

**MIT Art, Design and Technology University**

**MIT School of Computing, Pune**

**Department of Information Technology**

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

**Practical - Web Programming**

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

**Batch - DA-I**

**Prince Bankar**

**Mr./Ms.**

**A.Y. 2024 – 2025 (SEM-I**

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

**List of Experiments:**

In this series of assignments, you will create a Sneaker sphere website step by step. Each assignment will focus on a different aspect of the website, covering various HTML elements, CSS, JavaScript, PHP and MySQL concepts.

**Laboratory/Project Assignment Guidelines:**

1. Project Selection:
   * Each student must select a unique project topic for their laboratory assignments.
   * The chosen project topic should align with the concepts covered in the course syllabus.
   * The chosen project topic should be approved by the course coordinator/ subject teacher.
   * Students have the freedom to choose their project topics based on their interests and career aspirations.
   * Project topics may include but are not limited to:
     + E-commerce website
     + Blogging platform
     + Online booking system
     + Content management system (CMS)
     + Discussion forum
     + Social networking platform
     + Task management application
     + Portfolio website
2. Laboratory Assignments:
   * Throughout the course, students will complete laboratory assignments related to their chosen project topic.
3. Evaluation Criteria:
   * The laboratory assignments and the final project will be evaluated based on criteria such as Structure and Semantics, Content Organization, Forms and Inputs, Links and Navigation, Styling and Layout, Design Consistency, Functionality, Code Quality and adherence to project requirements.
   * Students are expected to demonstrate creativity, and a comprehensive understanding of web development principles in their projects.
   * The laboratory assignments based on chosen project topics will be assessed based on several key criteria that reflect both technical proficiency and creative application in web development. These include:

* Structure & Semantics: Proper use of HTML to create a logical, accessible structure with meaningful and semantically correct elements.
* Content Organization: Clear and intuitive organization of content, ensuring ease of navigation and logical flow throughout the site.
* Forms & User Input: Effective implementation of forms and user input elements that are functional, validated, and accessible.
* Links & Navigation: Well-structured navigation and functional links that provide a seamless user experience.
* Styling & Layout: Visually appealing and responsive design, with a well-executed layout that adapts to various screen sizes.
* Design Consistency: Uniformity in design elements, including colors, typography, and spacing, to maintain a cohesive look and feel across the site.
* Functionality: Full functionality of all interactive elements, ensuring a bug-free, smooth experience for users.
* Code Quality & Best Practices: Clean, well-organized, and efficient code that adheres to modern web development best practices and is easy to maintain.

1. Submission and Presentation:
   * The project and project report/journal must be submitted within the specified deadline and should meet the specified requirements outlined by the course coordinator/ subject teacher.

Experiment No.1

Problem Statement:

1. Create the basic structure of the community health center patient portal, including the home page layout with a header, main content area, and footer.

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

1. Brief information about the project.

Set the goals & deliverables.

1. Finalize the modules of the project.
2. Define the audience.
3. Describe pain points & the ideal experience (On the basis of existing systems)
4. Set the visual direction
5. Map out the Project structure.
6. Plan the content for each page.
7. Add ideas for content, images & layout.
8. Determine your site structure or Create content for your core patient portal pages:
9. Home page
10. About page
11. Medical Services page
12. Testimonial/review page
13. Support page
14. Starter blog posts
15. Create and collect design elements
16. These design elements define your brand personality and help patients feel what your brand represents through the use of:
17. Colors
18. Fonts and typography
19. Logo
20. Images and photos

Objective:

To design the basic structure of a community health center patient portal by planning its layout, content, and visual elements, ensuring it meets user needs and effectively represents the brand.

Theory:

Project Design and Plan Document for Clinic Store Website

1. Brief Information about the Project

The project is to create a user-friendly and visually appealing patient portal for a community health center. It aims to attract clinic enthusiasts, highlight services, and offer features such as customer testimonials and a contact platform. Additionally, the patient portal will support login and registration to personalize user experiences and allow secure access to exclusive features.

2. Goals and Deliverables

Goals

* Develop an engaging and functional patient portal for a community health center.
* Showcase the center's story, services, customer reviews, and contact details.
* Enable users to register, log in, and personalize their experience.
* Create a responsive patient portal that works across all devices.

Deliverables

* Website Pages:
  + Home Page
  + About Page
  + Services/Services Page
  + Patient Feedback Page
  + Contact Page
  + Patient Login Page
  + New Patient 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 community health center patient portal 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 patient portal welcomes users and highlights essential features. It sets the tone for the user experience.
* Features:
  + Hero section with the tagline and call-to-action buttons (e.g., "Book Now" or "Explore Menu").
  + Overview of featured services or promotions.
  + Navigation catalog linking to all patient portal sections (e.g., About, Services, Patient Feedback, Contact, Patient Login).
  + Footer with contact details, social links, and other information.

2. About Page Module

* Description:  
  Offers visitors a glimpse of the community health center's story, mission, and values.
* Features:
  + Introduction to the community health center's history and portable clinicsm.
  + Showcase the brand's principles like quality, sustainability, and customer service.
  + Engaging visuals to reflect the community health center's vibe.

3. Services/Services Page Module

* Description:  
  Displays the center's service offerings in a user-friendly way.
* Features:
  + Categorized catalog (e.g., Clinic, Snacks, Handhelds).
  + Images and details for each item, including price and description.
  + Option for filtering or searching services (future enhancement).

4. Patient Feedback Page Module

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

1. Contact Page Module

* Description:  
  Enables visitors to get in touch with the community health center.
* Features:
  + A form for user inquiries (fields: Name, Email, Subject, Message).
  + Embedded map for the physical center location.
  + Display contact details like phone number and working hours.

6. Patient Login Page Module

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

7. New Patient Registration Page Module

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

8. Footer Module

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

4. Define the audience

Target Audience

The patient portal for the community health center is designed to cater to a broad spectrum of visitors, each with specific needs and expectations. Understanding the audience ensures the patient portal design, content, and features meet their requirements. Below is a breakdown of the target audience:

a. Clinic Enthusiasts

* Characteristics:
  + Regular consumers of clinic.
  + Interested in exploring various clinic flavors, gaming methods, and exclusive services.
* Needs:
  + Detailed service descriptions with images and pricing.
  + Access to exclusive offers, rewards, or subscriptions for frequent books.

b. Professionals and Remote Workers

* Characteristics:
  + Need a quiet, comfortable space to work or meet clients.
  + May book clinic and snacks online or reserve seating.
* Needs:
  + Support page with opening hours and location details.
  + Online booking system for in-center pickups or delivery.
  + in-center testing area information or co-working amenities.

c. Students

* Characteristics:
  + Younger audience looking for affordable and quick snacks, beverages, and a social hangout space.
* Needs:
  + Discounts for students or groups.
  + Easy navigation to view catalogs and promotional deals.

d. Health-Conscious Customers

* Characteristics:
  + Focused on dietary requirements, organic options, or alternative milk and low-calorie beverages.
* Needs:
  + Highlight nutritional information on the catalog.
  + Clear categorization of vegan or healthy options.

e. Tourists and Travelers

* Characteristics:
  + Look for local centers offering an authentic experience.
  + Interested in souvenirs, gift cards, and specialty items.
* Needs:
  + Integration with map navigation to locate the center.
  + Details about specialty clinics or regional favorites.

f. New Users (Unfamiliar Customers)

* Characteristics:
  + New visitors who haven’t interacted with the brand before.
* Needs:
  + A professional and appealing "About Our Healthcare Center" page to share the center’s values and story.
  + A clean, user-friendly interface to build trust.

g. Online Storepers

* Characteristics:
  + Prefer the convenience of booking clinic services online, including healthcare services and gaming equipment.
* Needs:
  + Service page with a seamless healthcare booking experience.
  + Secure login and registration system for repeat centerping.

Website Features Mapped to Audience Needs:

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

Why Understanding the Audience is Important

* Helps in creating engaging and relevant content tailored to users’ preferences.
* Enhances the user experience (UX) by addressing specific pain points and ensuring seamless navigation.
* Builds brand trust and attracts loyal patients who resonate with the community health center’s story and mission.
* Leads to targeted marketing campaigns, such as student promotions, subscription offers for enthusiasts, or health-focused messaging.

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

1. Identifying Pain Points of Existing Systems

a. Pain Point: Poor Navigation and Cluttered Interface

* Issue: Many community health center patient portals have complicated or cluttered designs that make it hard for users to find what they are looking for.
* Impact: Users often leave the site due to frustration or lack of usability.

b. Pain Point: Limited Online Booking Functionality

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

c. Pain Point: Lack of Mobile Optimization

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

d. Pain Point: Insufficient Service Information

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

e. Pain Point: Weak Engagement Strategies

* Issue: Existing patient portals lack features like loyalty programs, student discounts, or engaging content like blogs.
* Impact: Missed opportunities for creating brand loyalty and retaining patients.

f. Pain Point: Inefficient Contact and Location Details

* Issue: Many patient portals fail to prominently display contact and location information, making it difficult for patients to find or connect with the community health center.
* Impact: Customers waste time searching and may opt for competitors insportable clinicsd.

g. Pain Point: No Personalization Options

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

2. Crafting the Ideal Experience

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

a. Intuitive Navigation and Clean Design

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

b. Seamless Online Booking

* Implement a robust healthcare booking system allowing patients to browse services, add items to a appointment summary, and complete books effortlessly.
* Provide features like "Book Now" buttons on the homepage and catalog pages.

c. Mobile-Responsive Design

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

d. Comprehensive Service Information

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

e. Customer Engagement Features

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

f. Easy Access to Contact and Location

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

g. Personalization

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

3. The Ideal User Journey

Step 1: Visiting the Website

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

Step 2: Browsing the Menu

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

Step 3: Placing an Book

* Users can seamlessly add items to their appointment summary and complete a book with minimal clicks.

Step 4: Finding Location or Contacting Support

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

Step 5: Engaging with Content

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

Step 6: Creating Loyalty

* Registered users receive personalized promotions or gain points through books.

6. Set the visual direction

1. Visual Design Goals

The visual design of the community health center patient portal should reflect its personality, build trust, and create an inviting experience for patients. It should align with the following principles:

* Welcoming and Comfortable: The patient portal should feel cozy and approachable, much like the service presentation of the community health center itself.
* Modern and Minimalistic: Clean layouts and modern design elements create a professional and user-friendly aesthetic.
* Brand Representation: The visual elements, including colors, typography, and images, should communicate the community health center's values and target audience.

2. Defining the Core Visual Elements

a. Color Palette

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

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

b. Typography

Fonts should be easy to read while reflecting the warm and inviting atmosphere of the community health center.

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

c. Logos and Branding

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

* Use a stylized clinic device, bean, or sportable clinicsm motif in the logo design.
* The logo should include the community health center's name in the selected typography.
* A monochrome version of the logo can be created for simplicity in headers or footers.

d. Imagery and Icons

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

* Photography:
  + Pictures of freshly useed clinic, healthcare services, cozy seating spaces, and happy patients.
  + Showcase specialty clinic clinics, desserts, and in-center service presentation.
* Icons:
  + Minimalistic icons for categories like catalog, location, testimonials, and contact.
* Hero Images:
  + Use a carousel or static hero banner on the homepage featuring key services or seasonal promotions.

3. Applying Visual Design to Pages

a. Home Page

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

b. About Page

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

c. Medical Services Page

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

d. Testimonial Page

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

e. Contact Page

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

f. Patient Login and New Patient Registration Pages

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

4. Layout and Design Hierarchy

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

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

5. Expected Impact of Visual Direction

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

7. Map out the Project structure

clinic\_center\_patient portal/

│

├── index.html             # Home page

├── about.html             # About page

├── services.html          # Services/Services page

├── testimonials.html      # Patient Feedback page

├── contact.html           # Support page

├── login.html             # Patient Login page

├── register.html          # New Patient Registration page

├── blog.html              # Optional blog page

│

├── assets/

│   ├── css/

│   │   ├── style.css        # Global CSS

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

│   │

│   ├── js/

│   │   ├── main.js          # Interactive scripts

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

│   │

│   ├── images/

│       ├── logo.png         # Website logo

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

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

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

│       ├── icons/           # Icons for UI elements

│

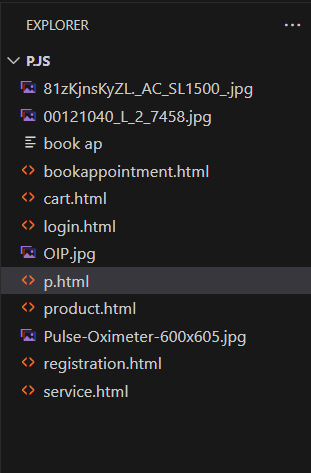
├── fonts/

│   ├── Poppins/            # Primary font

│   ├── OpenSans/           # Secondary font

│

└── README.md               # Project documentation file



8. Plan the content for each page

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

1. Home Page

Purpose:

* Welcome visitors.
* Highlight the community health center's offerings and features.

Content Plan:

* Header:
  + Logo on the left.
  + Navigation catalog: Home, About, Services, Patient Feedback, Contact.
  + Patient Login/Sign-Up button on the top right.
* Hero Section:
  + A high-quality banner image (community health center service presentation, a featured clinic).
  + A tagline like: “Brewed to Perfection!”.
  + CTA button: “Explore Our Menu”.
* Introduction Section:
  + Brief about the center (one or two sentences).
  + CTA: “Learn More About Our Healthcare Center” linking to the About page.
* Special Offer/Highlight Section:
  + Carousel or grid of featured clinics or offers.
  + Text: “Try Our New Seasonal Flavors!”.
* Footer:
  + Quick links, social media links, contact information.

2. About Page

Purpose:

* Share the story, vision, and people behind the community health center.

Content Plan:

* Header: (same as home page).
* About Our Healthcare Center Section:
  + A short introduction to the community health center’s history (e.g., when and why it was founded).
  + Emphasis on values such as sustainability or quality.
* Meet the Team Section:
  + Photos and short bios of the founders or portable clinicsm members.
* Special Features Section:
  + “Why Choose Us?”
  + Highlight USP (organic healthcare services, specialty uses, etc.).
* Footer: (same as home page).

3. Services Page

Purpose:

* Showcase the catalog and services offered.

Content Plan:

* Header: (same as home page).
* Menu Sections:
  + Category: e.g., Clinic, Tea, Snacks, Cables.
  + Service tiles: Images, names, short descriptions, prices.
* Highlight Section:
  + “Top Picks” or “Customer Favorites.”
* CTA Section:
  + Button: “Book Now” linking to Patient Login or New Patient Registration page.
* Footer: (same as home page).

4. Patient Feedback Page

Purpose:

* Build trust by showcasing feedback from happy patients.

Content Plan:

* Header: (same as home page).
* Customer Feedback Section:
  + Quotes or testimonials from existing patients.
  + Option to display Google or Yelp reviews.
  + Use star ratings for visual appeal.
* Submit a Testimonial:
  + Simple form to allow visitors to submit reviews.
* Footer: (same as home page).

5. Contact Page

Purpose:

* Allow patients to reach out easily for inquiries, reservations, or feedback.

Content Plan:

* Header: (same as home page).
* Contact Form:
  + Name, Email, Subject, Message.
  + Submit button with form validation.
* Location Section:
  + Embedded Google Map showing the community health center's location.
* Operating Hours Section:
  + Business hours listed clearly.
* Footer: (same as home page).

6. Patient Login Page

Purpose:

* Enable existing users to log in to their accounts.

Content Plan:

* Form:
  + Email and Password fields.
  + Submit button.
* Forgot Password Link:
  + Redirects to password recovery.
* CTA:
  + Link to the New Patient Registration page: “Don’t have an account? Sign Up Now!”.

7. New Patient Registration Page

Purpose:

* Allow new users to register for an account.

Content Plan:

* Form Fields:
  + Full Name.
  + Email.
  + Password and Confirm Password.
* Form Validation:
  + Password requirements.
* Submit Button:
  + Validates data and submits.
* Footer: (same as home page).

9. Add ideas for content, images & layout

Below are detailed ideas for crafting the content, selecting images, and structuring the layout for a community health center patient portal with at least five pages and login/registration functionality.

1. Home Page

Layout Ideas:

* Header Section:
  + Fixed navigation bar.
  + Logo on the left, catalog items in the center, and a Patient Login/Register button on the right.
* Hero Section:
  + Full-width background image of the community health center or clinic art.
  + Overlay text with a tagline (e.g., “Brewed with Love!”).
  + A button like “Explore Menu” or “Visit Us.”
* Featured Section:
  + Use cards or tiles to display 3-4 featured items (e.g., “Latte of the Day” or seasonal beverages).
  + Include enticing images and brief descriptions.

Content Ideas:

* Short, catchy welcome message.
* Announcements for specials or promotions, like “Happy Hour: 3 PM to 5 PM!”.
* Direct link to the catalog page.

Image Ideas:

* A banner of freshly useed clinic.
* Close-up shots of clinic healthcare services, PlayStation art, and cozy seating areas.

2. About Page

Layout Ideas:

* Story Section:
  + A timeline or column layout telling the story of the community health center.
  + Use dividers or icons to separate milestones.
* Team Section:
  + Grid layout featuring portable clinicsm member images and short bios.
* Feature Section:
  + Icons and text showcasing special features (e.g., Organic healthcare services, Sustainable practices, etc.).

Content Ideas:

* Mission statement, history, and values.
* Insights about sourcing and partnerships.
* Emphasis on community connection.

Image Ideas:

* Images of founders, the clinic-making process, and snapshots of community events.
* Collage of the center’s interior.

3. Services/Services Page

Layout Ideas:

* Categories Section:
  + Split the page into categories like Clinic, Accessories, Teas, and Special Offers.
  + Each category features service images, names, descriptions, and prices.
* Highlight Section:
  + Slider showing customer favorites or best sellers.
* CTA Section:
  + Book Now or Learn More buttons under each item.

Content Ideas:

* Detailed descriptions of items, focusing on unique flavors or preparation techniques.
* Highlight seasonal specialties (e.g., “Pumpkin Spice Latte”).

Image Ideas:

* High-resolution images of clinic devices, pastries, portable clinics varieties.
* Display trays, artisan-style packaging.

4. Patient Feedback/Reviews Page

Layout Ideas:

* Feedback Section:
  + Use a carousel to rotate customer reviews.
  + Display text alongside star ratings.
* CTA Section:
  + A “Submit Your Testimonial” form on the bottom.

Content Ideas:

* Customer feedback emphasizing service presentation, taste, and service.
* Pull quotes from external reviews (Google, Yelp, etc.).

Image Ideas:

* Happy patients enjoying clinic.
* Stylized images of written reviews.

5. Contact Page

Layout Ideas:

* Form Section:
  + A simple contact form (fields for name, email, subject, message).
* Map Section:
  + Embed Google Maps for easy navigation.
* Info Section:
  + Use icons to display the address, phone number, and operating hours.

Content Ideas:

* A friendly “We’d love to hear from you!” message.
* FAQs for common inquiries.

Image Ideas:

* Second-hand gaming clinic centerfront.
* Icons or illustrations for contact methods.

6. Patient Login Page

Layout Ideas:

* Minimalist form interface.
* A side panel or banner featuring the center logo or relevant imagery.

Content Ideas:

* Fields for email and password.
* A friendly reminder like: “New here? Sign up now!”

Image Ideas:

* Illustration of a clinic device or healthcare services.
* Background image with a transparent overlay for text fields.

7. New Patient Registration Page

Layout Ideas:

* Clean, centered form fields.
* Password strength indicator.

Content Ideas:

* A tagline: “Join us and enjoy exclusive rewards!”.
* Checkbox for opting into newsletters or special offers.

Image Ideas:

* A welcoming banner image.
* Icons for form fields (e.g., lock for password).

8. Visual Design Ideas for All Pages:

Colors:

* Earth tones: brown, cream, and green.
* Highlight color: gold or maroon for buttons and accents.

Fonts & Typography:

* Use a warm, inviting font for headers (e.g., Playfair Display).
* Clean sans-serif font for body text (e.g., Lato or Roboto).

Logos:

* Stylized clinic device with sportable clinicsm.

Images:

* Realistic, high-quality visuals with natural lighting.
* Use

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

1. Home page
2. About page
3. Medical Services page
4. Testimonial/review page
5. Support page
6. Starter blog posts

1. Home Page

The home page serves as the introduction to the community health center and provides navigation to the key sections of the patient portal.

Here's a revised version of the patient portal outline, reflecting the focus on a community health center:

1. Home Page

Header:

* Logo: A simple, bold representation of the community health center (e.g., a stylized healthcare service or controllers).
* Navigation Links: Menu, About, Services, Patient Feedback, Contact.
* Call-to-Action Button: "Buy Now" or "Join Us" (links to the book or registration page).

Hero Section:

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

About Section (Teaser):

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

Service Highlights Section:

* Featured Services: Grid showcasing 3-4 key services like "Clinic of the Week," "Featured Accessories," etc.
* Images and short descriptions with an option to learn more or make a book.

Social Proof Section (Testimonial Teaser):

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

Footer:

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

2. About Page

Introduction:

* Overview of the community health center’s story, mission, and values.
* Brief history of the business: "Founded in 2025 with a passion for gaming and sustainability..."

Meet the Team:

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

Our Promise:

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

Location Section:

* List of center locations with Google Maps integration.
* Hours of operation.

3. Medical Services Page

Clinic Categories:

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

Popular Items & Limited-Time Specials:

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

Book Online:

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

Footer (same as Home Page):

* Quick links, social media icons, center locations.

4. Patient Feedback/Review Page

Customer Reviews:

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

Featured Reviews:

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

Reviewing Process Section:

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

5. Contact Page

Contact Form:

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

Social Media & Address Section:

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

Interactive Map:

* Google Maps integration to guide patients to the center location.

Support Information:

* Contact details for customer support and FAQs.

6. Starter Blog Posts

Blog Categories:

* Clinic Knowledge: Articles like “How to Choose the Right Second-Hand Clinic” or “What to Look for in Refurbished Gaming Clinics.”
* Behind the Scenes: Features on clinic refurbishing processes, employee spotlights, or clinic culture.
* Sustainability Efforts: Articles on how the center contributes to sustainability by promoting second-hand gaming.
* Community Engagement: Stories about the center’s involvement in gaming events or charity support.

7. Patient Login & New Patient Registration Pages

Patient Login Page:

* Username/email and password fields.
* “Forgot password?” link.

New Patient Registration Page:

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

8. Overall Website Structure Map

* Home Page: Introductory page with links to featured services, testimonials, and social media.
* About Page: Overview of the center, team, and values.
* Medical Services Page: Showcase services with book options.
* Patient Feedback/Review Page: User feedback and submission form.
* Contact Page: Contact form and location details.
* Blog Section: Articles on gaming, sustainability, and community.
* Patient Login/New Patient Registration Page: For user accounts and updates.

9. Design Elements

Colors:

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

Fonts and Typography:

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

Logo:

* A simple logo with a visual element related to healthcare services (e.g., controllers, gaming icons).

Imagery and Photos:

* High-quality images of healthcare services, service close-ups, and the center ambiance.
* Lifestyle shots showing patients enjoying games in the center.

Interactive Elements:

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

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

2. Fonts and Typography

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

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

Impact: The mix of serif and sans-serif fonts maintains a balance between tradition and modernity, perfect for a community health center with a warm, upscale yet modern experience.

3. Logo

Your logo represents the visual identity of your community health center and sets the tone for your brand’s story.

* Logo Design: The logo should be simple but memorable, combining visual elements that represent clinic. Consider using stylized clinic devices, clinic healthcare services, or sportable clinicsm swirls. These visuals should clearly associate the logo with the essence of the center.
* Color Palette for the Logo: Use the primary colors like clinic brown and cappuccino beige, along with a touch of espresso black for contrast. If your community health center values organic ingredients, incorporating a bit of green could reinforce the sustainability aspect.

Logo Usage:

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

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

4. Imagery and Photos

Imagery on the community health center patient portal has the ability to build a stronger emotional connection by showcasing the clinic experience.

* Service Photography: High-quality images of clinic clinics, pastries, and desserts should dominate the site. Think close-ups of frothy cappuccinos, healthcare services being ground, sportable clinicsming devices of clinic, or beautiful PlayStation art.
  + For the catalog page, show clean, professional shots of the services with descriptions.
  + For the about page, images of the interior of your community health center, people enjoying their clinic, or portable clinicsm photos add authenticity and a sense of community.
* Ambiance Photography: Show the cozy community health center setting with soft lighting, wooden tables, and greenery. These images should showcase the atmosphere visitors will experience in person. Consider using candid photos of patients enjoying clinic together or a technician preparing a clinic.
* Lifestyle Photography: In addition to service-specific images, showing a lifestyle—people studying, working, or socializing in your center can be powerful. This reinforces the idea that your center is a place to gather, relax, and socialize.

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

5. Interactive Elements and Buttons

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

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

Conclusion:

The Clinic Store Website project serves as a comprehensive exercise in understanding and applying web development fundamentals. Through careful planning and implementation of design elements, content organization, and user experience considerations, the assignment ensures a functional, visually appealing, and user-friendly platform.

By creating essential pages like the Home, About, Medical Services, Testimonial, Contact, and Patient Login/New Patient Registration pages, the patient portal provides patients with intuitive navigation and engaging content. The visual direction, including the use of harmonious colors, typography, logos, and images, helps reinforce the brand's identity and appeal to the target audience.

This assignment highlights the importance of a structured approach to web design—starting with setting goals and mapping out the project structure, followed by planning and designing the content. By addressing audience needs and incorporating user-centric solutions, the project enhances the online presence of a community health center while delivering a seamless digital experience for patients.

`Experiment No.2

Problem Statement:

* Create a detailed home page for healthcare patient portal.
* Create a detailed service list/service page for the healthcare patient portal, listing all available items categorized appropriately.
* Create a appointment summary page that allows patients to review and manage the items they wish to book before proceeding to checkout.
* Create an about us page that provides detailed information about the healthcare history, mission, and team.
* Create a contact page that allows patients to easily get in touch with the healthcare through a form.
* Design and implement admin/user registration form for the healthcare patient portal.
* Design and implement admin/user login form for the healthcare patient portal.

Objective:

To create a Second-Hand Gaming Clinics webpage using HTML.

Introduction

In today's digital economy, healthcare booking platforms are essential for buying and selling services efficiently. This project focuses on creating a responsive and functional patient portal for a community health center. The platform caters to gamers looking for affordable alternatives to brand-new devices, promoting sustainability and cost-effectiveness.

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

1. Home Page

The home page serves as the landing page and provides a snapshot of the center’s offerings. It typically includes:

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

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

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

2. Service/Menu Page

This page is crucial as it displays the entire service inventory. Items are grouped into categories such as:

* Gaming Clinics (e.g., Blood Pressure Monitor, Stethoscope, Pulse Oximeter)
* Accessories (controllers, headsets, cables)
* Bundles or Combo Offers

Features include:

* Service image
* Title and specifications
* Condition (e.g., Good, Excellent, Refurbished)
* Price
* Add to Cart button

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

UX Consideration:  
Service filters (by brand, condition, or price range) improve usability and conversion rates.

🛒 3. Cart Page

The appointment summary system is a core part of the healthcare booking flow. It displays:

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

Real-world relevance:  
Gives users control over their books and supports decision-making before payment.

Optional enhancements:

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

4. About Our Healthcare Center Page

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

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

Purpose:  
Builds trust and authenticity with potential buyers, especially in a niche like second-hand electronics where quality assurance is crucial.

5. Contact Page

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

* Name
* Email
* Subject
* Message

Additional elements:

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

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

6. User/Admin New Patient Registration Form

This page allows new users and admins to create an account. It collects:

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

Functionality:

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

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

7. User/Admin Patient Login Form

This form validates users or admins against centerd credentials and redirects them to their respective dashboards.

Fields:

* Username/email
* Password
* Remember me checkbox
* Forgot password link

Security Considerations:

* Basic input validation
* In serviceion: hashing passwords, rate limiting, two-factor authentication

Differentiated Access:

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

Technological Stack Overview (Future Enhancement)

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

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

Sustainability Impact

The center promotes eco-conscious consumerism by extending the life cycle of electronics. It reduces electronic waste and supports circular economy practices by:

* Reselling quality-checked devices
* Offering affordable gaming experiences
* Educating users on reusability

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>Healthcare Info</title>

  <style>

    \* {

      margin: 0;

      padding: 0;

      box-sizing: border-box;

    }

    body {

      font-family: Arial, sans-serif;

      background-color: #f0f0f0;

      color: #333;

    }

    .navbar {

      background-color: #0077b6;

      display: flex;

      justify-content: space-between;

      align-items: center;

      padding: 10px 20px;

    }

    .navbar .logo {

      color: #fff;

      font-size: 24px;

      font-weight: bold;

    }

    .navbar ul {

      list-style: none;

      display: flex;

      gap: 20px;

    }

    .navbar ul li a {

      color: #fff;

      text-decoration: none;

      font-size: 16px;

      transition: color 0.3s;

    }

    .navbar ul li a:hover {

      color: #ff6f61;

      text-decoration: underline;

    }

    .navbar .auth-btn {

      padding: 6px 12px;

      border-radius: 4px;

      text-decoration: none;

      font-size: 16px;

    }

    .navbar .auth-btn {

      background-color: transparent;

      color: #fff;

    }

    header {

      background-color: #2e86de;

      color: white;

      padding: 40px 20px;

      text-align: center;

      background-image: url('header-image.jpg');

      background-size: cover;

      background-position: center;

      height: 300px;

      display: flex;

      flex-direction: column;

      justify-content: center;

    }

    header h1 {

      font-size: 40px;

      font-weight: 700;

    }

    header p {

      font-size: 20px;

    }

    section {

      padding: 40px 20px;

      text-align: center;

    }

    .services {

      display: flex;

      gap: 20px;

      flex-wrap: wrap;

      justify-content: center;

    }

    .service-box {

      border: 1px solid #ddd;

      padding: 20px;

      width: 250px;

      border-radius: 12px;

      background-color: #fff;

      box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

      transition: transform 0.3s ease, box-shadow 0.3s ease;

    }

    .service-box:hover {

      transform: translateY(-10px);

      box-shadow: 0 6px 12px rgba(0, 0, 0, 0.2);

    }

    footer {

      background-color: #2e86de;

      color: white;

      text-align: center;

      padding: 10px;

    }

    .contact input,

    .contact textarea {

      width: 100%;

      padding: 10px;

      margin-top: 5px;

      margin-bottom: 15px;

      border-radius: 8px;

      border: 1px solid #ddd;

    }

    .contact button {

      background-color: #0077b6;

      color: white;

      border: none;

      padding: 10px 20px;

      cursor: pointer;

      border-radius: 8px;

      transition: background-color 0.3s ease;

    }

    .contact button:hover {

      background-color: #023e8a;

    }

    .form-section input,

    .form-section select {

      width: 100%;

      padding: 8px;

      margin-bottom: 10px;

      border-radius: 8px;

      border: 1px solid #ddd;

    }

    .form-section button {

      background-color: #0077b6;

      color: white;

      border: none;

      padding: 8px 16px;

      cursor: pointer;

    }

    @media (max-width: 768px) {

      .services {

        flex-direction: column;

        align-items: center;

      }

      .navbar ul {

        flex-direction: column;

      }

    }

    @keyframes fadeIn {

      to {

        opacity: 1;

      }

    }

    .service-box {

      opacity: 0;

      animation: fadeIn 0.5s forwards;

    }

  </style>

</head>

<body>

    <div class="navbar">

        <div class="logo">My HealthCare</div>

        <ul>

          <li><a href="#service">Service</a></li>

          <li><a href="bookappointment.html">Book Appointment</a></li>

          <li><a href="#tips">Health Tips</a></li>

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

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

          <li><a href="registration.html">Register</a></li>

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

          <li><a href="#contact">Contact</a></li>

          <li><a href="#service">Service</a></li>

        </ul>

      </div>

  <header>

    <h1>Welcome to My HealthCare</h1>

    <p>Your Health is Our Priority</p>

  </header>

  <section id="services">

    <h2>Our Services</h2>

    <div class="services">

      <div class="service-box">

        <h3>Doctor Consultation</h3>

        <p>Talk to our expert doctors anytime.</p>

      </div>

      <div class="service-box">

        <h3>Lab Tests</h3>

        <p>Affordable and accurate lab testing.</p>

      </div>

      <div class="service-box">

        <h3>Home Care</h3>

        <p>Nurses and support at your doorstep.</p>

      </div>

      <div class="service-box">

        <h3>Emergency Services</h3>

        <p>24/7 ambulance and trauma support.</p>

      </div>

      <div class="service-box">

        <h3>Mental Health Counseling</h3>

        <p>Talk to certified counselors confidentially.</p>

      </div>

      <div class="service-box">

        <h3>Vaccination & Immunization</h3>

        <p>COVID-19, flu, and other vaccinations available.</p>

      </div>

      <div class="service-box">

        <h3>Online Prescription</h3>

        <p>Get your prescriptions online without hassle.</p>

      </div>

    </div>

  </section>

  <section id="tips">

    <h2>Health Tips</h2>

    <ul>

      <li>Drink plenty of water daily.</li>

      <li>Exercise for at least 30 minutes a day.</li>

      <li>Eat more fruits and vegetables.</li>

      <li>Avoid junk food and sugary drinks.</li>

    </ul>

  </section>

  <section id="contact" class="py-20 px-4 bg-white text-center">

    <h3 class="text-3xl font-bold mb-6">Get In Touch</h3>

    <p class="mb-8 text-lg text-gray-600">We’re here to answer any questions or provide assistance. Feel free to reach out to us.</p>

    <form class="max-w-xl mx-auto space-y-4">

      <input type="text" placeholder="Your Name" class="w-full p-3 border rounded-lg" required />

      <input type="email" placeholder="Your Email" class="w-full p-3 border rounded-lg" required />

      <textarea placeholder="Your Message" class="w-full p-3 border rounded-lg" rows="4" required></textarea>

      <button type="submit" class="bg-blue-600 text-white px-6 py-3 rounded-lg hover:bg-blue-700">

        Send Message

      </button>

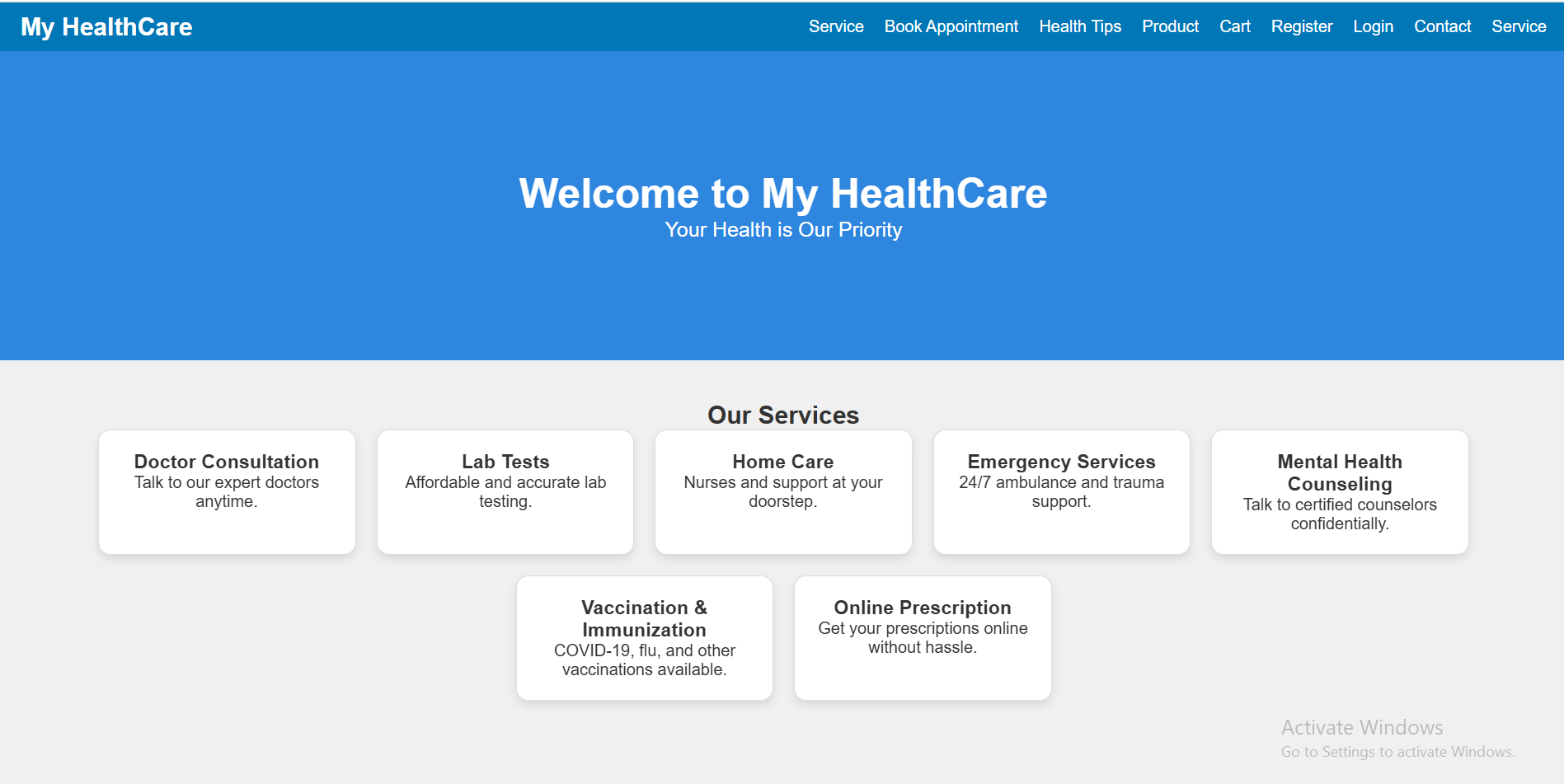
    </form>

  </section>

</body>

</html>

Output:



Code:

B. service page:

code:

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8" />

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

  <title>Healthcare Services</title>

  <style>

    body {

      font-family: Arial, sans-serif;

      margin: 0;

      padding: 0;

      background-color: #f8f8f8;

      color: #333;

    }

    .navbar {

      background-color: #0077b6;

      color: white;

      padding: 15px 20px;

      display: flex;

      justify-content: space-between;

      align-items: center;

    }

    .navbar .logo {

      font-size: 24px;

      font-weight: bold;

    }

    .navbar a {

      color: white;

      text-decoration: none;

      margin-left: 20px;

      font-weight: bold;

    }

    header {

      text-align: center;

      background-color: #2e86de;

      color: white;

      padding: 40px 20px;

    }

    header h1 {

      font-size: 36px;

      margin-bottom: 10px;

    }

    .services-container {

      display: flex;

      flex-wrap: wrap;

      gap: 20px;

      justify-content: center;

      padding: 40px 20px;

    }

    .service-card {

      background-color: white;

      border-radius: 10px;

      box-shadow: 0 4px 10px rgba(0,0,0,0.1);

      width: 280px;

      padding: 20px;

      transition: transform 0.3s ease;

    }

    .service-card:hover {

      transform: translateY(-8px);

    }

    .service-card h3 {

      color: #0077b6;

      margin-bottom: 10px;

    }

    .service-card p {

      font-size: 14px;

      color: #555;

    }

    footer {

      text-align: center;

      background-color: #0077b6;

      color: white;

      padding: 10px 20px;

    }

  </style>

</head>

<body>

  <div class="navbar">

    <div class="logo">My HealthCare</div>

    <div>

      <a href="p.html">Home</a>

      <a href="bookappointment.html">Book Appointment</a>

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

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

    </div>

  </div>

  <header>

    <h1>Our Healthcare Services</h1>

    <p>Explore the wide range of services we offer to keep you healthy and cared for.</p>

  </header>

  <section class="services-container">

    <div class="service-card">

      <h3>Doctor Consultation</h3>

      <p>Get expert advice and guidance from certified medical professionals at your convenience.</p>

    </div>

    <div class="service-card">

      <h3>Lab Tests</h3>

      <p>Book blood tests and health screenings online and receive accurate reports at home.</p>

    </div>

    <div class="service-card">

      <h3>Home Care</h3>

      <p>Compassionate nurses and medical attendants to assist you in your home.</p>

    </div>

    <div class="service-card">

      <h3>Emergency Services</h3>

      <p>24/7 ambulance, trauma support, and urgent medical assistance when you need it the most.</p>

    </div>

    <div class="service-card">

      <h3>Mental Health Counseling</h3>

      <p>Speak to trained counselors and mental health experts confidentially.</p>

    </div>

    <div class="service-card">

      <h3>Vaccination</h3>

      <p>Get immunized with flu shots, COVID vaccines, and more at your convenience.</p>

    </div>

    <div class="service-card">

      <h3>Online Prescriptions</h3>

      <p>Renew your prescriptions and get medication reminders directly from your doctor.</p>

    </div>

    <div class="service-card">

      <h3>Health Packages</h3>

      <p>Choose from affordable full-body checkups and preventive care packages.</p>

    </div>

  </section>

  <footer>

    &copy; 2025 My HealthCare. All Rights Reserved.

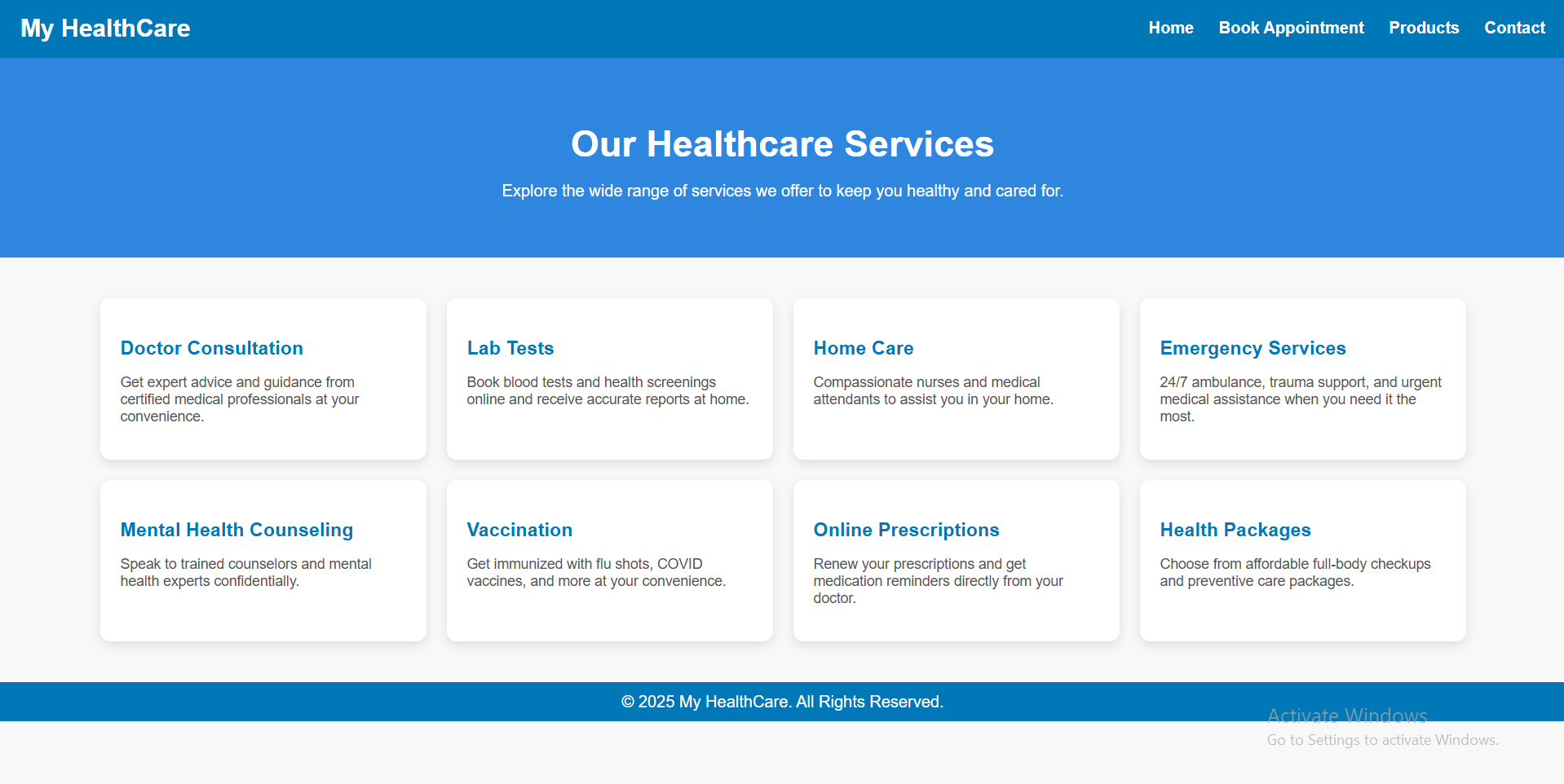
  </footer>

</body>

</html>

Output:

B. service page output:



Code:

C. appointment page:

code:

<!DOCTYPE html>

<html>

<head>

    <title>Shopping Cart</title>

    <style>

        body {

            font-family: Arial, sans-serif;

            background-color: #f4f4f4;

            text-align: center;

        }

        .appointment summary-container {

            background-color: white;

            padding: 20px;

            margin: 50px auto;

            width: 50%;

            border-radius: 8px;

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

        }

        .appointment summary-items {

            list-style: none;

            padding: 0;

        }

        .appointment summary-items li {

            display: flex;

            justify-content: space-between;

            padding: 10px;

            border-bottom: 1px solid #ddd;

        }

        .remove-item {

            background-color: red;

            color: white;

            border: none;

            padding: 5px 10px;

            cursor: pointer;

            border-radius: 5px;

        }

        .remove-item:hover {

            background-color: darkred;

        }

        .checkout-btn, .home-btn {

            display: block;

            margin: 10px auto;

            padding: 10px 15px;

            border: none;

            color: white;

            cursor: pointer;

            border-radius: 5px;

            width: 200px;

        }

        .checkout-btn {

            background-color: #007bff;

        }

        .checkout-btn:hover {

            background-color: #0056b3;

        }

        .home-btn {

            background-color: #28a745;

        }

        .home-btn:hover {

            background-color: #218838;

        }

    </style>

</head>

<body>

    <div class="appointment summary-container">

        <h2>Your Shopping Cart</h2>

        <ul id="appointment summary-items" class="appointment summary-items"></ul>

        <p id="total-price">Total Price: $0</p>

        <button class="checkout-btn" onclick="checkout()">Proceed to Checkout</button>

        <button class="home-btn" onclick="goHome()">Return to Home</button>

    </div>

    <script>

        let appointment summaryItems = JSON.parse(localStorage.getItem('appointment summary')) || [];

        let totalPrice = appointment summaryItems.reduce((sum, item) => sum + item.price, 0);

        function renderCart() {

            let appointment summaryList = document.getElementById('appointment summary-items');

            let totalPriceElement = document.getElementById('total-price');

            appointment summaryList.innerHTML = '';

            appointment summaryItems.forEach((item, index) => {

                let li = document.createElement('li');

                li.innerHTML = `${item.name} - $${item.price} <button class="remove-item" onclick="removeItem(${index})">Remove</button>`;

                appointment summaryList.appendChild(li);

            });

            totalPriceElement.innerText = `Total Price: $${totalPrice}`;

        }

        function removeItem(index) {

            totalPrice -= appointment summaryItems[index].price;

            appointment summaryItems.splice(index, 1);

            localStorage.setItem('appointment summary', JSON.stringify(appointment summaryItems));

            renderCart();

        }

        function checkout() {

            alert("Proceeding to checkout...");

        }

        function goHome() {

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

        }

        renderCart();

    </script>

</body>

</html><!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8" />

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

  <title>Book Appointment</title>

  <style>

    body {

      font-family: Arial, sans-serif;

      display: flex;

      justify-content: center;

      align-items: center;

      height: 100vh;

      margin: 0;

      background-color: #f4f4f4;

    }

    .appointment-form {

      background-color: white;

      padding: 20px;

      border-radius: 8px;

      box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

      width: 300px;

    }

    .appointment-form h2 {

      text-align: center;

      margin-bottom: 20px;

    }

    .appointment-form input,

    .appointment-form select {

      width: 100%;

      padding: 8px;

      margin-bottom: 15px;

      border: 1px solid #ddd;

      border-radius: 4px;

    }

    .appointment-form button {

      width: 100%;

      padding: 10px;

      background-color: #0077b6;

      color: white;

      border: none;

      border-radius: 4px;

      cursor: pointer;

    }

    .appointment-form button:hover {

      background-color: #023e8a;

    }

  </style>

</head>

<body>

  <div class="appointment-form">

    <h2>Book an Appointment</h2>

    <form>

      <input type="text" placeholder="Full Name" required />

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

      <input type="tel" placeholder="Phone Number" required />

      <select required>

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

        <option>General Physician</option>

        <option>Cardiology</option>

        <option>Neurology</option>

      </select>

      <input type="date" required />

      <button type="submit">Book Now</button>

    </form>

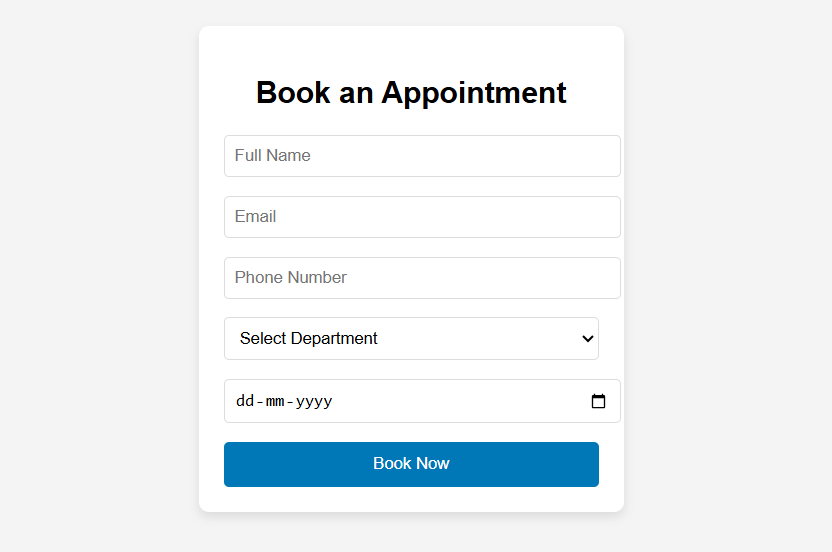
  </div>

</body>

</html>

Output:

C. appointment page  output:



Code:

D. Product page:

code:

<!DOCTYPE html>

<html>

<head>

  <title>Products - My HealthCare</title>

  <style>

    body {

      font-family: sans-serif;

      padding: 20px;

    }

    .navbar {

      background-color: #0077b6;

      padding: 10px;

      color: white;

    }

    .navbar a {

      color: white;

      margin-right: 15px;

      text-decoration: none;

    }

    .product {

      border: 1px solid #ccc;

      padding: 15px;

      margin: 10px 0;

      text-align: center;

    }

    .product img {

      max-width: 200px;

      height: auto;

      margin-bottom: 10px;

    }

    .product button {

      background-color: #0077b6;

      color: white;

      border: none;

      padding: 5px 10px;

      cursor: pointer;

    }

    footer {

      margin-top: 20px;

      text-align: center;

      color: #777;

    }

  </style>

</head>

<body>

  <div class="navbar">

    <a href="index.html">Home</a>

    <a href="cart.html">Cart (<span id="cartCount">0</span>)</a>

  </div>

  <h2>Products</h2>

  <div class="product">

    <img src="00121040\_L\_2\_7458.jpg" alt="Digital Thermometer">

    <h3>Digital Thermometer</h3>

    <p>Check body temperature easily.</p>

    <button onclick="addToCart('Digital Thermometer')">Add to Cart</button>

  </div>

  <div class="product">

    <img src="Pulse-Oximeter-600x605.jpg" alt="Pulse Oximeter">

    <h3>Pulse Oximeter</h3>

    <p>Measures oxygen level and pulse rate.</p>

    <button onclick="addToCart('Pulse Oximeter')">Add to Cart</button>

  </div>

  <div class="product">

    <img src="81zKjnsKyZL.\_AC\_SL1500\_.jpg" alt="Face Mask">

    <h3>Face Mask (Pack of 50)</h3>

    <p>Protective 3-layer face masks.</p>

    <button onclick="addToCart('Face Mask')">Add to Cart</button>

  </div>

  <div class="product">

    <img src="OIP.jpg" alt="Hand Sanitizer">

    <h3>Hand Sanitizer</h3>

    <p>Keep your hands clean and germ-free.</p>

    <button onclick="addToCart('Hand Sanitizer')">Add to Cart</button>

  </div>

  <footer>

    <p></p>

  </footer>

  <script>

    function addToCart(productName) {

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

      cart.push({name: productName});

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

      updateCartCount();

    }

    function updateCartCount() {

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

      document.getElementById('cartCount').textContent = cart.length;

    }

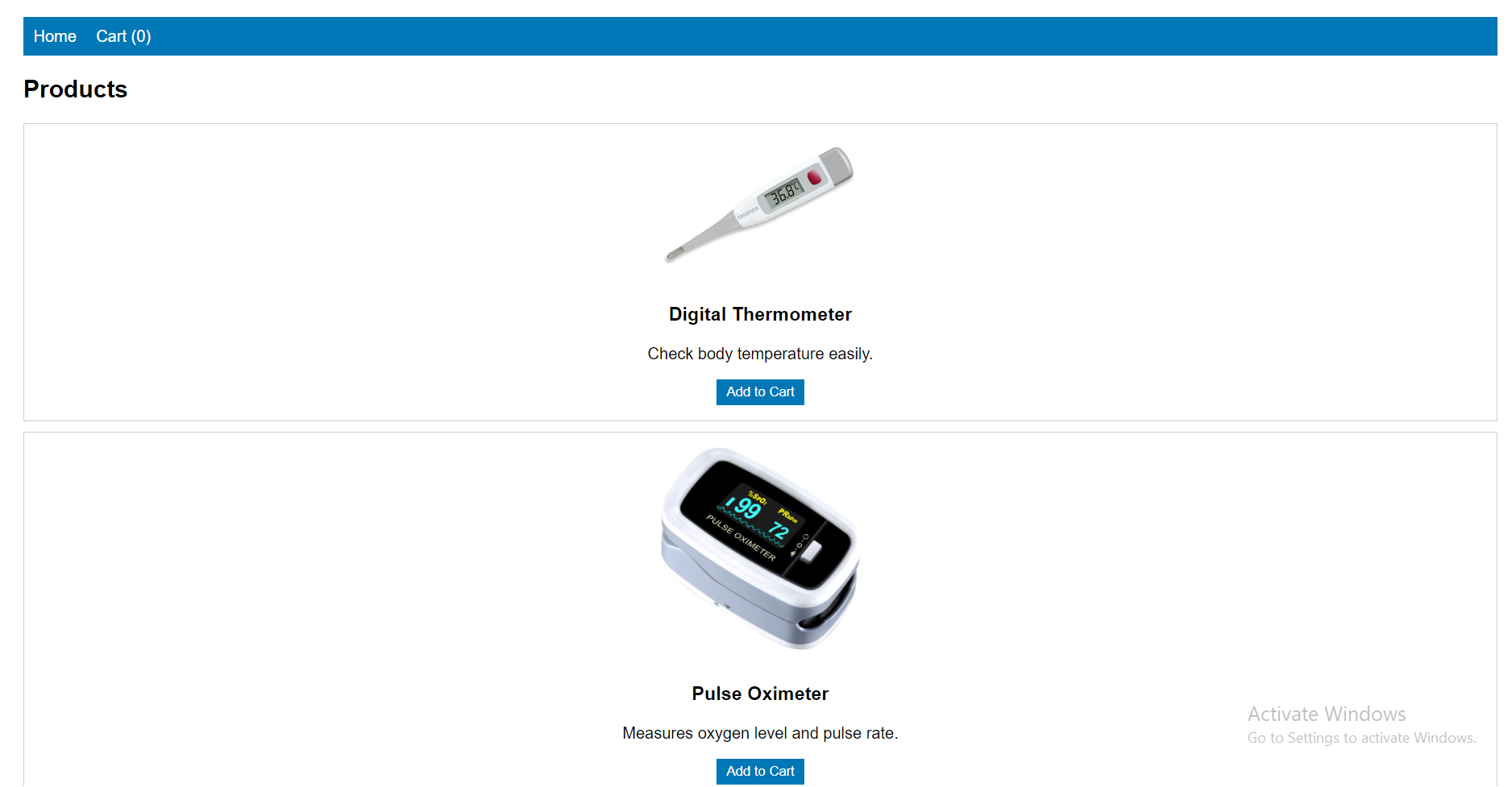
    updateCartCount();

  </script>

</body>

</html>

Output:



Code:

G. login page:

code:

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8" />

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

  <title>Login - Simple</title>

  <style>

    body {

      font-family: Arial, sans-serif;

      background-color: #f0f0f0;

      color: #333;

      margin: 0;

      padding: 0;

    }

    .login-container {

      width: 100%;

      max-width: 400px;

      margin: 100px auto;

      background-color: white;

      padding: 30px;

      border-radius: 8px;

      box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

    }

    h2 {

      text-align: center;

      margin-bottom: 20px;

    }

    input {

      width: 100%;

      padding: 12px;

      margin: 8px 0;

      border: 1px solid #ddd;

      border-radius: 4px;

    }

    button {

      width: 100%;

      padding: 12px;

      background-color: #0077b6;

      color: white;

      border: none;

      border-radius: 4px;

      cursor: pointer;

    }

    button:hover {

      background-color: #005f8a;

    }

    .forgot-password {

      text-align: center;

      margin-top: 10px;

    }

    .forgot-password a {

      text-decoration: none;

      color: #0077b6;

    }

  </style>

</head>

<body>

  <div class="login-container">

    <h2>Login</h2>

    <form action="#" method="POST">

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

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

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

    </form>

    <div class="forgot-password">

      <a href="#">Forgot password?</a>

    </div>

  </div>

</body>

</html>

Output:

A screenshot of a login box

AI-generated content may be incorrect.

Registration page

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8" />

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

  <title>Register - My HealthCare</title>

  <style>

    \* {

      margin: 0;

      padding: 0;

      box-sizing: border-box;

    }

    body {

      font-family: Arial, sans-serif;

      background-color: #f4f4f4;

      color: #333;

    }

    .navbar {

      background-color: #0077b6;

      display: flex;

      justify-content: space-between;

      align-items: center;

      padding: 10px 20px;

    }

    .navbar .logo {

      color: #fff;

      font-size: 24px;

      font-weight: bold;

    }

    .navbar ul {

      list-style: none;

      display: flex;

      gap: 20px;

    }

    .navbar ul li a {

      color: #fff;

      text-decoration: none;

      font-size: 16px;

    }

    .navbar ul li a:hover {

      text-decoration: underline;

    }

    .navbar .auth-btn {

      padding: 6px 12px;

      border-radius: 4px;

      text-decoration: none;

      font-size: 16px;

    }

    .navbar .auth-btn {

      background-color: transparent;

      color: #fff;

    }

    header {

      background-color: #2e86de;

      color: white;

      padding: 20px;

      text-align: center;

    }

    section {

      padding: 20px;

    }

    .form-section {

      background-color: white;

      padding: 20px;

      border-radius: 8px;

      width: 40%;

      margin: 20px auto;

      box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

    }

    .form-section input,

    .form-section select {

      width: 100%;

      padding: 10px;

      margin-bottom: 15px;

      border: 1px solid #ccc;

      border-radius: 4px;

    }

    .form-section button {

      background-color: #0077b6;

      color: white;

      border: none;

      padding: 10px 20px;

      cursor: pointer;

      border-radius: 4px;

      width: 100%;

    }

    .form-section button:hover {

      background-color: #023e8a;

    }

    .form-section p {

      text-align: center;

    }

    .form-section a {

      color: #0077b6;

      text-decoration: none;

    }

    .form-section a:hover {

      text-decoration: underline;

    }

  </style>

</head>

<body>

  <div class="navbar">

    <div class="logo">My HealthCare</div>

    <ul>

      <li><a href="#services">Services</a></li>

      <li><a href="#appointment">Book Appointment</a></li>

      <li><a href="#tips">Health Tips</a></li>

      <li><a href="#login" class="auth-btn login">Login</a></li>

      <li><a href="#contact">Contact</a></li>

    </ul>

  </div>

  <header>

    <h1>Register - My HealthCare</h1>

    <p>Join us and get access to premium healthcare services.</p>

  </header>

  <section>

    <div class="form-section">

      <h2>Create an Account</h2>

      <form>

        <input type="text" placeholder="Full Name" required />

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

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

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

        <input type="tel" placeholder="Phone Number" required />

        <select required>

          <option value="">Select Your Gender</option>

          <option>Male</option>

          <option>Female</option>

          <option>Other</option>

        </select>

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

      </form>

      <p>Already have an account? <a href="#login">Login here</a></p>

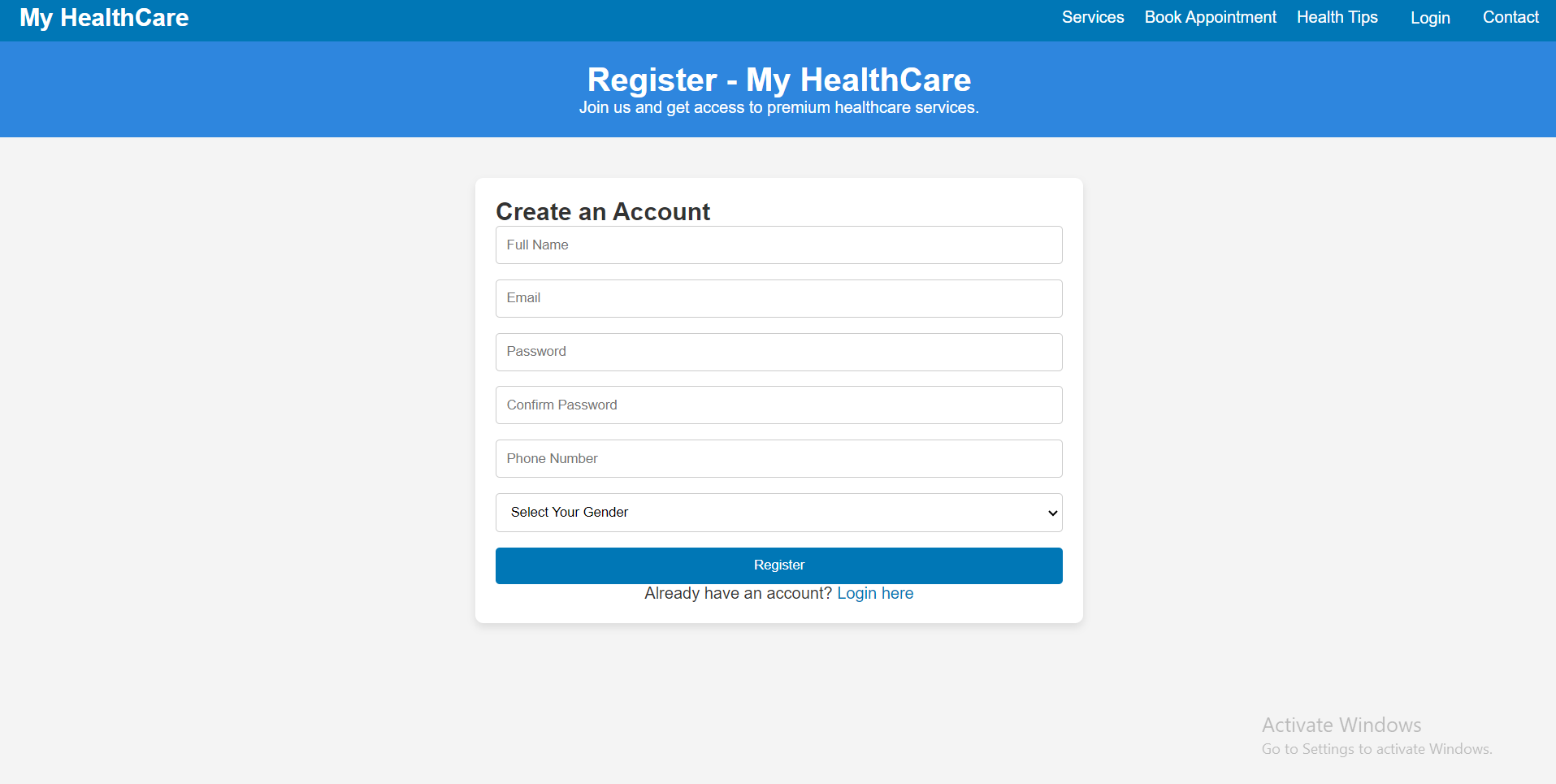
    </div>

  </section>

</body>

</html>

Output



Cart code

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

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

  <title>Shopping Cart - My HealthCare</title>

  <style>

    body {

      font-family: sans-serif;

      padding: 20px;

    }

    .navbar {

      background-color: #0077b6;

      padding: 10px;

      color: white;

    }

    .navbar a {

      color: white;

      margin-right: 15px;

      text-decoration: none;

    }

    .cart-items {

      margin-top: 20px;

    }

    .cart-item {

      border: 1px solid #ccc;

      padding: 15px;

      margin: 10px 0;

      display: flex;

      justify-content: space-between;

      align-items: center;

    }

    .cart-item button {

      background-color: #ff6f61;

      color: white;

      border: none;

      padding: 5px 10px;

      cursor: pointer;

    }

    .cart-item button:hover {

      background-color: #e05e4f;

    }

    .total {

      margin-top: 20px;

      font-size: 18px;

      font-weight: bold;

    }

    footer {

      margin-top: 20px;

      text-align: center;

      color: #777;

    }

  </style>

</head>

<body>

  <div class="navbar">

    <a href="p.html">Home</a>

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

    <a href="cart.html">Cart (<span id="cartCount">0</span>)</a>

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

    <a href="login.html">Login</a>

    <a href="#contact">Contact</a>

  </div>

  <h2>Shopping Cart</h2>

  <div class="cart-items" id="cartItems">

  </div>

  <div class="total" id="total">

    Total: ₹<span id="totalAmount">0.00</span>

  </div>

  <footer>

  </footer>

  <script>

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

    function updateCartCount() {

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

      cartCount.textContent = cart.length;

    }

    function displayCartItems() {

      const cartItemsContainer = document.getElementById('cartItems');

      const totalAmountElement = document.getElementById('totalAmount');

      let totalAmount = 0;

      cartItemsContainer.innerHTML = '';

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

        const cartItemElement = document.createElement('div');

        cartItemElement.classList.add('cart-item');

        cartItemElement.innerHTML = `

          <span>${product.name}</span>

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

        `;

        cartItemsContainer.appendChild(cartItemElement);

        totalAmount += 10 \* 82;

      });

      totalAmountElement.textContent = totalAmount.toFixed(2);

    }

    function addToCart(productName) {

      const existingProduct = cart.find(item => item.name === productName);

      if (!existingProduct) {

        cart.push({ name: productName });

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

      }

      updateCartCount();

      displayCartItems();

    }

    function removeFromCart(index) {

      cart.splice(index, 1);

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

      displayCartItems();

      updateCartCount();

    }

    displayCartItems();

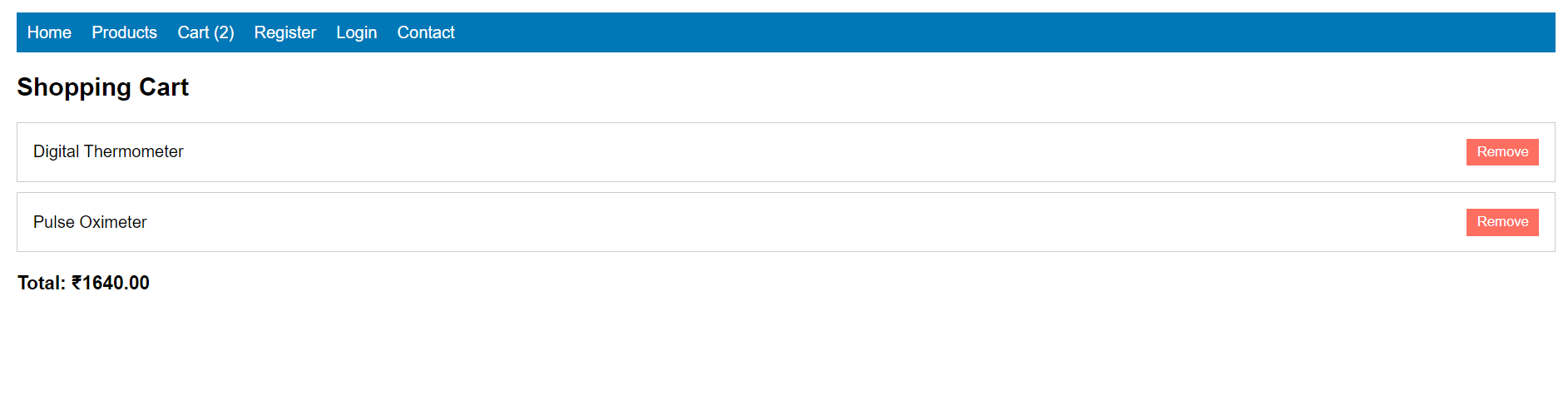
    updateCartCount();

  </script>

</body>

</html>

output



Conclusion

The community health center patient portal combines practical healthcare booking features with a sustainability-driven mission. The use of structured web design, user-friendly forms, and clear service categorization ensures a seamless experience for both shoppers and admins. With further backend integration, it can evolve into a fully operational online business supporting green technology use and community engagement.

Experiment No.3

Problem Statement:

CSS

1. Enhance the layout of the healthcare website using CSS Grid for the home page.
2. Use CSS Grid to layout the menu/product items in a structured and style the menu categories with appropriate headings, spacing, separators, images, descriptions, and prices.

Theory:

CSS Theory for Enhancing the Layout of a Second-Hand Gaming Clinic Website using CSS Grid

Introduction to CSS Grid

CSS Grid Layout is a two-dimensional layout system optimized for web interfaces. Unlike Flexbox (which is one-dimensional), CSS Grid allows layout control both across rows and columns, making it ideal for complex responsive layouts such as those found in healthcare booking patient portals.

Using CSS Grid, designers and developers can create clean, consistent, and responsive page structures. This is particularly helpful for:

* Landing pages with multiple content blocks (like a homepage)
* Service listings in multiple categories (like a service list page)
* Cart or gallery layouts with structured data display

Why CSS Grid for this Website?

In a second-hand healthcare service healthcare booking site, service presentation and layout are key to user satisfaction and engagement. Customers need to easily browse clinics, compare services, and take quick actions.

CSS Grid is used to:

* Arrange clinic items in a neat grid (3x3 or 4x4 etc.)
* Create sections like “Featured Clinics”, “Latest Deals”, or “Accessories” in distinct, well-defined grid blocks
* Ensure consistent alignment of images, text, and price details
* Support responsive design for mobile, tablet, and desktop screens

1. Home Page Layout with CSS Grid

The homepage is structured into visually defined areas using CSS Grid:

* A navigation header spanning full width
* A hero section with a large featured image or banner
* A three-column highlight section for featured categories or deals
* A testimonial section laid out in a row
* A footer with contact info and social links

Grid Benefits on Home Page:

* Easy to define large areas and control layout positions
* Aligns different components (text, images, buttons) in a consistent way
* Makes the layout scalable and responsive without relying heavily on media queries

2. Menu/Service Page Layout Using CSS Grid

This page displays the actual healthcare services and accessories in a structured manner. Items are grouped into categories like:

* PlayStation
* Xbox
* Nintendo
* Accessories
* Bundles

Each service is displayed as a card, and all cards are arranged using CSS Grid for better responsiveness and visual balance.

Key Grid Features on Service Page:

* Uniform item widths and spacing
* Grid gaps for breathing space between items
* Text (name, description, price) aligned properly under images
* Easily allows 2, 3, or 4 columns depending on screen size

Example CSS Grid Layout for Service Items:

.services-grid {

display: grid;

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

grid-gap: 30px;

padding: 20px;

}

Each service-card inside this grid will have:

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

Additional Styling Concepts:

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

Mobile Responsiveness with CSS Grid

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

Benefits for mobile users:

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

Code:

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8" />

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

  <title>Healthcare Info</title>

  <style>

    \* {

      margin: 0;

      padding: 0;

      box-sizing: border-box;

    }

    body {

      font-family: Arial, sans-serif;

      background-color: #f0f0f0;

      color: #333;

    }

    .navbar {

      background-color: #0077b6;

      display: flex;

      justify-content: space-between;

      align-items: center;

      padding: 10px 20px;

    }

    .navbar .logo {

      color: #fff;

      font-size: 24px;

      font-weight: bold;

    }

    .navbar ul {

      list-style: none;

      display: flex;

      gap: 20px;

    }

    .navbar ul li a {

      color: #fff;

      text-decoration: none;

      font-size: 16px;

      transition: color 0.3s;

    }

    .navbar ul li a:hover {

      color: #ff6f61;

      text-decoration: underline;

    }

    .navbar .auth-btn {

      padding: 6px 12px;

      border-radius: 4px;

      text-decoration: none;

      font-size: 16px;

    }

    .navbar .auth-btn {

      background-color: transparent;

      color: #fff;

    }

    header {

      background-color: #2e86de;

      color: white;

      padding: 40px 20px;

      text-align: center;

      background-image: url('header-image.jpg');

      background-size: cover;

      background-position: center;

      height: 300px;

      display: flex;

      flex-direction: column;

      justify-content: center;

    }

    header h1 {

      font-size: 40px;

      font-weight: 700;

    }

    header p {

      font-size: 20px;

    }

    section {

      padding: 40px 20px;

      text-align: center;

    }

    .services {

      display: flex;

      gap: 20px;

      flex-wrap: wrap;

      justify-content: center;

    }

    .service-box {

      border: 1px solid #ddd;

      padding: 20px;

      width: 250px;

      border-radius: 12px;

      background-color: #fff;

      box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

      transition: transform 0.3s ease, box-shadow 0.3s ease;

    }

    .service-box:hover {

      transform: translateY(-10px);

      box-shadow: 0 6px 12px rgba(0, 0, 0, 0.2);

    }

    footer {

      background-color: #2e86de;

      color: white;

      text-align: center;

      padding: 10px;

    }

    .contact input,

    .contact textarea {

      width: 100%;

      padding: 10px;

      margin-top: 5px;

      margin-bottom: 15px;

      border-radius: 8px;

      border: 1px solid #ddd;

    }

    .contact button {

      background-color: #0077b6;

      color: white;

      border: none;

      padding: 10px 20px;

      cursor: pointer;

      border-radius: 8px;

      transition: background-color 0.3s ease;

    }

    .contact button:hover {

      background-color: #023e8a;

    }

    .form-section input,

    .form-section select {

      width: 100%;

      padding: 8px;

      margin-bottom: 10px;

      border-radius: 8px;

      border: 1px solid #ddd;

    }

    .form-section button {

      background-color: #0077b6;

      color: white;

      border: none;

      padding: 8px 16px;

      cursor: pointer;

    }

    @media (max-width: 768px) {

      .services {

        flex-direction: column;

        align-items: center;

      }

      .navbar ul {

        flex-direction: column;

      }

    }

    @keyframes fadeIn {

      to {

        opacity: 1;

      }

    }

    .service-box {

      opacity: 0;

      animation: fadeIn 0.5s forwards;

    }

  </style>

</head>

<body>

    <div class="navbar">

        <div class="logo">My HealthCare</div>

        <ul>

          <li><a href="service.html">Service</a></li>

          <li><a href="bookappointment.html">Book Appointment</a></li>

          <li><a href="#tips">Health Tips</a></li>

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

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

          <li><a href="registration.html">Register</a></li>

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

          <li><a href="#contact">Contact</a></li>

          <li><a href="#service">Service</a></li>

        </ul>

      </div>

  <header>

    <h1>Welcome to My HealthCare</h1>

    <p>Your Health is Our Priority</p>

  </header>

  <section id="services">

    <h2>Our Services</h2>

    <div class="services">

      <div class="service-box">

        <h3>Doctor Consultation</h3>

        <p>Talk to our expert doctors anytime.</p>

      </div>

      <div class="service-box">

        <h3>Lab Tests</h3>

        <p>Affordable and accurate lab testing.</p>

      </div>

      <div class="service-box">

        <h3>Home Care</h3>

        <p>Nurses and support at your doorstep.</p>

      </div>

      <div class="service-box">

        <h3>Emergency Services</h3>

        <p>24/7 ambulance and trauma support.</p>

      </div>

      <div class="service-box">

        <h3>Mental Health Counseling</h3>

        <p>Talk to certified counselors confidentially.</p>

      </div>

      <div class="service-box">

        <h3>Vaccination & Immunization</h3>

        <p>COVID-19, flu, and other vaccinations available.</p>

      </div>

      <div class="service-box">

        <h3>Online Prescription</h3>

        <p>Get your prescriptions online without hassle.</p>

      </div>

    </div>

  </section>

  <section id="tips">

    <h2>Health Tips</h2>

    <ul>

      <li>Drink plenty of water daily.</li>

      <li>Exercise for at least 30 minutes a day.</li>

      <li>Eat more fruits and vegetables.</li>

      <li>Avoid junk food and sugary drinks.</li>

    </ul>

  </section>

  <section id="contact" class="py-20 px-4 bg-white text-center">

    <h3 class="text-3xl font-bold mb-6">Get In Touch</h3>

    <p class="mb-8 text-lg text-gray-600">We’re here to answer any questions or provide assistance. Feel free to reach out to us.</p>

    <form class="max-w-xl mx-auto space-y-4">

      <input type="text" placeholder="Your Name" class="w-full p-3 border rounded-lg" required />

      <input type="email" placeholder="Your Email" class="w-full p-3 border rounded-lg" required />

      <textarea placeholder="Your Message" class="w-full p-3 border rounded-lg" rows="4" required></textarea>

      <button type="submit" class="bg-blue-600 text-white px-6 py-3 rounded-lg hover:bg-blue-700">

        Send Message

      </button>

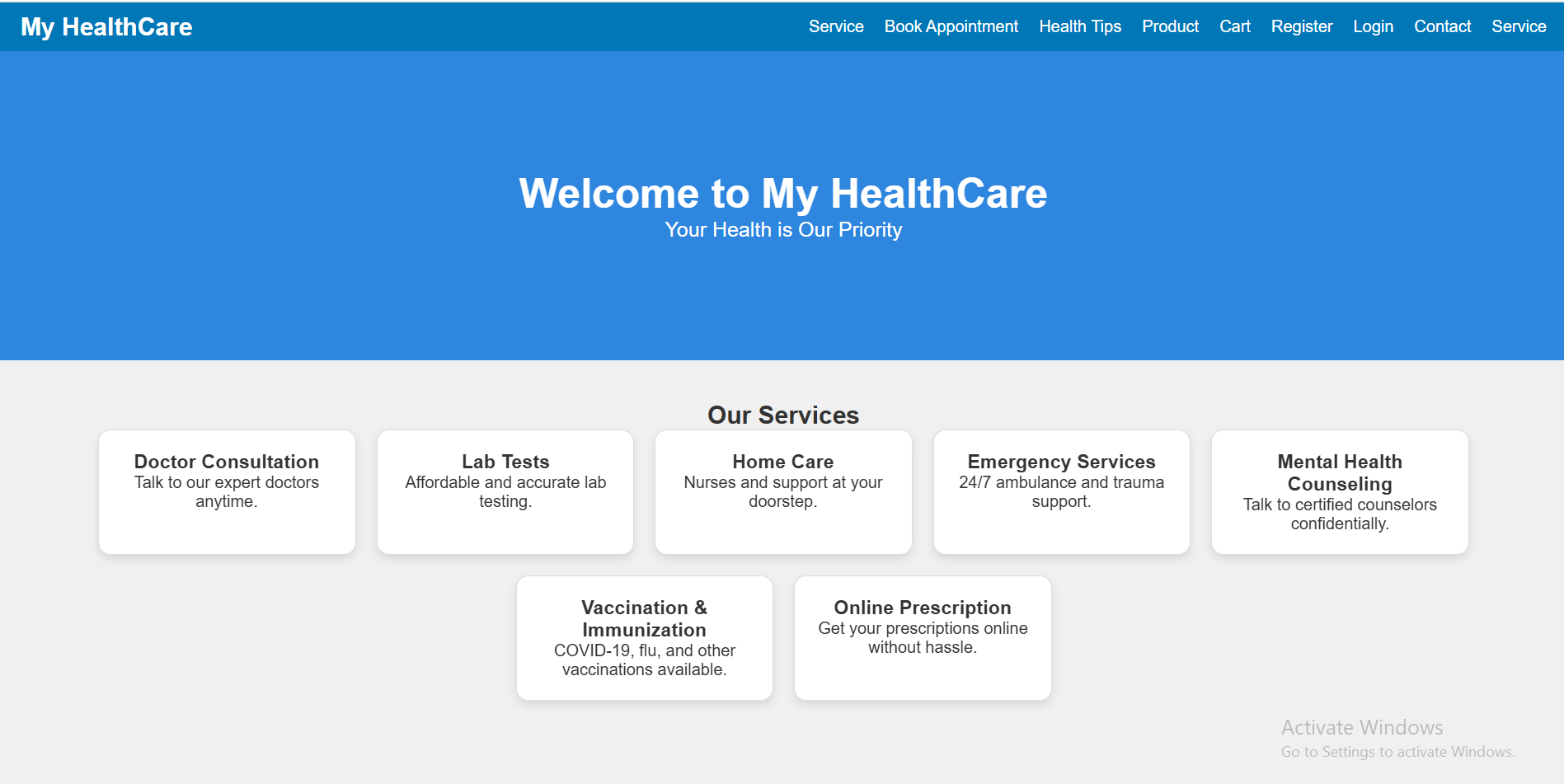
    </form>

  </section>

</body>

</html>

Output



Conclusion

CSS Grid is a powerful tool for building modern, responsive, and structured patient portals. In the case of a community health center, CSS Grid simplifies complex layout structures, enhances visual clarity, and provides a clean, user-friendly interface.

By using CSS Grid:

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

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

Experiment No.4

Problem Statement:

1. Enhance the cart page to make it user-friendly and visually appealing. Style the cart items with appropriate margins, paddings, and input field styles to provide a seamless shopping experience.
2. Enhance and style the about us page with appropriate margins, paddings, and input field styles.
3. Enhance and style the contact page to make it user-friendly and visually appealing. Style the contact form with appropriate margins, paddings, and input field styles.
4. Enhance and style the admin/user registration form with appropriate margins, paddings, and input field styles.
5. Enhance and style the admin/user login form with appropriate margins, paddings, and input field styles.

Theory

1. Why CSS Styling Matters in E-commerce Websites

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

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

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

Page-wise CSS Styling Theory

1. Cart Page

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

Key Styling Techniques:

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

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

2. About Our Healthcare Center Page

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

Key Styling Techniques:

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

Result: A professional and inviting presentation that builds trust.

3. Contact Page

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

Key Styling Techniques:

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

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

4. Admin/User New Patient Registration Form

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

Key Styling Techniques:

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

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

5. Admin/User Patient Login Form

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

Key Styling Techniques:

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

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

Code:

Cart page:

code:

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

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

  <title>Shopping Cart - My HealthCare</title>

  <style>

    body {

      font-family: sans-serif;

      padding: 20px;

    }

    .navbar {

      background-color: #0077b6;

      padding: 10px;

      color: white;

    }

    .navbar a {

      color: white;

      margin-right: 15px;

      text-decoration: none;

    }

    .cart-items {

      margin-top: 20px;

    }

    .cart-item {

      border: 1px solid #ccc;

      padding: 15px;

      margin: 10px 0;

      display: flex;

      justify-content: space-between;

      align-items: center;

    }

    .cart-item button {

      background-color: #ff6f61;

      color: white;

      border: none;

      padding: 5px 10px;

      cursor: pointer;

    }

    .cart-item button:hover {

      background-color: #e05e4f;

    }

    .total {

      margin-top: 20px;

      font-size: 18px;

      font-weight: bold;

    }

    footer {

      margin-top: 20px;

      text-align: center;

      color: #777;

    }

  </style>

</head>

<body>

  <div class="navbar">

    <a href="p.html">Home</a>

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

    <a href="cart.html">Cart (<span id="cartCount">0</span>)</a>

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

    <a href="login.html">Login</a>

    <a href="#contact">Contact</a>

  </div>

  <h2>Shopping Cart</h2>

  <div class="cart-items" id="cartItems">

  </div>

  <div class="total" id="total">

    Total: ₹<span id="totalAmount">0.00</span>

  </div>

  <footer>

  </footer>

  <script>

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

    function updateCartCount() {

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

      cartCount.textContent = cart.length;

    }

    function displayCartItems() {

      const cartItemsContainer = document.getElementById('cartItems');

      const totalAmountElement = document.getElementById('totalAmount');

      let totalAmount = 0;

      cartItemsContainer.innerHTML = '';

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

        const cartItemElement = document.createElement('div');

        cartItemElement.classList.add('cart-item');

        cartItemElement.innerHTML = `

          <span>${product.name}</span>

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

        `;

        cartItemsContainer.appendChild(cartItemElement);

        totalAmount += 10 \* 82;

      });

      totalAmountElement.textContent = totalAmount.toFixed(2);

    }

    function addToCart(productName) {

      const existingProduct = cart.find(item => item.name === productName);

      if (!existingProduct) {

        cart.push({ name: productName });

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

      }

      updateCartCount();

      displayCartItems();

    }

    function removeFromCart(index) {

      cart.splice(index, 1);

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

      displayCartItems();

      updateCartCount();

    }

    displayCartItems();

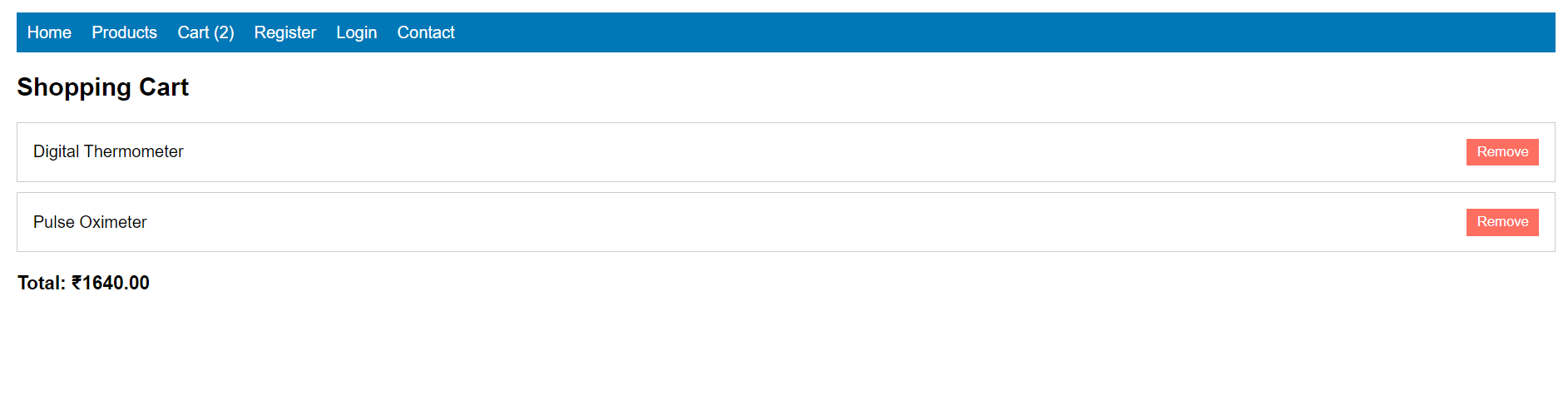
    updateCartCount();

  </script>

</body>

</html>

Output:



Code:

registration page:

code:

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8" />

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

  <title>Register - My HealthCare</title>

  <style>

    \* {

      margin: 0;

      padding: 0;

      box-sizing: border-box;

    }

    body {

      font-family: Arial, sans-serif;

      background-color: #f4f4f4;

      color: #333;

    }

    .navbar {

      background-color: #0077b6;

      display: flex;

      justify-content: space-between;

      align-items: center;

      padding: 10px 20px;

    }

    .navbar .logo {

      color: #fff;

      font-size: 24px;

      font-weight: bold;

    }

    .navbar ul {

      list-style: none;

      display: flex;

      gap: 20px;

    }

    .navbar ul li a {

      color: #fff;

      text-decoration: none;

      font-size: 16px;

    }

    .navbar ul li a:hover {

      text-decoration: underline;

    }

    .navbar .auth-btn {

      padding: 6px 12px;

      border-radius: 4px;

      text-decoration: none;

      font-size: 16px;

    }

    .navbar .auth-btn {

      background-color: transparent;

      color: #fff;

    }

    header {

      background-color: #2e86de;

      color: white;

      padding: 20px;

      text-align: center;

    }

    section {

      padding: 20px;

    }

    .form-section {

      background-color: white;

      padding: 20px;

      border-radius: 8px;

      width: 40%;

      margin: 20px auto;

      box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

    }

    .form-section input,

    .form-section select {

      width: 100%;

      padding: 10px;

      margin-bottom: 15px;

      border: 1px solid #ccc;

      border-radius: 4px;

    }

    .form-section button {

      background-color: #0077b6;

      color: white;

      border: none;

      padding: 10px 20px;

      cursor: pointer;

      border-radius: 4px;

      width: 100%;

    }

    .form-section button:hover {

      background-color: #023e8a;

    }

    .form-section p {

      text-align: center;

    }

    .form-section a {

      color: #0077b6;

      text-decoration: none;

    }

    .form-section a:hover {

      text-decoration: underline;

    }

  </style>

</head>

<body>

  <div class="navbar">

    <div class="logo">My HealthCare</div>

    <ul>

      <li><a href="#services">Services</a></li>

      <li><a href="#appointment">Book Appointment</a></li>

      <li><a href="#tips">Health Tips</a></li>

      <li><a href="#login" class="auth-btn login">Login</a></li>

      <li><a href="#contact">Contact</a></li>

    </ul>

  </div>

  <header>

    <h1>Register - My HealthCare</h1>

    <p>Join us and get access to premium healthcare services.</p>

  </header>

  <section>

    <div class="form-section">

      <h2>Create an Account</h2>

      <form>

        <input type="text" placeholder="Full Name" required />

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

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

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

        <input type="tel" placeholder="Phone Number" required />

        <select required>

          <option value="">Select Your Gender</option>

          <option>Male</option>

          <option>Female</option>

          <option>Other</option>

        </select>

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

      </form>

      <p>Already have an account? <a href="#login">Login here</a></p>

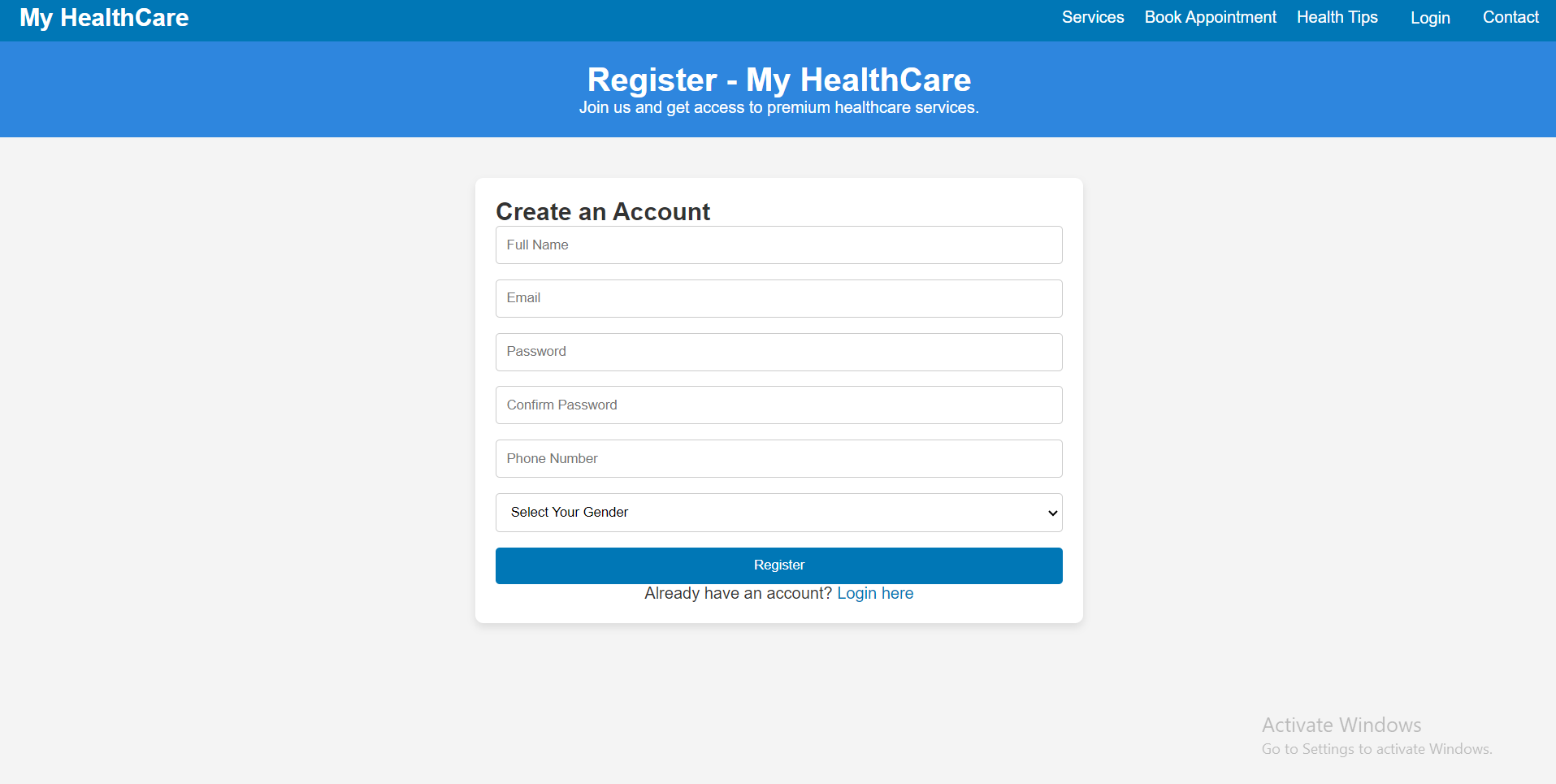
    </div>

  </section>

</body>

</html>

Output:



Code:

login page:

code:

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8" />

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

  <title>Login - Simple</title>

  <style>

    body {

      font-family: Arial, sans-serif;

      background-color: #f0f0f0;

      color: #333;

      margin: 0;

      padding: 0;

    }

    .login-container {

      width: 100%;

      max-width: 400px;

      margin: 100px auto;

      background-color: white;

      padding: 30px;

      border-radius: 8px;

      box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

    }

    h2 {

      text-align: center;

      margin-bottom: 20px;

    }

    input {

      width: 100%;

      padding: 12px;

      margin: 8px 0;

      border: 1px solid #ddd;

      border-radius: 4px;

    }

    button {

      width: 100%;

      padding: 12px;

      background-color: #0077b6;

      color: white;

      border: none;

      border-radius: 4px;

      cursor: pointer;

    }

    button:hover {

      background-color: #005f8a;

    }

    .forgot-password {

      text-align: center;

      margin-top: 10px;

    }

    .forgot-password a {

      text-decoration: none;

      color: #0077b6;

    }

  </style>

</head>

<body>

  <div class="login-container">

    <h2>Login</h2>

    <form action="#" method="POST">

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

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

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

    </form>

    <div class="forgot-password">

      <a href="#">Forgot password?</a>

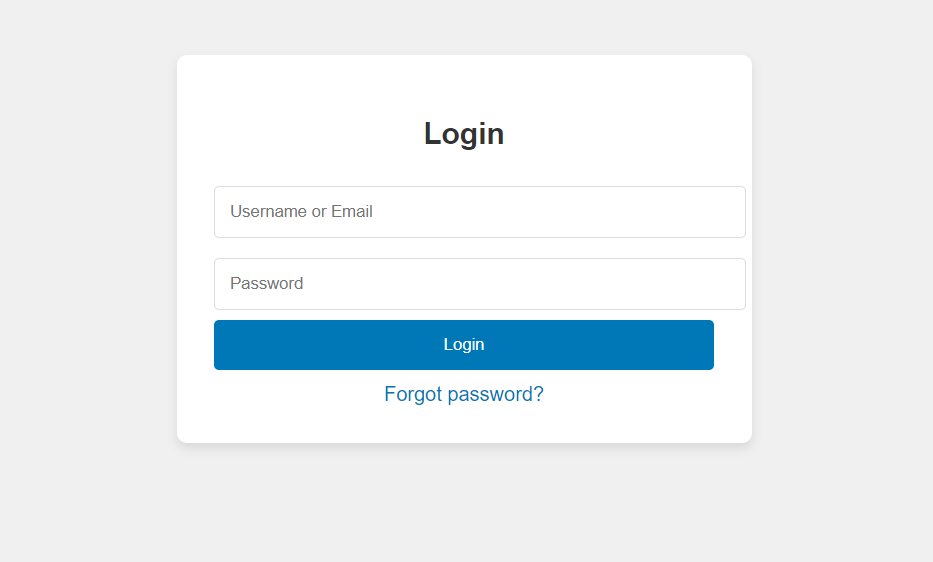
    </div>

  </div>

</body>

</html>

Output:



Conclusion

The visual and functional success of any healthcare booking platform, especially one like your second-hand healthcare service patient portal, relies heavily on how well the pages are styled using CSS. Applying appropriate margins, paddings, spacing, and input field enhancements ensures:

* Better user experience (UX)
* Improved readability and accessibility
* A more polished, professional appearance
* Higher engagement, conversion, and trust

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

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

Experiment No.5

Problem Statement:

1. Implement user registration and login forms for the healthcare website. These forms will allow users to create an account, log in, and access personalized features, such as saving favorite items or viewing order history.
2. User Registration Form will allow new customers to sign up and create an account on the website. The form will capture basic user details, including the name, email address, and password (not limited to these fields).
3. User Login Form will allow registered users to log into their accounts. The form will require an email address and a password to authenticate the user.
4. Provide validations for user registration and login forms to validate the input to ensure that all required fields are filled and that the email format is valid. (Contents beyond Syllabus)
5. Develop cart functionality to allow users to add items, update quantities, and remove items.

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

Introduction

In modern web development, client-side scripting using JavaScript is essential for creating interactive, responsive, and user-friendly applications. For an healthcare booking patient portal, particularly one focusing on second-hand healthcare services, implementing registration, login, form validation, and shopping appointment summary functionality is a core requirement to facilitate smooth user engagement and personalized services.

1. User New Patient Registration and Patient Login Forms

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

New Patient Registration Form

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

Key responsibilities of JavaScript in registration:

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

Patient Login Form

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

Key responsibilities of JavaScript in login:

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

2. JavaScript Form Validations

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

Typical validation tasks include:

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

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

3. Cart Functionality

The shopping appointment summary is an essential component of any healthcare booking patient portal. It allows users to review their selections, modify quantities, and proceed to checkout.

Key JavaScript implementations for the appointment summary include:

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

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

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>Register - My HealthCare</title>

  <style>

    \* {

      margin: 0;

      padding: 0;

      box-sizing: border-box;

    }

    body {

      font-family: Arial, sans-serif;

      background-color: #f4f4f4;

      color: #333;

    }

    .navbar {

      background-color: #0077b6;

      display: flex;

      justify-content: space-between;

      align-items: center;

      padding: 10px 20px;

    }

    .navbar .logo {

      color: #fff;

      font-size: 24px;

      font-weight: bold;

    }

    .navbar ul {

      list-style: none;

      display: flex;

      gap: 20px;

    }

    .navbar ul li a {

      color: #fff;

      text-decoration: none;

      font-size: 16px;

    }

    .navbar ul li a:hover {

      text-decoration: underline;

    }

    .navbar .auth-btn {

      padding: 6px 12px;

      border-radius: 4px;

      text-decoration: none;

      font-size: 16px;

    }

    .navbar .auth-btn {

      background-color: transparent;

      color: #fff;

    }

    header {

      background-color: #2e86de;

      color: white;

      padding: 20px;

      text-align: center;

    }

    section {

      padding: 20px;

    }

    .form-section {

      background-color: white;

      padding: 20px;

      border-radius: 8px;

      width: 40%;

      margin: 20px auto;

      box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

    }

    .form-section input,

    .form-section select {

      width: 100%;

      padding: 10px;

      margin-bottom: 15px;

      border: 1px solid #ccc;

      border-radius: 4px;

    }

    .form-section button {

      background-color: #0077b6;

      color: white;

      border: none;

      padding: 10px 20px;

      cursor: pointer;

      border-radius: 4px;

      width: 100%;

    }

    .form-section button:hover {

      background-color: #023e8a;

    }

    .form-section p {

      text-align: center;

    }

    .form-section a {

      color: #0077b6;

      text-decoration: none;

    }

    .form-section a:hover {

      text-decoration: underline;

    }

  </style>

</head>

<body>

  <div class="navbar">

    <div class="logo">My HealthCare</div>

    <ul>

      <li><a href="#services">Services</a></li>

      <li><a href="#appointment">Book Appointment</a></li>

      <li><a href="#tips">Health Tips</a></li>

      <li><a href="#login" class="auth-btn login">Login</a></li>

      <li><a href="#contact">Contact</a></li>

    </ul>

  </div>

  <header>

    <h1>Register - My HealthCare</h1>

    <p>Join us and get access to premium healthcare services.</p>

  </header>

  <section>

    <div class="form-section">

      <h2>Create an Account</h2>

      <form>

        <input type="text" placeholder="Full Name" required />

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

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

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

        <input type="tel" placeholder="Phone Number" required />

        <select required>

          <option value="">Select Your Gender</option>

          <option>Male</option>

          <option>Female</option>

          <option>Other</option>

        </select>

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

      </form>

      <p>Already have an account? <a href="#login">Login here</a></p>

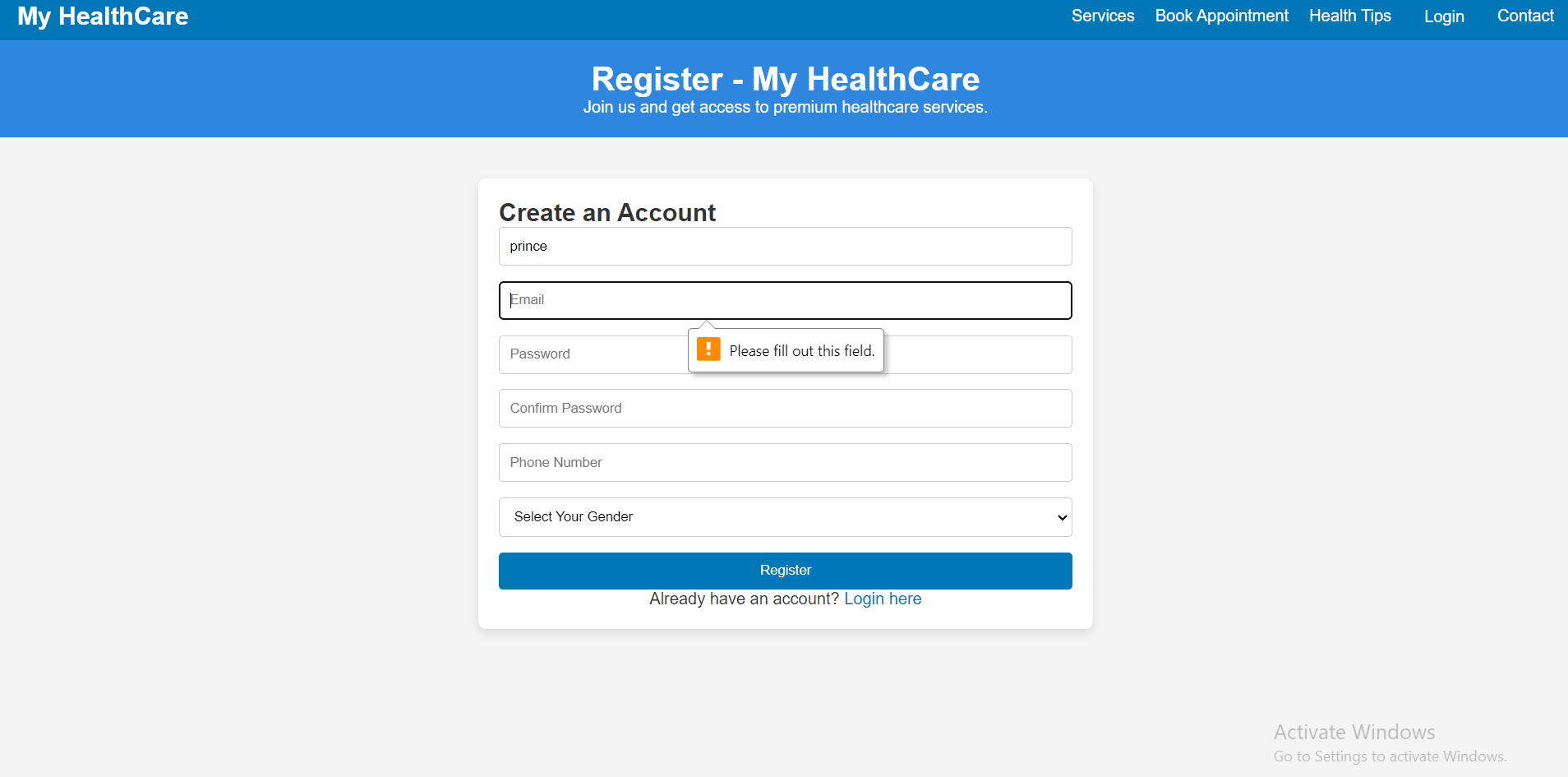
    </div>

  </section>

</body>

</html>

Output:



Code:

G. login page:

code:

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8" />

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

  <title>Login - Simple</title>

  <style>

    body {

      font-family: Arial, sans-serif;

      background-color: #f0f0f0;

      color: #333;

      margin: 0;

      padding: 0;

    }

    .login-container {

      width: 100%;

      max-width: 400px;

      margin: 100px auto;

      background-color: white;

      padding: 30px;

      border-radius: 8px;

      box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

    }

    h2 {

      text-align: center;

      margin-bottom: 20px;

    }

    input {

      width: 100%;

      padding: 12px;

      margin: 8px 0;

      border: 1px solid #ddd;

      border-radius: 4px;

    }

    button {

      width: 100%;

      padding: 12px;

      background-color: #0077b6;

      color: white;

      border: none;

      border-radius: 4px;

      cursor: pointer;

    }

    button:hover {

      background-color: #005f8a;

    }

    .forgot-password {

      text-align: center;

      margin-top: 10px;

    }

    .forgot-password a {

      text-decoration: none;

      color: #0077b6;

    }

  </style>

</head>

<body>

  <div class="login-container">

    <h2>Login</h2>

    <form action="#" method="POST">

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

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

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

    </form>

    <div class="forgot-password">

      <a href="#">Forgot password?</a>

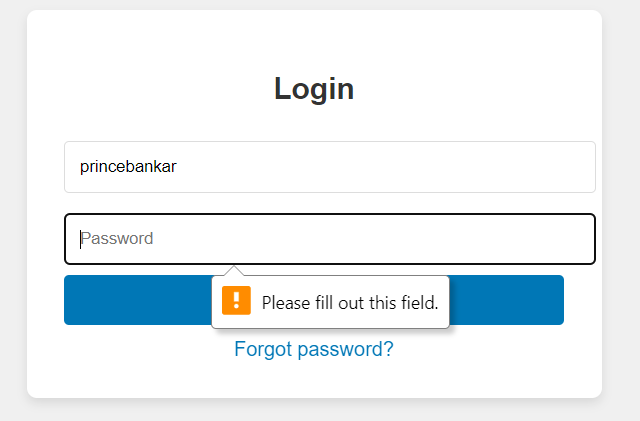
    </div>

  </div>

</body>

</html>

Output:



Conclusion

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

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

JavaScript, therefore, plays a critical role in transforming a static service listing site into a dynamic and functional healthcare booking platform.

Experiment No.6

Problem Statement:

1. The user login form will allow registered users to log into their accounts. The form will require an email address and a password to authenticate the user.
2. If the login is successful, the user should be redirected to the homepage or their user dashboard. (Contents beyond Syllabus)
3. Use localStorage or sessionStorage to store authentication data, such as the user’s email and login status. This ensures that once a user is logged in, they remain authenticated even after the page reloads or when they visit the site again. (Contents beyond Syllabus)
4. Save the cart data to local storage when items are added, updated, or removed. Retrieve and load the cart data from local storage when the page loads. (Contents beyond Syllabus)

Theory:

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

Introduction

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

1. Persistent Patient 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 centerd 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 centers:
  + 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 centerd 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 appointment summarys are central to any healthcare booking patient portal. 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 appointment summary.

Implementation Features:

* Every time a user adds, removes, or updates a service in the appointment summary:
  + JavaScript serializes the appointment summary array/object into JSON
  + This data is saved to localStorage
* On page load:
  + JavaScript checks if appointment summary data exists in localStorage
  + If it does, it parses and loads it into the appointment summary view
* The appointment summary 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 healthcare booking simulations or prototypes for portfolio projects

Code:

A. Home page:

code:

<!DOCTYPE html>

<html>

<head>

    <title>Patient Login</title>

    <style>

        body {

            font-family: Arial, sans-serif;

            background-color: #f4f4f4;

            display: flex;

            justify-content: center;

            align-items: center;

            height: 100vh;

        }

        form {

            background-color: white;

            padding: 25px;

            border-radius: 8px;

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

            width: 300px;

        }

        h2 {

            margin-bottom: 15px;

            text-align: center;

        }

        label {

            display: block;

            margin-top: 10px;

            font-weight: bold;

        }

        input.box {

            width: 100%;

            padding: 8px;

            margin-top: 5px;

            border: 1px solid #ccc;

            border-radius: 4px;

        }

        #show-pass {

            margin-top: 10px;

            background: none;

            border: none;

            color: blue;

            cursor: pointer;

            text-decoration: underline;

        }

        #submit-btn {

            margin-top: 15px;

            width: 100%;

            padding: 10px;

            background-color: #333;

            color: white;

            border: none;

            border-radius: 4px;

            cursor: pointer;

            opacity: 0.5;

        }

        #submit-btn:enabled {

            opacity: 1;

        }

        .msg {

            margin-top: 10px;

            font-weight: bold;

            text-align: center;

        }

    </style>

</head>

<body>

    <form>

        <h2>Patient Login Form</h2>

        <label for="username">User Name</label>

        <input type="text" class="box" placeholder="Enter User name" id="username" name="username">

        <label for="pass">Password</label>

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

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

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

        <div class="msg"></div>

    </form>

    <script>

        const submit = document.getElementById('submit-btn');

        const msgElement = document.querySelector('.msg');

        const showPassBtn = document.getElementById('show-pass');

        const usernameInput = document.getElementById('username');

        const passwordInput = document.getElementById('pass');

        const validUser = "OjasUmate";

        const validPass = "Ojas@123";

        usernameInput.addEventListener('input', validateForm);

        passwordInput.addEventListener('input', validateForm);

        function validateForm() {

            if (usernameInput.value.trim() && passwordInput.value.trim()) {

                submit.disabled = false;

            } else {

                submit.disabled = true;

            }

        }

        showPassBtn.addEventListener('click', function (e) {

            e.preventDefault();

            passwordInput.type = passwordInput.type === "password" ? "text" : "password";

            showPassBtn.textContent = passwordInput.type === "password" ? "Show Password" : "Hide Password";

        });

        submit.addEventListener('click', function (e) {

            e.preventDefault();

            let enteredUser = usernameInput.value.trim();

            let enteredPass = passwordInput.value;

            if (enteredUser === validUser && enteredPass === validPass) {

                msgElement.style.color = 'green';

                msgElement.textContent = 'Successfully logged in';

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

                setTimeout(() => {

                    window.location.href = "homepage.html";

                }, 2000);

            } else {

                msgElement.style.color = 'red';

                msgElement.textContent = 'Invalid Username or Password';

            }

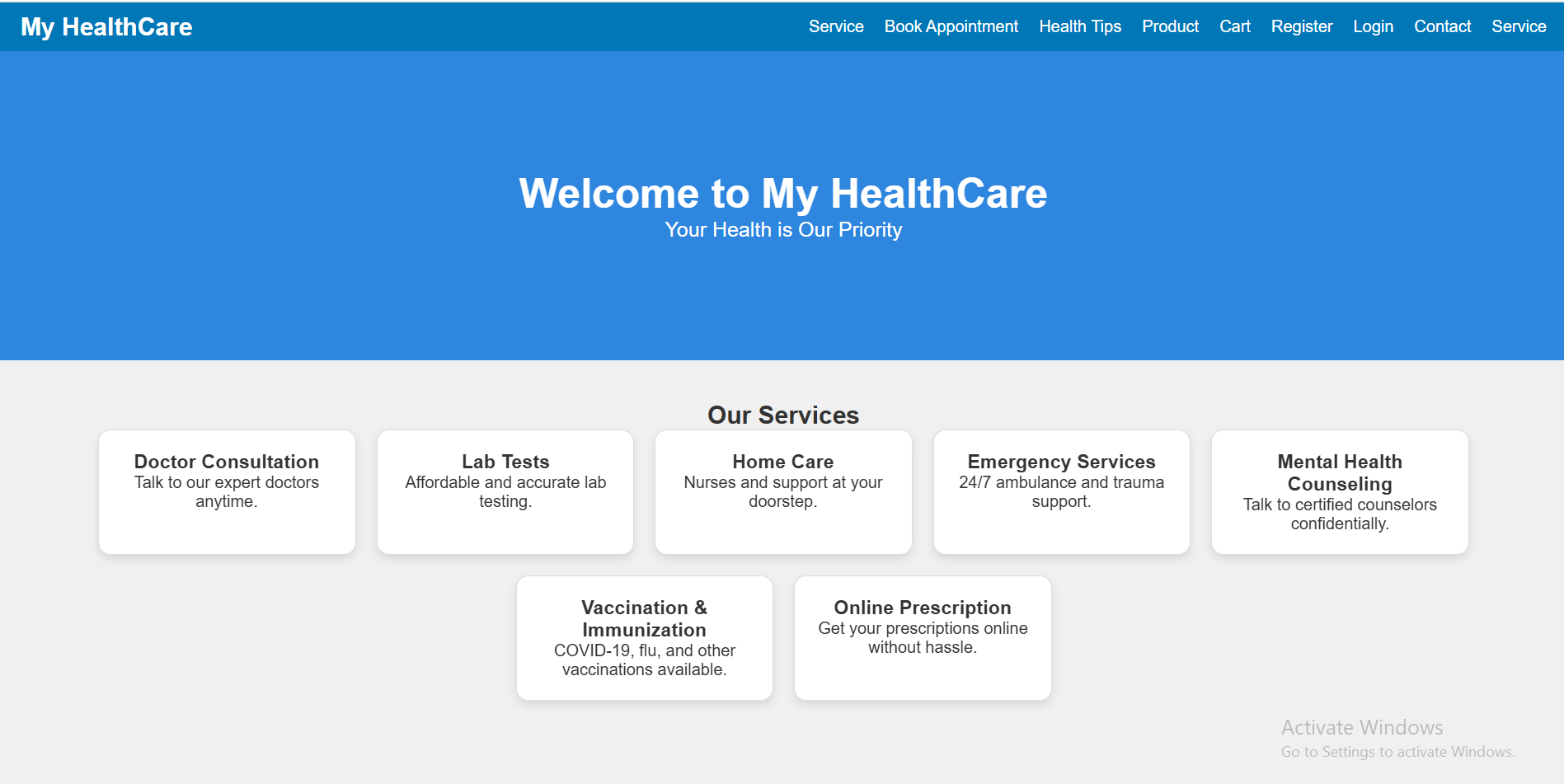
        });

    </script>

</body>

</html>

Output:



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 healthcare service patient portal, implementing persistent login and appointment summary functionality ensures that users have a smooth, uninterrupted interaction with the site.

By storing authentication states and appointment summary data locally:

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

These techniques mimic real-world behavior found in professional healthcare booking 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:

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

Theory:

User registration is a fundamental component of web applications, particularly in healthcare booking platforms like your second-hand healthcare service patient portal. 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 centers them securely into a database. To maintain security, user passwords are hashed before storage.

Core Elements of the PHP New Patient 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 center user data.
5. Error Handling: Displays messages for missing fields or registration failures.
6. User Feedback: Provides confirmation or redirection upon success.

CODE:-

<?php

$host = 'localhost';

$user = 'root';

$password = '';

$dbname = 'hostipsl\_center';

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

if ($conn->connect\_error) {

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

}

?>

New Patient 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"]);

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

} else {

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

}

$stmt->close();

} else {

foreach ($errors as $error) {

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

}

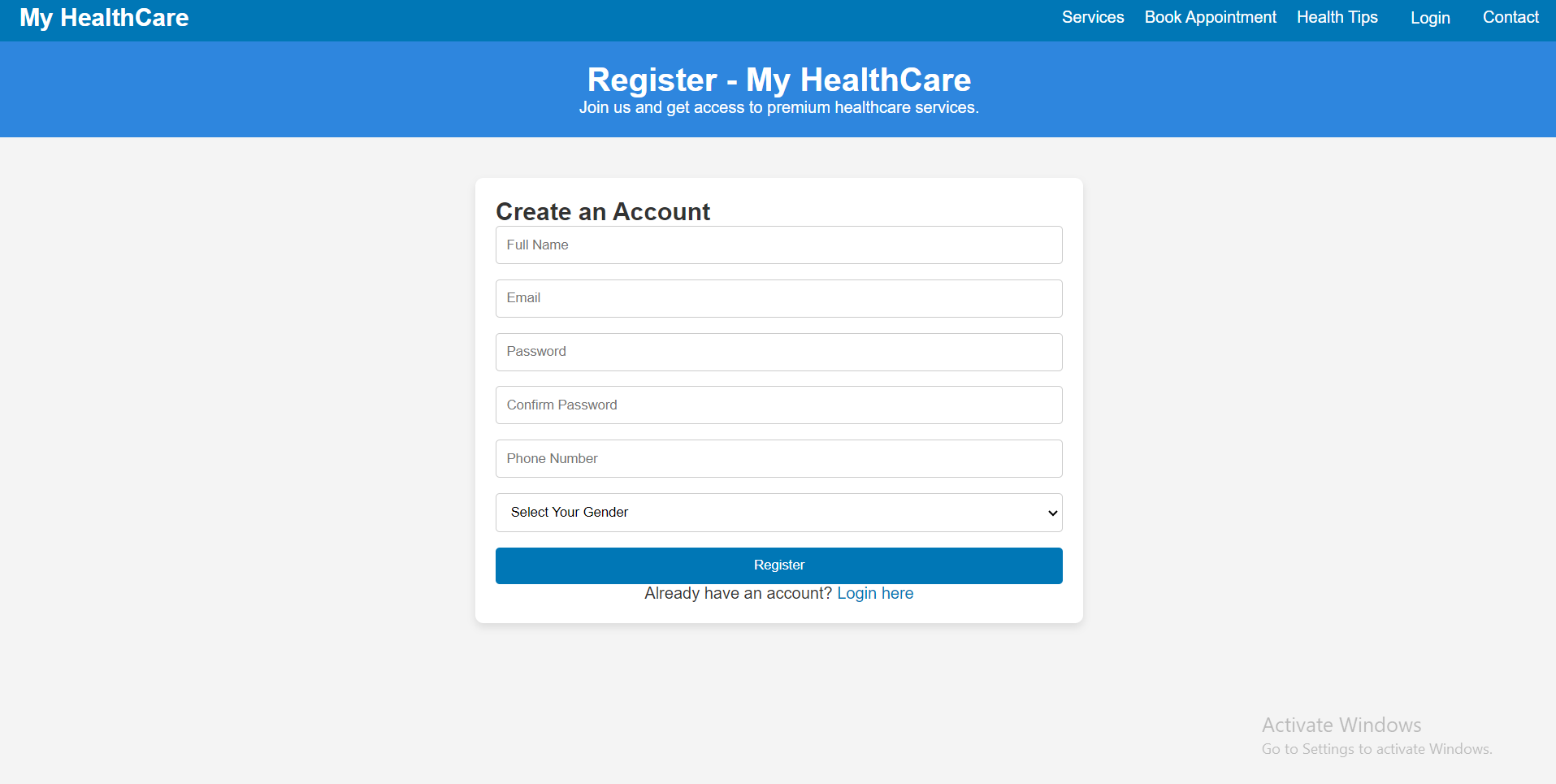
}

$conn->close();

}

?>

Output



Conclusion

Implementing user registration with PHP provides the backbone of user management in your patient portal. 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:

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

Theory:

PHP Patient Login System

A user login system is a fundamental component of most patient portals, especially healthcare booking platforms. It enables secure access to personalized features like managing appointment summarys, tracking orders, or viewing saved services. In PHP, login functionality typically involves:

* Capturing login credentials via a form (email and password).
* Validating inputs.
* Comparing credentials against centerd 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 centerd 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"]);

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

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

}

if (empty($password)) {

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

}

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();

}

foreach ($errors as $error) {

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

}

?>  
  
Patient Login form:-

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

<h2>Patient 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="Patient Login">

</form>

Dashboard:-

<?php

session\_start();

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

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

exit();

}

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

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

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

?>

<?php

session\_start();

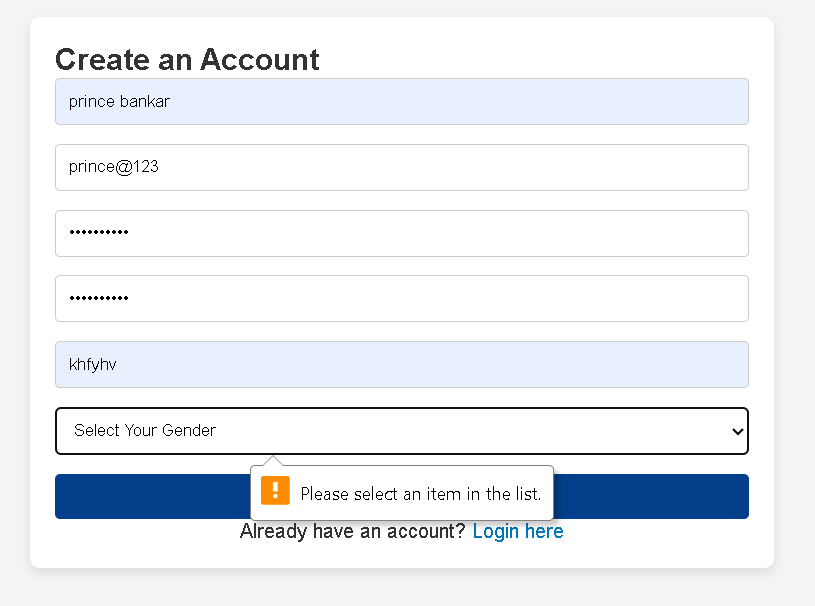
session\_destroy();

header("Location: login.html");

exit();

?>

Output



Conclusion

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

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

This login system:

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

Experiment 9

Problem Statement:

1. Develop a PHP script that allows users to manage their shopping cart for an healthcare website. The script should allow users to add items to their cart, view their cart contents, and remove items if needed.
2. Develop a PHP script to manage the shopping cart for an healthcare 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 appointment summary is a core component of any healthcare booking platform. It serves as a temporary storage space where users can collect and manage the items they wish to book. In the case of a second-hand healthcare services patient portal, where services can be unique and availability may be limited to single units, the shopping appointment summary 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 center appointment summary data in memory while the user is browsing. It is useful for fast prototyping and requires no database interaction.

Key Characteristics:

* Cart data is centerd in $\_SESSION.
* Data persists during the browsing session.
* No need to log in to use the appointment summary.
* Items are lost if the session expires or the browser is closed.

Operations Supported:

* Add to Cart: Add items by storing service ID, name, quantity, and price in session.
* View Cart: Display the contents centerd 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 appointment summary data is centerd in a MySQL database. It allows appointment summary contents to persist across user sessions, devices, and logins.

Key Characteristics:

* Each user has a unique appointment summary identified by user ID.
* Cart contents are centerd in a appointment summary table, and optionally a appointment summary\_items table for item details.
* Requires user login or session management.

Operations Supported:

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

Advantages:

* Cart is persistent and user-specific.
* Works across sessions and devices.
* Enables appointment summary analytics and user behavior tracking.

Limitations:

* Requires more setup and error handling.
* Needs secure login system to link appointment summary with user.

CODE:-

CREATE TABLE appointment summary\_items (

id INT AUTO\_INCREMENT PRIMARY KEY,

user\_id INT NOT NULL,

service\_id INT NOT NULL,

service\_name VARCHAR(255),

quantity INT DEFAULT 1,

price DECIMAL(10, 2),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

Code

<?php

session\_start();

require 'db.php';

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

    die("Please log in to add health items to your cart.");

}

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

$product\_id = $\_POST['product\_id'];

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

$sql = "SELECT \* FROM healthcare\_cart WHERE user\_id = ? AND product\_id = ?";

$stmt = $conn->prepare($sql);

$stmt->bind\_param("ii", $user\_id, $product\_id);

$stmt->execute();

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

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

    $row = $result->fetch\_assoc();

    $new\_quantity = $row['quantity'] + $quantity;

    $update = $conn->prepare("UPDATE healthcare\_cart SET quantity = ? WHERE user\_id = ? AND product\_id = ?");

    $update->bind\_param("iii", $new\_quantity, $user\_id, $product\_id);

    $update->execute();

} else {

    $insert = $conn->prepare("INSERT INTO healthcare\_cart (user\_id, product\_id, quantity) VALUES (?, ?, ?)");

    $insert->bind\_param("iii", $user\_id, $product\_id, $quantity);

    $insert->execute();

}

echo "Health item added to cart.";

?>

Remove code- <?php

session\_start();

require 'cart\_db.php';

$user\_id = 1;

if (isset($\_POST['add'])) {

    $stmt = $mysqli->prepare("INSERT INTO healthcare\_cart (user\_id, product\_id, name, price, qty) VALUES (?, ?, ?, ?, ?)");

    $stmt->bind\_param("iisdi", $user\_id, $\_POST['id'], $\_POST['name'], $\_POST['price'], $\_POST['qty']);

    $stmt->execute();

}

if (isset($\_GET['remove'])) {

    $stmt = $mysqli->prepare("DELETE FROM healthcare\_cart WHERE user\_id=? AND product\_id=?");

    $stmt->bind\_param("ii", $user\_id, $\_GET['remove']);

    $stmt->execute();

}

$result = $mysqli->query("SELECT \* FROM healthcare\_cart WHERE user\_id = $user\_id");

echo "<h2>Your Healthcare Cart</h2>";

if ($result->num\_rows === 0) {

    echo "Your cart is empty.";

} else {

    echo "<ul>";

    while ($item = $result->fetch\_assoc()) {

        echo "<li>{$item['name']} - ₹{$item['price']} x {$item['qty']}

              <a href='healthcare\_cart.php?remove={$item['product\_id']}'>Remove</a></li>";

    }

    echo "</ul>";

}

?>

Output

A screenshot of a computer

AI-generated content may be incorrect.

Conclusion

A shopping appointment summary system, whether session-based or database-driven, is essential for enhancing the user experience and improving sales on your second-hand healthcare service patient portal.

When using PHP:

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

For a fully functional and scalable patient portal, the MySQL-based appointment summary is highly recommended, as it:

* Improves user experience with persistent appointment summarys.
* Enables personalization and user analytics.
* Supports consistent item tracking (especially when each clinic unit is unique).

Experiment 10

Problem Statement:

1. 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.
2. 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 healthcare booking platform. It translates the user’s appointment summary 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 healthcare service patient portal, where services 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 centerd in the session ($\_SESSION['appointment summary']).
* On checkout, a confirmation message is shown.
* Useful for simple or demo applications. Workflow:

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

Advantages:

* Quick to implement.
* Minimal setup required.

Limitations:

* Data not persistent.
* Not scalable or serviceion-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 appointment summary 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 appointment summary.

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,

service\_id INT,

service\_name VARCHAR(255),

quantity INT,

price DECIMAL(10, 2)

);

Checkout session:-

<?php

session\_start();

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

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

echo "Your appointment summary 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['appointment summary'] 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>";

unset($\_SESSION['appointment summary']);

} else {

echo "Invalid request method.";

}

?>

MySQL-Based PHP Checkout Script:-

<?php

session\_start();

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

if ($conn->connect\_error) {

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

}

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

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

echo "Your appointment summary 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['appointment summary'] 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, service\_id, service\_name, quantity, price) VALUES (?, ?, ?, ?, ?)");

foreach ($\_SESSION['appointment summary'] 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['appointment summary']);

} else {

echo "Checkout failed. Please try again.";

}

$stmt->close();

$conn->close();

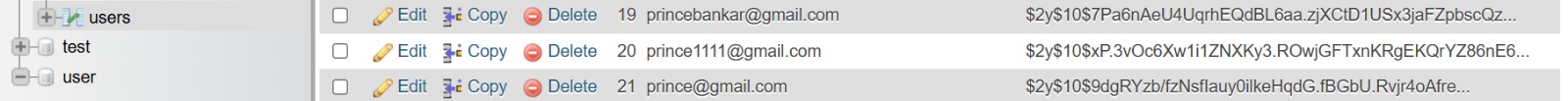
} else {

echo "Invalid request.";

}

?>

Output



Conclusion

The checkout process is the most vital component of an healthcare booking platform—it turns intent into action. For your second-hand healthcare services patient portal:

Use Case Importance:

* Unique item inventory means precise, real-time appointment summary 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.