;
  align-items: center;
  padding: 15px;
  z-index: 1000;
  gap: 15px;
}

nav a {
  color: white;
  text-decoration: none;
  padding: 12px 20px;
  font-size: 18px;
  transition: 0.3s;
}

nav a:hover {
  background: #e74c3c;
  border-radius: 5px;
}

#welcome {
  color: white;
```

```
  font-weight: bold;

  font-size: 18px;

}


header {

  background: url('https://images.unsplash.com/photo-1517248135467-4c7edcad34c4?ixlib=rb-1.2.1&auto=format&fit=crop&w=1950&q=80') no-repeat center center/cover;

  height: 70vh;

  display: flex;

  justify-content: center;

  align-items: center;

  color: #fff;

  text-align: center;

  position: relative;

  margin-top: 60px;

}


header::before {

  content: '';

  position: absolute;

  top: 0;

  left: 0;

  width: 100%;

  height: 100%;

  background: rgba(0, 0, 0, 0.5);

}


header h1 {
```

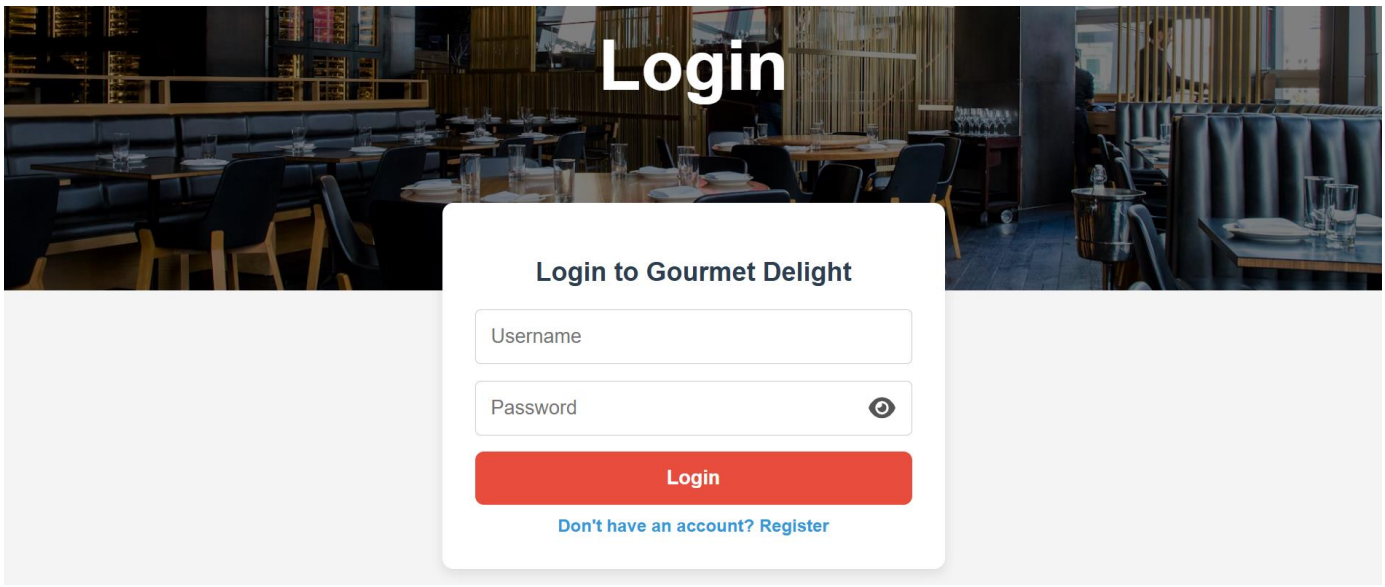
```
  font-size: 4rem;
  z-index: 1;
  animation: fadeIn 2s ease-in-out;
  margin: 0;
}

@keyframes fadeIn {
  from { opacity: 0; }
  to { opacity: 1; }
}

.login-container {
  background: white;
  padding: 30px;
  border-radius: 10px;
  color: #333;
  width: 400px;
  text-align: center;
  box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
  margin: -80px auto 40px;
  position: relative;
  z-index: 10;
}

.login-container h2 {
  color: #2c3e50;
  margin-bottom: 20px;
}
```

```
.input-container {
  position: relative;
  text-align: left;
  margin-bottom: 15px;
}

input {
  width: 100%;
  padding: 14px;
  border-radius: 5px;
  border: 1px solid #ccc;
  font-size: 18px;
  display: block;
  transition: border 0.3s;
  box-sizing: border-box;
}

input.correct {
  border: 2px solid green;
}

input.incorrect {
  border: 2px solid red;
}

.password-container {
  position: relative;
```

```
  width: 100%;
}

.toggle-password {
  position: absolute;
  right: 15px;
  top: 50%;
  transform: translateY(-50%);
  cursor: pointer;
  font-size: 22px;
  color: #555;
}

.error-text {
  font-size: 14px;
  color: red;
  margin-top: 5px;
  display: none;
}

button {
  width: 100%;
  padding: 14px;
  background: #e74c3c;
  color: white;
  border: none;
  border-radius: 8px;
  cursor: pointer;
```

```
  font-size: 18px;
  font-weight: bold;
  transition: 0.3s;
  box-sizing: border-box;
}

button:hover {
  background: #c0392b;
  transform: scale(1.05);
}

.back {
  color: #3498db;
  text-decoration: none;
  display: block;
  margin-top: 10px;
  font-weight: bold;
}

.back:hover {
  text-decoration: underline;
}
```

Output:



Conclusion

CSS is essential for enhancing the design and functionality of a Hotel Menu System Website. By employing grid and flexbox layouts, utilizing card-style designs, adding hover effects, and ensuring mobile responsiveness, you can create an engaging, accessible, and aesthetically pleasing website. CSS helps in structuring content, improving readability, and ensuring that the user experience is seamless across all devices.

Experiment No.5

Problem statement

JavaScript Theory: User Registration, Login, Validation, and Bookmarking Functionality for shopping – shopping platform

Theory

In the digital era, interactive features are essential for providing a seamless and personalized user experience—especially for a Hotel Menu System Website. JavaScript plays a key role in enabling client-side functionalities like user registration, login, form validation, and managing a virtual cart, ensuring the platform is dynamic, responsive, and user-friendly.

1. User Registration and Login Forms

These pages form the foundation of user account creation and access for placing orders or saving preferences.

Registration Form

The registration form gathers user information such as name, email, and password. JavaScript enhances this process by:

- **Validating empty fields** to ensure no required input is missed
- **Using Regular Expressions (RegEx)** to confirm proper email format
- **Verifying password strength** (e.g., length, special characters)
- **Matching password and confirm password fields**
- **Providing real-time visual feedback** (e.g., green checkmarks or red warning text)

This results in a more intuitive and error-free registration process.

Login Form

JavaScript also makes the login form secure and interactive by:

- Checking if **input fields are left blank**
- Verifying entered credentials against saved data (locally or via backend)
- Showing user-friendly **error messages** for failed attempts
- **Redirecting users** to the menu or dashboard page after successful login

2. JavaScript Form Validations

Form validation ensures only correct and complete data is processed. This improves the accuracy of user inputs and reduces unnecessary server requests.

Typical Validations Include:

- Required field detection
- Proper **email syntax** using RegEx
- Password rules: **minimum characters**, use of **numbers/symbols**, etc.
- **Confirm password check**
- Displaying inline messages like "Password too short" or "Email invalid"

Client-side validation via JavaScript ensures faster feedback for users, a smoother experience, and protection against invalid data submissions.

3. Add to Cart (Virtual Cart for Menu Items)

Just like e-commerce platforms, a Hotel Menu System can implement a virtual **cart** to let users collect food items before placing an order.

JavaScript Enhances Cart Features by:

- **Capturing user clicks** on "Add to Cart" buttons
- **Storing dish details (name, price, quantity)** in an array or local storage
- **Dynamically updating cart totals and quantities** as the user interacts
- **Allowing real-time item removal** from the cart without page reload
- **Redirecting to a final checkout or confirmation page**

These features mimic real restaurant ordering systems, creating a seamless digital dining experience.

4. Page Interactivity and Dynamic Updates

Beyond forms and cart logic, JavaScript can further enhance interactivity across all key pages:

- **About Page**: Toggle team member bios or display hotel story segments interactively.
- **Contact Page**: Enable form submissions with real-time validation, dynamic message confirmation, or map interactions.
- **Menu Page**: Filter dishes by category (e.g., Starters, Main Course, Desserts) without page reload.
- **Cart Page**: Dynamically render selected items and allow quantity updates or removals on the fly.

Code:

Registration page

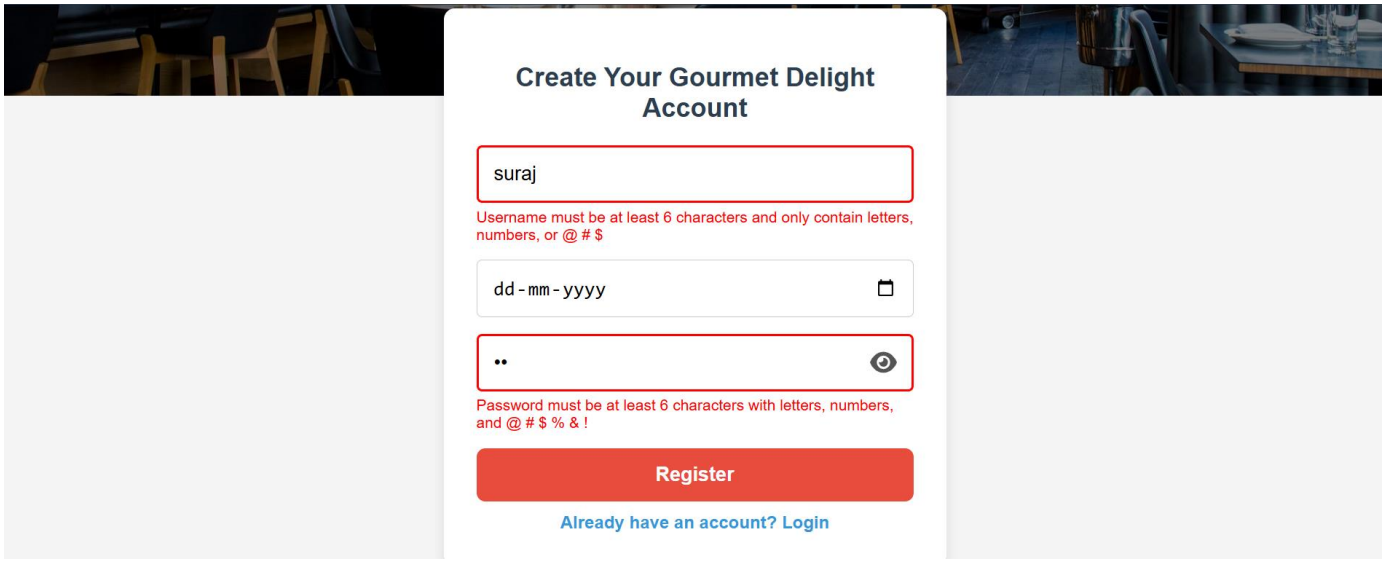
code:

```
function togglePassword() {
  const passwordField = document.getElementById('password');
  const toggleButton = document.querySelector('.toggle-password');

  if (passwordField.type === 'password') {
    passwordField.type = 'text';
    toggleButton.classList.remove('fa-eye');
    toggleButton.classList.add('fa-eye-slash');
  } else {
    passwordField.type = 'password';
    toggleButton.classList.remove('fa-eye-slash');
    toggleButton.classList.add('fa-eye');
  }
}

function validateUsername() {
  const username = document.getElementById('username');
  const usernameError = document.getElementById('username-error');
  const isValid = /^[a-zA-Z0-9@#$]{6,}$/.test(username.value.trim());

  if (isValid) {
    username.classList.remove("incorrect");
    username.classList.add("correct");
    usernameError.style.display = "none";
  } else {
    username.classList.remove("correct");
    username.classList.add("incorrect");
    usernameError.style.display = "block";
```

```
  }
}

function validatePassword() {
  const password = document.getElementById('password');
  const passwordError = document.getElementById('password-error');
  const isValid = /^(?=.*[a-zA-Z])(?=.*\d)(?=.*[@#$%&!]).{6,}$/.test(password.value);

  if (isValid) {
    password.classList.remove("incorrect");
    password.classList.add("correct");
    passwordError.style.display = "none";
  } else {
    password.classList.remove("correct");
    password.classList.add("incorrect");
    passwordError.style.display = "block";
  }
}

function register(event) {
  event.preventDefault();

  const username = document.getElementById('username').value.trim();
  const dob = document.getElementById('dob').value;
  const password = document.getElementById('password').value;

  const usernameField = document.getElementById('username');
  const passwordField = document.getElementById('password');
```

```
  const usernameValid = /^[a-zA-Z0-9@#$]{6,}$/.test(username);
  const passwordValid = /^(?=.*[a-zA-Z])(?=.*\d)(?=.*[@#$%&!]).{6,}$/.test(password);

  if (!usernameValid) {
    usernameField.classList.add("incorrect");
    return;
  }

  if (!passwordValid) {
    passwordField.classList.add("incorrect");
    return;
  }

  const users = JSON.parse(localStorage.getItem('users')) || [];

  const userExists = users.find(user => user.username === username);

  if (userExists) {
    alert('Username already exists. Please choose another.');
    return;
  }

  users.push({ username, dob, password });
  localStorage.setItem('users', JSON.stringify(users));

  alert('Registration successful! Redirecting to login page.');
  window.location.href = 'login.html';
}
```

Output:



Login Page

code:

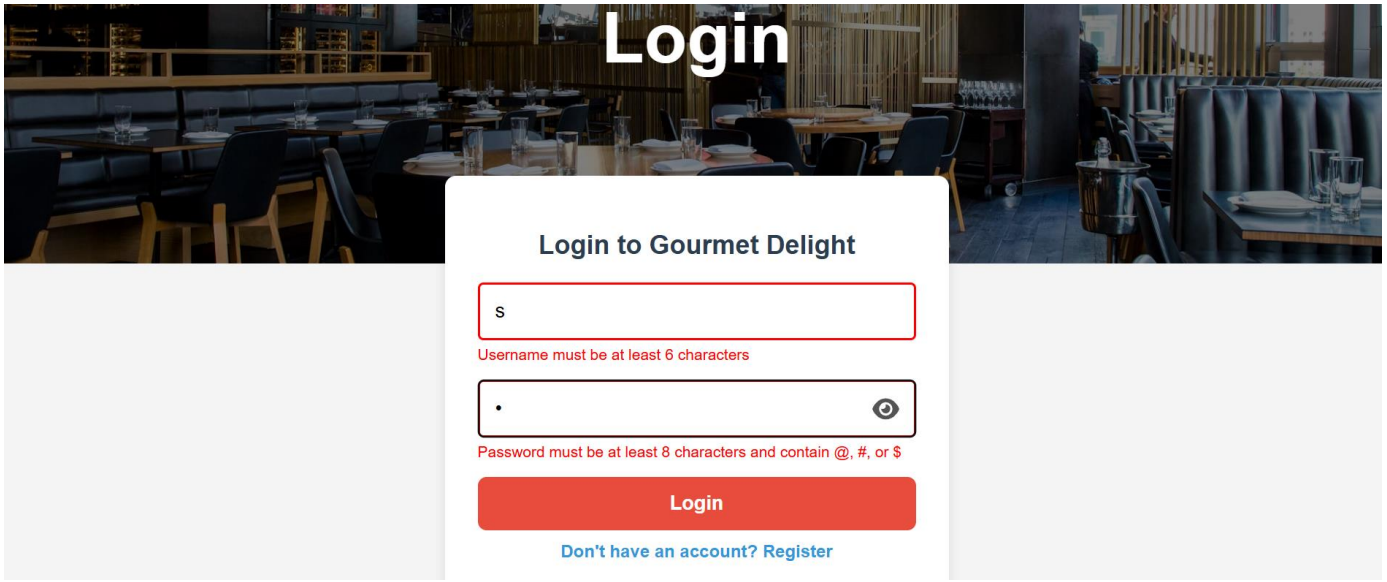
```
function togglePassword() {
  const passwordField = document.getElementById('password');
  const toggleButton = document.querySelector('.toggle-password');

  if (passwordField.type === 'password') {
    passwordField.type = 'text';
    toggleButton.classList.remove('fa-eye');
    toggleButton.classList.add('fa-eye-slash');
  } else {
    passwordField.type = 'password';
    toggleButton.classList.remove('fa-eye-slash');
    toggleButton.classList.add('fa-eye');
  }
}
function validateUsername() {
  const username = document.getElementById('username');
  const usernameError = document.getElementById('username-error');
```

```
  if (username.value.length >= 6) {
    username.classList.remove("incorrect");
    username.classList.add("correct");
    usernameError.style.display = "none";
  } else {
    username.classList.remove("correct");
    username.classList.add("incorrect");
    usernameError.style.display = "block";
  }
}
function validatePassword() {
  const password = document.getElementById('password');
  const passwordError = document.getElementById('password-error');
  const passwordRegex = /[@#$]/;

  if (password.value.length >= 8 && passwordRegex.test(password.value)) {
    password.classList.remove("incorrect");
    password.classList.add("correct");
    passwordError.style.display = "none";
  } else {
    password.classList.remove("correct");
    password.classList.add("incorrect");
    passwordError.style.display = "block";
  }
}
function login(event) {
  event.preventDefault();
```

```
const username = document.getElementById('username').value;
const password = document.getElementById('password').value;
const usernameInput = document.getElementById('username');
const passwordInput = document.getElementById('password');

if (username.length < 6) {
  usernameInput.classList.add("incorrect");
  return;
}

const passwordRegex = /[@#$]/;
if (password.length < 8 || !passwordRegex.test(password)) {
  passwordInput.classList.add("incorrect");
  return;
}

const users = JSON.parse(localStorage.getItem('users')) || [];
const validUser = users.find(user => user.username === username && user.password === password);

if (validUser) {
  usernameInput.classList.add("correct");
  passwordInput.classList.add("correct");
  alert('Login successful! Redirecting to Home page.');

  // Store user details for welcome message
  localStorage.setItem('userDetails', JSON.stringify({ username }));

  window.location.href = 'index.html';
} else {
```

```
    usernameInput.classList.add("incorrect");
    passwordInput.classList.add("incorrect");
    alert('Invalid credentials. Please try again.');
  }
}
function init() {
  const userDetails = JSON.parse(localStorage.getItem('userDetails') || 'null');
  if (userDetails) {
    document.getElementById("login").style.display = 'none';
    document.getElementById("register").style.display = 'none';
    document.getElementById("welcome").textContent = `Welcome ${userDetails.username}`;
  }
}
window.onload = init;
```

Output:



Conclusion

JavaScript is a vital part of the Hotel Menu System's interactivity and functionality. From verifying form inputs and managing user sessions to dynamically handling cart items, it ensures the site operates efficiently and responsively. By leveraging JavaScript properly, the platform becomes more engaging, reduces user frustration, and delivers a restaurant-like digital experience on both desktop and mobile devices.

Experiment No.6

Problem statement

JavaScript Theory: Persistent Login and Cart Functionality using Web Storage API

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

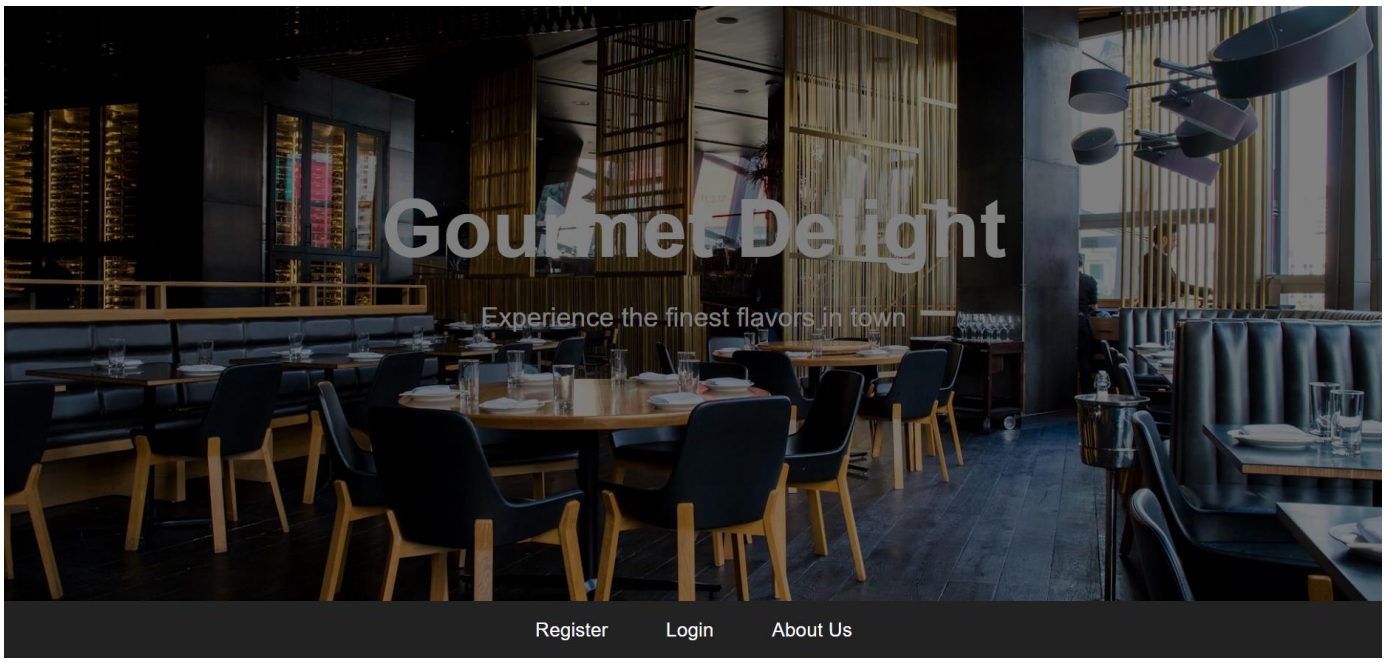
Code:

A. Home page:

```
function init() {
  const userDetails = JSON.parse(localStorage.getItem('userDetails') || null);

  if (userDetails) {
    // Hide guest links
    document.getElementById("login").style.display = 'none';
    document.getElementById("register").style.display = 'none';
    document.getElementById("about").style.display = 'none';

    // Show user links
    document.getElementById("menu").style.display = 'inline';
    document.getElementById("welcome").style.display = 'inline';
```

```
    document.getElementById("logout").style.display = 'inline';
    document.getElementById("welcome").textContent = `Welcome, ${userDetails.username}`;
  }
  document.getElementById("logout").addEventListener("click", function (e) {
    e.preventDefault();
    localStorage.removeItem("userDetails");
    location.reload();
  });
}
window.onload = init;
```

Output:



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

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();
}
?>
```

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

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();
?>
```

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 No.9

Problem Statement:

A. Develop a PHP script that allows users to manage their shopping cart for an Hotel menu system 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 Hotel Menu system 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
);
```

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 No.10

Problem Statement

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

Theory: PHP Checkout Process

The **checkout process** is the final and most crucial step in any 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.";
}
?>
```

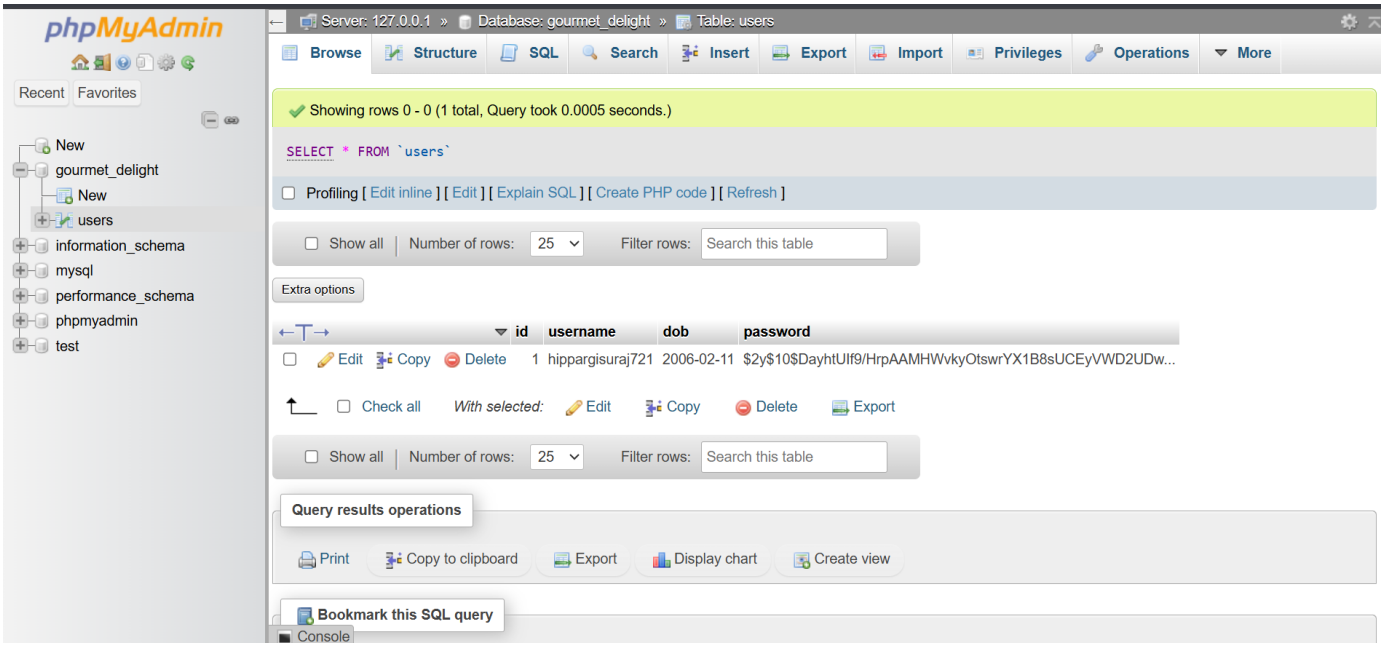
MySQL-Based PHP Checkout Script:-

```
<?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:



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.