AMRITA
VISHWA VIDYAPEETHAM

22AIE457

FULL STACK DEVELOPMENT

LAB REPORT

Regn. No: CH.EN.U4AIE22046

LAB EXPERIMENT - 1

Git and GitHub Tutorial – Version Control for Beginners

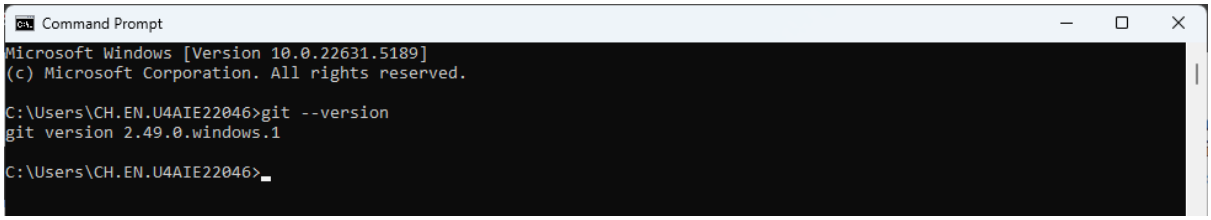
Aim: To study and perform basic Git operations including repository creation, committing, branching, merging, pushing, and cloning.

Software Requirements:

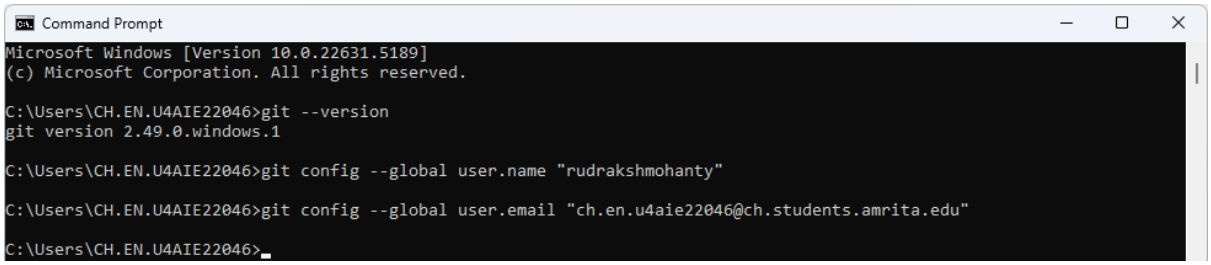
- ✓ Windows 11 / 11
- ✓ Git (version 2.50.0 or later)
- ✓ GitHub account

Commands Executed & Explanation:

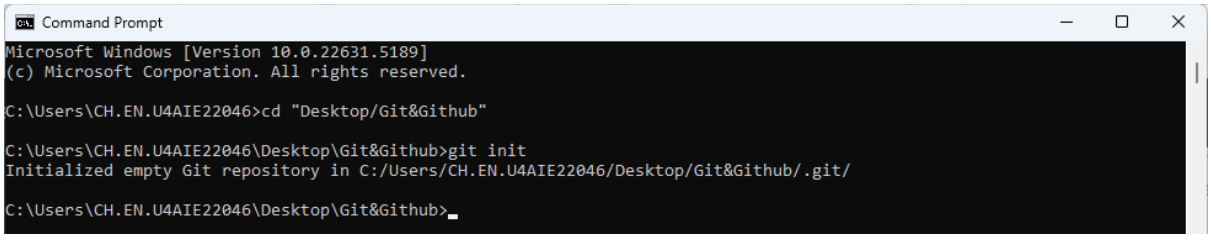
1. Configure Git



➢ Checks Git installation and sets username & email globally.



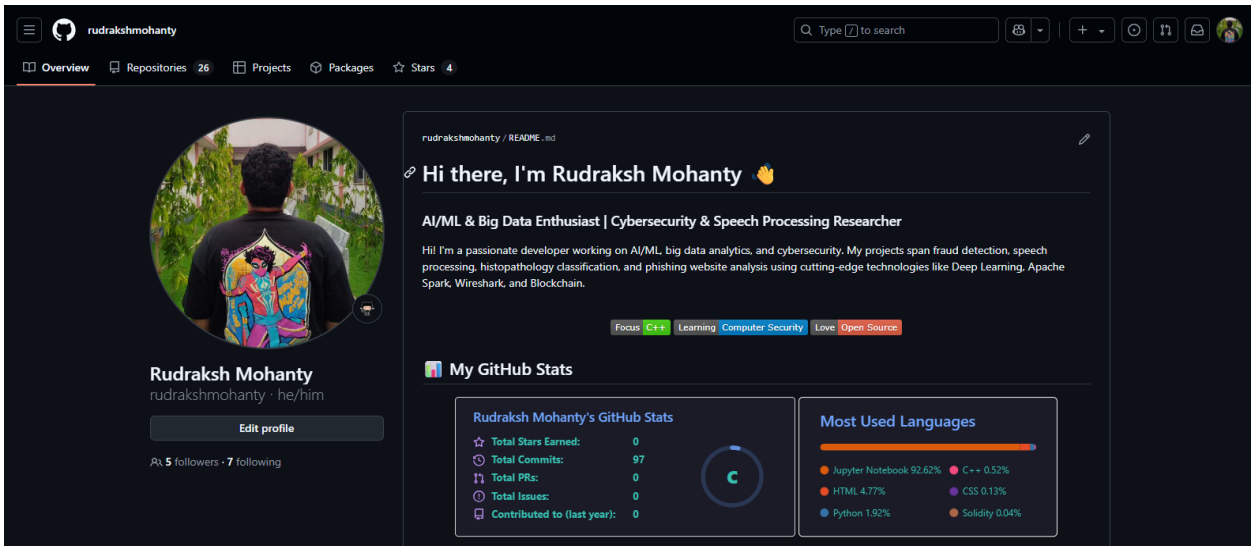
2. Create Local Repository



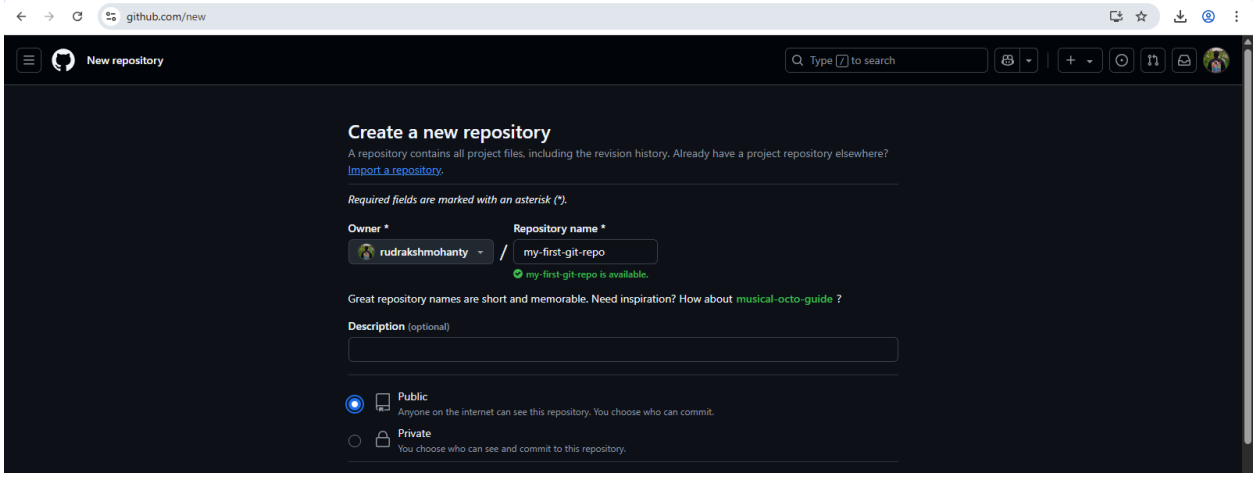
➢ Creates new folder and initializes Git repository.

Push A Repository To GitHub:

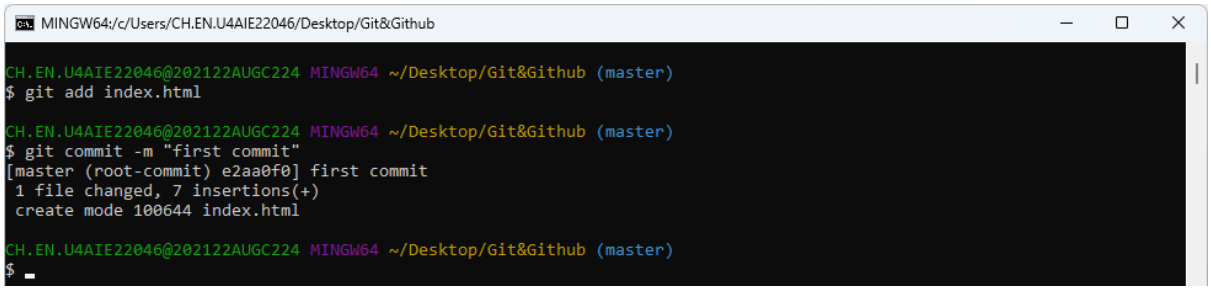
→ **Step-1:** Create a GitHub Account:



→ **Step-2:** Create a Repository:

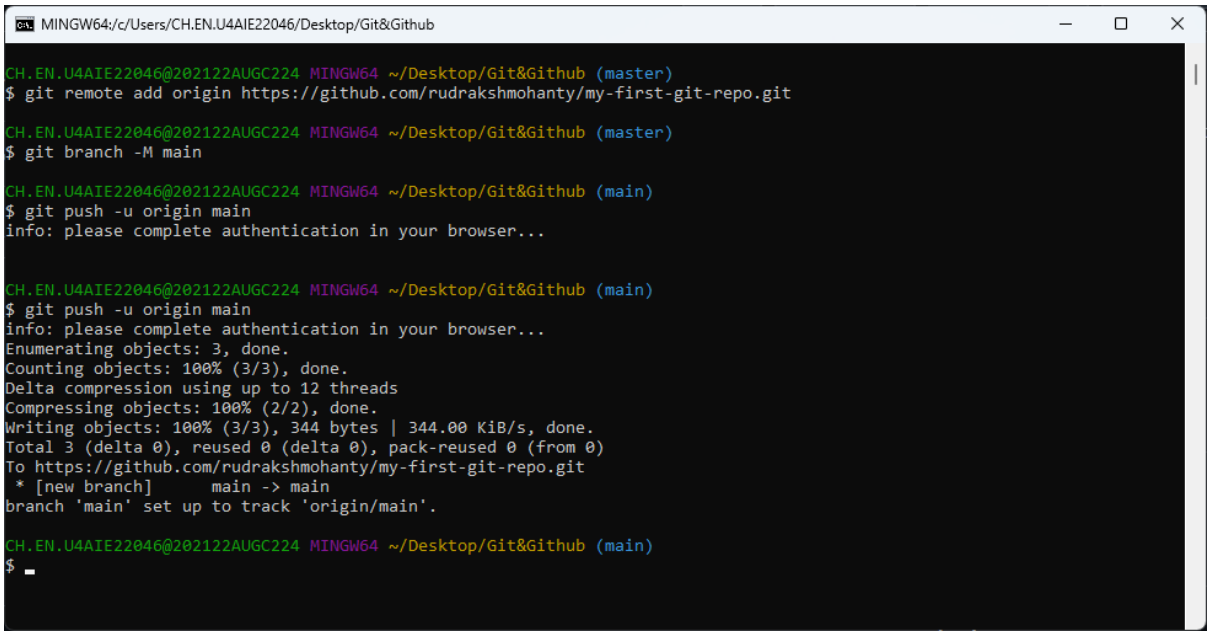


3. Add File and First Commit



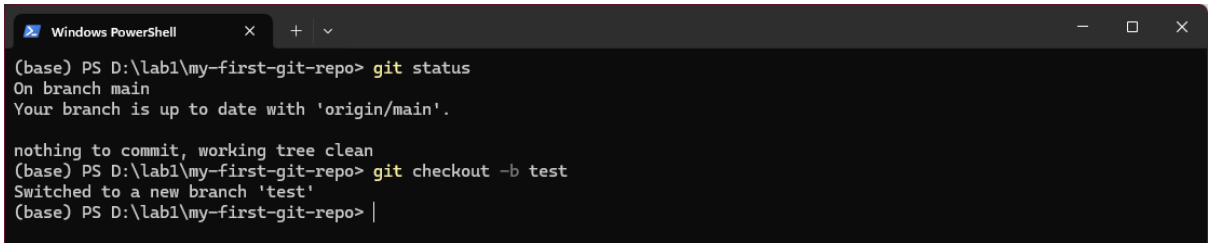
➢ Creates a text file and commits it to the repository.

4. Connect to Remote GitHub Repository



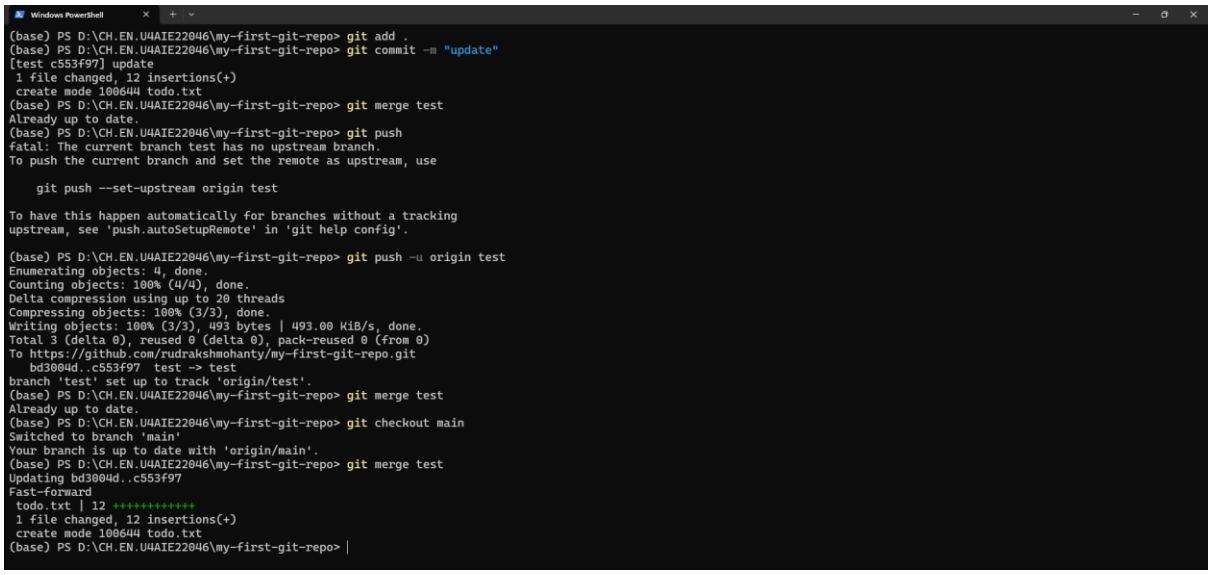
Status:

6. Create and Work on a New Branch



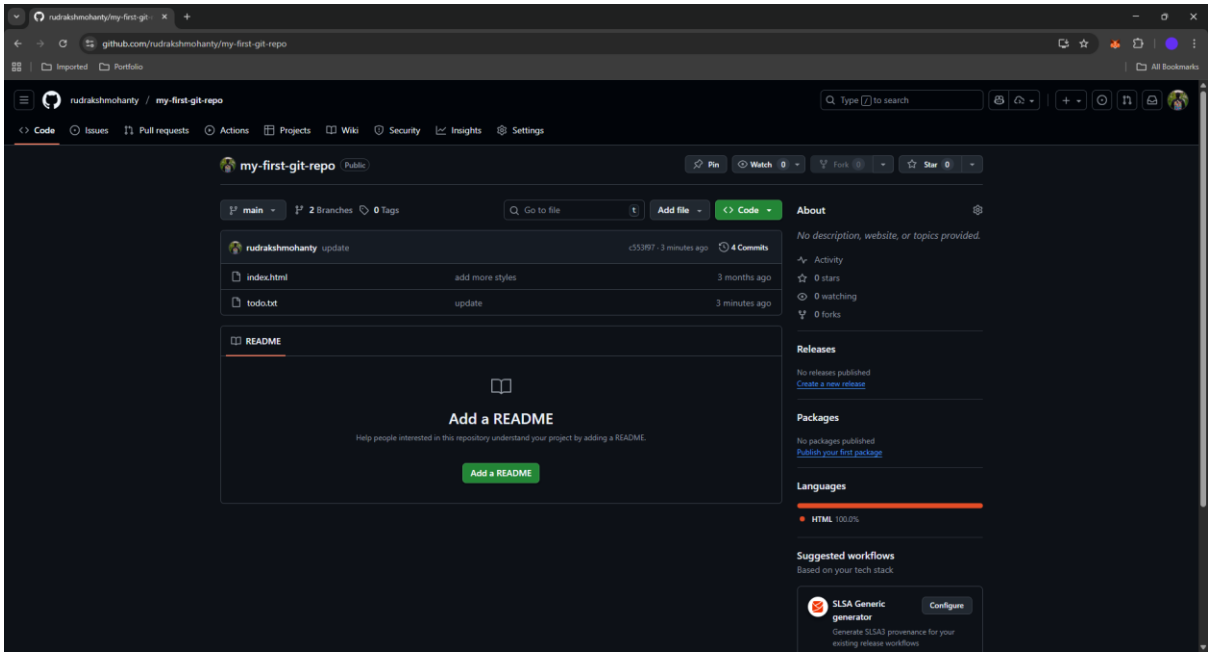
- Creates `test` branch, adds content, and pushes to GitHub.

7. Merge Branch into Main



- Merges changes from test branch into main.

Output:



LAB EXPERIMENT - 2

The objective of this lab is to install Node.js on a Windows 11 system, verify the installation, update npm, and run a sample Node.js application.

Software Requirements:

- ✓ Windows 11 operating system
- ✓ Node.js .msi installer
- ✓ Visual Studio Code (VS Code)
- ✓ Command Prompt / PowerShell

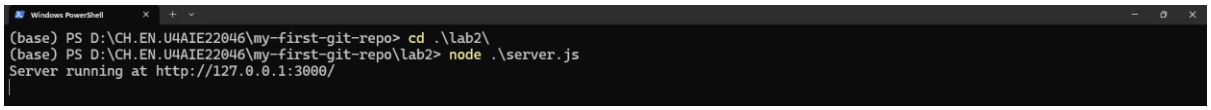
Procedure:

Step 1: Running a Sample Node.js Web Server

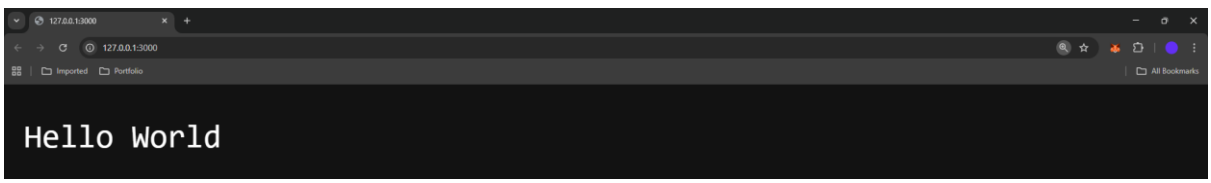
1. Create a file named server.js with the following code:

```
const { createServer } = require('node:http');
const hostname = '127.0.0.1';
const port = 3000;
const server = createServer((req, res) => {
 res.statusCode = 200;
 res.setHeader('Content-Type', 'text/plain');
 res.end('Hello World');
});
server.listen(port, hostname, () => {
console.log(`Server running at http://${hostname}:${port}/`);
});
```

2. Open the terminal in the folder containing server.js.
3. Run the command:



4. Open a web browser and go to http://127.0.0.1:3000



Step 2: Running Node.js Script in VS Code

1. Create a folder for the project.

2. Inside the folder, create a file named First.js with the following content:

```
const x = "My first web page";
console.log(x);
```

3. Open VS Code terminal and run:

Observations:

- ✓ Node.js successfully executed a basic HTTP server.
- ✓ Browser displayed the correct server response.
- ✓ VS Code terminal successfully ran a JavaScript file using Node.js.

Conclusion:

Node.js can be used to run server-side applications and execute JavaScript code on Windows 11. The lab confirmed that Node.js applications work as expected, with HTTP server responses and script execution verified.

LAB EXPERIMENT - 3

Objective:

1. To understand the basic structure of an HTML document and use common HTML tags.
2. To design static web pages for an online bookstore (Home, Login, and Catalogue pages).
3. To apply CSS to enhance webpage presentation, including fonts, backgrounds, links, layers, and custom cursors.

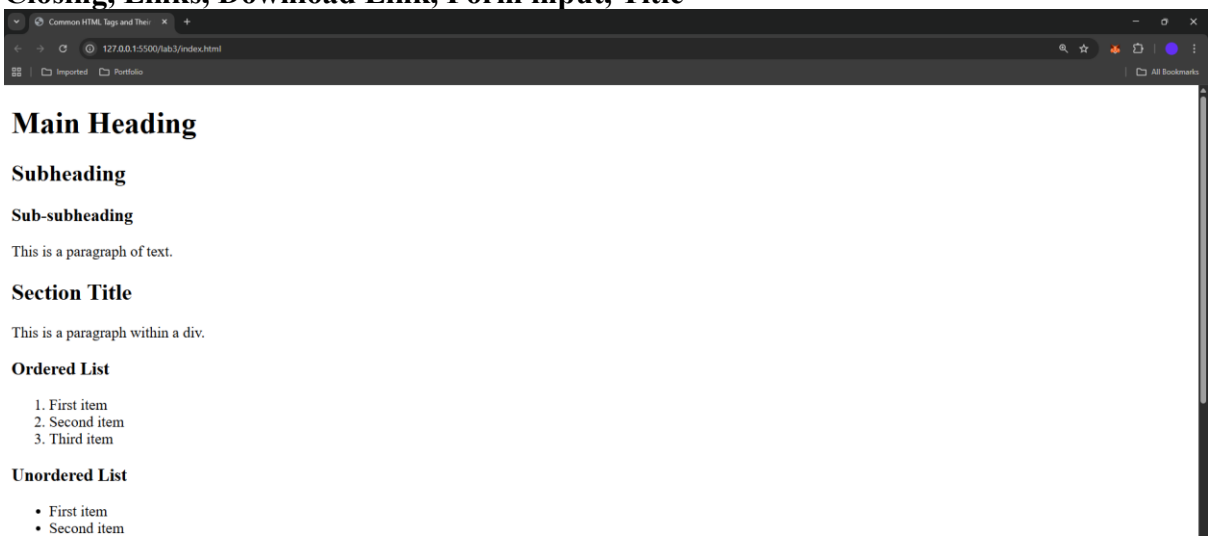
3(a) Basic Structure of an HTML Document

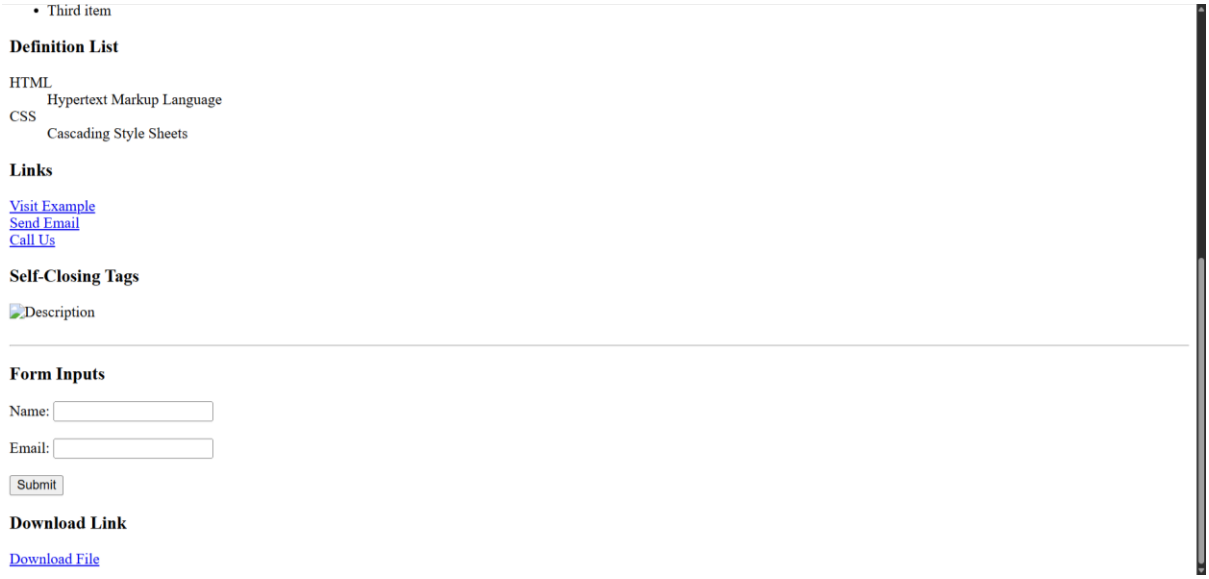
```
<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="UTF-8">
 <title>Common HTML Tags and Their Usage</title>
</head>
<body>

 <!-- Headers -->
 <h1>Main Heading</h1>
 <h2>Subheading</h2>
 <h3>Sub-subheading</h3>
 <!-- Paragraphs -->
 <p>This is a paragraph of text.</p>
 <!-- Comments -->
 <!-- This is a comment -->
 <!-- Divisions -->
 <div>
  <h2>Section Title</h2>
  <p>This is a paragraph within a div.</p>
 </div>
 <!-- Ordered List -->
 <h3>Ordered List</h3>
 <ol>
  <li>First item</li>
  <li>Second item</li>
  <li>Third item</li>
 </ol>
 <!-- Unordered List -->
 <h3>Unordered List</h3>
 <ul>
  <li>First item</li>
  <li>Second item</li>
  <li>Third item</li>
 </ul>
 <!-- Definition List -->
 <h3>Definition List</h3>
 <dl>
```

```
  <dt>HTML</dt>
  <dd>Hypertext Markup Language</dd>
  <dt>CSS</dt>
  <dd>Cascading Style Sheets</dd>
 </dl>
 <!-- Links -->
 <h3>Links</h3>
 <a href="https://www.example.com">Visit Example</a><br>
 <a href="mailto:someone@example.com">Send Email</a><br>
 <a href="tel:+1234567890">Call Us</a>
 <!-- Self-Closing Tags -->
 <h3>Self-Closing Tags</h3>
 <img src="image.jpg" alt="Description"><br>
 <br>
 <hr>
 <!-- Input Tags -->
 <h3>Form Inputs</h3>
 <form>
  <label for="name">Name:</label>
  <input type="text" id="name" name="name"><br><br>
  <label for="email">Email:</label>
  <input type="email" id="email" name="email"><br><br>
  <input type="submit" value="Submit">
 </form>
 <!-- Download Link -->
 <h3>Download Link</h3>
 <a href="file.zip" download>Download File</a>
</body>
</html>
```

Common HTML Tags:

Headings, Paragraphs, Division, Ordered and unordered lists, Definition lists, Self Closing, Links, Download Link, Form input, Title





3(b) Static Web Pages for Online Bookstore

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Online Book Store - Enhanced UI</title>
<link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
<link href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;600;700&display=swap"
rel="stylesheet">
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.5.1/css/all.min.css">
<style>
  /* --- CSS Variables for Easy Theme Management --- */
  :root {
    --primary-color: #00796b; /* Teal */
    --secondary-color: #004d40; /* Darker Teal */
    --accent-color: #ffc107; /* Amber */
    --background-color: #f4f6f8;
    --text-color: #333;
    --light-gray: #e0e0e0;
    --white: #ffffff;
    --font-family: 'Poppins', sans-serif;
  }

  /* --- General Body & Reset Styles --- */
  * {
    box-sizing: border-box;
```

```
  margin: 0;
  padding: 0;
}

body {
  font-family: var(--font-family);
  background-color: var(--background-color);
  color: var(--text-color);
  display: flex;
  flex-direction: column;
  min-height: 100vh;
}

/* --- Header --- */
header {
  background: linear-gradient(90deg, var(--secondary-color), var(--primary-color));
  color: var(--white);
  padding: 20px 0;
  text-align: center;
  font-size: 1.8em;
  font-weight: 700;
  letter-spacing: 2px;
  box-shadow: 0 4px 10px rgba(0, 77, 64, 0.3);
}

/* --- Navigation Bar --- */
nav {
  background: var(--white);
  padding: 15px 0;
  text-align: center;
  box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
  position: sticky;
  top: 0;
  z-index: 1000;
}

nav a {
  color: var(--primary-color);
  margin: 0 18px;
  text-decoration: none;
  font-weight: 600;
  font-size: 1em;
  transition: color 0.3s ease, transform 0.3s ease;
  padding: 5px 10px;
  border-radius: 5px;
}

nav a:hover {
```

```
  color: var(--secondary-color);
  background-color: #e0f2f1; /* Light teal background on hover */
  transform: translateY(-2px);
}

nav a .fa-solid {
  margin-right: 8px;
}

/* --- Main Content Area --- */
main {
  display: flex;
  flex-grow: 1;
  margin: 20px;
  gap: 20px;
}

/* --- Sidebar (Aside) --- */
aside {
  width: 240px;
  background: var(--white);
  padding: 25px;
  border-radius: 10px;
  box-shadow: 0 4px 15px rgba(0, 0, 0, 0.08);
  flex-shrink: 0; /* Prevents sidebar from shrinking */
}

aside h3 {
  color: var(--secondary-color);
  margin-bottom: 25px;
  font-weight: 700;
  text-align: center;
  border-bottom: 2px solid var(--primary-color);
  padding-bottom: 10px;
}

aside a {
  display: block;
  margin-bottom: 15px;
  color: var(--primary-color);
  font-weight: 600;
  text-decoration: none;
  padding: 12px 15px;
  border-radius: 8px;
  transition: background 0.3s ease, color 0.3s ease, transform 0.2s ease;
}

aside a:hover {
```

```
  background: var(--primary-color);
  color: var(--white);
  transform: translateX(5px);
}

/* --- Main Section --- */
section {
  flex-grow: 1;
  padding: 30px;
  overflow-y: auto;
  background: var(--white);
  border-radius: 10px;
  box-shadow: 0 4px 15px rgba(0, 0, 0, 0.08);
}

section h2 {
  color: var(--secondary-color);
  margin-bottom: 20px;
  font-weight: 700;
}

/* --- Book Grid & Card Styling --- */
.book-grid {
  display: grid;
  grid-template-columns: repeat(auto-fill, minmax(220px, 1fr));
  gap: 30px;
}

.book-card {
  background: #f9f9f9;
  border-radius: 10px;
  overflow: hidden;
  box-shadow: 0 4px 12px rgba(0,0,0,0.1);
  display: flex;
  flex-direction: column;
  transition: transform 0.3s ease, box-shadow 0.3s ease;
  text-align: center;
}

.book-card:hover {
  transform: translateY(-8px);
  box-shadow: 0 8px 20px rgba(0, 77, 64, 0.2);
}

.book-card img {
  width: 100%;
  height: 250px;
  object-fit: cover;
```

```
}

.book-card-content {
  padding: 15px;
  flex-grow: 1;
  display: flex;
  flex-direction: column;
  justify-content: space-between;
}

.book-card h4 {
  font-size: 1.1em;
  font-weight: 600;
  margin-bottom: 5px;
  color: var(--secondary-color);
}

.book-card .author {
  font-size: 0.9em;
  color: #777;
  margin-bottom: 15px;
}

.book-card .price {
  font-size: 1.2em;
  font-weight: 700;
  color: var(--primary-color);
  margin-bottom: 15px;
}

/* --- Form Styling --- */
.form-container {
  max-width: 400px;
  margin: 40px auto;
  padding: 30px;
  border: 1px solid var(--light-gray);
  border-radius: 12px;
  background: #fafdfd;
  box-shadow: 0 5px 15px rgba(0, 0, 0, 0.07);
}

.form-container label {
  display: block;
  margin-bottom: 8px;
  font-weight: 600;
  color: var(--secondary-color);
}
```

```
input[type=text], input[type=password], input[type=email] {
  width: 100%;
  padding: 12px;
  margin-bottom: 20px;
  border: 1px solid var(--light-gray);
  border-radius: 8px;
  font-size: 1em;
  font-family: var(--font-family);
  transition: border-color 0.3s ease, box-shadow 0.3s ease;
}

input[type=text]:focus, input[type=password]:focus, input[type=email]:focus {
  border-color: var(--primary-color);
  outline: none;
  box-shadow: 0 0 8px rgba(0, 121, 107, 0.2);
}

/* --- Universal Button Style --- */
button {
  width: 100%;
  padding: 12px 15px;
  background: var(--primary-color);
  border: none;
  color: var(--white);
  font-weight: 600;
  font-size: 1em;
  border-radius: 8px;
  cursor: pointer;
  transition: background 0.3s ease, transform 0.2s ease;
}

button:hover {
  background: var(--secondary-color);
  transform: scale(1.02);
}

/* --- Footer --- */
footer {
  background: var(--secondary-color);
  color: var(--white);
  text-align: center;
  padding: 20px 0;
  font-weight: 400;
  font-size: 0.9em;
  letter-spacing: 1px;
  margin-top: auto; /* Pushes footer to the bottom */
}
</style>
```

```
</head>
<body>

<header>
  Online Book Store 
</header>
<nav>
  <a href="#" onclick="loadPage('home')"><i class="fa-solid fa-house"></i>Home</a>
  <a href="#" onclick="loadPage('login')"><i class="fa-solid fa-right-to-bracket"></i>Login</a>
  <a href="#" onclick="loadPage('register')"><i class="fa-solid fa-user-plus"></i>Registration</a>
  <a href="#" onclick="loadPage('catalogue')"><i class="fa-solid fa-book-open"></i>Catalogue</a>
  <a href="#" onclick="loadPage('cart')"><i class="fa-solid fa-cart-shopping"></i>Cart</a>
</nav>
<main>
  <aside>
    <h3>Departments</h3>
    <a href="#" onclick="loadPage('cse')">Computer Science</a>
    <a href="#" onclick="loadPage('ece')">Electronics & Comm.</a>
    <a href="#" onclick="loadPage('eee')">Electrical</a>
    <a href="#" onclick="loadPage('civil')">Civil Engineering</a>
  </aside>
  <section id="content">
    </section>
</main>

<footer>
  © 2025 Online Book Store | Email: support@bookstore.com
</footer>
<script>
  // Helper function to create a book card
  function createBookCard(imgSrc, title, author, price) {
    return `
      <div class="book-card">
        <img src="${imgSrc}" alt="${title}">
        <div class="book-card-content">
          <div>
            <h4>${title}</h4>
            <p class="author">by ${author}</p>
          </div>
          <div>
            <p class="price">₹${price}</p>
            <button>Add to Cart</button>
          </div>
        </div>
      </div>
    `;
  }
```

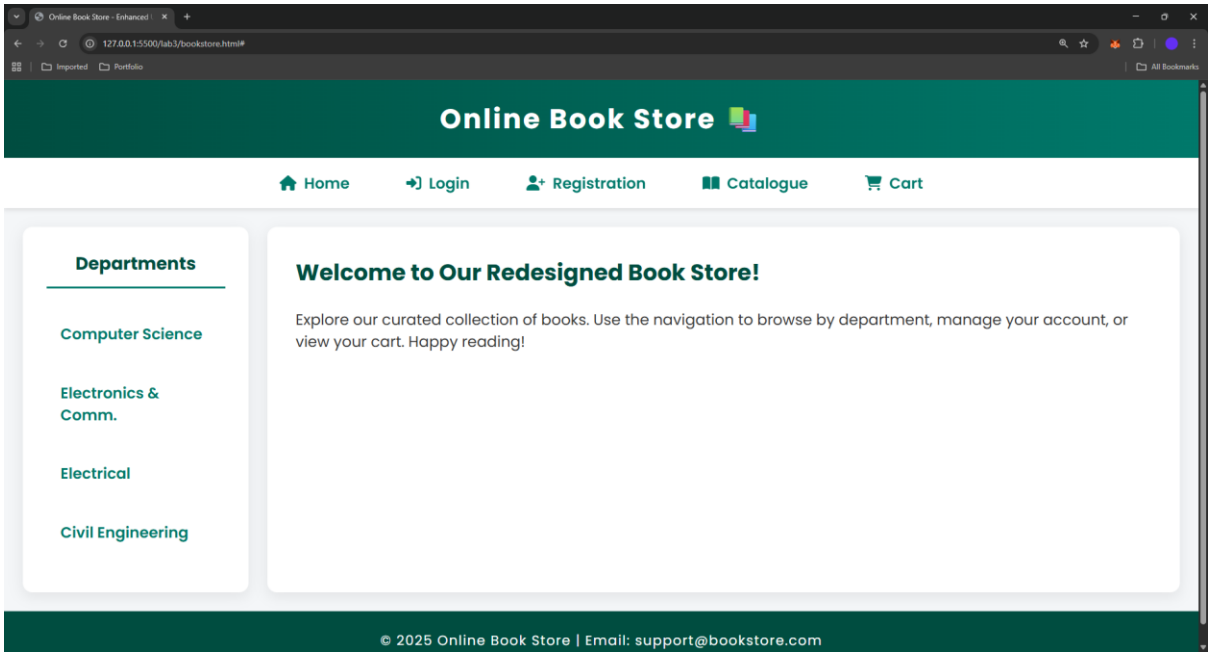
```
const pages = {
  home: `
    <h2>Welcome to Our Redesigned Book Store!</h2>
    <p>Explore our curated collection of books. Use the navigation to browse by department, manage your account, or view your cart. Happy reading!</p>
  `,
  login: `
    <div class="form-container">
      <h2 style="text-align:center;">User Login</h2>
      <form>
        <label for="username">Username:</label>
        <input type="text" id="username" name="username" required>
        <label for="password">Password:</label>
        <input type="password" id="password" name="password" required>
        <button type="submit">Login</button>
      </form>
    </div>
  `,
  register: `
    <div class="form-container">
      <h2 style="text-align:center;">Create an Account</h2>
      <form>
        <label for="name">Name:</label>
        <input type="text" id="name" name="name" required>
        <label for="email">Email:</label>
        <input type="email" id="email" name="email" required>
        <label for="reg-password">Password:</label>
        <input type="password" id="reg-password" name="password" required>
        <button type="submit">Register</button>
      </form>
    </div>
  `,
  cart: `
    <h2>Your Shopping Cart</h2>
    <p>Your cart is currently empty. Start browsing the catalogue to add some books!</p>
  `,
  catalogue: `
    <h2>Full Book Catalogue</h2>
    <div class="book-grid">
      ${createBookCard('https://placehold.co/400x600/00796b/white?text=Programming\\nin C%23', 'Programming in C#', 'Harsh Bhasin', '200')}
      ${createBookCard('https://placehold.co/400x600/004d40/white?text=Computer\\nNetworks', 'Computer Networks', 'A. S. Tanenbaum', '400')}
      ${createBookCard('https://placehold.co/400x600/ffc107/black?text=Web\\nProgramming', 'Web Programming', 'Chris Bates', '599')}
      ${createBookCard('https://placehold.co/400x600/333/white?text=Data\\nStructures', 'Data Structures & Algos', 'Ravi Raj', '450')}
```

```
      ${createBookCard('https://placehold.co/400x600/00796b/white?text=Operating\\nSystems', 'Operating Systems', 'Philip C. Johnson', '550')}
      ${createBookCard('https://placehold.co/400x600/004d40/white?text=Fantasy\\nProgramming', 'Fantasy Programming', 'J.K. Rowling', '650')}
    </div>
  `,
  cse: `
    <h2>Computer Science Books</h2>
    <div class="book-grid">
      ${createBookCard('https://placehold.co/400x600/00796b/white?text=Programming\\nin C%23', 'Programming in C#', 'Harsh Bhasin', '200')}
      ${createBookCard('https://placehold.co/400x600/004d40/white?text=Computer\\nNetworks', 'Computer Networks', 'A. S. Tanenbaum', '400')}
      ${createBookCard('https://placehold.co/400x600/333/white?text=Data\\nStructures', 'Data Structures & Algos', 'Ravi Raj', '450')}
      ${createBookCard('https://placehold.co/400x600/00796b/white?text=Operating\\nSystems', 'Operating Systems', 'Philip C. Johnson', '550')}
    </div>
  `,
  ece: `
    <h2>Electronics & Comm. Books</h2>
    <div class="book-grid">
      ${createBookCard('https://placehold.co/400x600/00796b/white?text=Signals\\nand Systems', 'Signals and Systems', 'Simon Haykin', '500')}
      ${createBookCard('https://placehold.co/400x600/004d40/white?text=Digital\\nSignal\\nProcessing', 'Digital Signal Processing', 'John G. Proakis', '650')}
    </div>
  `,
  eee: `
    <h2>Electrical Books</h2>
    <div class="book-grid">
      ${createBookCard('https://placehold.co/400x600/00796b/white?text=Electronic\\nPrinciples', 'Electronic Principles', 'Albert Malvino', '480')}
      ${createBookCard('https://placehold.co/400x600/004d40/white?text=Basic\\nElectronics', 'Basic Electronics', 'Malvino', '520')}
    </div>
  `,
  civil: `
    <h2>Civil Engineering Books</h2>
    <div class="book-grid">
      ${createBookCard('https://placehold.co/400x600/00796b/white?text=Structural\\nAnalysis', 'Structural Analysis', 'R.C. Hibbeler', '550')}
      ${createBookCard('https://placehold.co/400x600/004d40/white?text=Building\\nConstruction', 'Building Construction', 'James Ambrose', '600')}
    </div>
  `
};
```

```
function loadPage(page) {
  const content = document.getElementById('content');
  if (pages[page]) {
    content.innerHTML = pages[page];
  } else {
    content.innerHTML = `<h2>Page Not Found</h2><p>The page you requested does not exist.</p>`;
  }
}

// Load the home page by default when the script runs
document.addEventListener('DOMContentLoaded', () => {
    loadPage('home');
});
</script>
</body>
</html>
```

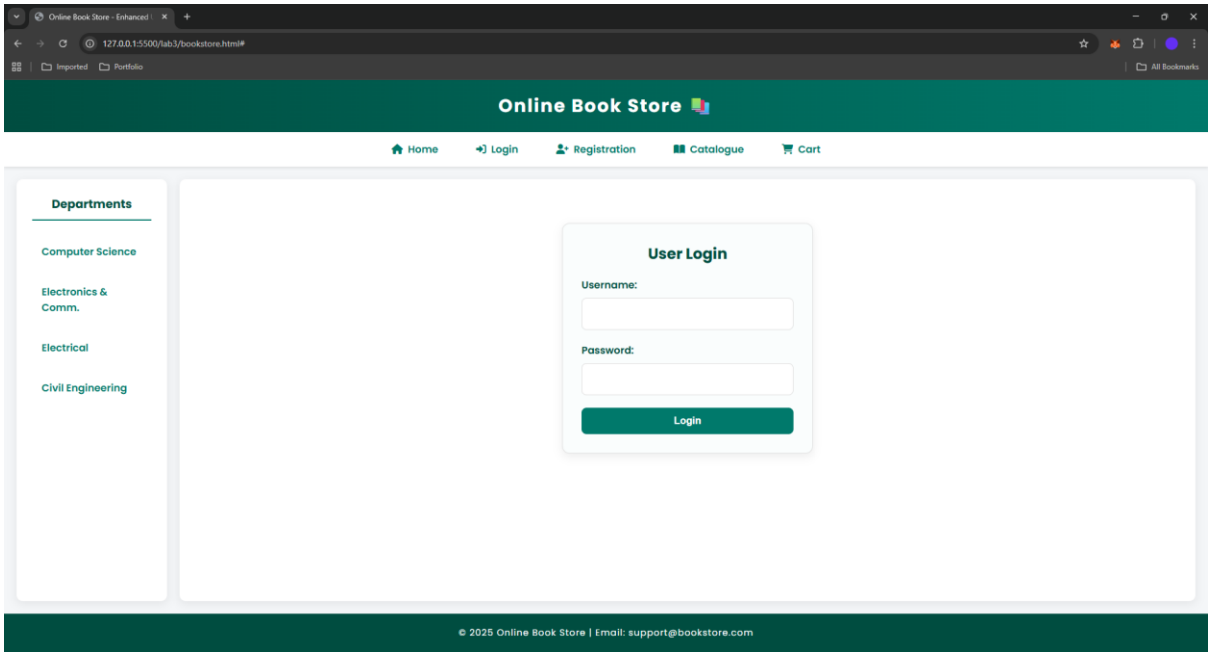
1. Home Page

- Top Frame: Logo, college name, navigation links (Home, Login, Registration, Catalogue, Cart)
- Left Frame: Navigation links for book categories (e.g., CSE, ECE, ME, Civil)
- Right Frame: Description of the website or catalogue page (loads content based on left-frame selection)

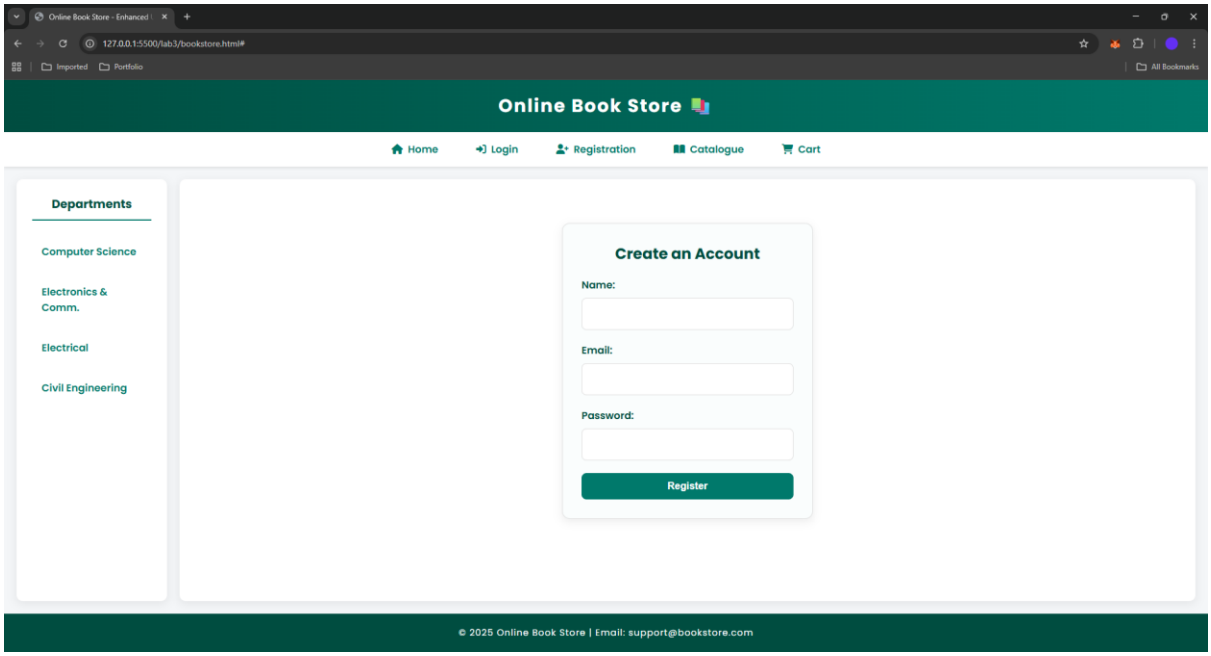


2. Login Page

- Simple login form with fields for username/email and password, plus a submit button.



Registration Page:

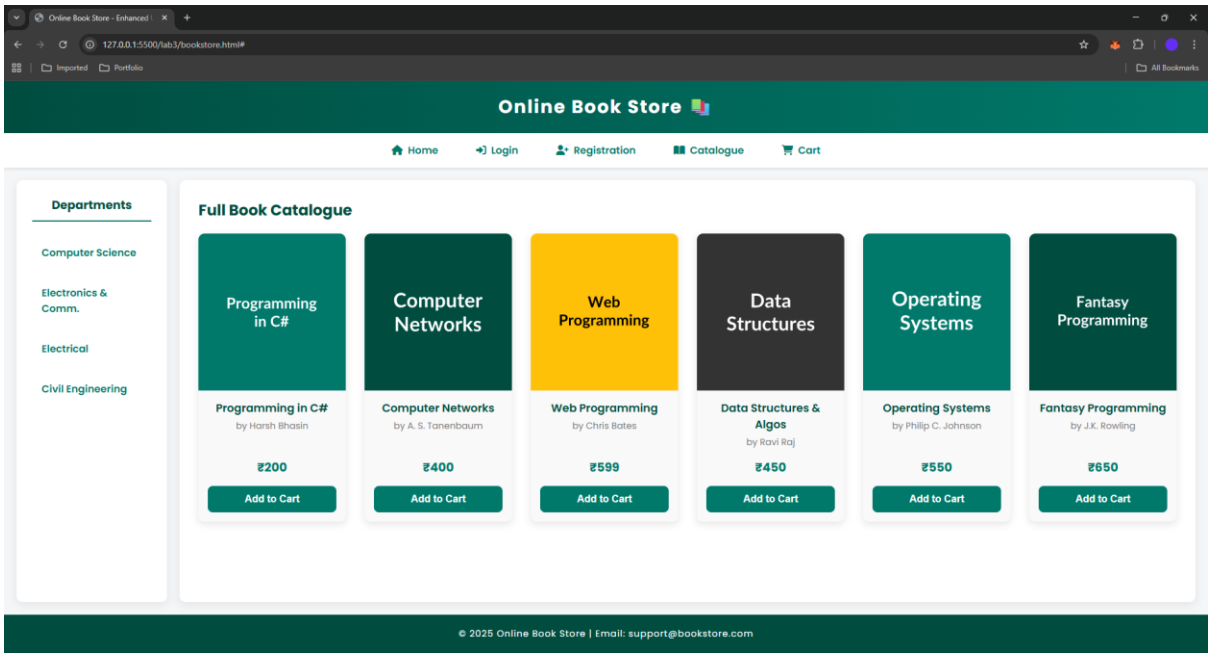


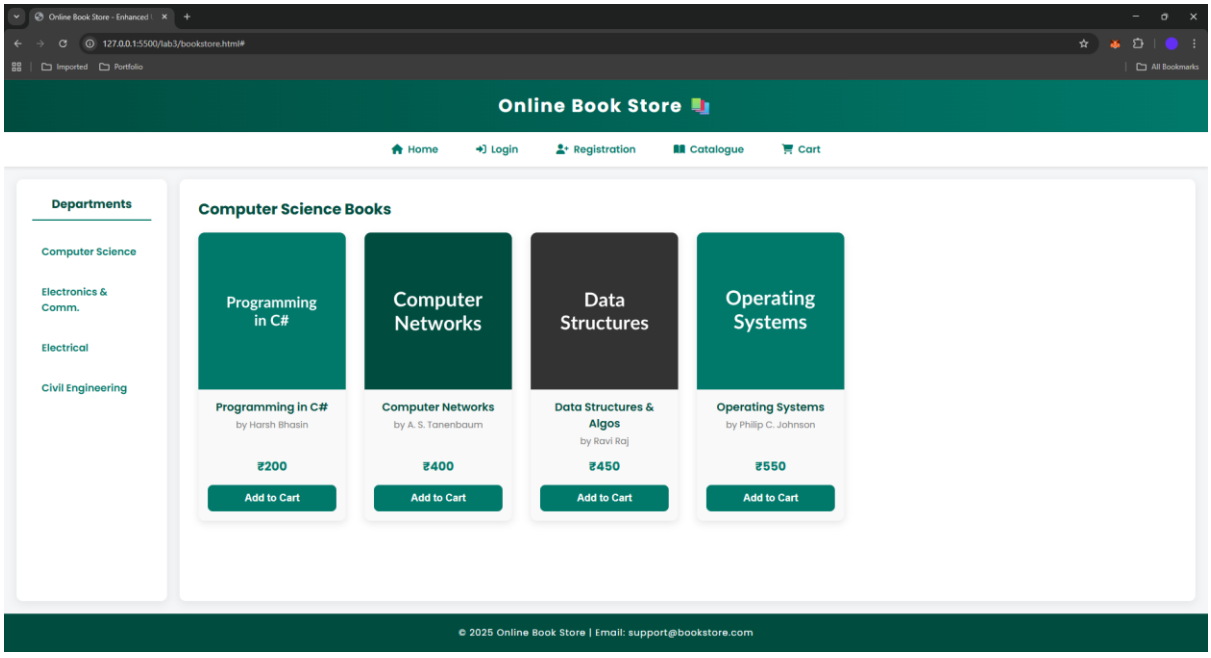
3. Catalogue Page

Table displaying all books with:

- ✓ Cover image
- ✓ Author name

- ✓ Publisher
- ✓ Price
- ✓ Add to Cart button





3(c) CSS Web Page Design

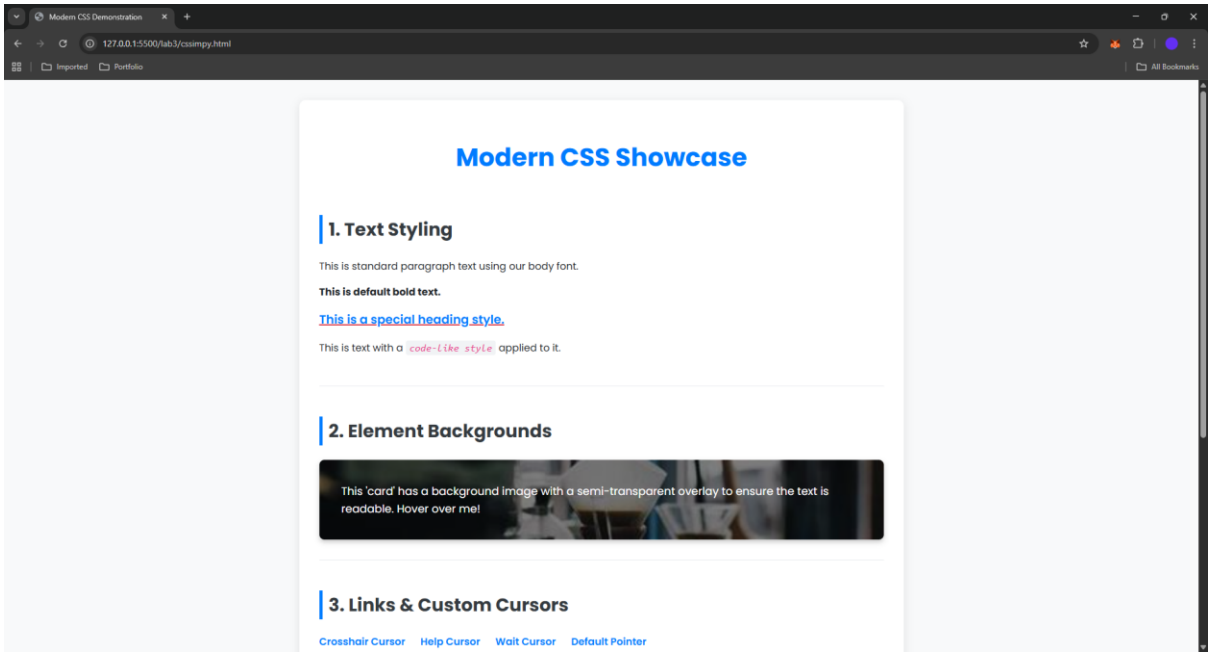
```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Modern CSS Demonstration</title>
   <link rel="preconnect" href="https://fonts.googleapis.com">
   <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
   <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;600;700&display=swap"
rel="stylesheet">
   <style>
      /* * Modern CSS Best Practices:
       * - CSS Variables for easy theme management.
       * - A clean, readable font from Google Fonts.
       * - A structured and spaced-out layout.
       */
      :root {
         --primary-color: #007BFF;
         --secondary-color: #343a40;
         --background-color: #f8f9fa;
         --text-color: #212529;
         --card-bg: #ffffff;
         --accent-color: #dc3545;
         --font-family: 'Poppins', sans-serif;
      }

      body {
         font-family: var(--font-family);
         background-color: var(--background-color);
         color: var(--text-color);
         margin: 0;
         padding: 2em;
         line-height: 1.6;
      }

      .container {
         max-width: 900px;
         margin: 0 auto;
         background: var(--card-bg);
         padding: 2em;
         border-radius: 10px;
         box-shadow: 0 4px 15px rgba(0, 0, 0, 0.1);
      }
```

```
section {
    margin-bottom: 3em;
    border-bottom: 1px solid #e9ecef;
    padding-bottom: 2em;
}

section:last-child {
    border-bottom: none;
    margin-bottom: 0;
}

h1, h2 {
    font-family: 'Poppins', sans-serif;
    font-weight: 700;
    color: var(--secondary-color);
}

h1 {
    text-align: center;
    color: var(--primary-color);
    margin-bottom: 1.5em;
    font-size: 2.5em;
}

h2 {
    font-size: 1.8em;
    margin-bottom: 0.8em;
    border-left: 5px solid var(--primary-color);
    padding-left: 10px;
}

/* 1. Different Fonts and Text Styles */
.special-heading {
    color: var(--primary-color);
    font-size: 1.2em;
    font-weight: 600;
    text-decoration: underline;
    text-decoration-color: var(--accent-color);
    text-decoration-thickness: 2px;
}

.code-text {
    font-family: Consolas, "Courier New", monospace;
    font-style: italic;
    color: #e83e8c; /* A softer pink for code */
    background-color: #f1f3f5;
    padding: 2px 6px;
    border-radius: 4px;
```

```
    }

    /* 2. Background Image for a Single Element */
    .card-with-bg {
        background-image: linear-gradient(rgba(0,0,0,0.5), rgba(0,0,0,0.5)),
url("https://picsum.photos/id/1060/400/200");
        background-repeat: no-repeat;
        background-size: cover;
        background-position: center;
        padding: 2em;
        color: white;
        font-size: 1.1em;
        border-radius: 8px;
        box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);
        transition: transform 0.3s ease, box-shadow 0.3s ease;
    }

    .card-with-bg:hover {
        transform: translateY(-5px);
        box-shadow: 0 8px 16px rgba(0, 0, 0, 0.2);
    }

    /* 3. Styles for Links & Customized Cursors */
    .link-showcase a {
        font-weight: 600;
        text-decoration: none;
        padding: 5px 0;
        margin-right: 20px;
        position: relative;
        transition: color 0.3s ease;
    }

    .link-showcase a::after {
        content: '';
        position: absolute;
        width: 100%;
        height: 2px;
        bottom: 0;
        left: 0;
        background-color: var(--accent-color);
        transform: scaleX(0);
        transform-origin: bottom right;
        transition: transform 0.3s ease-out;
    }

    .link-showcase a:hover::after {
        transform: scaleX(1);
        transform-origin: bottom left;
```

```
}

.link-showcase a:link    { color: var(--primary-color); }
.link-showcase a:visited { color: #6f42c1; /* Purple */ }
.link-showcase a:active  { color: var(--accent-color); }
.link-showcase a:hover   { color: var(--secondary-color); }

.cursor-crosshair { cursor: crosshair; }
.cursor-help      { cursor: help; }
.cursor-pointer   { cursor: pointer; }
.cursor-wait      { cursor: wait; }

/* 4. Working with Layers (z-index) */
.layer-container {
    position: relative; /* Establishes a stacking context */
    height: 200px;
    background: #e9ecef;
    border-radius: 8px;
    padding: 1em;
}

.layer {
    position: absolute;
    width: 150px;
    height: 150px;
    border-radius: 8px;
    display: flex;
    align-items: center;
    justify-content: center;
    font-size: 1.2em;
    font-weight: 600;
    color: white;
}

.layer-back {
    background: var(--secondary-color);
    top: 20px;
    left: 80px;
    z-index: 1; /* Lower z-index, appears behind */
}

.layer-front {
    background: var(--accent-color);
    top: 40px;
    left: 30px;
    z-index: 2; /* Higher z-index, appears in front */
}
```

```
  </style>
</head>
<body>
  <div class="container">
    <h1>Modern CSS Showcase</h1>

    <section id="text-styles">
      <h2>1. Text Styling</h2>
      <p>This is standard paragraph text using our body font.</p>
      <p><b>This is default bold text.</b></p>
      <p><b class="special-heading">This is a special heading style.</b></p>
      <p>This is text with a <span class="code-text">code-like style</span> applied to it.</p>
    </section>

    <section id="backgrounds">
      <h2>2. Element Backgrounds</h2>
      <div class="card-with-bg">
        This 'card' has a background image with a semi-transparent overlay to ensure the text is readable. Hover over me!
      </div>
    </section>

    <section id="links-and-cursors">
      <h2>3. Links & Custom Cursors</h2>
      <p class="link-showcase">
        <a href="#" class="cursor-crosshair">Crosshair Cursor</a>
        <a href="#" class="cursor-help">Help Cursor</a>
        <a href="#" class="cursor-wait">Wait Cursor</a>
        <a href="#" class="cursor-pointer">Default Pointer</a>
      </p>
    </section>

    <section id="layers">
      <h2>4. Stacking with Z-Index</h2>
      <p>The red box has a higher <code class="code-text">z-index</code> (2) than the dark box (1), so it appears on top.</p>
      <div class="layer-container">
        <div class="layer layer-front">Front (z-index: 2)</div>
        <div class="layer layer-back">Back (z-index: 1)</div>
      </div>
    </section>

  </div>
</body>
</html>
```



Observations:

- ✓ HTML pages were successfully created with proper headings, paragraphs, links, lists, forms, and tables.
- ✓ Frames, tables, and forms were functioning as expected.
- ✓ CSS successfully applied styles, background images, layers, and custom cursors.
- ✓ Links responded correctly to hover and active states.

Conclusion

The lab demonstrated practical usage of HTML and CSS in designing web pages. We were able to:

- ✓ Create structured HTML documents with standard tags.
- ✓ Design a small online bookstore with Home, Login, and Catalogue pages.
- ✓ Apply CSS for visual enhancement including fonts, backgrounds, layers, and custom cursors.

LAB EXPERIMENT - 4

Objective:

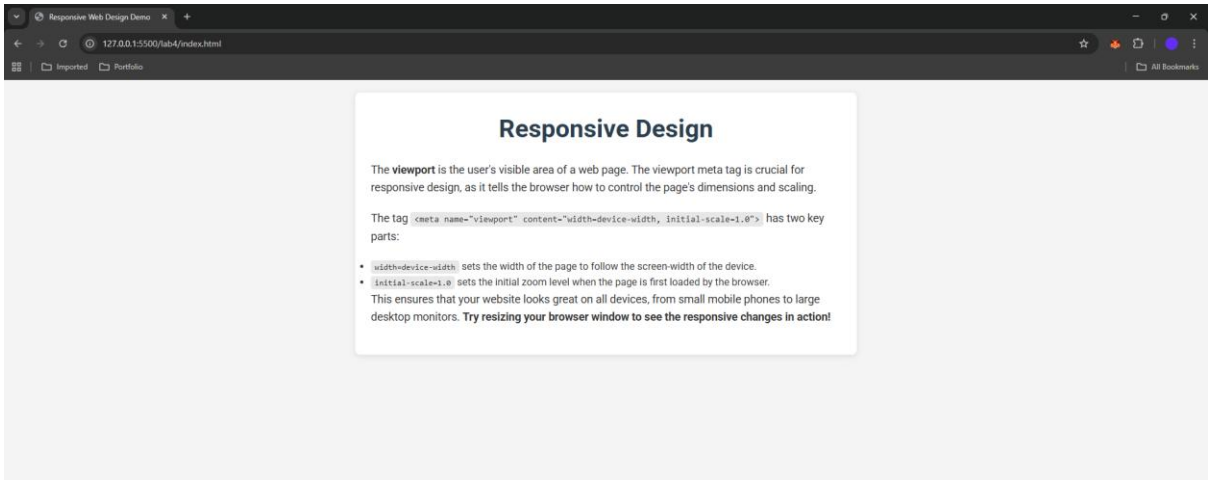
To design responsive web pages that adapt to different devices (desktop, tablet, and mobile) using HTML viewport, responsive images, responsive texts, CSS media queries, and layouts with Flexbox and Multicolumn.

1. HTML Viewport Meta Tag for Responsive Web Design

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

    <link rel="preconnect" href="https://fonts.googleapis.com">
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
    <link href="https://fonts.googleapis.com/css2?family=Roboto:wght@400;700&display=swap"
rel="stylesheet">

    <style>
        /* Basic reset and body styling */
        * {
            box-sizing: border-box;
            margin: 0;
            padding: 0;
        }

        body {
            font-family: 'Roboto', sans-serif;
            line-height: 1.6;
            background-color: #f4f4f4;
            color: #333;
        }

        /* Container to center content and set a max-width */
        .container {
            max-width: 800px;
            margin: 20px auto;
            padding: 25px;
            background-color: #ffffff;
            border-radius: 8px;
            box-shadow: 0 2px 10px rgba(0,0,0,0.1);
        }

        h1 {
```

```
            color: #2c3e50; /* Dark Blue */
            text-align: center;
            font-size: 2.5em; /* 40px */
            font-weight: 700;
            margin-bottom: 20px;
        }

        p {
            font-size: 1.1em; /* 18px */
            margin-bottom: 1.2em;
        }

        /* Responsive image styling */
        img {
            max-width: 100%;
            height: auto;
            border-radius: 5px;
            display: block;
            margin: 20px 0;
        }

        code {
            background-color: #e8e8e8;
            padding: 3px 6px;
            border-radius: 4px;
            font-family: monospace;
        }

        /* --- Media Query for Responsiveness --- */
        /* This CSS will only apply when the screen width is 600px or less */
        @media (max-width: 600px) {
            body {
                background-color: #eef1f5; /* Change background on mobile */
            }
            .container {
                margin: 10px;
                padding: 15px;
            }
            h1 {
                font-size: 2em; /* Decrease heading size on smaller screens */
            }
            p {
                font-size: 1em;
            }
        }
    </style>
</head>
<body>
```

```
  <div class="container">
    <h1>Responsive Design</h1>



    <p>
      The <strong>viewport</strong> is the user's visible area of a web page. The viewport meta tag is
crucial for responsive design, as it tells the browser how to control the page's dimensions and scaling.
    </p>
    <p>
      The tag <code>&lt;meta name="viewport" content="width=device-width, initial-
scale=1.0"&gt;</code> has two key parts:
    </p>
    <ul>
      <li><code>width=device-width</code> sets the width of the page to follow the screen-width of the
device.</li>
      <li><code>initial-scale=1.0</code> sets the initial zoom level when the page is first loaded by the
browser.</li>
    </ul>
    <p>
      This ensures that your website looks great on all devices, from small mobile phones to large
desktop monitors. <strong>Try resizing your browser window to see the responsive changes in
action!</strong>
    </p>
  </div>
</body>
</html>
```



- A large green centered heading.

- A smaller centered subheading.
- A justified paragraph adapting to different screen widths.

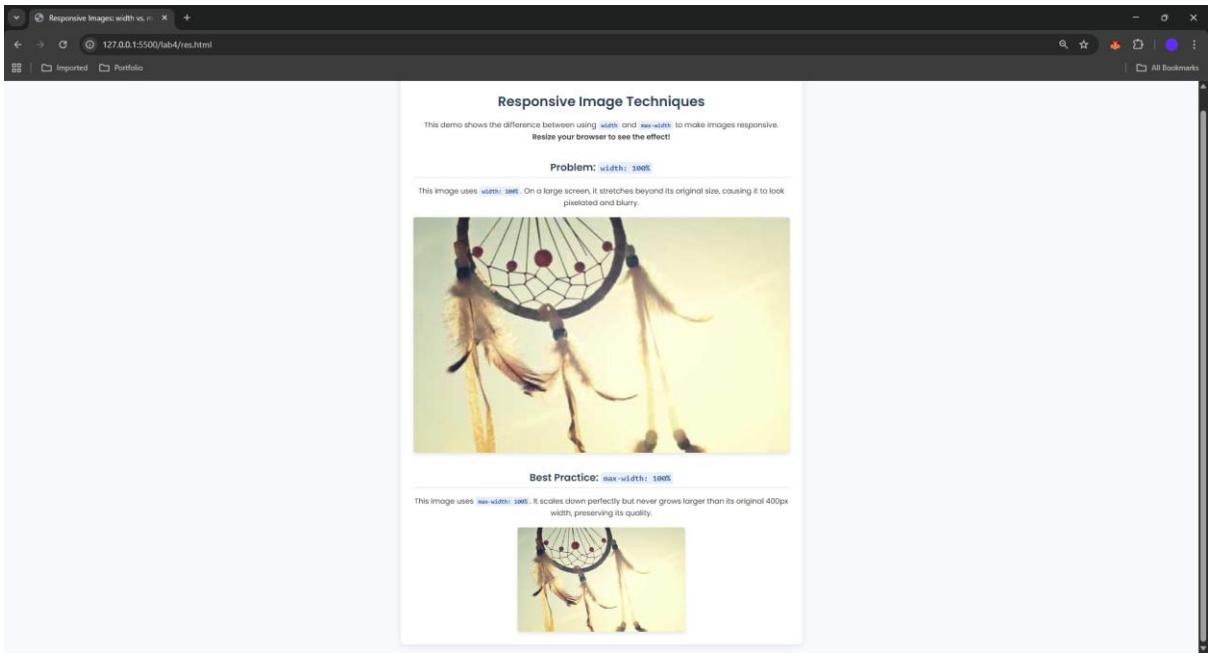
2. Responsive Images

2.1 Using width Property (100%)

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Responsive Images: width vs. max-width</title>

   <link rel="preconnect" href="https://fonts.googleapis.com">
   <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
   <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;600&display=swap"
rel="stylesheet">

   <style>
      body {
         font-family: 'Poppins', sans-serif;
         background-color: #f7f9fc;
         color: #333;
         margin: 0;
         padding: 20px;
      }

      .container {
         max-width: 900px;
         margin: 0 auto;
         background-color: #ffffff;
         padding: 30px;
         border-radius: 10px;
         box-shadow: 0 4px 15px rgba(0,0,0,0.08);
      }

      h1, h2 {
         text-align: center;
         color: #2c3e50;
      }

      h1 {
         margin-bottom: 20px;
      }

      h2 {
         font-size: 1.5em;
```

```
        margin-top: 40px;
        margin-bottom: 10px;
        border-bottom: 2px solid #e0e0e0;
        padding-bottom: 5px;
    }

    p {
        text-align: center;
        font-size: 1.1em;
        line-height: 1.6;
        margin-bottom: 20px;
    }

    /* Common image styles */
    img {
        height: auto; /* Maintain aspect ratio */
        display: block; /* Remove bottom space */
        margin: 0 auto; /* Center the image */
        border-radius: 5px;
        box-shadow: 0 4px 8px rgba(0,0,0,0.1);
    }

    /* * Problematic Method: width: 100%
     * This forces the image to always fill its container,
     * which can stretch a small image and make it blurry.
    */
    .problem-image {
        width: 100%;
    }

    /* * Best Practice: max-width: 100%
     * This allows the image to shrink if the container is smaller,
     * but it will never grow larger than its original size.
     * This prevents pixelation.
    */
    .best-practice-image {
        max-width: 100%;
    }

    code {
        background-color: #e8f0fe;
        color: #34559b;
        padding: 3px 6px;
        border-radius: 4px;
        font-weight: 600;
    }

</style>
```

```
</head>
<body>

  <div class="container">
    <h1>Responsive Image Techniques</h1>
    <p>This demo shows the difference between using <code>width</code> and <code>max-width</code> to make images responsive. <strong>Resize your browser to see the effect!</strong></p>

    <section>
      <h2>Problem: <code>width: 100%</code></h2>
      <p>This image uses <code>width: 100%</code>. On a large screen, it stretches beyond its original size, causing it to look pixelated and blurry.</p>

      <img src="https://picsum.photos/id/104/400/250" alt="Pixelated landscape" class="problem-image">
    </section>

    <section>
      <h2>Best Practice: <code>max-width: 100%</code></h2>
      <p>This image uses <code>max-width: 100%</code>. It scales down perfectly but never grows larger than its original 400px width, preserving its quality.</p>

      <img src="https://picsum.photos/id/104/400/250" alt="Responsive landscape" class="best-practice-image">
    </section>
  </div>

</body>
</html>
```

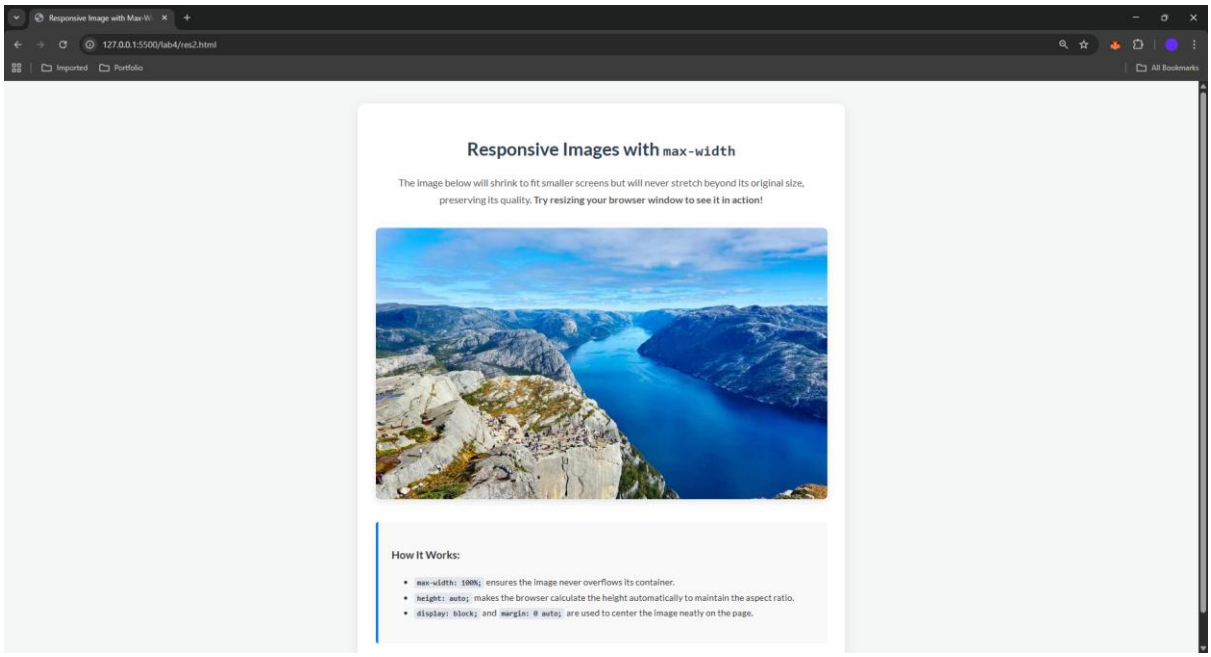


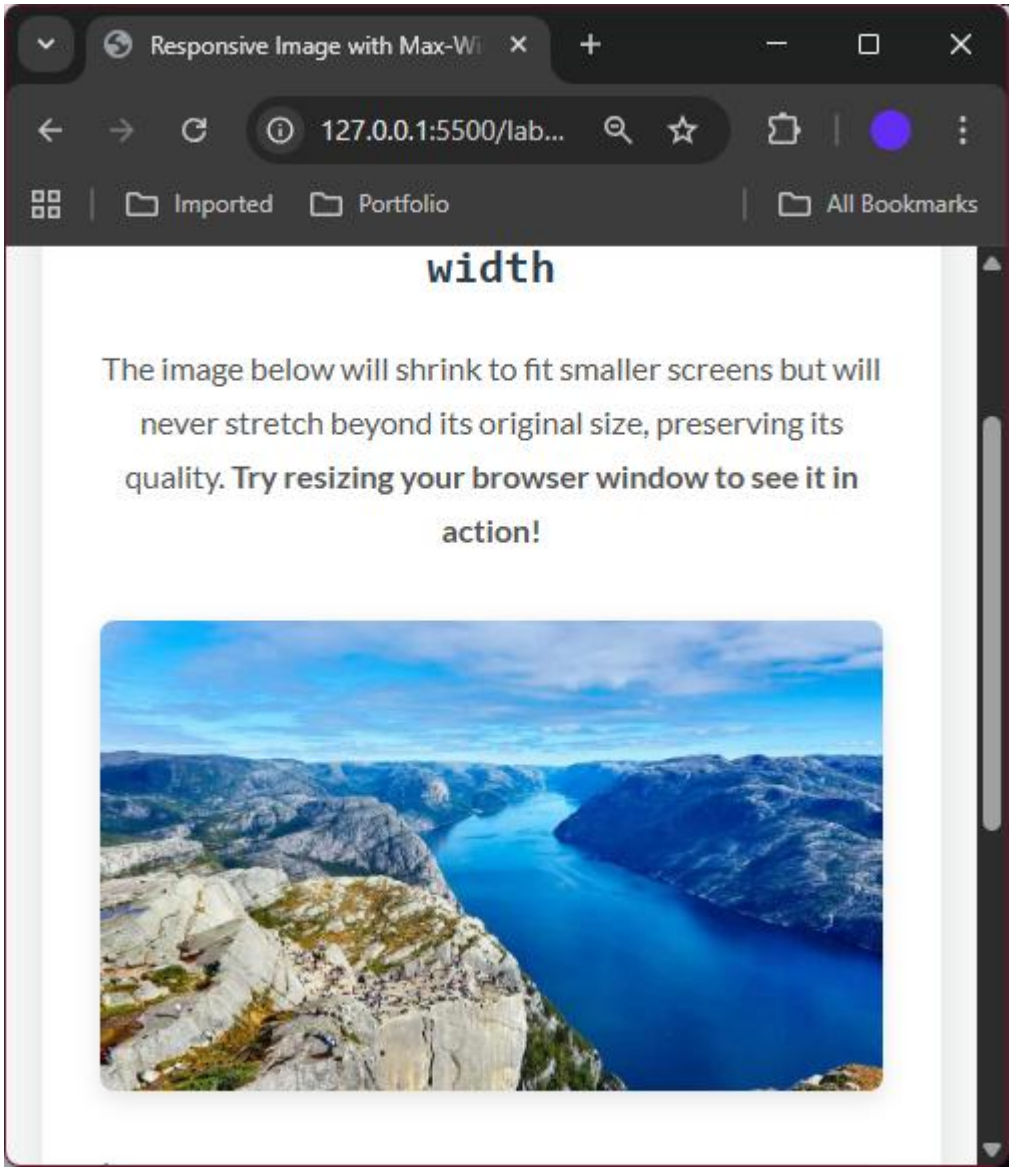
2.2 Using max-width Property

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Responsive Image with Max-Width</title>

    <link rel="preconnect" href="https://fonts.googleapis.com">
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
    <link href="https://fonts.googleapis.com/css2?family=Lato:wght@400;700&display=swap"
rel="stylesheet">

    <style>
        /* General page styling for a modern look */
        body {
            font-family: 'Lato', sans-serif;
            background-color: #f4f7f6;
            color: #333;
            line-height: 1.7;
            margin: 0;
            padding: 20px;
        }

        /* A container to center the content and give it a nice frame */
        .container {
            max-width: 800px;
            margin: 20px auto;
            padding: 2em;
            background-color: #ffffff;
            border-radius: 12px;
            box-shadow: 0 5px 20px rgba(0,0,0,0.08);
        }

        h1 {
            color: #2d3e50;
            text-align: center;
            margin-bottom: 0.5em;
        }

        p {
            font-size: 1.1em;
            text-align: center;
            color: #555;
            margin-bottom: 2em;
```

```
    }

    /* The core styles for the responsive image */
    img {
      /* This is the key property for responsive images.
         The image will scale down but never grow larger than its actual size. */
      max-width: 100%;

      /* This maintains the image's original aspect ratio as it scales. */
      height: auto;

      /* These two properties center the image horizontally. */
      display: block;
      margin: 0 auto;

      /* Adding some nice visual touches */
      border-radius: 8px;
      box-shadow: 0 4px 12px rgba(0,0,0,0.1);
    }

    /* Styling for the explanation section */
    .explanation {
      margin-top: 2.5em;
      padding: 1.5em;
      background-color: #f9f9f9;
      border-left: 5px solid #007bff;
      border-radius: 5px;
    }

    .explanation code {
      background-color: #e2e8f0;
      padding: 2px 5px;
      border-radius: 4px;
      font-family: monospace;
      font-weight: 600;
    }

  </style>
</head>
<body>

  <div class="container">
    <h1>Responsive Images with <code>max-width</code></h1>
    <p>The image below will shrink to fit smaller screens but will never stretch beyond its original size, preserving its quality. <strong>Try resizing your browser window to see it in action!</strong></p>


    <img src="https://picsum.photos/id/1015/1000/600" alt="Responsive River Landscape">
```

```
    <div class="explanation">
      <h3>How It Works:</h3>
      <ul>
        <li><code>max-width: 100%;</code> ensures the image never overflows its container.</li>
        <li><code>height: auto;</code> makes the browser calculate the height automatically to maintain the aspect ratio.</li>
        <li><code>display: block;</code> and <code>margin: 0 auto;</code> are used to center the image neatly on the page.</li>
      </ul>
    </div>
  </div>

</body>
</html>
```



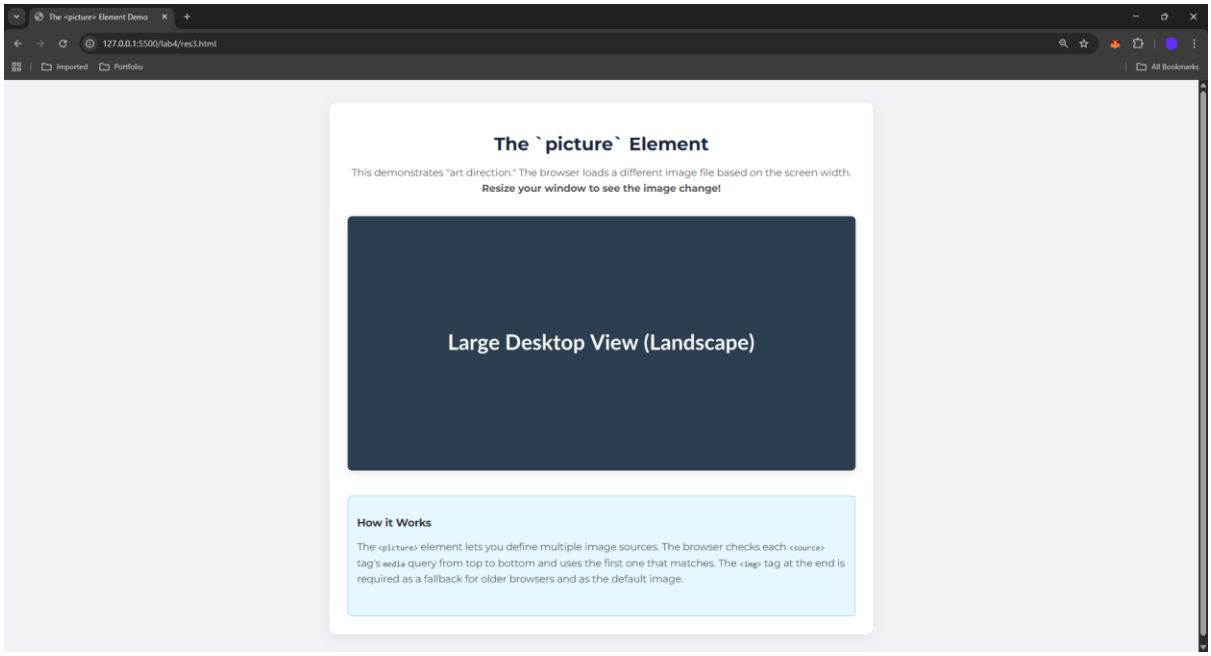


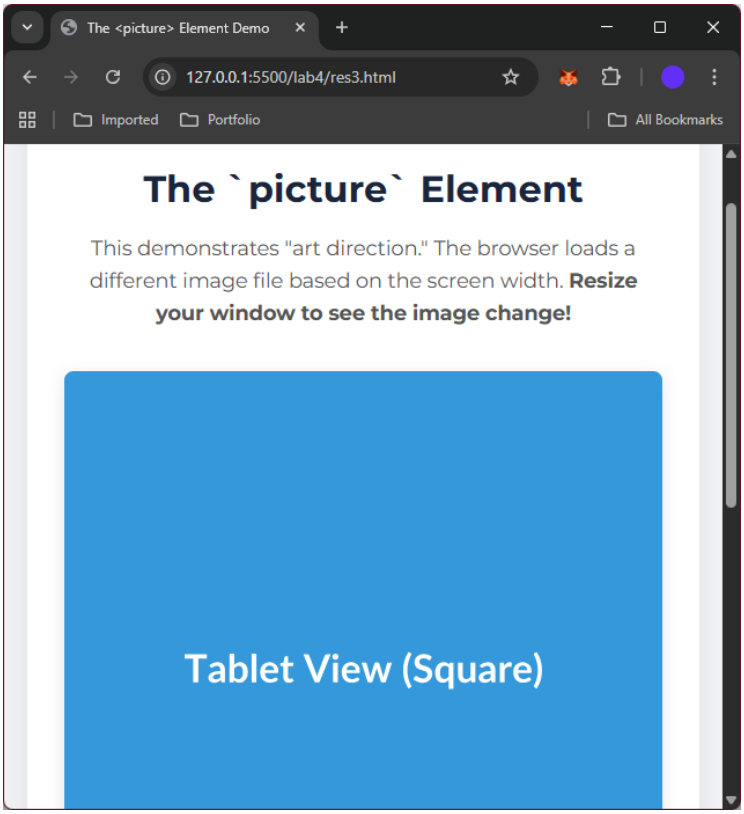
2.3 Using <picture> Element

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

    <link rel="preconnect" href="https://fonts.googleapis.com">
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
    <link href="https://fonts.googleapis.com/css2?family=Montserrat:wght@400;700&display=swap"
rel="stylesheet">

    <style>
        body {
            font-family: 'Montserrat', sans-serif;
```

```
        background-color: #f0f2f5;
        color: #333;
        margin: 0;
        padding: 20px;
        text-align: center;
    }

    .container {
        max-width: 900px;
        margin: 20px auto;
        padding: 2em;
        background-color: #ffffff;
        border-radius: 12px;
        box-shadow: 0 6px 20px rgba(0,0,0,0.07);
    }

    h1 {
        color: #1a253c;
        margin-bottom: 0.5em;
    }

    p {
        font-size: 1.1em;
        line-height: 1.6;
        color: #555;
        margin-bottom: 2em;
    }

    /* The image will be responsive within its container */
    img {
        max-width: 100%;
        height: auto;
        border-radius: 8px;
        box-shadow: 0 4px 15px rgba(0,0,0,0.1);
    }

    .info-box {
        margin-top: 2.5em;
        padding: 1em;
        background-color: #e6f7ff;
        border: 1px solid #91d5ff;
        border-radius: 8px;
        text-align: left;
    }

  </style>
</head>
<body>
```

```
  <div class="container">
    <h1>The `picture` Element</h1>
    <p>This demonstrates "art direction." The browser loads a different image file based on the screen width. <strong>Resize your window to see the image change!</strong></p>

    <picture>
      <source
        media="(min-width: 1000px)"
        srcset="https://placehold.co/1200x600/2c3e50/ffffff?text=Large+Desktop+View+(Landscape)">

      <source
        media="(min-width: 600px)"
        srcset="https://placehold.co/600x600/3498db/ffffff?text=Tablet+View+(Square)">


      <img
        src="https://placehold.co/400x600/e74c3c/ffffff?text=Mobile+View+(Portrait)"
        alt="A responsive image that changes based on screen size">
    </picture>

    <div class="info-box">
      <h3>How it Works</h3>
      <p>The <code>&lt;picture&gt;</code> element lets you define multiple image sources. The browser checks each <code>&lt;source&gt;</code> tag's <code>media</code> query from top to bottom and uses the first one that matches. The <code>&lt;img&gt;</code> tag at the end is required as a fallback for older browsers and as the default image.</p>
    </div>
  </div>

</body>
</html>
```



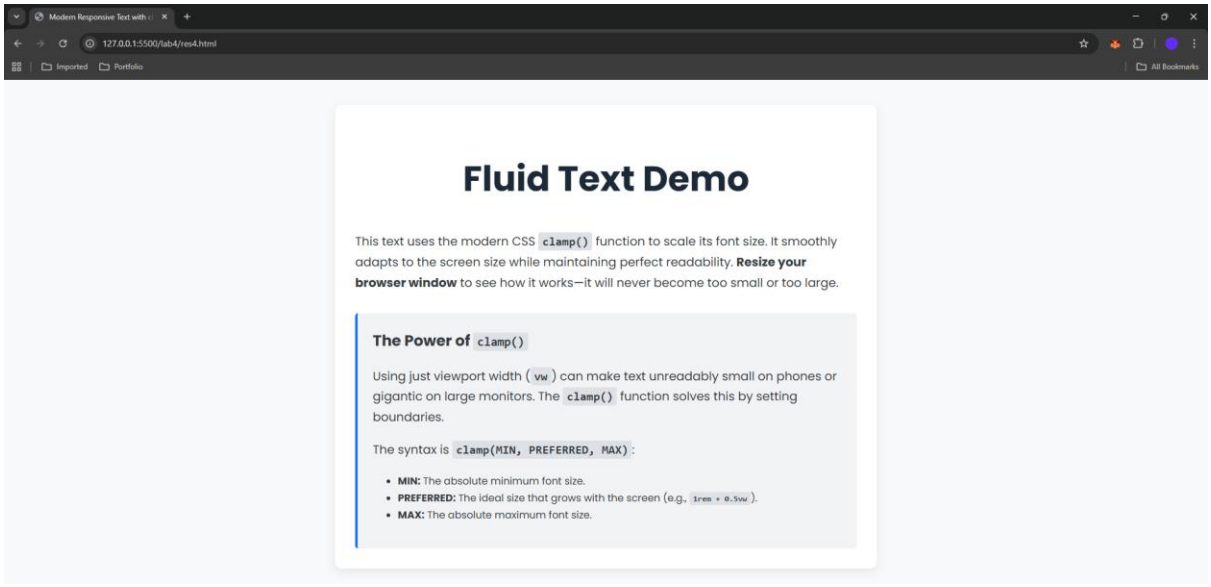


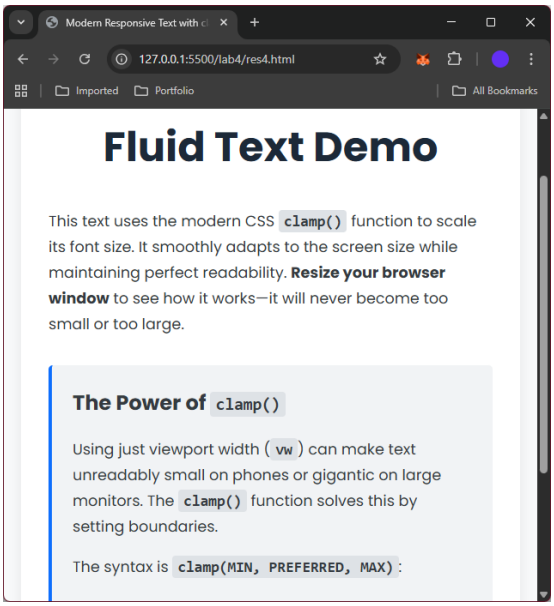
3. Responsive Texts

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Modern Responsive Text with clamp()</title>
```

```
  <link rel="preconnect" href="https://fonts.googleapis.com">
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
  <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;700&display=swap"
rel="stylesheet">

  <style>
    body {
      font-family: 'Poppins', sans-serif;
      background-color: #f8f9fa;
      color: #343a40;
      line-height: 1.7;
      margin: 0;
      padding: 20px;
    }

    .container {
      max-width: 800px;
      margin: 20px auto;
      padding: 2em;
      background-color: #ffffff;
      border-radius: 10px;
      box-shadow: 0 5px 15px rgba(0,0,0,0.08);
    }

    /*
     * The Modern Way: Using clamp() for fluid typography.
     * clamp(MINIMUM, PREFERRED, MAXIMUM);
     * The font size will scale between the minimum and maximum values.
    */
    h1 {
      text-align: center;
      color: #1c2938;
      /* Font will be at least 2rem, will try to be 1rem + 5vw, but will not exceed 3.5rem */
      font-size: clamp(2rem, 1rem + 5vw, 3.5rem);
    }

    p {
      /* Font will be at least 1rem, will try to be 0.9rem + 0.5vw, but will not exceed 1.2rem */
      font-size: clamp(1rem, 0.9rem + 0.5vw, 1.2rem);
    }

    .explanation {
      margin-top: 2em;
      padding: 1.5em;
      background-color: #f1f3f5;
      border-left: 4px solid #0d6efd;
      border-radius: 5px;
    }
```

```
      .explanation h2 {
        margin-top: 0;
        font-size: 1.5em;
      }

      code {
        background-color: #dee2e6;
        padding: 3px 6px;
        border-radius: 4px;
        font-family: monospace;
        font-weight: 600;
      }

  </style>
</head>
<body>

  <div class="container">
    <h1>Fluid Text Demo</h1>

    <p>
      This text uses the modern CSS <code>clamp()</code> function to scale its font size. It smoothly adapts to the screen size while maintaining perfect readability. <strong>Resize your browser window</strong> to see how it works—it will never become too small or too large.
    </p>

    <div class="explanation">
      <h2>The Power of <code>clamp()</code></h2>
      <p>
        Using just viewport width (<code>vw</code>) can make text unreadably small on phones or gigantic on large monitors. The <code>clamp()</code> function solves this by setting boundaries.
      </p>
      <p>
        The syntax is <code>clamp(MIN, PREFERRED, MAX)</code>:
      </p>
      <ul>
        <li><strong>MIN:</strong> The absolute minimum font size.</li>
        <li><strong>PREFERRED:</strong> The ideal size that grows with the screen (e.g., <code>1rem + 0.5vw</code>).</li>
        <li><strong>MAX:</strong> The absolute maximum font size.</li>
      </ul>
    </div>
  </div>

</body>
</html>
```





➢ Text sizes adjust dynamically with screen width.

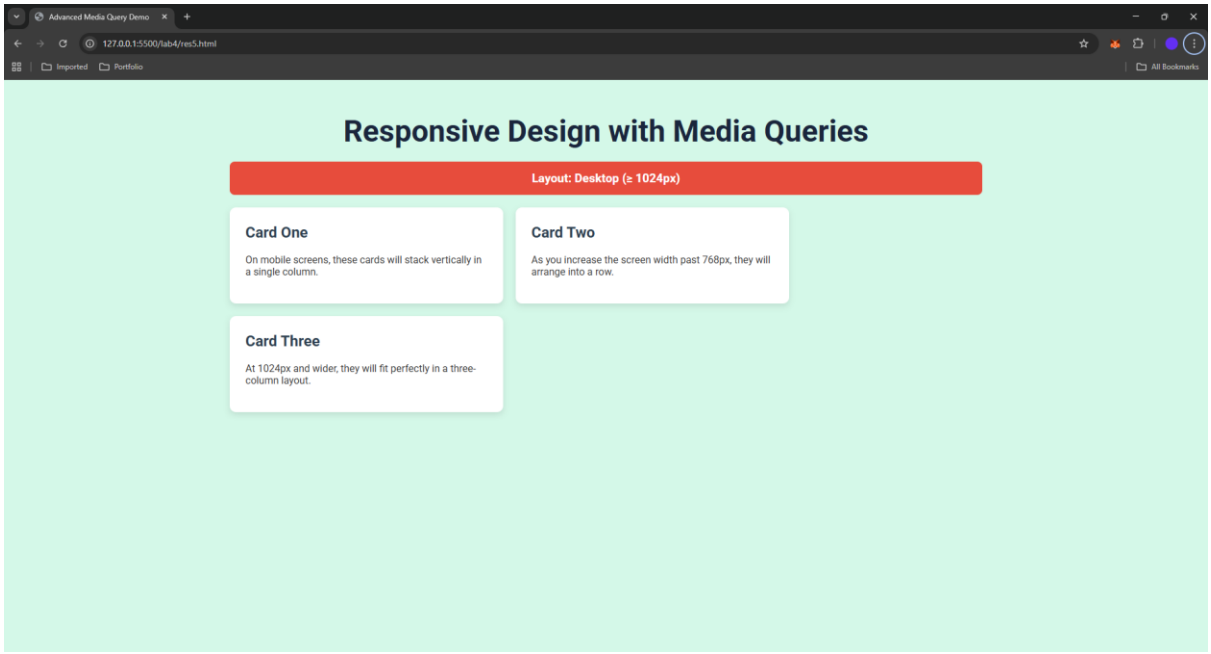
4. CSS Media Queries

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

    <link rel="preconnect" href="https://fonts.googleapis.com">
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
    <link href="https://fonts.googleapis.com/css2?family=Roboto:wght@400;700&display=swap"
rel="stylesheet">
```

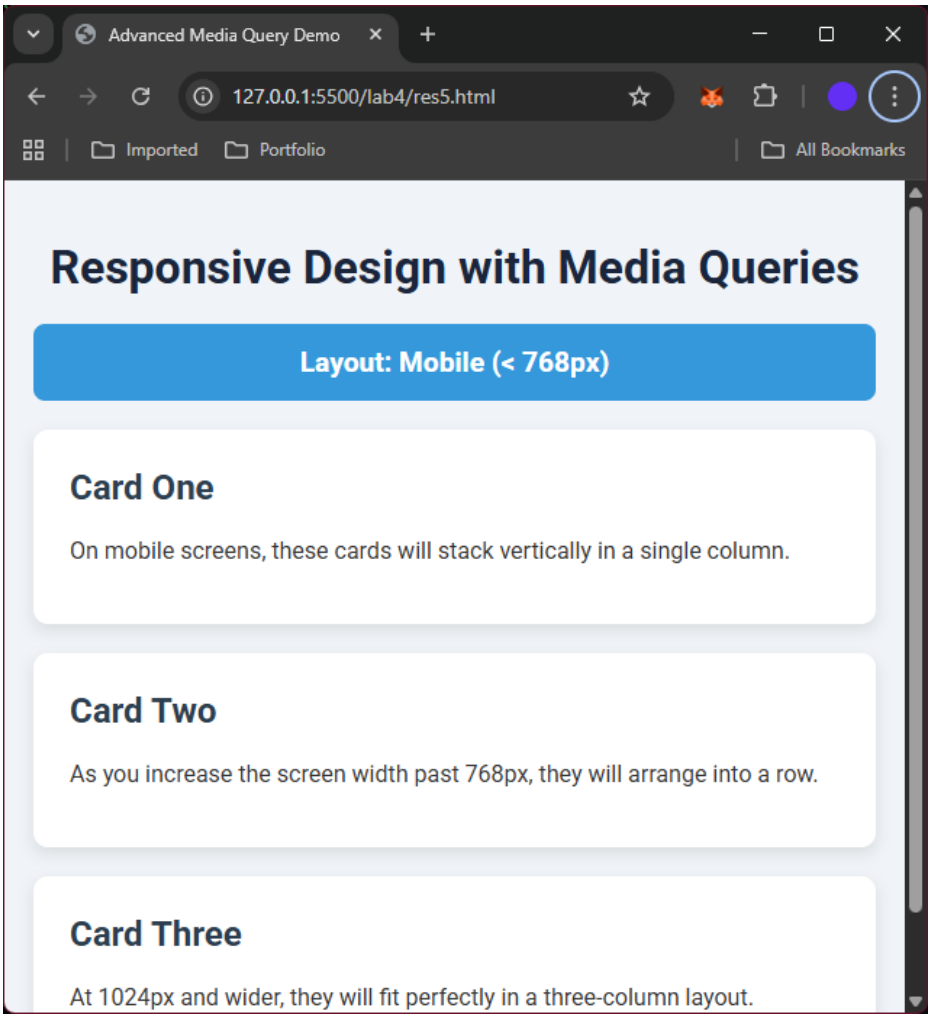
```
<style>
    /* --- Mobile First Styles (Default) --- */
    body {
        font-family: 'Roboto', sans-serif;
        background-color: #f0f4f8;
        color: #333;
        margin: 0;
        padding: 20px;
        transition: background-color 0.3s ease;
    }

    .container {
        max-width: 1200px;
        margin: 0 auto;
    }

    h1 {
        font-size: 2em;
        text-align: center;
        color: #1a253c;
        margin-bottom: 20px;
    }

    /* This box will show which layout is active */
    .breakpoint-indicator {
        padding: 15px;
        margin-bottom: 20px;
        background-color: #3498db;
        color: white;
        text-align: center;
        font-weight: 700;
        border-radius: 8px;
        font-size: 1.2em;
    }

    .breakpoint-indicator::before {
        content: 'Layout: Mobile (< 768px)';
    }

    /* On mobile, cards are in a single column */
    .card-container {
        display: flex;
        flex-direction: column;
        gap: 20px;
    }

    .card {
        background-color: #ffffff;
```

```
    padding: 25px;
    border-radius: 10px;
    box-shadow: 0 4px 10px rgba(0,0,0,0.1);
}

.card h2 {
  margin-top: 0;
  color: #2c3e50;
}

/* --- Tablet Breakpoint --- */
@media screen and (min-width: 768px) {
    body {
        background-color: #e0e7ff; /* Lighter blue for tablet */
    }
    h1 {
        font-size: 2.5em;
    }
    .breakpoint-indicator {
        background-color: #2ecc71; /* Green for tablet */
    }
    .breakpoint-indicator::before {
        content: 'Layout: Tablet (≥ 768px)';
    }
    /* Switch to a row layout for the cards */
    .card-container {
        flex-direction: row;
        flex-wrap: wrap; /* Allow cards to wrap to the next line */
    }
    .card {
        /* Each card takes up roughly half the width */
        flex-basis: calc(50% - 10px);
    }
}

/* --- Desktop Breakpoint --- */
@media screen and (min-width: 1024px) {
    body {
        background-color: #d4f8e8; /* Lighter green for desktop */
    }
    h1 {
        font-size: 3em;
    }
    .breakpoint-indicator {
        background-color: #e74c3c; /* Red for desktop */
    }
    .breakpoint-indicator::before {
        content: 'Layout: Desktop (≥ 1024px)';
```

```
        }
        .card {
            /* Each card takes up roughly a third of the width */
            flex-basis: calc(33.333% - 14px);
        }
    }
  </style>
</head>
<body>
  <div class="container">
    <h1>Responsive Design with Media Queries</h1>
    <div class="breakpoint-indicator"></div>

    <div class="card-container">
      <div class="card">
        <h2>Card One</h2>
        <p>On mobile screens, these cards will stack vertically in a single column.</p>
      </div>
      <div class="card">
        <h2>Card Two</h2>
        <p>As you increase the screen width past 768px, they will arrange into a row.</p>
      </div>
      <div class="card">
        <h2>Card Three</h2>
        <p>At 1024px and wider, they will fit perfectly in a three-column layout.</p>
      </div>
    </div>
  </div>
</body>
</html>
```



- Normal mode: white background, large fonts.
- When resized below 800px: background turns aqua, text shrinks.



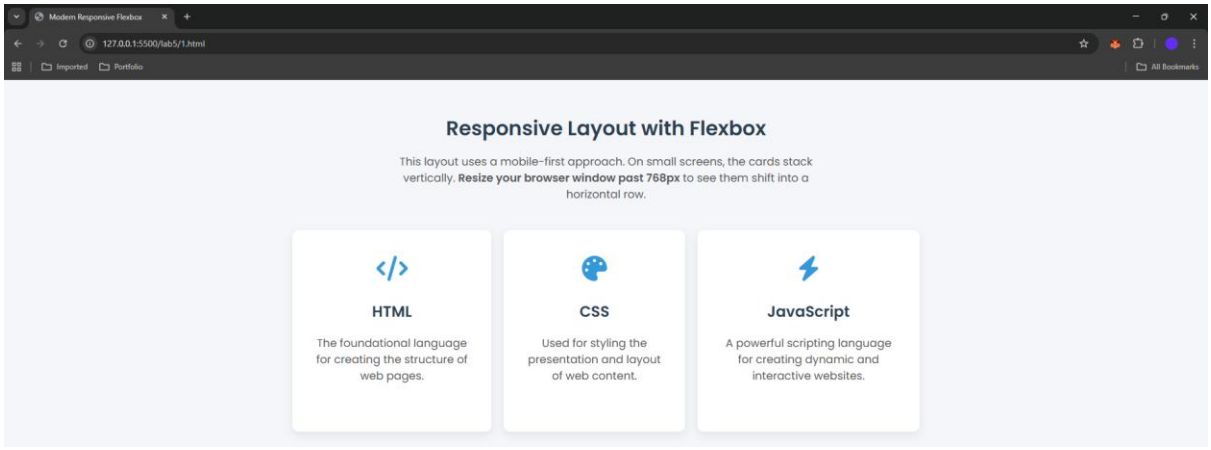
5. Responsive Layouts

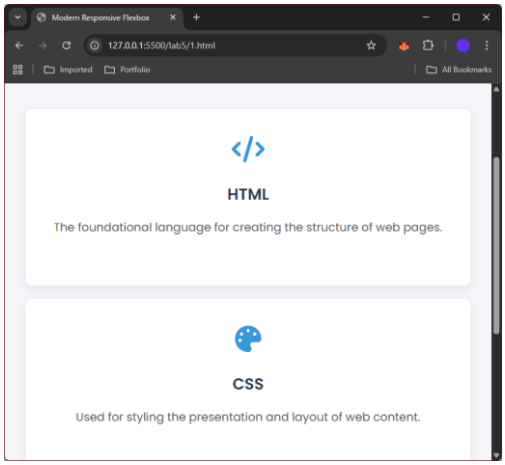
5.1 Using Flexbox

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

    <link rel="preconnect" href="https://fonts.googleapis.com">
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
    <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;600&display=swap"
rel="stylesheet">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.5.1/css/all.min.css">
```

```
<style>
    /* --- Mobile First (Default) Styles --- */
    body {
        font-family: 'Poppins', sans-serif;
        background-color: #f4f6f9;
        color: #333;
        margin: 0;
        padding: 2em;
    }

    .container {
        max-width: 1000px;
        margin: 0 auto;
        text-align: center;
    }

    h1 {
        color: #2c3e50;
        margin-bottom: 0.5em;
    }

    p {
        font-size: 1.1em;
        color: #555;
        max-width: 700px;
        margin: 0 auto 2.5em auto;
    }

    .flex-container {
        display: flex;
        /* On mobile, cards stack vertically */
        flex-direction: column;
        gap: 20px;
    }

    .card {
        background-color: #ffffff;
        padding: 2em;
        border-radius: 10px;
        box-shadow: 0 5px 15px rgba(0,0,0,0.08);
        text-align: center;
        transition: transform 0.3s ease, box-shadow 0.3s ease;
    }

    .card:hover {
        transform: translateY(-8px);
        box-shadow: 0 8px 25px rgba(0,0,0,0.12);
    }
```

```
        .card .icon {
            font-size: 2.5em;
            color: #3498db;
            margin-bottom: 0.5em;
        }

        .card h2 {
            margin-top: 0;
            color: #2c3e50;
        }

        /* --- Desktop Breakpoint --- */
        /* When the screen is 768px or wider, apply these styles */
        @media screen and (min-width: 768px) {
            .flex-container {
                /* On larger screens, arrange cards in a row */
                flex-direction: row;
            }
        }
    </style>
</head>
<body>
    <div class="container">
        <h1>Responsive Layout with Flexbox</h1>
        <p>This layout uses a mobile-first approach. On small screens, the cards stack vertically.
<strong>Resize your browser window past 768px</strong> to see them shift into a horizontal row.</p>

        <div class="flex-container">
            <div class="card">
                <div class="icon"><i class="fa-solid fa-code"></i></div>
                <h2>HTML</h2>
                <p>The foundational language for creating the structure of web pages.</p>
            </div>
            <div class="card">
                <div class="icon"><i class="fa-solid fa-palette"></i></div>
                <h2>CSS</h2>
                <p>Used for styling the presentation and layout of web content.</p>
            </div>
            <div class="card">
                <div class="icon"><i class="fa-solid fa-bolt"></i></div>
                <h2>JavaScript</h2>
                <p>A powerful scripting language for creating dynamic and interactive websites.</p>
            </div>
        </div>
    </div>
</body>

</html>
```



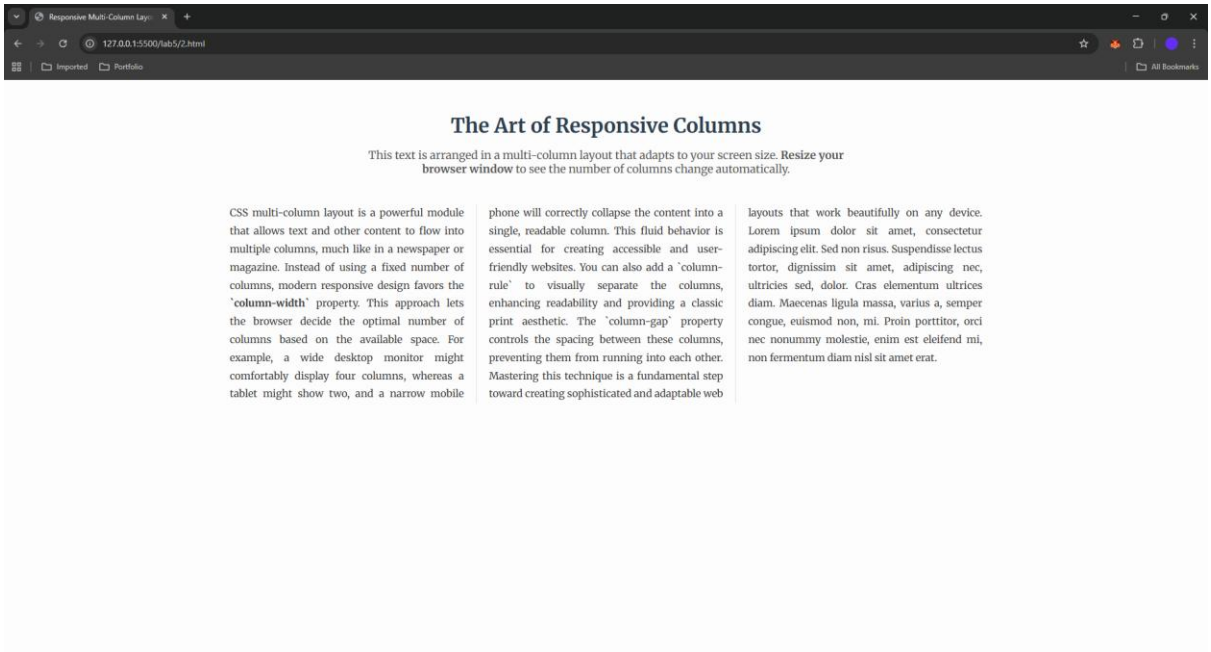


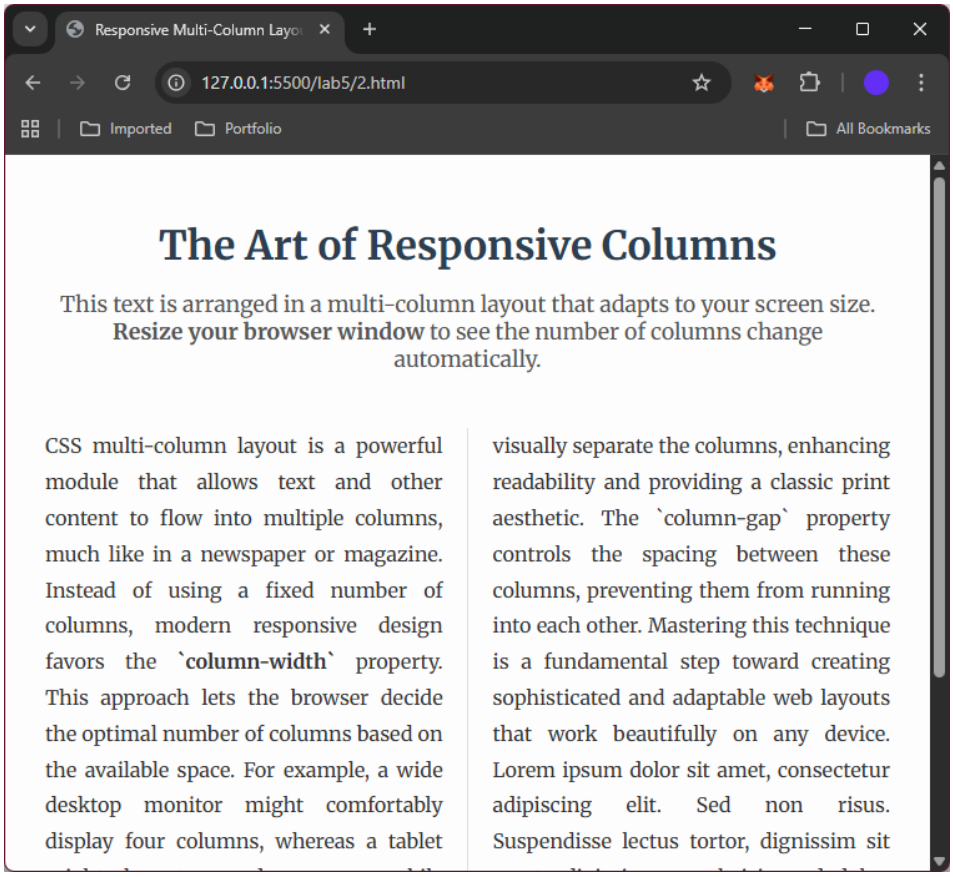
5.2 Using MultiColumn

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

    <link rel="preconnect" href="https://fonts.googleapis.com">
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
    <link href="https://fonts.googleapis.com/css2?family=Merriweather:wght@400;700&display=swap"
rel="stylesheet">

    <style>
        body {
            font-family: 'Merriweather', serif;
            background-color: #fdfdfd;
            color: #333;
            margin: 0;
            padding: 2em;
        }
```

```
      .container {
        max-width: 1200px;
        margin: 0 auto;
      }

      h1 {
        text-align: center;
        color: #2c3e50;
        margin-bottom: 0.5em;
        font-weight: 700;
      }

      p.intro {
        text-align: center;
        font-size: 1.1em;
        color: #555;
        max-width: 800px;
        margin: 0 auto 2.5em auto;
      }

      .content {
        /* This is the key for a responsive column layout.
           The browser will create columns that are AT LEAST 300px wide.
           It automatically adjusts the column count to fit the container. */
        column-width: 300px;

        /* Sets the space between columns */
        column-gap: 40px;

        /* Adds a decorative vertical line between columns */
        column-rule: 1px solid #e0e0e0;

        /* Justify text for a clean, newspaper-like look */
        text-align: justify;
        line-height: 1.8;
      }
    </style>
</head>
<body>
    <div class="container">
      <h1>The Art of Responsive Columns</h1>
      <p class="intro">
        This text is arranged in a multi-column layout that adapts to your screen size.
        <strong>Resize your browser window</strong> to see the number of columns change
automatically.
      </p>
```

```
    <div class="content">
        CSS multi-column layout is a powerful module that allows text and other content to flow into multiple columns, much like in a newspaper or magazine. Instead of using a fixed number of columns, modern responsive design favors the <strong>`column-width`</strong> property. This approach lets the browser decide the optimal number of columns based on the available space. For example, a wide desktop monitor might comfortably display four columns, whereas a tablet might show two, and a narrow mobile phone will correctly collapse the content into a single, readable column. This fluid behavior is essential for creating accessible and user-friendly websites. You can also add a `column-rule` to visually separate the columns, enhancing readability and providing a classic print aesthetic. The `column-gap` property controls the spacing between these columns, preventing them from running into each other. Mastering this technique is a fundamental step toward creating sophisticated and adaptable web layouts that work beautifully on any device. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed non risus. Suspendisse lectus tortor, dignissim sit amet, adipiscing nec, ultricies sed, dolor. Cras elementum ultrices diam. Maecenas ligula massa, varius a, semper congue, euismod non, mi. Proin porttitor, orci nec nonummy molestie, enim est eleifend mi, non fermentum diam nisl sit amet erat.
    </div>
  </div>
</body>
</html>
```





Observations

- ✓ Viewport tag ensured responsiveness across devices.
- ✓ Images scaled smoothly with width, max-width, and <picture>.
- ✓ Text scaled dynamically using vw units.
- ✓ Media queries allowed adaptive design changes at different breakpoints.
- ✓ Flexbox and MultiColumn layouts provided flexible, device-friendly structures.

Conclusion

This lab demonstrated HTML responsive web design techniques, including viewport, images, texts, media queries, and layouts. These methods ensure webpages adjust seamlessly across multiple screen sizes, improving accessibility and user experience.

LAB EXPERIMENT - 5

PHP Installation and React Setup in VS Code

Aim:

- To install PHP on Windows 11 and verify its installation.
- To create and run a React application using Visual Studio Code.
- To demonstrate debugging, linting, and live editing in React.

Software Requirements:

- Windows 11
- Git (optional for project version control)
- PHP (latest version)
- Node.js and npm
- Visual Studio Code
- ESLint extension in VS Code

Part A: Installing PHP on Windows 11

Steps & Commands:

1. Download PHP ZIP from https://www.php.net/downloads (Thread Safe version)
2. Extract the folder and copy it to C:\Program Files\PHP.
3. Add PHP folder to system Path:

- Start → Search Edit the system environment variables → Environment Variables → Path → New → Paste PHP folder path → OK.

4. Verify installation

```
(base) PS D:\CH.EN.U4AIE22046\my-first-git-repo> php -v
PHP 8.4.12 (cli) (built: Aug 26 2025 21:18:21) (ZTS Visual C++ 2022 x64)
Copyright (c) The PHP Group
Zend Engine v4.4.12, Copyright (c) Zend Technologies
(base) PS D:\CH.EN.U4AIE22046\my-first-git-repo>
```

Part B: Using React in VS Code

1. Create React App

```
(base) PS D:\CH.EN.U4AIE22046\my-first-git-repo\lab5> node -v
v22.18.0
(base) PS D:\CH.EN.U4AIE22046\my-first-git-repo\lab5> npm -v
10.9.3
(base) PS D:\CH.EN.U4AIE22046\my-first-git-repo\lab5> npx create-react-app test-app
Need to install the following packages:
create-react-app@5.1.0
Ok to proceed? (y) y

npm warn deprecated inflight@1.0.6: This module is not supported, and leaks memory. Do not use it. Check out lru-cache if you want a g
ood and tested way to coalesce async requests by a key value, which is much more comprehensive and powerful.
npm warn deprecated rimraf@2.7.1: Rimraf versions prior to v4 are no longer supported
npm warn deprecated fstream-ignore@1.0.5: This package is no longer supported.
npm warn deprecated glob@7.2.3: Glob versions prior to v9 are no longer supported
npm warn deprecated fstream@1.0.12: This package is no longer supported.
npm warn deprecated uid-number@0.0.6: This package is no longer supported.
npm warn deprecated tar@2.2.2: This version of tar is no longer supported, and will not receive security updates. Please upgrade asap.
create-react-app is deprecated.

You can find a list of up-to-date React frameworks on react.dev
For more info see:https://react.dev/link/cra

This error message will only be shown once per install.

Creating a new React app in D:\CH.EN.U4AIE22046\my-first-git-repo\lab5\test-app.

Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts with cra-template...
```

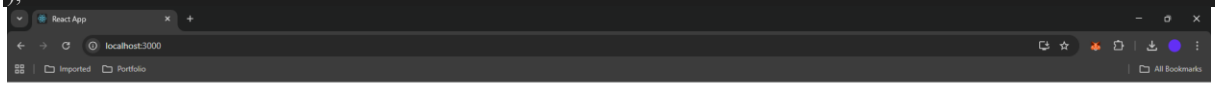
2. Run React App

```
(base) PS D:\CH.EN.U4AIE22046\my-first-git-repo\lab5> cd .\test-app\
(base) PS D:\CH.EN.U4AIE22046\my-first-git-repo\lab5\test-app> npm start

> test-app@0.1.0 start
> react-scripts start

(node:22300) [DEP_WEBPACK_DEV_SERVER_ON_AFTER_SETUP_MIDDLEWARE] DeprecationWarning: 'onAfterSetupMiddleware' option is deprecated. Ple
ase use the 'setupMiddlewares' option.
(Use `node --trace-deprecation ...` to show where the warning was created)
(node:22300) [DEP_WEBPACK_DEV_SERVER_ON_BEFORE_SETUP_MIDDLEWARE] DeprecationWarning: 'onBeforeSetupMiddleware' option is deprecated. P
lease use the 'setupMiddlewares' option.
Starting the development server...
Compiled successfully!

You can now view test-app in the browser.

  Local:            http://localhost:3000
  On Your Network:  http://192.168.56.1:3000

Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
```

3. Modify App to Show “Hello, world!

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';

function HelloWorld() {
  return <h1 className="greeting">Hello, world!</h1>;
}
const root = ReactDOM.createRoot(document.getElementById('root'));
```

```
root.render(
  <React.StrictMode>
    <HelloWorld />
  </React.StrictMode>
);
```



Hello, world!

4. Debug React App

Steps:

- Open Run and Debug in VS Code (Ctrl+Shift+D).
- Click create a launch.json file → Select Web App (Edge).
- Change url to http://localhost:3000:

```
{
 "version": "0.2.0",
 "configurations": [
   {
    "type": "msedge",
    "request": "launch",
    "name": "Launch Edge against localhost",
    "url": "http://localhost:3000",
    "webRoot": "${workspaceFolder}"
   }
 ]}
```

- Set breakpoints in index.js → Press **F5** → Refresh browser.

5. ESLint Setup and Linting

Commands: npm install -g eslint

Create .eslintrc.js:

```
module.exports = {
  env: {
    browser: true,
    es2020: true,
  },
  extends: ['eslint:recommended', 'plugin:react/recommended'],
  parserOptions: {
    ecmaFeatures: { jsx: true },
    ecmaVersion: 11,
    sourceType: 'module',
  },
  plugins: ['react'],
  rules: {
    "no-extra-semi": "error",
    "react/prop-types": "off"
  },
```

```
};
```

Test ESLint:

- Add extra semicolon ;; → Red squiggly line appears.
- Check Problems panel in VS Code.
- Screenshot: ESLint error highlighted in VS Code.

Result

- ✓ PHP successfully installed and verified on Windows 11.
- ✓ React application created and running successfully in VS Code.
- ✓ Code debugging, linting, and live editing demonstrated.

LAB EXPERIMENT - 6

6.1 Write a JavaScript program to compute the sum of an array of integers

Aim:

To compute the sum of an array of integers using JavaScript.

Source Code:

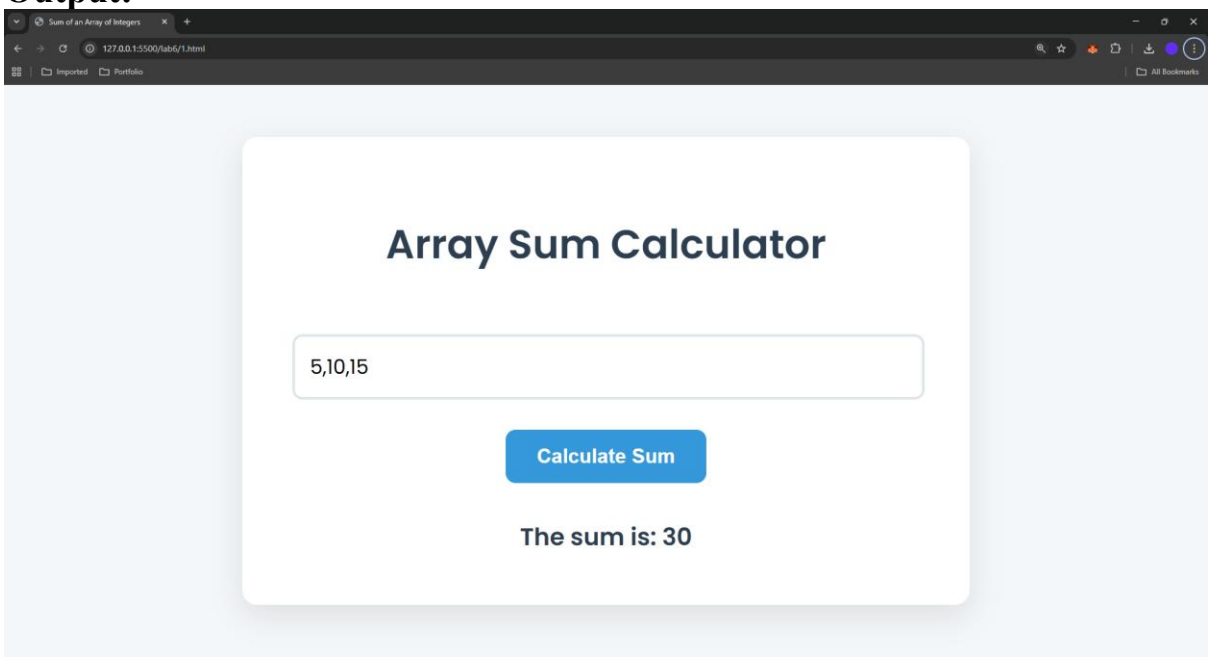
```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Sum of an Array of Integers</title>
  <!-- Google Font for a modern look -->
  <link rel="preconnect" href="https://fonts.googleapis.com">
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
  <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;600&display=swap"
rel="stylesheet">
  <style>
    body {
      font-family: 'Poppins', sans-serif;
      background-color: #f4f7f9;
      color: #333;
      display: flex;
      justify-content: center;
      align-items: center;
      min-height: 100vh;
      margin: 0;
    }

    .container {
      background-color: #ffffff;
      padding: 2.5em;
      border-radius: 12px;
      box-shadow: 0 8px 25px rgba(0, 0, 0, 0.1);
      width: 100%;
      max-width: 500px;
      text-align: center;
    }

    h1 {
      color: #2c3e50;
```

```
        margin-bottom: 1.5em;
    }

    #arrayInput {
        width: calc(100% - 24px); /* Account for padding */
        padding: 12px;
        font-size: 1em;
        border: 2px solid #dfe6e9;
        border-radius: 8px;
        margin-bottom: 1.5em;
        font-family: 'Poppins', sans-serif;
        transition: border-color 0.3s ease;
    }

    #arrayInput:focus {
        outline: none;
        border-color: #3498db;
    }

    button {
        background-color: #3498db;
        color: white;
        border: none;
        padding: 12px 25px;
        font-size: 1em;
        font-weight: 600;
        border-radius: 8px;
        cursor: pointer;
        transition: background-color 0.3s ease, transform 0.2s ease;
    }

    button:hover {
        background-color: #2980b9;
        transform: translateY(-2px);
    }

    #result {
        margin-top: 1.5em;
        font-size: 1.2em;
        font-weight: 600;
        color: #2c3e50;
        min-height: 25px;
    }

    .error {
        color: #e74c3c;
    }
</style>
```

```
</head>
<body>

  <div class="container">
    <h1>Array Sum Calculator</h1>
    <input type="text" id="arrayInput" placeholder="Enter numbers, e.g., 5, 8, 12, 4">
    <button onclick="calculateSum()">Calculate Sum</button>
    <div id="result"></div>
  </div>

  <script>
    function calculateSum() {
      const inputString = document.getElementById('arrayInput').value;
      const resultDiv = document.getElementById('result');

      // Clear previous result
      resultDiv.textContent = '';
      resultDiv.classList.remove('error');

      if (inputString.trim() === '') {
        resultDiv.textContent = 'Please enter some numbers.';
        resultDiv.classList.add('error');
        return;
      }

      // Split the string by commas and convert each part to a number
      const stringArray = inputString.split(',');
      const numbers = [];

      for (let i = 0; i < stringArray.length; i++) {
        const num = parseInt(stringArray[i].trim(), 10);
        if (isNaN(num)) {
          resultDiv.textContent = `Error: "${stringArray[i].trim()}" is not a valid integer.`;
          resultDiv.classList.add('error');
          return;
        }
        numbers.push(num);
      }

      // --- Method 1: Using the reduce() method (most modern and concise) ---
      const sum = numbers.reduce((accumulator, currentValue) => accumulator + currentValue, 0);

      /*
      // --- Method 2: Using a for loop (classic approach) ---
      let sum = 0;
      for (let i = 0; i < numbers.length; i++) {
        sum += numbers[i];
      }
```

```
        */

        /*
        // --- Method 3: Using forEach() method ---
        let sum = 0;
        numbers.forEach(number => {
            sum += number;
        });
        */

        // Display the result
        resultDiv.textContent = `The sum is: ${sum}`;
    }
  </script>

</body>
</html>
```

Output:



6.2 Write a JavaScript program to determine whether a given year is a leap year

Aim:

To check if a given year is a leap year or not.

Source Code:

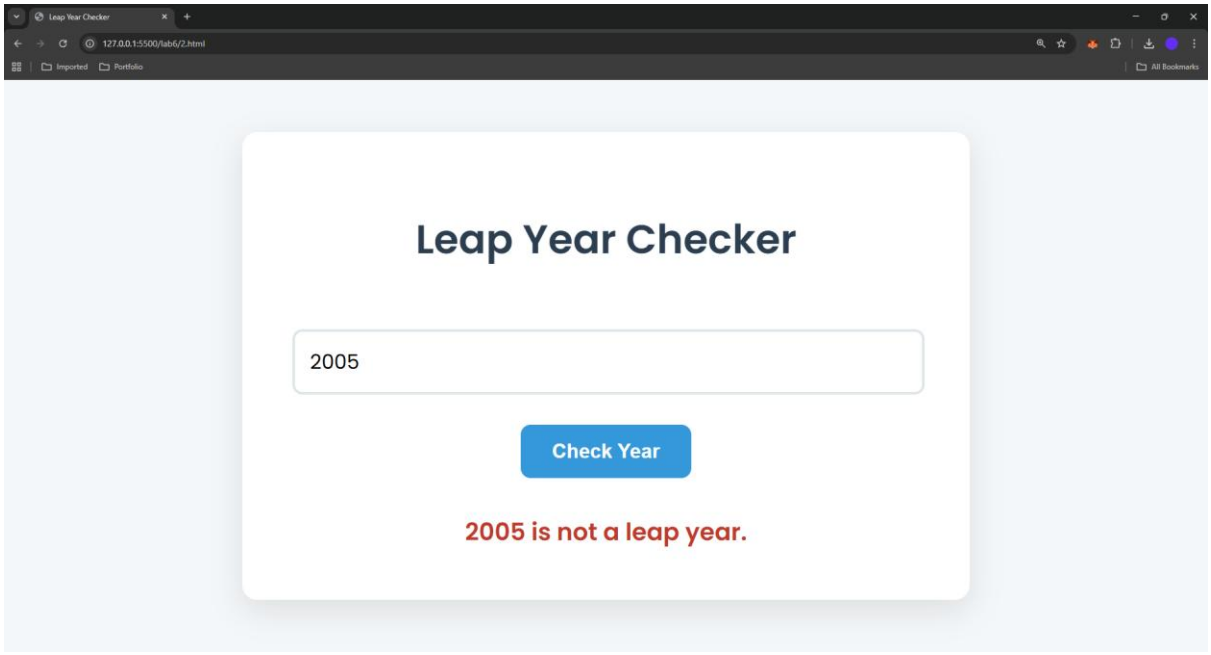
```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Leap Year Checker</title>
   <!-- Google Font for a modern look -->
   <link rel="preconnect" href="https://fonts.googleapis.com">
   <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
   <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;600&display=swap"
rel="stylesheet">
   <style>
      body {
         font-family: 'Poppins', sans-serif;
         background-color: #f4f7f9;
         color: #333;
         display: flex;
         justify-content: center;
         align-items: center;
         min-height: 100vh;
         margin: 0;
      }

      .container {
         background-color: #ffffff;
         padding: 2.5em;
         border-radius: 12px;
         box-shadow: 0 8px 25px rgba(0, 0, 0, 0.1);
         width: 100%;
         max-width: 500px;
         text-align: center;
      }

      h1 {
         color: #2c3e50;
         margin-bottom: 1.5em;
      }

      #yearInput {
         width: calc(100% - 24px); /* Account for padding */
```

```
    padding: 12px;
    font-size: 1em;
    border: 2px solid #dfe6e9;
    border-radius: 8px;
    margin-bottom: 1.5em;
    font-family: 'Poppins', sans-serif;
    transition: border-color 0.3s ease;
}

#yearInput:focus {
    outline: none;
    border-color: #3498db;
}

button {
    background-color: #3498db;
    color: white;
    border: none;
    padding: 12px 25px;
    font-size: 1em;
    font-weight: 600;
    border-radius: 8px;
    cursor: pointer;
    transition: background-color 0.3s ease, transform 0.2s ease;
}

button:hover {
    background-color: #2980b9;
    transform: translateY(-2px);
}

#result {
    margin-top: 1.5em;
    font-size: 1.2em;
    font-weight: 600;
    min-height: 25px;
}

.leap {
    color: #27ae60; /* Green for leap year */
}

.not-leap {
    color: #c0392b; /* Red for non-leap year */
}

.error {
    color: #e74c3c;
```

```
    }
  </style>
</head>
<body>

  <div class="container">
    <h1>Leap Year Checker</h1>
    <input type="text" id="yearInput" placeholder="Enter a year, e.g., 2024">
    <button onclick="checkLeapYear()">Check Year</button>
    <div id="result"></div>
  </div>

  <script>
    function checkLeapYear() {
      const yearString = document.getElementById('yearInput').value;
      const resultDiv = document.getElementById('result');

      // Clear previous result and classes
      resultDiv.textContent = '';
      resultDiv.classList.remove('leap', 'not-leap', 'error');

      if (yearString.trim() === '') {
        resultDiv.textContent = 'Please enter a year.';
        resultDiv.classList.add('error');
        return;
      }

      const year = parseInt(yearString, 10);

      if (isNaN(year) || year < 0) {
        resultDiv.textContent = 'Please enter a valid positive year.';
        resultDiv.classList.add('error');
        return;
      }

      // A year is a leap year if it is divisible by 4,
      // except for end-of-century years, which must be divisible by 400.
      // This logic covers all cases.
      if ((year % 4 === 0 && year % 100 !== 0) || (year % 400 === 0)) {
        resultDiv.textContent = `${year} is a leap year!`;
        resultDiv.classList.add('leap');
      } else {
        resultDiv.textContent = `${year} is not a leap year.`;
        resultDiv.classList.add('not-leap');
      }
    }
  </script>
```

```
</body>
</html>
```

Output:



6.3 Write a JavaScript function that checks whether a string is palindrome

Aim:

To check if the given string is a palindrome.

Source Code:

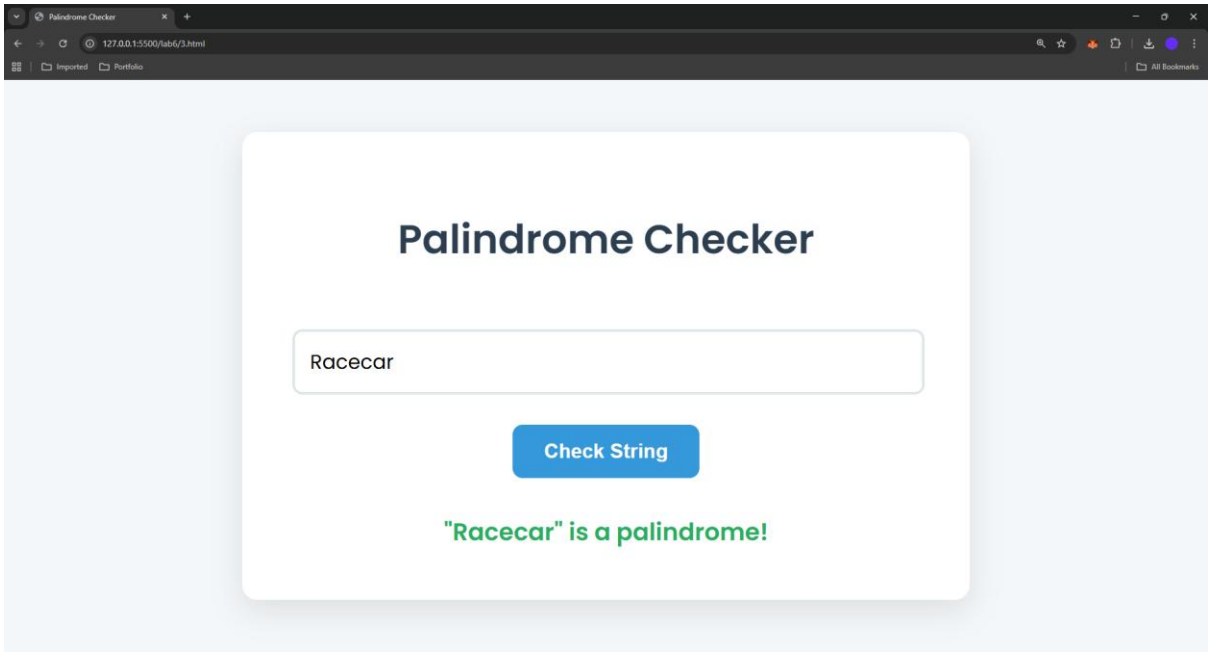
```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Palindrome Checker</title>
   <!-- Google Font for a modern look -->
   <link rel="preconnect" href="https://fonts.googleapis.com">
   <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
   <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;600&display=swap"
rel="stylesheet">
   <style>
      body {
         font-family: 'Poppins', sans-serif;
         background-color: #f4f7f9;
         color: #333;
         display: flex;
         justify-content: center;
         align-items: center;
         min-height: 100vh;
         margin: 0;
      }

      .container {
         background-color: #ffffff;
         padding: 2.5em;
         border-radius: 12px;
         box-shadow: 0 8px 25px rgba(0, 0, 0, 0.1);
         width: 100%;
         max-width: 500px;
         text-align: center;
      }

      h1 {
         color: #2c3e50;
         margin-bottom: 1.5em;
      }

      #stringInput {
         width: calc(100% - 24px); /* Account for padding */
         padding: 12px;
```

```
    font-size: 1em;
    border: 2px solid #dfe6e9;
    border-radius: 8px;
    margin-bottom: 1.5em;
    font-family: 'Poppins', sans-serif;
    transition: border-color 0.3s ease;
}

#stringInput:focus {
    outline: none;
    border-color: #3498db;
}

button {
    background-color: #3498db;
    color: white;
    border: none;
    padding: 12px 25px;
    font-size: 1em;
    font-weight: 600;
    border-radius: 8px;
    cursor: pointer;
    transition: background-color 0.3s ease, transform 0.2s ease;
}

button:hover {
    background-color: #2980b9;
    transform: translateY(-2px);
}

#result {
    margin-top: 1.5em;
    font-size: 1.2em;
    font-weight: 600;
    min-height: 25px;
}

.palindrome {
    color: #27ae60; /* Green for palindrome */
}

.not-palindrome {
    color: #c0392b; /* Red for not a palindrome */
}

.error {
    color: #e74c3c;
}
```

```
  </style>
</head>
<body>

  <div class="container">
    <h1>Palindrome Checker</h1>
    <input type="text" id="stringInput" placeholder="e.g., A man, a plan, a canal: Panama">
    <button onclick="checkPalindrome()">Check String</button>
    <div id="result"></div>
  </div>

  <script>
    function checkPalindrome() {
      const originalString = document.getElementById('stringInput').value;
      const resultDiv = document.getElementById('result');

      // Clear previous result and classes
      resultDiv.textContent = '';
      resultDiv.classList.remove('palindrome', 'not-palindrome', 'error');

      if (originalString.trim() === '') {
        resultDiv.textContent = 'Please enter a string.';
        resultDiv.classList.add('error');
        return;
      }

      // 1. Sanitize the string:
      //    - Convert to lowercase to ignore case (e.g., 'Racecar' vs 'racecar').
      //    - Remove all non-alphanumeric characters (spaces, punctuation, etc.).
      const cleanedString = originalString.toLowerCase().replace(/[^a-z0-9]/g, '');

      // 2. Reverse the cleaned string:
      //    - .split('') turns the string into an array of characters.
      //    - .reverse() reverses that array.
      //    - .join('') joins the characters back into a string.
      const reversedString = cleanedString.split('').reverse().join('');

      // 3. Compare the cleaned string with its reversed version.
      if (cleanedString === reversedString) {
        resultDiv.textContent = `"${originalString}" is a palindrome!`;
        resultDiv.classList.add('palindrome');
      } else {
        resultDiv.textContent = `"${originalString}" is not a palindrome.`;
        resultDiv.classList.add('not-palindrome');
      }
    }
  </script>
```

```
</body>
</html>
```

Output:



6.4 Write a JavaScript program to test if the first character of a string is uppercase

Aim:

To test if the first character of a string is uppercase.

Source Code:

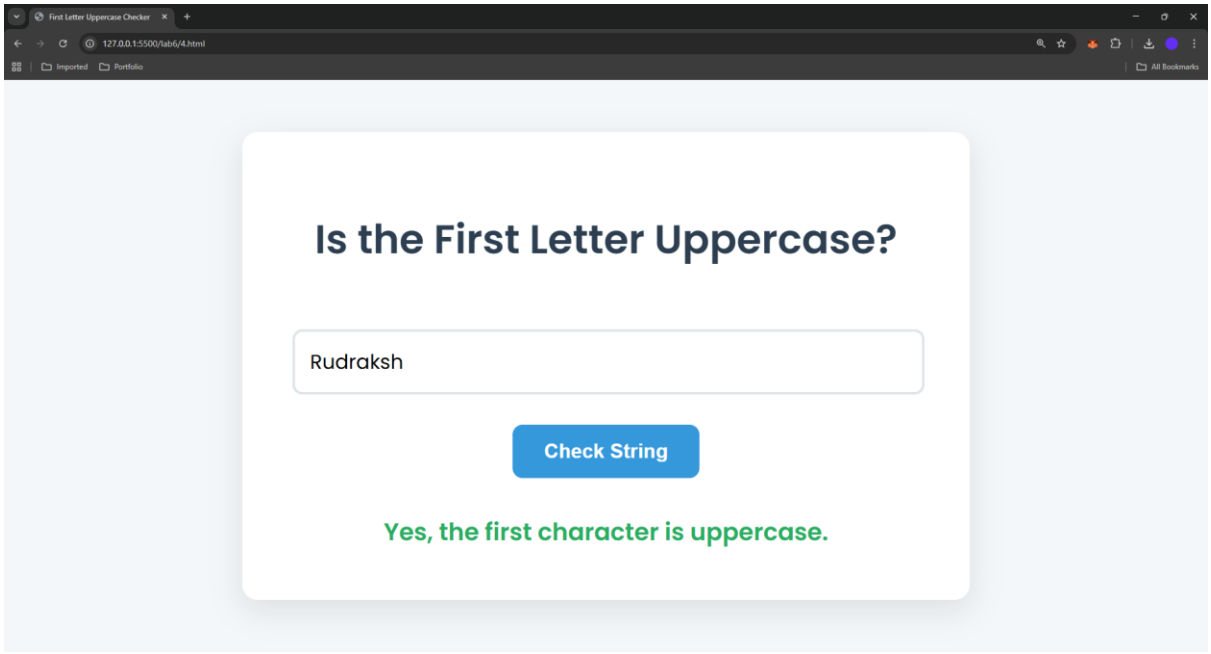
```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>First Letter Uppercase Checker</title>
   <!-- Google Font for a modern look -->
   <link rel="preconnect" href="https://fonts.googleapis.com">
   <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
   <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;600&display=swap"
rel="stylesheet">
   <style>
      body {
         font-family: 'Poppins', sans-serif;
         background-color: #f4f7f9;
         color: #333;
         display: flex;
         justify-content: center;
         align-items: center;
         min-height: 100vh;
         margin: 0;
      }

      .container {
         background-color: #ffffff;
         padding: 2.5em;
         border-radius: 12px;
         box-shadow: 0 8px 25px rgba(0, 0, 0, 0.1);
         width: 100%;
         max-width: 500px;
         text-align: center;
      }

      h1 {
         color: #2c3e50;
         margin-bottom: 1.5em;
      }

      #stringInput {
```

```
    width: calc(100% - 24px); /* Account for padding */
    padding: 12px;
    font-size: 1em;
    border: 2px solid #dfe6e9;
    border-radius: 8px;
    margin-bottom: 1.5em;
    font-family: 'Poppins', sans-serif;
    transition: border-color 0.3s ease;
}

#stringInput:focus {
    outline: none;
    border-color: #3498db;
}

button {
    background-color: #3498db;
    color: white;
    border: none;
    padding: 12px 25px;
    font-size: 1em;
    font-weight: 600;
    border-radius: 8px;
    cursor: pointer;
    transition: background-color 0.3s ease, transform 0.2s ease;
}

button:hover {
    background-color: #2980b9;
    transform: translateY(-2px);
}

#result {
    margin-top: 1.5em;
    font-size: 1.2em;
    font-weight: 600;
    min-height: 25px;
}

.uppercase {
    color: #27ae60; /* Green */
}

.not-uppercase {
    color: #c0392b; /* Red */
}

.error {
```

```
        color: #e74c3c;
    }
  </style>
</head>
<body>

  <div class="container">
    <h1>Is the First Letter Uppercase?</h1>
    <input type="text" id="stringInput" placeholder="Enter any string...">
    <button onclick="checkFirstChar()">Check String</button>
    <div id="result"></div>
  </div>

  <script>
    function checkFirstChar() {
      const inputString = document.getElementById('stringInput').value;
      const resultDiv = document.getElementById('result');

      // Clear previous result and classes
      resultDiv.textContent = '';
      resultDiv.classList.remove('uppercase', 'not-uppercase', 'error');

      if (inputString.trim() === '') {
        resultDiv.textContent = 'Please enter a string.';
        resultDiv.classList.add('error');
        return;
      }

      // Get the first character
      const firstChar = inputString[0];

      // Regular expression to check if the character is a letter
      const isLetter = /[a-zA-Z]/.test(firstChar);

      // Check if the first character is an uppercase letter
      if (isLetter && firstChar === firstChar.toUpperCase()) {
        resultDiv.textContent = 'Yes, the first character is uppercase.';
        resultDiv.classList.add('uppercase');
      } else if (isLetter) {
        resultDiv.textContent = 'No, the first character is not uppercase.';
        resultDiv.classList.add('not-uppercase');
      } else {
        resultDiv.textContent = 'The first character is not a letter.';
        resultDiv.classList.add('error');
      }
    }
  </script>
```

```
</body>
</html>
```

Output:



6.5 Write a JavaScript program to set the background colour of a paragraph

Aim:

To change the background color of a paragraph using JavaScript.

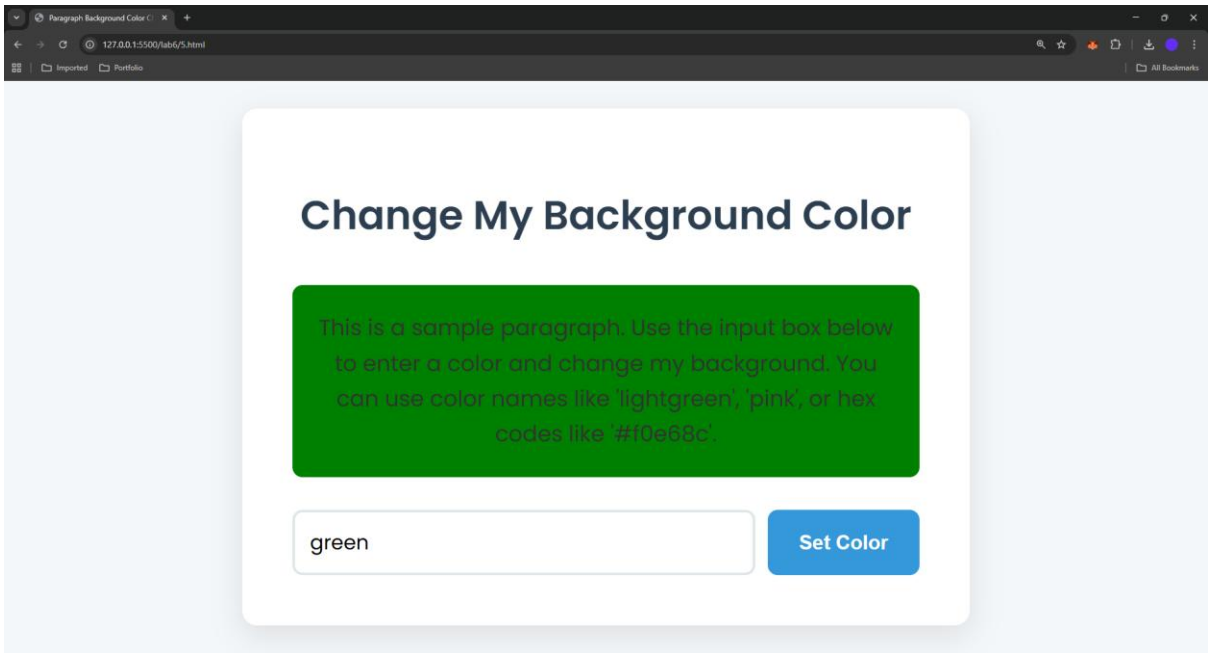
Source Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Paragraph Background Color Changer</title>
    <!-- Google Font for a modern look -->
    <link rel="preconnect" href="https://fonts.googleapis.com">
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
    <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;600&display=swap"
rel="stylesheet">
    <style>
        body {
            font-family: 'Poppins', sans-serif;
            background-color: #f4f7f9;
            color: #333;
            display: flex;
            justify-content: center;
            align-items: center;
            min-height: 100vh;
            margin: 0;
        }

        .container {
            background-color: #ffffff;
            padding: 2.5em;
            border-radius: 12px;
            box-shadow: 0 8px 25px rgba(0, 0, 0, 0.1);
            width: 100%;
            max-width: 500px;
            text-align: center;
        }

        h1 {
            color: #2c3e50;
            margin-bottom: 1em;
        }

        #textParagraph {
```

```
        background-color: #e9ecef;
        padding: 20px;
        border-radius: 8px;
        font-size: 1.1em;
        line-height: 1.6;
        margin-bottom: 1.5em;
        transition: background-color 0.4s ease;
    }

    .controls {
        display: flex;
        gap: 10px;
        justify-content: center;
    }

    #colorInput {
        flex-grow: 1;
        padding: 12px;
        font-size: 1em;
        border: 2px solid #dfe6e9;
        border-radius: 8px;
        font-family: 'Poppins', sans-serif;
        transition: border-color 0.3s ease;
    }

    #colorInput:focus {
        outline: none;
        border-color: #3498db;
    }

    button {
        background-color: #3498db;
        color: white;
        border: none;
        padding: 12px 25px;
        font-size: 1em;
        font-weight: 600;
        border-radius: 8px;
        cursor: pointer;
        transition: background-color 0.3s ease, transform 0.2s ease;
    }

    button:hover {
        background-color: #2980b9;
        transform: translateY(-2px);
    }

</style>
```

```
</head>
<body>

  <div class="container">
    <h1>Change My Background Color</h1>

    <p id="textParagraph">
      This is a sample paragraph. Use the input box below to enter a color and change my background.
You can use color names like 'lightgreen', 'pink', or hex codes like '#f0e68c'.
    </p>

    <div class="controls">
      <input type="text" id="colorInput" placeholder="Enter a color...">
      <button onclick="changeBackgroundColor()">Set Color</button>
    </div>
  </div>

  <script>
    function changeBackgroundColor() {
      // 1. Get the paragraph element by its ID.
      const paragraph = document.getElementById('textParagraph');

      // 2. Get the value from the color input field.
      const colorValue = document.getElementById('colorInput').value;

      // 3. Set the paragraph's background color style to the new value.
      //    The browser will automatically handle valid color names, hex codes, etc.
      if (colorValue) { // Only change color if the input is not empty
        paragraph.style.backgroundColor = colorValue;
      }
    }
  </script>

</body>
</html>
```

Output:


Change My Background Color
This is a sample paragraph. Use the input box below to enter a color and change my background. You can use color names like 'lightgreen', 'pink', or hex codes like '#f0e68c'.
green
Set Color

6.6 Write a JavaScript program to check if a given number is a mobile number using form

Aim:

To validate if the entered number is a mobile number.

Source Code:

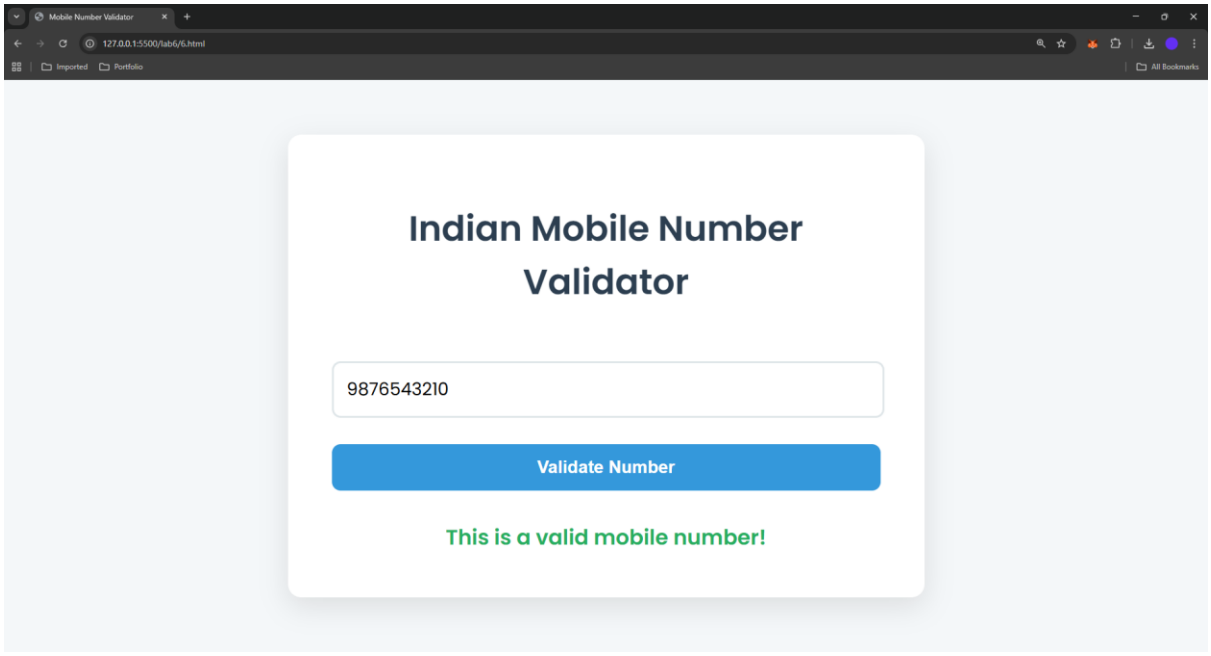
```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Mobile Number Validator</title>
   <!-- Google Font for a modern look -->
   <link rel="preconnect" href="https://fonts.googleapis.com">
   <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
   <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;600&display=swap"
rel="stylesheet">
   <style>
      body {
         font-family: 'Poppins', sans-serif;
         background-color: #f4f7f9;
         color: #333;
         display: flex;
         justify-content: center;
         align-items: center;
         min-height: 100vh;
         margin: 0;
      }

      .container {
         background-color: #ffffff;
         padding: 2.5em;
         border-radius: 12px;
         box-shadow: 0 8px 25px rgba(0, 0, 0, 0.1);
         width: 100%;
         max-width: 500px;
         text-align: center;
      }

      h1 {
         color: #2c3e50;
         margin-bottom: 1.5em;
      }

      #phoneInput {
```

```
    width: calc(100% - 24px); /* Account for padding */
    padding: 12px;
    font-size: 1em;
    border: 2px solid #dfe6e9;
    border-radius: 8px;
    margin-bottom: 1.5em;
    font-family: 'Poppins', sans-serif;
    transition: border-color 0.3s ease;
}

#phoneInput:focus {
    outline: none;
    border-color: #3498db;
}

button {
    background-color: #3498db;
    color: white;
    border: none;
    padding: 12px 25px;
    font-size: 1em;
    font-weight: 600;
    border-radius: 8px;
    cursor: pointer;
    width: 100%;
    transition: background-color 0.3s ease, transform 0.2s ease;
}

button:hover {
    background-color: #2980b9;
    transform: translateY(-2px);
}

#result {
    margin-top: 1.5em;
    font-size: 1.2em;
    font-weight: 600;
    min-height: 25px;
}

.valid {
    color: #27ae60; /* Green */
}

.invalid {
    color: #c0392b; /* Red */
}
```

```
    .error {
      color: #e74c3c;
    }
  </style>
</head>
<body>

  <div class="container">
    <h1>Indian Mobile Number Validator</h1>
    <form id="validatorForm" onsubmit="validatePhoneNumber(event)">
      <input type="tel" id="phoneInput" placeholder="e.g., +91 9876543210">
      <button type="submit">Validate Number</button>
    </form>
    <div id="result"></div>
  </div>

  <script>
    function validatePhoneNumber(event) {
      // Prevent the form from reloading the page
      event.preventDefault();

      const phoneInput = document.getElementById('phoneInput').value;
      const resultDiv = document.getElementById('result');

      // Clear previous result and classes
      resultDiv.textContent = '';
      resultDiv.classList.remove('valid', 'invalid', 'error');

      if (phoneInput.trim() === '') {
        resultDiv.textContent = 'Please enter a mobile number.';
        resultDiv.classList.add('error');
        return;
      }

      // This regular expression is designed for Indian mobile numbers.
      // It checks for:
      // ^               - Start of the string
      // (?:\+91|0)?     - An optional non-capturing group for "+91" or "0"
      // [6-9]           - The first digit must be between 6 and 9
      // \d{9}           - Followed by exactly 9 digits
      // $               - End of the string
      const regex = /^(?:\+91|0)?[6-9]\d{9}$/;

      // First, remove any spaces or hyphens from the input for easier validation
      const cleanedNumber = phoneInput.replace(/[\s-]/g, '');

      if (regex.test(cleanedNumber)) {
        resultDiv.textContent = 'This is a valid mobile number!';
```

```
            resultDiv.classList.add('valid');
        } else {
            resultDiv.textContent = 'This is not a valid mobile number.';
            resultDiv.classList.add('invalid');
        }
    }
  </script>

</body>
</html>
```

Output:



6.7 Design a student registration form and apply validation using external JavaScript

Aim:

To design a student registration form and apply validation using external JS file.

Source Code:

form.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Student Registration Form</title>
   <!-- Google Font -->
   <link rel="preconnect" href="https://fonts.googleapis.com">
   <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
   <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;600&display=swap"
rel="stylesheet">

   <style>
      /* General Body Styles */
      body {
         font-family: 'Poppins', sans-serif;
         background-color: #f4f7f9;
         color: #333;
         display: flex;
         justify-content: center;
         align-items: center;
         min-height: 100vh;
         padding: 2em 0;
      }

      /* Main Container */
      .container {
         background-color: #ffffff;
         padding: 2.5em;
         border-radius: 12px;
         box-shadow: 0 8px 30px rgba(0, 0, 0, 0.1);
         width: 100%;
         max-width: 550px;
      }

      h1 {
         color: #2c3e50;
```

```
    margin-bottom: 1.5em;
    text-align: center;
}

/* Form Grouping */
.form-group {
    margin-bottom: 1.5em;
}

label {
    display: block;
    margin-bottom: 8px;
    font-weight: 600;
    color: #555;
}

input[type="text"],
input[type="email"],
input[type="tel"],
input[type="password"],
select {
    width: 100%;
    padding: 12px;
    font-size: 1em;
    border: 2px solid #dfe6e9;
    border-radius: 8px;
    font-family: 'Poppins', sans-serif;
    transition: border-color 0.3s ease, box-shadow 0.3s ease;
    box-sizing: border-box; /* Important for consistent sizing */
}

input:focus,
select:focus {
    outline: none;
    border-color: #3498db;
    box-shadow: 0 0 5px rgba(52, 152, 219, 0.3);
}

/* Button Styling */
button {
    background-color: #3498db;
    color: white;
    border: none;
    padding: 14px 25px;
    font-size: 1.1em;
    font-weight: 600;
    border-radius: 8px;
    cursor: pointer;
```

```
            width: 100%;
            transition: background-color 0.3s ease, transform 0.2s ease;
        }

        button:hover {
            background-color: #2980b9;
            transform: translateY(-2px);
        }

        /* Error and Success Message Styling */
        .error-message {
            color: #c0392b;
            font-size: 0.9em;
            margin-top: 6px;
            min-height: 1em; /* Prevents layout shifts */
        }

        .success-message {
            color: #27ae60;
            font-weight: 600;
            text-align: center;
            margin-top: 1em;
        }
    </style>
</head>
<body>

    <div class="container">
        <h1>Student Registration</h1>
        <form id="registrationForm" novalidate>
            <!-- Full Name -->
            <div class="form-group">
                <label for="fullName">Full Name</label>
                <input type="text" id="fullName" name="fullName" placeholder="Enter your full name"
required>
                <div class="error-message" id="nameError"></div>
            </div>

            <!-- Email -->
            <div class="form-group">
                <label for="email">Email Address</label>
                <input type="email" id="email" name="email" placeholder="e.g., name@example.com"
required>
                <div class="error-message" id="emailError"></div>
            </div>

            <!-- Mobile Number -->
            <div class="form-group">
```

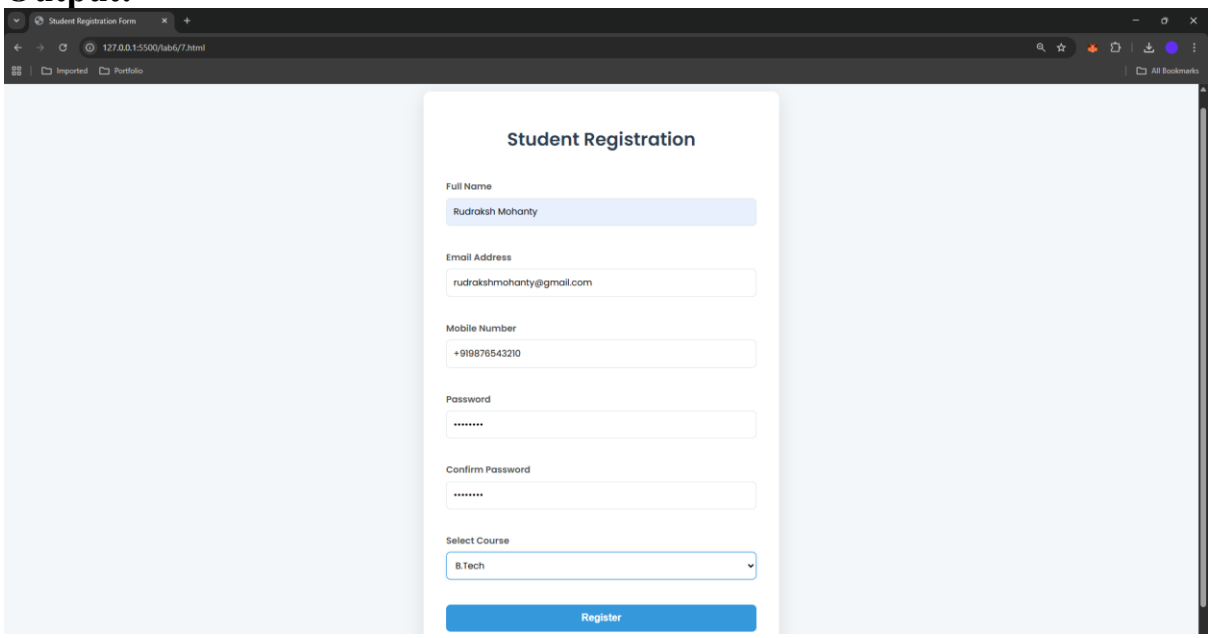
```
            <label for="mobile">Mobile Number</label>
            <input type="tel" id="mobile" name="mobile" placeholder="+91 XXXXXXXXXX" required>
            <div class="error-message" id="mobileError"></div>
        </div>

        <!-- Password -->
        <div class="form-group">
            <label for="password">Password</label>
            <input type="password" id="password" name="password" placeholder="Create a strong
password" required>
            <div class="error-message" id="passwordError"></div>
        </div>

        <!-- Confirm Password -->
        <div class="form-group">
            <label for="confirmPassword">Confirm Password</label>
            <input type="password" id="confirmPassword" name="confirmPassword" placeholder="Re-
enter your password" required>
            <div class="error-message" id="confirmPasswordError"></div>
        </div>

        <!-- Course Selection -->
        <div class="form-group">
            <label for="course">Select Course</label>
            <select id="course" name="course" required>
                <option value="" disabled selected>-- Choose a course --</option>
                <option value="btech">B.Tech</option>
                <option value="mtech">M.Tech</option>
                <option value="bca">BCA</option>
                <option value="mca">MCA</option>
            </select>
            <div class="error-message" id="courseError"></div>
        </div>

        <!-- Submit Button -->
        <button type="submit">Register</button>
        <div id="successMessage" class="success-message"></div>
    </form>
</div>

<script>
    document.addEventListener('DOMContentLoaded', () => {
        const form = document.getElementById('registrationForm');

        // Get all input fields
        const fullName = document.getElementById('fullName');
        const email = document.getElementById('email');
        const mobile = document.getElementById('mobile');
```

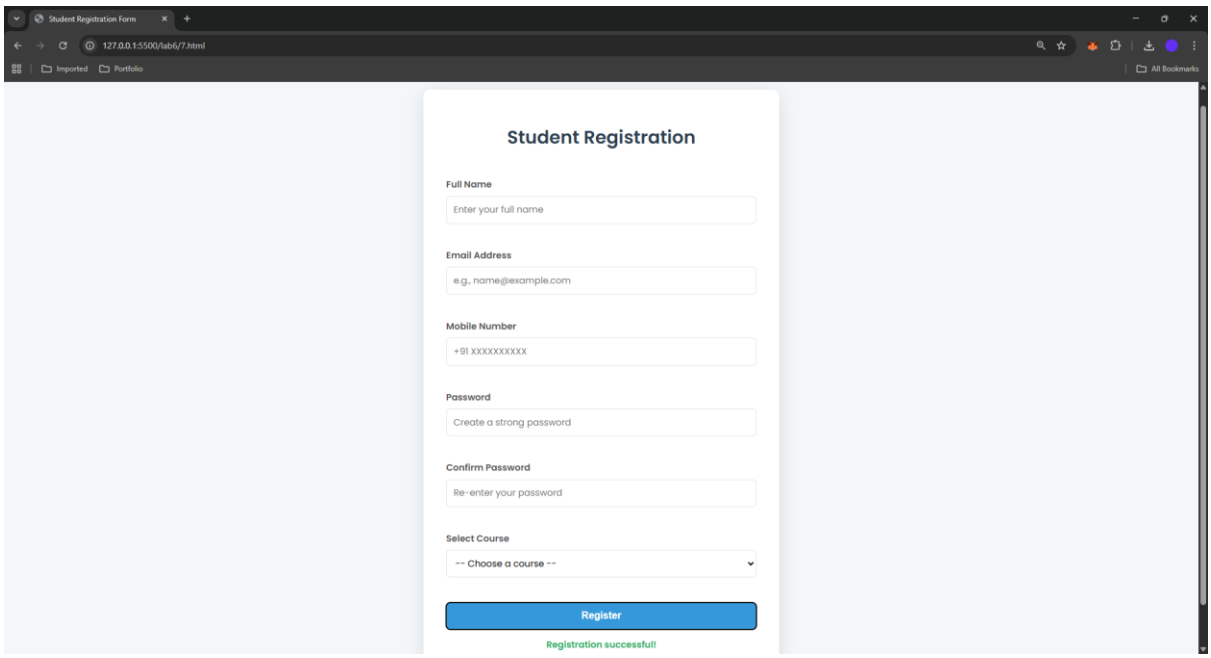
```
        const password = document.getElementById('password');
        const confirmPassword = document.getElementById('confirmPassword');
        const course = document.getElementById('course');

        // Get all error message containers
        const nameError = document.getElementById('nameError');
        const emailError = document.getElementById('emailError');
        const mobileError = document.getElementById('mobileError');
        const passwordError = document.getElementById('passwordError');
        const confirmPasswordError = document.getElementById('confirmPasswordError');
        const courseError = document.getElementById('courseError');
        const successMessage = document.getElementById('successMessage');

        form.addEventListener('submit', (event) => {
            // Prevent the form from submitting by default
            event.preventDefault();

            // Clear previous success message
            successMessage.textContent = '';

            // Perform validation
            const isNameValid = validateName();
            const isEmailValid = validateEmail();
            const isMobileValid = validateMobile();
            const isPasswordValid = validatePassword();
            const isConfirmPasswordValid = validateConfirmPassword();
            const isCourseValid = validateCourse();

            // If all validations pass, show success message
            if (isNameValid && isEmailValid && isMobileValid && isPasswordValid &&
isConfirmPasswordValid && isCourseValid) {
                successMessage.textContent = 'Registration successful!';
                form.reset(); // Optional: clear the form
            }
        });

        // --- Validation Functions ---

        function validateName() {
            if (fullName.value.trim() === '') {
                nameError.textContent = 'Full Name is required.';
                return false;
            }
            nameError.textContent = '';
            return true;
        }

        function validateEmail() {
```

```
    const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
    if (!emailRegex.test(email.value)) {
        emailError.textContent = 'Please enter a valid email address.';
        return false;
    }
    emailError.textContent = '';
    return true;
}

function validateMobile() {
    // Regex for Indian mobile numbers
    const mobileRegex = /^(?:\+91|0)?[6-9]\d{9}$/;
    const cleanedNumber = mobile.value.replace(/[\s-]/g, '');
    if (!mobileRegex.test(cleanedNumber)) {
        mobileError.textContent = 'Please enter a valid 10-digit Indian mobile number.';
        return false;
    }
    mobileError.textContent = '';
    return true;
}

function validatePassword() {
    if (password.value.length < 8) {
        passwordError.textContent = 'Password must be at least 8 characters long.';
        return false;
    }
    passwordError.textContent = '';
    return true;
}

function validateConfirmPassword() {
    if (confirmPassword.value !== password.value) {
        confirmPasswordError.textContent = 'Passwords do not match.';
        return false;
    }
    if (confirmPassword.value === '') {
        confirmPasswordError.textContent = 'Please confirm your password.';
        return false;
    }
    confirmPasswordError.textContent = '';
    return true;
}

function validateCourse() {
    if (course.value === '') {
        courseError.textContent = 'Please select a course.';
        return false;
    }
```

```
            courseError.textContent = '';
            return true;
        }
    });
  </script>
</body>
</html>
```

Output:

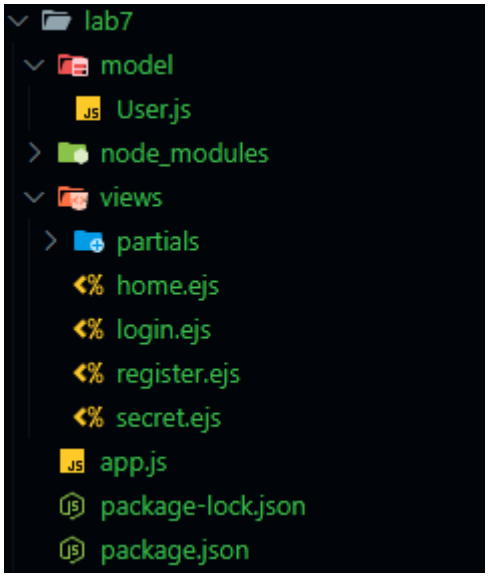




LAB EXPERIMENT - 7

The objective is to create a login form using Node.js, Express, MongoDB, and Passport for authentication, allowing users to sign up and log in to a website.

Project Structure:



Code:

- **User.js:**

```
const mongoose = require('mongoose');
const passportLocalMongoose = require('passport-local-mongoose');

const UserSchema = new mongoose.Schema({});

UserSchema.plugin(passportLocalMongoose);

module.exports = mongoose.model('User', UserSchema);
```

- **Home.ejs**

```
<%- include('partials/header') %>

<div class="p-5 mb-4 bg-white rounded-3 shadow-sm text-center">
  <div class="container-fluid py-5">
    <h1 class="display-5 fw-bold">Welcome to AuthApp</h1>
    <p class="fs-4">This is a beautiful, secure, and modern authentication system built with Node.js and Passport.</p>
    <a href="/register" class="btn btn-primary btn-lg" type="button">Get Started</a>
  </div>
</div>
```

```
<%- include('partials/footer') %>
```

- **Login.ejs:**

```
<%- include('partials/header') %>

<div class="row justify-content-center">
  <div class="col-md-6 col-lg-4">
    <div class="card shadow-sm">
      <div class="card-body">
        <h1 class="card-title text-center mb-4">Login</h1>
        <form action="/login" method="POST">
          <div class="mb-3">
            <label for="username" class="form-label">Username</label>
            <input type="text" name="username" class="form-control" id="username"
placeholder="Enter username" required>
          </div>
          <div class="mb-3">
            <label for="password" class="form-label">Password</label>
            <input type="password" name="password" class="form-control" id="password"
placeholder="Enter password" required>
          </div>
          <div class="d-grid">
            <button type="submit" class="btn btn-primary">Login</button>
          </div>
        </form>
        <div class="text-center mt-3">
          <a href="/register">Don't have an account? Sign Up</a>
        </div>
      </div>
    </div>
  </div>
</div>

<%- include('partials/footer') %>
```

- **Register.ejs:**

```
<%- include('partials/header') %>

<div class="row justify-content-center">
  <div class="col-md-6 col-lg-4">
    <div class="card shadow-sm">
      <div class="card-body">
        <h1 class="card-title text-center mb-4">Sign Up</h1>
        <form action="/register" method="POST">
          <div class="mb-3">
            <label for="username" class="form-label">Username</label>
```

```
                <input type="text" name="username" class="form-control" id="username"
placeholder="Choose a username" required>
            </div>
            <div class="mb-3">
                <label for="password" class="form-label">Password</label>
                <input type="password" name="password" class="form-control" id="password"
placeholder="Choose a password" required>
            </div>
            <div class="d-grid">
                <button type="submit" class="btn btn-primary">Sign Up</button>
            </div>
        </form>
        <div class="text-center mt-3">
            <a href="/login">Already have an account? Login</a>
        </div>
    </div>
  </div>
</div>

<%- include('partials/footer') %>
```

- **Secret.ejs:**

```
<%- include('partials/header') %>

<div class="p-5 mb-4 bg-white rounded-3 shadow-sm">
  <div class="container-fluid py-5">
    <h1 class="display-5 fw-bold">Welcome, <%= currentUser.username %>!</h1>
    <p class="fs-4">This is the secret page. You have successfully authenticated. ✨</p>
    <p>Only logged-in users are able to see this content. You can now access all the protected resources
of this application.</p>
    <a href="/logout" class="btn btn-danger" type="button">Logout</a>
  </div>
</div>

<%- include('partials/footer') %>
```

- **Header.ejs:**

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Beautiful Auth App</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
  <style>
    body {
      background-color: #f8f9fa;
```

```
    }
  </style>
</head>
<body>

<nav class="navbar navbar-expand-lg navbar-light bg-light shadow-sm">
  <div class="container">
    <a class="navbar-brand" href="/">AuthApp</a>
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarNav">
      <ul class="navbar-nav ms-auto">
        <% if(!currentUser){ %>
          <li class="nav-item">
            <a class="nav-link" href="/login">Login</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="/register">Sign Up</a>
          </li>
        <% } else { %>
          <li class="nav-item">
            <a class="nav-link" href="#">Signed In As <%= currentUser.username %></a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="/logout">Logout</a>
          </li>
        <% } %>
      </ul>
    </div>
  </div>
</nav>

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

- **Footer.ejs:**

```
</main>

<footer class="text-center text-muted mt-5">
  <p>&copy; 2025 AuthApp</p>
</footer>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>
</body>
</html>
```

Steps to run the application:

Step 1: To run the above code you should first have the mongoose server running If you do not have the mongoose folder setup follow this article. After setting up mongoDB start the server using following command mongod

Step 2: Type the following command in terminal of your project directory

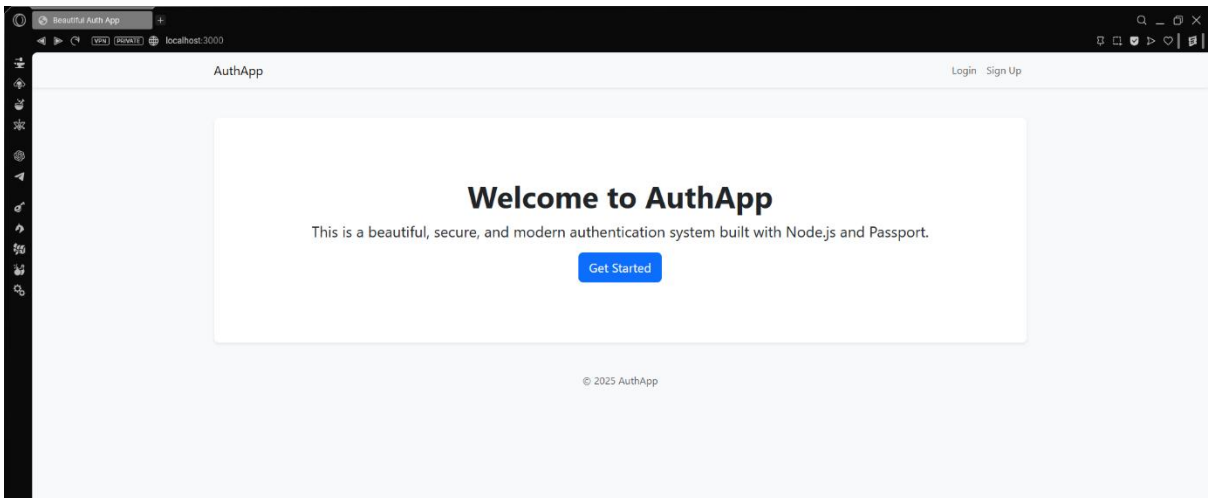
node app.js

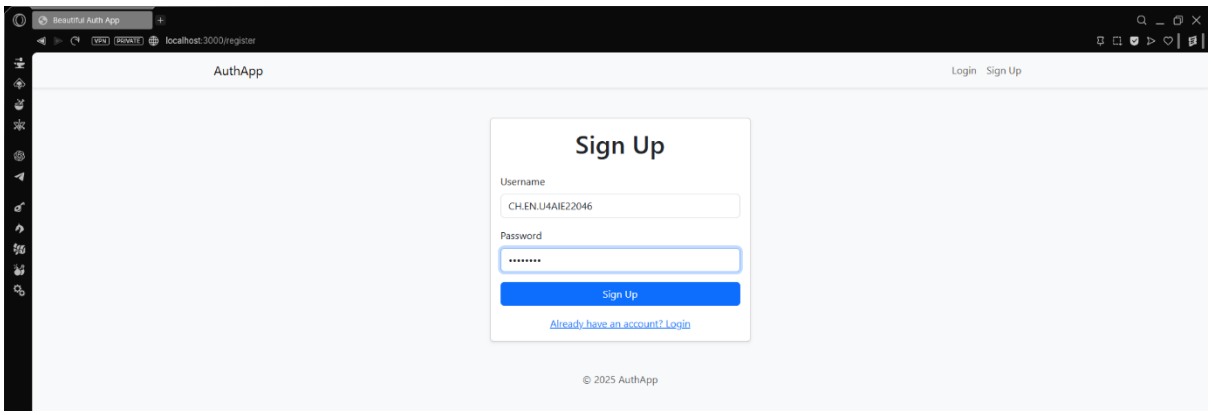
Step 3: Open your web browser and type the following address in the URL bar.

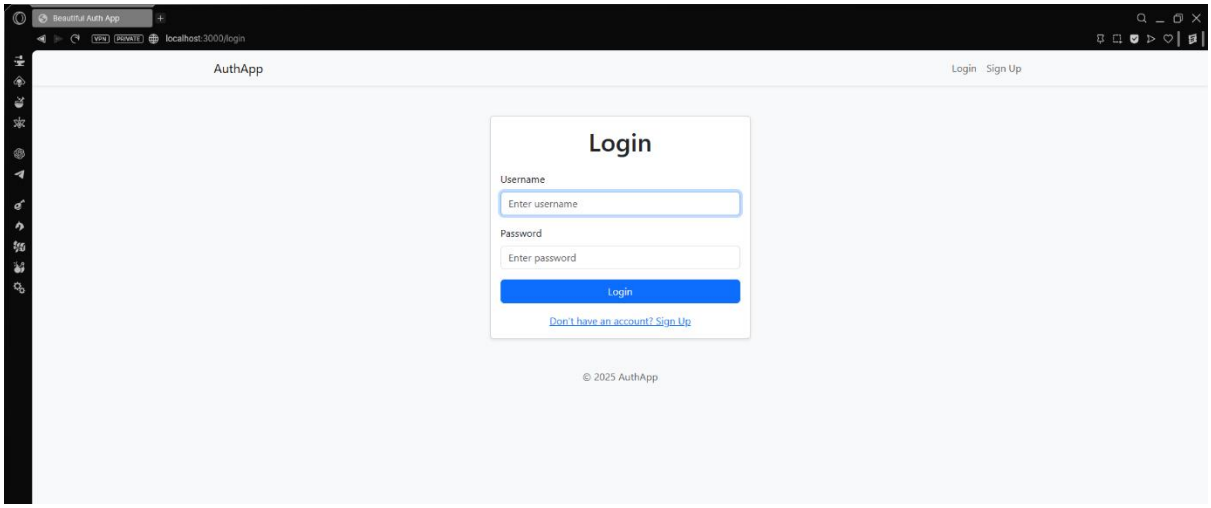
http://localhost:3000/

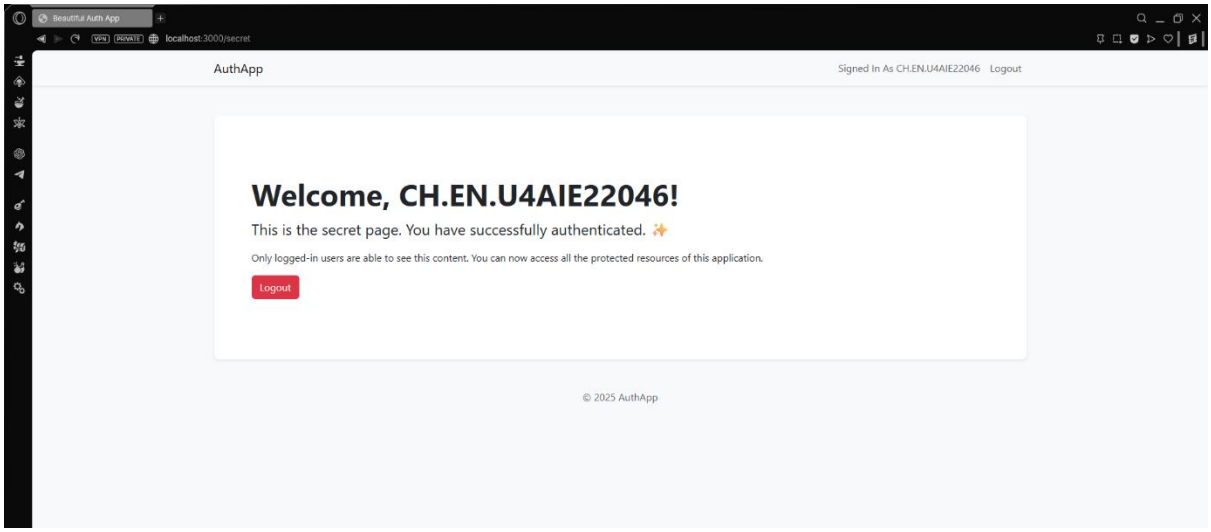
```
D:\CH.EN.U4AIE22046\my-first-git-repo\lab7>node app.js
Server is running on port 3000
Connected to DB!
```

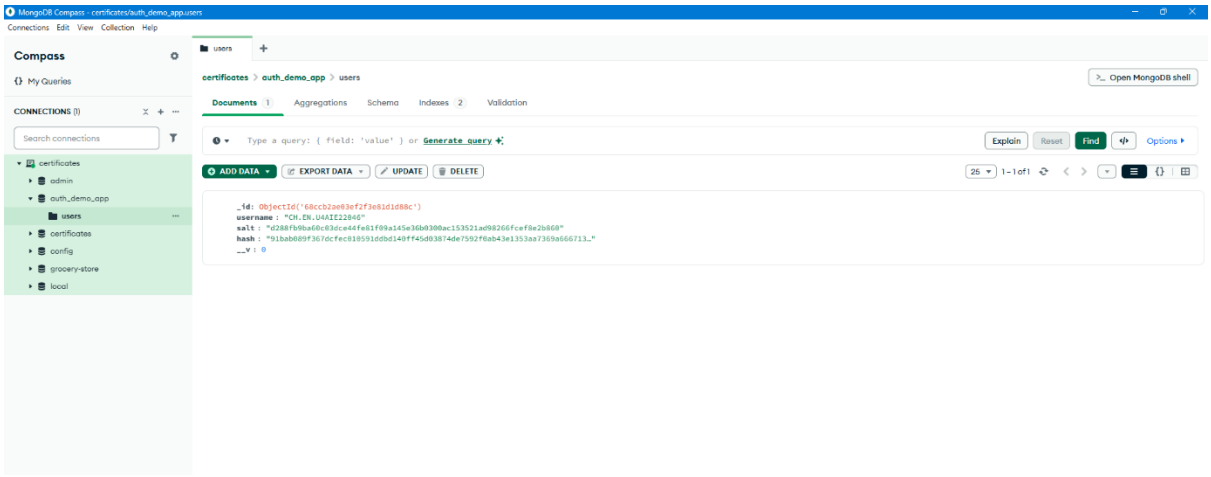
Output:






Beautiful Auth App
localhost:3000/login
AuthApp
Login
Sign Up
Login
Username
Enter username
Password
Enter password
Login
Don't have an account? Sign Up
© 2025 AuthApp


Beautiful Auth App
localhost:3000/secret
AuthApp
Signed In As CH.EN.U4AIE22046
Logout
Welcome, CH.EN.U4AIE22046!
This is the secret page. You have successfully authenticated. ✨
Only logged-in users are able to see this content. You can now access all the protected resources of this application.
Logout
© 2025 AuthApp


MongoDB Compass - certificates/auth_demo_app.users
Compass
My Queries
CONNECTIONS (1)
Search connections
certificates
admin
auth_demo_app
users
certificates
config
grocery-store
local
certificates > auth_demo_app > users
Open MongoDB shell
Documents 1
Aggregations
Schema
Indexes 2
Validation
Explain
Reset
Find
Options
ADD DATA
EXPORT DATA
UPDATE
DELETE

LAB EXPERIMENT - 8

Basic Login System with Node.js, Express, and MySQL

1. **Why create a login system with Node.js as opposed to PHP?**

   Node.js is a powerful open-source server environment that leverages JavaScript as its core scripting language. As more people become aware of Node.js, it is becoming increasingly popular in the development of web applications. Therefore, if you plan to develop applications for the future web, I highly suggest you enhance your knowledge with Node.js. Node.js's package manager (NPM) already has over 450,000 packages available for you to download. Those numbers alone indicate how fast it's growing.

2. **Setup & File Structure**

   - Create a new directory called **lab8**, which can be created anywhere on your environment.
   - Open the command line as administrator, and navigate to your new directory with the following command:
   - **cd c:\nodeprojects\nodelogin**
   - **Run the command: npm init - it will prompt us to enter a package name, enter: login.**
   - **When it prompts to enter the entry point, enter login.js.**

3. **Requirements**
   - MySQL Server >= 5.6
   - Node.js
   - Express - Install with command: **npm install express --save.**
   - Express Sessions - Install with command: **npm install express-session --save**.
   - MySQL for Node.js - Install with command: **npm install mysql –save.**

Code:

1. home.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Home</title>
    <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;500;600&displa
y=swap" rel="stylesheet">
    <link rel="stylesheet" href="/style.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/5.15.4/css/all.min.css">
</head>
```

```
<body>
    <div class="home-container">
        <div class="home-content">
            <i class="fas fa-check-circle success-icon"></i>
            <h1>Welcome, {{username}}!</h1>
            <p>You have successfully logged in to your account.</p>
            <a href="/logout" class="logout-button">Logout</a>
        </div>
    </div>
</body>
</html>
```

2. login.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Modern Login</title>
    <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;500;600&displa
y=swap" rel="stylesheet">
    <link rel="stylesheet" href="/style.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/5.15.4/css/all.min.css">
</head>
<body>
    <div class="login-container">
        <div class="login-form">
            <h2>Welcome Back!</h2>
            <p>Please enter your details to sign in.</p>
            <form action="/auth" method="post">
                <div class="input-group">
                    <i class="fas fa-user"></i>
                    <input type="text" name="username" placeholder="Username"
required>
                </div>
                <div class="input-group">
                    <i class="fas fa-lock"></i>
                    <input type="password" name="password"
placeholder="Password" required>
                </div>
                <button type="submit">Login</button>
            </form>
            <div class="bottom-text">
                <p>Don't have an account? <a href="/signup">Sign Up</a></p>
            </div>
        </div>
        <div class="login-art">
```

```
            <img
src="https://placehold.co/400x550/8e9aaf/ffffff?text=Secure+Login" alt="Login
Art">
        </div>
    </div>
</body>
</html>
```

3. Signup.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Modern Signup</title>
    <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;500;600&displa
y=swap" rel="stylesheet">
    <link rel="stylesheet" href="/style.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/5.15.4/css/all.min.css">
</head>
<body>
    <div class="login-container">
        <div class="login-form">
            <h2>Create Account</h2>
            <p>Join us today! It's free and only takes a minute.</p>
            <form action="/register" method="post">
                <div class="input-group">
                    <i class="fas fa-user"></i>
                    <input type="text" name="username" placeholder="Username"
required>
                </div>
                 <div class="input-group">
                    <i class="fas fa-envelope"></i>
                    <input type="email" name="email" placeholder="Email
Address" required>
                </div>
                <div class="input-group">
                    <i class="fas fa-lock"></i>
                    <input type="password" name="password"
placeholder="Password" required>
                </div>
                <button type="submit">Sign Up</button>
            </form>
            <div class="bottom-text">
                <p>Already have an account? <a href="/">Login</a></p>
            </div>
        </div>
```

```
        <div class="login-art">
            <img
src="https://placehold.co/400x550/5c677d/ffffff?text=Get+Started" alt="Signup
Art">
        </div>
    </div>
</body>
</html>
```

4. Login.js:

```
// Import required modules
const express = require('express');
const session = require('express-session');
const mysql = require('mysql');
const path = require('path');
const fs = require('fs'); // File System module to read HTML files

// Create a MySQL connection
const connection = mysql.createConnection({
    host: 'localhost',
    user: 'root', // Your MySQL username
    password: 'Rudrax@3017', // Your MySQL password
    database: 'lab8'
});

connection.connect((err) => {
    if (err) {
        console.error('Error connecting to database:', err);
        return;
    }
    console.log('Connected to the MySQL server.');
});

// Initialize the Express app
const app = express();

// Set up the session middleware
app.use(session({
    secret: 'your-secret-key', // Change this to a random secret key
    resave: true,
    saveUninitialized: true
}));

// Middleware to parse POST request bodies and serve static files
app.use(express.json());
app.use(express.urlencoded({ extended: true }));
app.use(express.static(path.join(__dirname, 'static')));
```

```
// --- ROUTES ---

// Route for the main login page
app.get('/', (req, res) => {
    res.sendFile(path.join(__dirname, 'login.html'));
});

// Route to serve the signup page
app.get('/signup', (req, res) => {
    res.sendFile(path.join(__dirname, 'signup.html'));
});

// Route to handle login authentication
app.post('/auth', (req, res) => {
    const { username, password } = req.body;
    if (username && password) {
        connection.query('SELECT * FROM accounts WHERE username = ? AND
password = ?', [username, password], (error, results) => {
            if (error) throw error;
            if (results.length > 0) {
                req.session.loggedin = true;
                req.session.username = username;
                res.redirect('/home');
            } else {
                res.send('Incorrect Username and/or Password!');
            }
            res.end();
        });
    } else {
        res.send('Please enter Username and Password!');
        res.end();
    }
});

// Route to handle new user registration
app.post('/register', (req, res) => {
    const { username, password, email } = req.body;
    if (username && password && email) {
        connection.query('SELECT * FROM accounts WHERE username = ?',
[username], (error, results) => {
            if (error) throw error;
            if (results.length > 0) {
                res.send('Username already exists!');
            } else {
                connection.query('INSERT INTO accounts (username, password,
email) VALUES (?, ?, ?)', [username, password, email], (err, result) => {
                    if (err) throw err;
                    req.session.loggedin = true;
```

```
                    req.session.username = username;
                    res.redirect('/home');
                });
            }
            res.end();
        });
    } else {
        res.send('Please fill out all fields!');
        res.end();
    }
});


// Route for the protected home page
app.get('/home', (req, res) => {
    if (req.session.loggedin) {
        // Read the home.html file
        fs.readFile(path.join(__dirname, 'home.html'), 'utf8', (err, data) =>
{
            if (err) {
                res.status(500).send("Error loading the page.");
                return;
            }
            // Replace placeholder with the actual username and send the file
            const personalizedHtml = data.replace('{{username}}',
req.session.username);
            res.send(personalizedHtml);
        });
    } else {
        res.redirect('/');
    }
});

// Route for logging out
app.get('/logout', (req, res) => {
    req.session.destroy((err) => {
        if (err) {
            return console.log(err);
        }
        res.redirect('/');
    });
});


// Start the server
const port = 3000;
app.listen(port, () => {
    console.log(`Server is running on http://localhost:${port}`);
});
```

5. Style.css:

```
/* Import Google Font */
@import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@400;500;600&display
=swap');

/* --- Base Styles --- */
body {
    font-family: 'Poppins', sans-serif;
    display: flex;
    justify-content: center;
    align-items: center;
    min-height: 100vh;
    margin: 0;
    background: linear-gradient(135deg, #ece9e6, #ffffff);
}

/* --- Login Container --- */
.login-container {
    display: flex;
    width: 900px;
    height: 550px;
    background: #fff;
    border-radius: 20px;
    box-shadow: 0 15px 30px rgba(0, 0, 0, 0.1);
    overflow: hidden;
}

/* --- Login Form Section --- */
.login-form {
    flex: 1;
    padding: 60px;
    display: flex;
    flex-direction: column;
    justify-content: center;
}

.login-form h2 {
    font-size: 28px;
    font-weight: 600;
    margin-bottom: 10px;
    color: #333;
}

.login-form p {
    color: #777;
    margin-bottom: 30px;
}
```

```
/* --- Input Groups --- */
.input-group {
    position: relative;
    margin-bottom: 25px;
}

.input-group i {
    position: absolute;
    left: 15px;
    top: 50%;
    transform: translateY(-50%);
    color: #aaa;
}

.input-group input {
    width: 100%;
    padding: 15px 15px 15px 45px; /* Add padding for the icon */
    border: 1px solid #ddd;
    border-radius: 10px;
    font-size: 16px;
    transition: border-color 0.3s;
}

.input-group input:focus {
    outline: none;
    border-color: #8e9aaf;
}

/* --- Login Button --- */
button {
    width: 100%;
    padding: 15px;
    border: none;
    border-radius: 10px;
    background-color: #5c677d;
    color: #fff;
    font-size: 18px;
    font-weight: 500;
    cursor: pointer;
    transition: background-color 0.3s, transform 0.2s;
}

button:hover {
    background-color: #4a5467;
    transform: translateY(-2px);
}

/* --- Login Art Section --- */
```

```
.login-art {
    flex: 1;
    display: flex;
    justify-content: center;
    align-items: center;
    background-color: #f0f2f5;
}

.login-art img {
    width: 100%;
    height: 100%;
    object-fit: cover;
}

/* --- Responsive Design --- */
@media (max-width: 920px) {
    .login-container {
        flex-direction: column;
        width: 90%;
        max-width: 450px;
        height: auto;
    }

    .login-art {
        display: none; /* Hide the image on smaller screens */
    }

    .login-form {
        padding: 40px;
    }
}
```

Mysql:

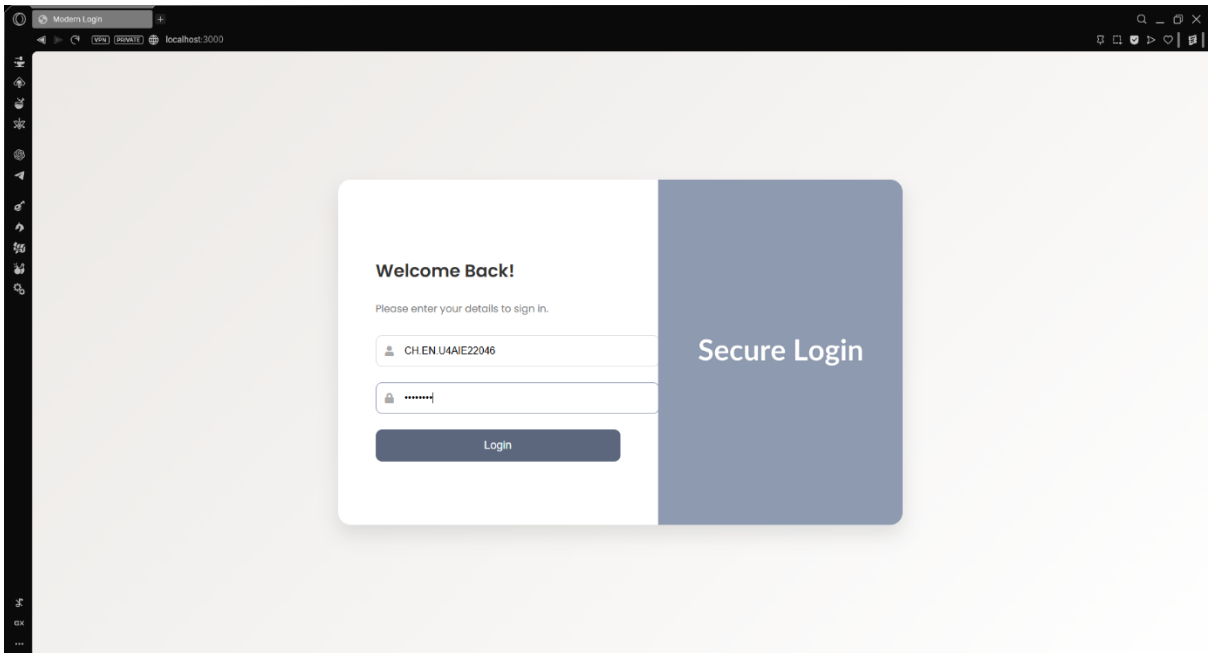
```
create database lab8;
use lab8;

CREATE TABLE IF NOT EXISTS `accounts` (
    `id` INT AUTO_INCREMENT PRIMARY KEY,
    `username` VARCHAR(50) NOT NULL,
    `password` VARCHAR(255) NOT NULL,
    `email` VARCHAR(100) NOT NULL
);

INSERT INTO `accounts` (`username`, `password`, `email`) VALUES ('test',
'test', 'test@example.com');
```

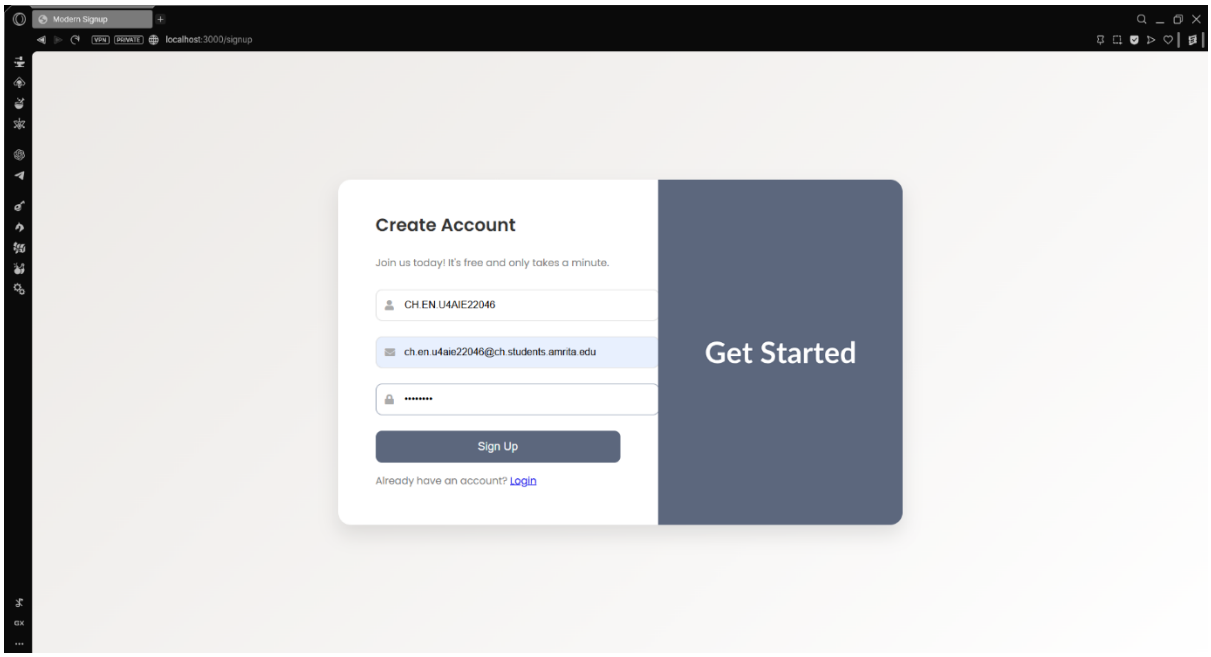
How to run the project:

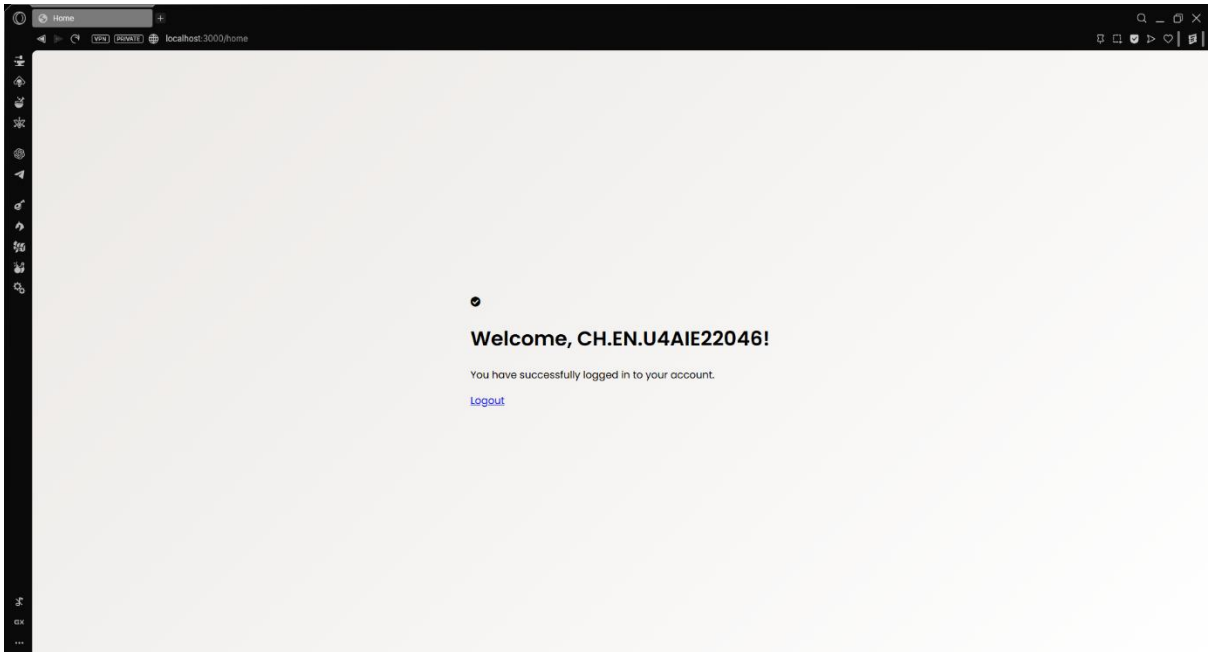
- ✓ Make sure Mysql is installed
- ✓ Then follow this:

```
D:\CH.EN.U4AIE22046\my-first-git-repo\lab8>node login.js
Server is running on http://localhost:3000
Connected to the MySQL server.
```

Output:






Home
localhost:3000/home
Welcome, CH.EN.U4AIE22046!
You have successfully logged in to your account.
Logout

LAB EXPERIMENT – 9

Create Live Editable Table with jQuery, PHP and MySQL

Live HTML table edit or inline table edit is a very user friendly feature that enable users to edit HTML table value directly by clicking on table cells. HTML table with jQuery and PHP, so the file structure for this experiment is following.

- index.php
- custom_table_edit.js
- live_edit.php

Codes:

Custom_table_edit.js:

```
$(document).ready(function(){
    $('#data_table').Tabledit({
        url: 'live_edit.php',
        editButton: false, // We will edit on cell click, so we don't need a
separate button
        deleteButton: true,
        hideIdentifier: false, // Show the ID column for reference
        columns: {
            identifier: [0, 'id'],
            editable: [
                [1, 'name'],
                [2, 'gender', '{"Male": "Male", "Female": "Female", "Other":
"Other"}'], // Creates a dropdown
                [3, 'age'],
                [4, 'designation'],
                [5, 'address']
            ]
        },
        onSuccess: function(data, textStatus, jqXHR) {
            console.log('Successfully updated:', data);
        },
        onFail: function(jqXHR, textStatus, errorThrown) {
            console.error('Update failed:', errorThrown);
        }
    });
});
```

Index.php:

```
<?php
// Database connection
$servername = "localhost";
$username = "root";
$password = "";
```

```
$dbname = "company";

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

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

// Fetch data from the database
$sql = "SELECT id, name, gender, age, designation, address FROM developers";
$result = $conn->query($sql);
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Live Editable Table</title>
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet">
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <div class="container my-5">
        <div class="card shadow-sm">
            <div class="card-header">
                <h2 class="card-title text-center">Developer Directory</h2>
                <p class="card-subtitle text-muted text-center">Click on a
cell to edit the content directly.</p>
            </div>
            <div class="card-body">
                <div class="table-responsive">
                    <table id="data_table" class="table table-striped table-
hover">
                        <thead class="table-dark">
                            <tr>
                                <th>Id</th>
                                <th>Name</th>
                                <th>Gender</th>
                                <th>Age</th>
                                <th>Designation</th>
                                <th>Address</th>
                            </tr>
                        </thead>
                        <tbody>
                            <?php
```

```
                            if ($result->num_rows > 0) {
                                while ($row = $result->fetch_assoc()) {
                                    echo "<tr>";
                                    echo "<td>" . $row["id"] . "</td>";
                                    echo "<td>" . $row["name"] . "</td>";
                                    echo "<td>" . $row["gender"] . "</td>";
                                    echo "<td>" . $row["age"] . "</td>";
                                    echo "<td>" . $row["designation"] .
"</td>";
                                    echo "<td>" . $row["address"] . "</td>";
                                    echo "</tr>";
                                }
                            } else {
                                echo "<tr><td colspan='6' class='text-
center'>No records found</td></tr>";
                            }
                            ?>
                        </tbody>
                    </table>
                </div>
            </div>
        </div>
    </div>

    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></scrip
t>
    <script src="https://cdn.jsdelivr.net/npm/jquery-
tabledit@1.0.0/jquery.tabledit.min.js"></script>
    <script src="custom_table_edit.js"></script>
</body>
</html>

<?php
$conn->close();
?>
```

Live_edit.php:

```
<?php
// Database connection
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "company";

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

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

```php
}

// Get the input data from the AJAX request
$input = filter_input_array(INPUT_POST);

if ($input['action'] === 'edit') {
    // Sanitize and prepare the update query
    $name = mysqli_real_escape_string($conn, $input['name']);
    $gender = mysqli_real_escape_string($conn, $input['gender']);
    $age = mysqli_real_escape_string($conn, $input['age']);
    $designation = mysqli_real_escape_string($conn, $input['designation']);
    $address = mysqli_real_escape_string($conn, $input['address']);
    $id = mysqli_real_escape_string($conn, $input['id']);

    $query = "
        UPDATE developers
        SET name = '".$name."',
            gender = '".$gender."',
            age = '".$age."',
            designation = '".$designation."',
            address = '".$address."'
        WHERE id = '".$id."'
    ";

    if (mysqli_query($conn, $query)) {
        echo json_encode(['status' => 'success']);
    } else {
        echo json_encode(['status' => 'error', 'message' =>
mysqli_error($conn)]);
    }

} else if ($input['action'] === 'delete') {
    // Sanitize and prepare the delete query
    $id = mysqli_real_escape_string($conn, $input['id']);

    $query = "
        DELETE FROM developers
        WHERE id = '".$id."'
    ";

    if (mysqli_query($conn, $query)) {
        echo json_encode(['status' => 'success']);
    } else {
        echo json_encode(['status' => 'error', 'message' =>
mysqli_error($conn)]);
    }
}
```

```php
$conn->close();

// Set the content type to application/json
header('Content-Type: application/json');
?>
```

Styles.css:

```css
@import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@400;500;600&display
=swap');

body {
    font-family: 'Poppins', sans-serif;
    background-color: #f8f9fa;
}

.container {
    max-width: 1000px;
}

.card-header {
    background-color: #343a40;
    color: white;
}

.card-title {
    margin-bottom: 0.25rem;
}

.table thead th {
    vertical-align: middle;
}

.table tbody tr:hover {
    background-color: #f1f1f1;
    cursor: pointer;
}

/* Tabledit Plugin Styles */
.tabledit-input {
    border-radius: 0.25rem;
    border: 1px solid #ced4da;
    padding: 0.375rem 0.75rem;
}

.tabledit-save-button, .tabledit-confirm-button {
    background-color: #28a745 !important;
    border: none !important;
}
```

```
.tabledit-edit-button, .tabledit-delete-button {
    background-color: #007bff !important;
    border: none !important;
}

.tabledit-delete-button {
     background-color: #dc3545 !important;
}

.tabledit-toolbar-btn {
    margin: 0 2px !important;
}
```

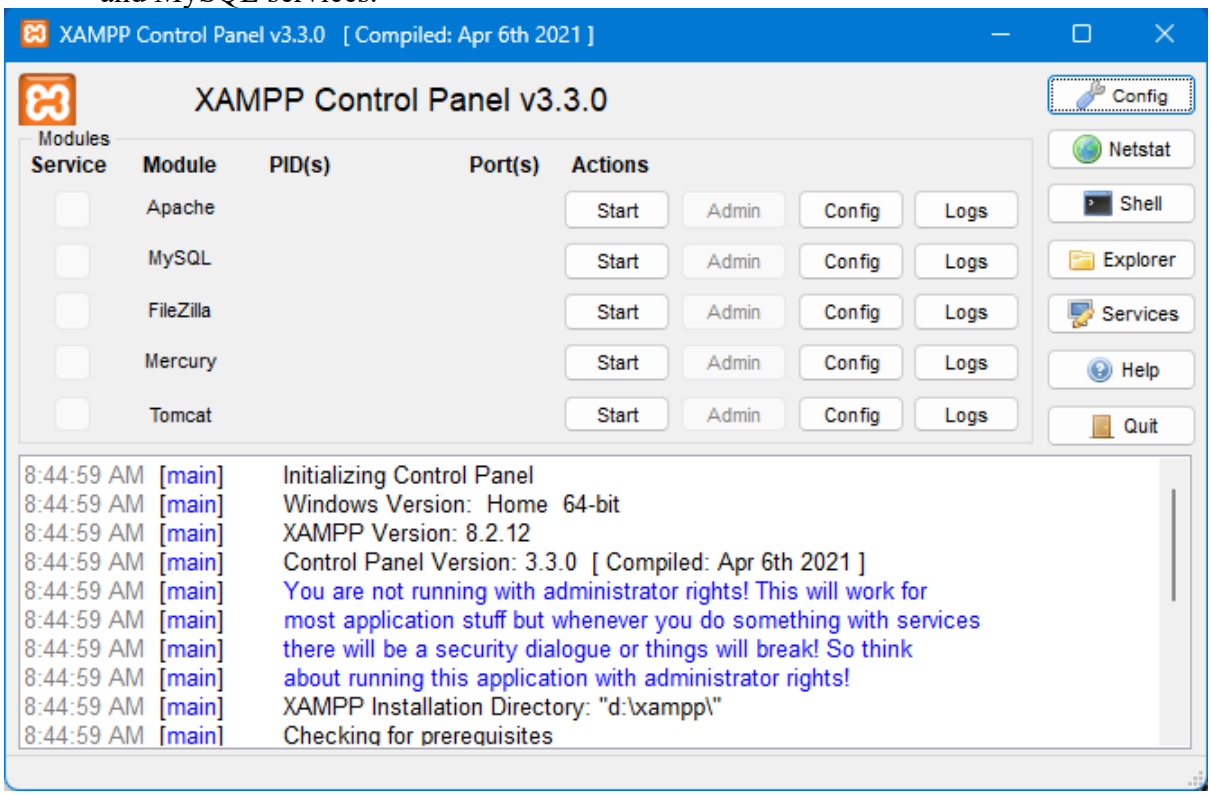
Sql:

```
-- Create a new database named 'lab9'
CREATE DATABASE IF NOT EXISTS lab9;

-- Use the 'company' database
USE lab9;

-- Create the 'developers' table
CREATE TABLE `developers` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(255) NOT NULL,
  `gender` varchar(50) NOT NULL,
  `age` int(11) NOT NULL,
  `designation` varchar(255) NOT NULL,
  `address` varchar(255) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- Insert some sample data into the table
INSERT INTO `developers` (`id`, `name`, `gender`, `age`, `designation`,
`address`) VALUES
(1, 'John Smith', 'Male', 30, 'Web Developer', '123 Main St, New York'),
(2, 'Jane Doe', 'Female', 28, 'UI/UX Designer', '456 Market St, San
Francisco'),
(3, 'Peter Jones', 'Male', 35, 'Project Manager', '789 Oak Ave, Chicago'),
(4, 'Mary Johnson', 'Female', 25, 'Frontend Developer', '321 Pine St, Los
Angeles');
```

How to run the experiment:

Step 1- Set Up Your Local Server:

1. **Install XAMPP:** Download and install XAMPP from the official website. It's a free package that includes Apache (the web server), MySQL (the database), and PHP.
2. **Start Apache and MySQL:** Open the XAMPP Control Panel and start the Apache and MySQL services.



Step 2- Place the Project Files:

1. Navigate to the XAMPP installation directory on your computer (usually C:\xampp).
2. Find the web server's root folder, which is named **htdocs**.
3. Inside **htdocs**, create a new folder.
4. Place all the project files (index.php, live_edit.php, custom_table_edit.js, and style.css) inside this new folder.

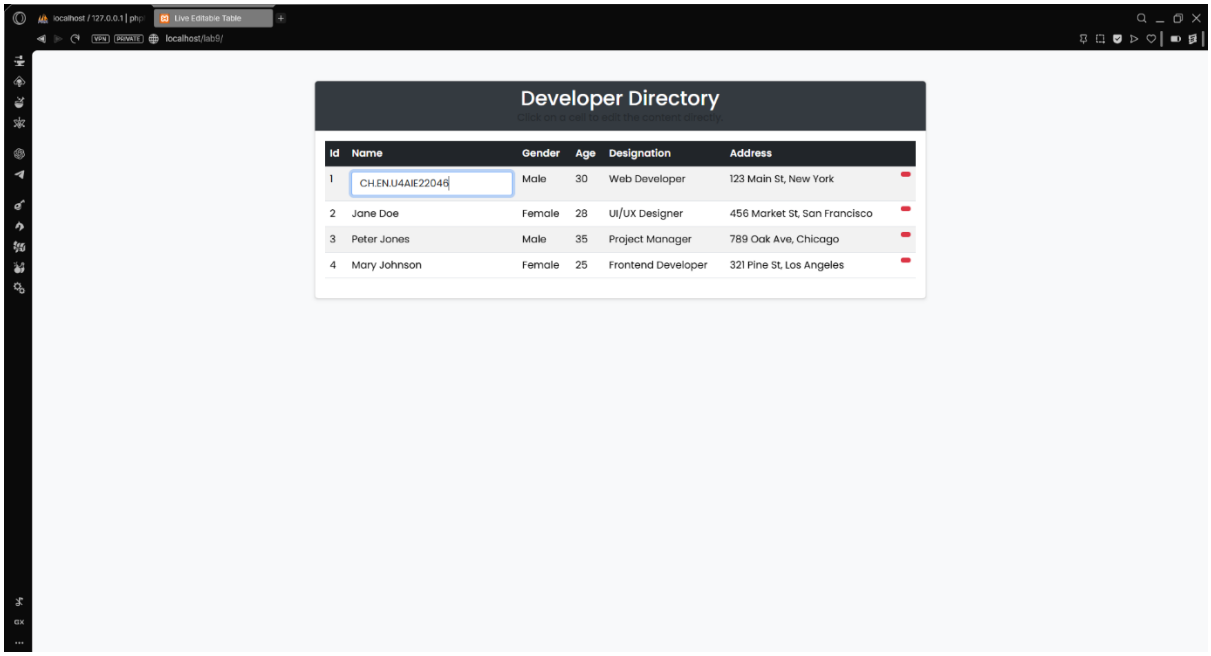
   C:\xampp\htdocs\live-table

Step 3- Create the Database:

1. Open your web browser and go to **http://localhost/phpmyadmin/.**
2. Click on the "SQL" tab at the top.
3. Copy the entire content of the db.sql file I provided earlier, paste it into the SQL query box, and click the "Go" button. This will create the company database, the developers table, and insert the sample data.

Step 4- Run the Application:

1. Open a new tab in your web browser.
2. Navigate to the project's URL. The URL is **http://localhost/** followed by the name of the folder you created in **htdocs**.
   http://localhost/folder_name/


Live Editable Table
localhost/lab9/
Developer Directory
Id
Name
Gender
Age
Designation
Address
1
CH.EN.U4AIE22046
Male
30
Web Developer
123 Main St, New York
2
Jane Doe
Female
28
UI/UX Designer
456 Market St, San Francisco
3
Peter Jones
Male
35
Project Manager
789 Oak Ave, Chicago
4
Mary Johnson
Female
25
Frontend Developer
321 Pine St, Los Angeles

LAB EXPERIMENT – 10

How to Submit AJAX Forms with JQuery.

Introduction How to Submit AJAX Forms with JQuery:

jQuery can be paired with form submission to handle validation. This has the benefit of providing users with feedback on any errors in their input.

Prerequisites:

- PHP installed locally and are able to run the built-in web server.
- Some familiarity with selectors and methods from the jQuery library.
- Some familiarity with classes from the Bootstrap library.
- A code editor.
- A modern web browser.

Codes:

1. Style.css:

```
/* --- Global Styles --- */
body {
    font-family: 'Poppins', sans-serif;
    background-color: #f0f2f5;
    display: flex;
    align-items: center;
    justify-content: center;
    min-height: 100vh;
    padding: 2rem;
}

/* --- Form Card Styles --- */
.contact-form-card {
    border: none;
    border-radius: 1rem;
    box-shadow: 0 10px 30px rgba(0, 0, 0, 0.1);
    background: #ffffff;
    padding: 1.5rem;
}

.card-title {
    font-weight: 600;
    color: #333;
}

/* --- Input & Button Styles --- */
.form-control, .input-group-text {
    border-radius: 0.5rem;
    padding: 0.85rem 1rem;
```

```
}

.form-control:focus {
    border-color: #0d6efd;
    box-shadow: 0 0 0 0.25rem rgba(13, 110, 253, 0.25);
}

.input-group-text {
    background-color: #e9ecef;
    border: 1px solid #ced4da;
}

.btn-primary {
    padding: 0.85rem 1rem;
    font-weight: 500;
    border-radius: 0.5rem;
    transition: background-color 0.3s, transform 0.2s;
}

.btn-primary:hover {
    background-color: #0b5ed7;
    transform: translateY(-2px);
}

/* --- Form Messages Styles --- */
#form-messages {
    border-radius: 0.5rem;
    padding: 1rem;
    font-size: 1rem;
}
```

2. form-handler.js:

```
$(function() {
    // Get the form.
    var form = $('#ajax-contact');

    // Get the messages div.
    var formMessages = $('#form-messages');

    // Set up an event listener for the contact form.
    $(form).submit(function(event) {
        // Stop the browser from submitting the form.
        event.preventDefault();

        // Serialize the form data.
        var formData = $(form).serialize();

        // Submit the form using AJAX.
        $.ajax({
            type: 'POST',
```

```
            url: $(form).attr('action'),
            data: formData
        })
        .done(function(response) {
            // Make sure that the formMessages div has the 'success' class.
            $(formMessages).removeClass('alert alert-danger');
            $(formMessages).addClass('alert alert-success');

            // Set the message text.
            $(formMessages).text(response.message);

            // Clear the form.
            $('#name').val('');
            $('#email').val('');
            $('#message').val('');
        })
        .fail(function(data) {
            // Make sure that the formMessages div has the 'error' class.
            $(formMessages).removeClass('alert alert-success');
            $(formMessages).addClass('alert alert-danger');

            // Set the message text.
            if (data.responseJSON && data.responseJSON.message) {
                $(formMessages).text(data.responseJSON.message);
            } else {
                $(formMessages).text('Oops! An error occurred and your
message could not be sent.');
            }
        });
    });
});
```

3. process_form.php:

```
<?php

// Set the content type to application/json for the response
header('Content-Type: application/json');

// This will hold our response data
$response = array(
    'success' => false,
    'message' => ''
);

// Only process POST requests.
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // Get the form fields and remove whitespace.
    $name = trim($_POST["name"]);
    $email = trim($_POST["email"]);
    $message = trim($_POST["message"]);
```

```
    // --- Validation ---

    // Check that data was sent.
    if ( empty($name) || empty($message) || !filter_var($email,
FILTER_VALIDATE_EMAIL)) {
        // Set a 400 (bad request) response code and exit.
        http_response_code(400);
        $response['message'] = 'Oops! Please complete all fields and use a
valid email.';
        echo json_encode($response);
        exit;
    }

    // Set a 200 (okay) response code.
    http_response_code(200);
    $response['success'] = true;
    $response['message'] = 'Thank You! Your message has been sent.';

} else {
    // Not a POST request, set a 403 (forbidden) response code.
    http_response_code(403);
    $response['message'] = 'There was a problem with your submission, please
try again.';
}

// Echo the JSON response
echo json_encode($response);

?>
```

4. index.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>AJAX Form with jQuery & PHP</title>
    <!-- Bootstrap 5 CSS -->
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet">
    <!-- Font Awesome for Icons -->
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.4.0/css/all.min.css">
    <!-- Google Fonts -->
    <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;500;600&displa
y=swap" rel="stylesheet">
    <!-- Custom CSS -->
```

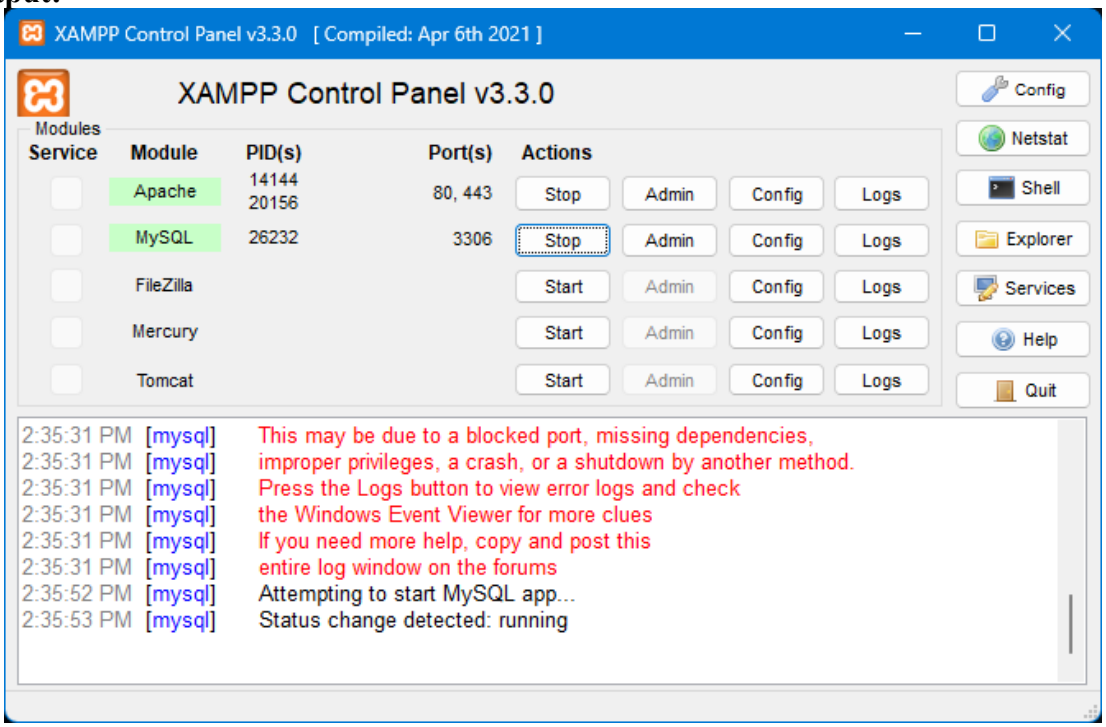
```
    <link rel="stylesheet" href="assets/css/style.css">
</head>
<body>

    <div class="container">
        <div class="row justify-content-center">
            <div class="col-lg-7 col-md-9">
                <div class="contact-form-card">
                    <div class="card-body">
                        <h2 class="card-title text-center mb-2">Get In
Touch</h2>
                        <p class="card-subtitle text-center text-muted mb-
4">We'd love to hear from you! Submit the form below.</p>

                        <!-- Form Messages Area -->
                        <div id="form-messages" class="mb-3"></div>

                        <!-- The Form -->
                        <form id="ajax-contact" method="post"
action="process_form.php">
                            <!-- Name Field -->
                            <div class="mb-3">
                                <label for="name" class="form-
label">Name</label>
                                <div class="input-group">
                                    <span class="input-group-text"><i
class="fas fa-user"></i></span>
                                    <input type="text" class="form-control"
id="name" name="name" placeholder="Your Name" required>
                                </div>
                            </div>

                            <!-- Email Field -->
                            <div class="mb-3">
                                <label for="email" class="form-label">Email
Address</label>
                                 <div class="input-group">
                                    <span class="input-group-text"><i
class="fas fa-envelope"></i></span>
                                    <input type="email" class="form-control"
id="email" name="email" placeholder="your.email@example.com" required>
                                </div>
                            </div>

                            <!-- Message Field -->
                            <div class="mb-3">
                                <label for="message" class="form-
label">Message</label>
```

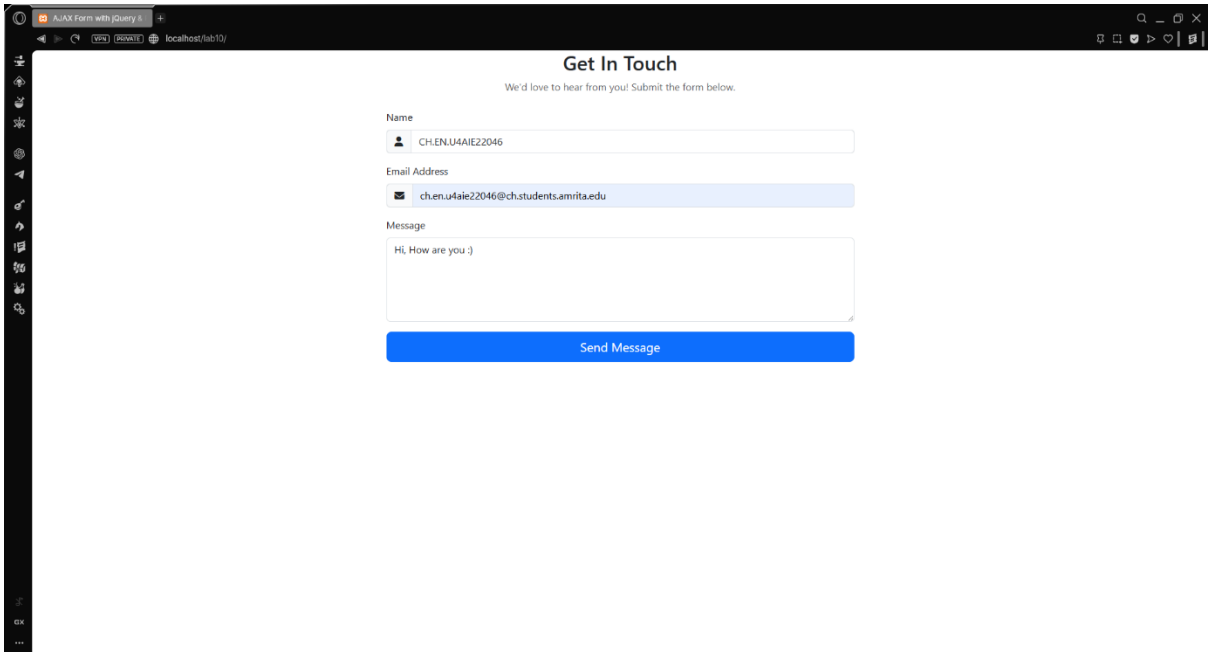
```
                                <textarea class="form-control" id="message"
name="message" rows="5" placeholder="Your message..." required></textarea>
                            </div>
                            <!-- Submit Button -->
                            <div class="d-grid">
                                <button type="submit" class="btn btn-primary
btn-lg">Send Message</button>
                            </div>
                        </form>
                    </div>
                </div>
            </div>
        </div>
    </div>
    <!-- jQuery -->
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <!-- Custom JS for AJAX -->
    <script src="assets/js/form-handler.js"></script>
</body>
</html>
```

How to Run This Project

1. **Set up a local server** (XAMPP).
2. Place the entire `lab11` folder inside your server's root directory (e.g., `C:\xampp\htdocs\`).
3. Open your browser and navigate to `http://localhost/lab11/`.

Output:




AJAX Form with jQuery &
localhost/lab10/
Get In Touch
We'd love to hear from you! Submit the form below.
Name
CH.EN.U4AIE22046
Email Address
ch.en.u4aie22046@ch.students.amrita.edu
Message
Hi, How are you :)
Send Message

LAB EXPERIMENT – 11

AJAX Database Operations

To perform this task we will create some sample entries in our MySQL database. Start the XAMPP control panel and start the MySQL service and open the MySQL Admin page.

Codes:

1. Style.css:

```
@import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@400;500;600&display
=swap');

body {
    font-family: 'Poppins', sans-serif;
    background-color: #f0f2f5;
}

.card-header {
    background-color: #343a40;
    color: white;
}

.card-title {
    font-weight: 600;
}

.table thead th {
    font-weight: 600;
}

.btn i {
    margin-right: 0.5rem;
}

.modal-header {
    background-color: #f8f9fa;
    border-bottom: 1px solid #dee2e6;
}

.form-control:focus {
    border-color: #86b7fe;
    box-shadow: 0 0 0 0.25rem rgba(13, 110, 253, 0.25);
}
```

2. Main.js:

```
$(document).ready(function() {
    // --- 1. Fetch and Display Users on Page Load ---
    fetchUsers();

    function fetchUsers() {
        $.ajax({
            url: 'api.php?action=fetch_all',
            type: 'GET',
            dataType: 'json',
            success: function(users) {
                let tableBody = $('#userTableBody');
                tableBody.empty(); // Clear existing rows
                users.forEach(user => {
                    tableBody.append(`
                        <tr>
                            <td>${user.id}</td>
                            <td>${user.name}</td>
                            <td>${user.email}</td>
                            <td>${user.phone}</td>
                            <td class="text-center">
                                <button class="btn btn-sm btn-info editBtn"
data-id="${user.id}"><i class="fas fa-edit"></i> Edit</button>
                                <button class="btn btn-sm btn-danger
deleteBtn" data-id="${user.id}"><i class="fas fa-trash"></i> Delete</button>
                            </td>
                        </tr>
                    `);
                });
            }
        });
    }

    // --- 2. Handle Add/Update Form Submission ---
    $('#userForm').on('submit', function(e) {
        e.preventDefault();
        let formData = $(this).serialize();
        let userId = $('#userId').val();
        let url = userId ? 'api.php' : 'api.php';
        formData += userId ? '&action=update_user' : '&action=add_user';

        $.ajax({
            url: url,
            type: 'POST',
            data: formData,
            dataType: 'json',
            success: function(response) {
                if(response.success) {
                    $('#userModal').modal('hide');
```

```
                fetchUsers(); // Refresh the table
            }
        }
    });
});

// --- 3. Handle 'Add User' Button Click ---
$('#addUserBtn').on('click', function() {
    $('#userForm')[0].reset();
    $('#userId').val('');
    $('#modalTitle').text('Add New User');
});

// --- 4. Handle 'Edit' Button Click ---
$('#userTableBody').on('click', '.editBtn', function() {
    let userId = $(this).data('id');
    $.ajax({
        url: `api.php?action=fetch_single&id=${userId}`,
        type: 'GET',
        dataType: 'json',
        success: function(user) {
            $('#modalTitle').text('Edit User');
            $('#userId').val(user.id);
            $('#userName').val(user.name);
            $('#userEmail').val(user.email);
            $('#userPhone').val(user.phone);
            $('#userModal').modal('show');
        }
    });
});

// --- 5. Handle 'Delete' Button Click ---
$('#userTableBody').on('click', '.deleteBtn', function() {
    let userId = $(this).data('id');
    if (confirm('Are you sure you want to delete this user?')) {
        $.ajax({
            url: 'api.php',
            type: 'POST',
            data: { action: 'delete_user', id: userId },
            dataType: 'json',
            success: function(response) {
                if (response.success) {
                    fetchUsers(); // Refresh the table
                }
            }
        });
    }
});
```

```
});
```

3. api.php:

```
<?php
// --- Database Connection ---
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "lab11";

$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// Set the header to return JSON
header('Content-Type: application/json');

// --- API Logic ---
$action = isset($_POST['action']) ? $_POST['action'] : (isset($_GET['action'])
? $_GET['action'] : '');

switch ($action) {
    case 'fetch_all':
        $sql = "SELECT * FROM users ORDER BY id DESC";
        $result = $conn->query($sql);
        $users = [];
        if ($result->num_rows > 0) {
            while ($row = $result->fetch_assoc()) {
                $users[] = $row;
            }
        }
        echo json_encode($users);
        break;

    case 'add_user':
        $name = $_POST['name'];
        $email = $_POST['email'];
        $phone = $_POST['phone'];
        $sql = "INSERT INTO users (name, email, phone) VALUES (?, ?, ?)";
        $stmt = $conn->prepare($sql);
        $stmt->bind_param("sss", $name, $email, $phone);
        if ($stmt->execute()) {
            echo json_encode(['success' => true, 'message' => 'User added
successfully!']);
        } else {
            echo json_encode(['success' => false, 'message' => 'Failed to add
user.']);
        }
```

```
        $stmt->close();
        break;

    case 'fetch_single':
        $id = $_GET['id'];
        $sql = "SELECT * FROM users WHERE id = ?";
        $stmt = $conn->prepare($sql);
        $stmt->bind_param("i", $id);
        $stmt->execute();
        $result = $stmt->get_result();
        $user = $result->fetch_assoc();
        echo json_encode($user);
        $stmt->close();
        break;

    case 'update_user':
        $id = $_POST['id'];
        $name = $_POST['name'];
        $email = $_POST['email'];
        $phone = $_POST['phone'];
        $sql = "UPDATE users SET name = ?, email = ?, phone = ? WHERE id = ?";
        $stmt = $conn->prepare($sql);
        $stmt->bind_param("sssi", $name, $email, $phone, $id);
        if ($stmt->execute()) {
            echo json_encode(['success' => true, 'message' => 'User updated
successfully!']);
        } else {
            echo json_encode(['success' => false, 'message' => 'Failed to
update user.']);
        }
        $stmt->close();
        break;

    case 'delete_user':
        $id = $_POST['id'];
        $sql = "DELETE FROM users WHERE id = ?";
        $stmt = $conn->prepare($sql);
        $stmt->bind_param("i", $id);
        if ($stmt->execute()) {
            echo json_encode(['success' => true, 'message' => 'User deleted
successfully!']);
        } else {
            echo json_encode(['success' => false, 'message' => 'Failed to
delete user.']);
        }
        $stmt->close();
        break;
```

```
    default:
        echo json_encode(['success' => false, 'message' => 'Invalid
action.']);
        break;
}

$conn->close();
?>
```

4. index.php:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>AJAX CRUD Operations</title>
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.4.0/css/all.min.css">
    <link rel="stylesheet" href="assets/css/style.css">
</head>
<body>

    <div class="container my-5">
        <div class="card shadow-sm">
            <div class="card-header d-flex justify-content-between align-
items-center">
                <h2 class="card-title mb-0">User Management</h2>
                <button class="btn btn-primary" data-bs-toggle="modal" data-
bs-target="#userModal" id="addUserBtn">
                    <i class="fas fa-plus"></i> Add New User
                </button>
            </div>
            <div class="card-body">
                <div class="table-responsive">
                    <table class="table table-hover align-middle">
                        <thead class="table-dark">
                            <tr>
                                <th>ID</th>
                                <th>Name</th>
                                <th>Email</th>
                                <th>Phone</th>
                                <th class="text-center">Actions</th>
                            </tr>
                        </thead>
                        <tbody id="userTableBody">
                            </tbody>
```

```
                    </table>
                </div>
            </div>
        </div>
    </div>

    <div class="modal fade" id="userModal" tabindex="-1">
        <div class="modal-dialog">
            <div class="modal-content">
                <form id="userForm">
                    <div class="modal-header">
                        <h5 class="modal-title" id="modalTitle">Add New
User</h5>
                        <button type="button" class="btn-close" data-bs-
dismiss="modal"></button>
                    </div>
                    <div class="modal-body">
                        <input type="hidden" id="userId" name="id">
                        <div class="mb-3">
                            <label for="userName" class="form-
label">Name</label>
                            <input type="text" class="form-control"
id="userName" name="name" required>
                        </div>
                        <div class="mb-3">
                            <label for="userEmail" class="form-
label">Email</label>
                            <input type="email" class="form-control"
id="userEmail" name="email" required>
                        </div>
                        <div class="mb-3">
                            <label for="userPhone" class="form-
label">Phone</label>
                            <input type="tel" class="form-control"
id="userPhone" name="phone" required>
                        </div>
                    </div>
                    <div class="modal-footer">
                        <button type="button" class="btn btn-secondary" data-
bs-dismiss="modal">Close</button>
                        <button type="submit" class="btn btn-primary">Save
Changes</button>
                    </div>
                </form>
            </div>
        </div>
    </div>
```

```
    <script src="https://code.jquery.com/jquery-3.7.1.min.js"></script>
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min
.js"></script>
    <script src="assets/js/main.js"></script>

</body>
</html>
```

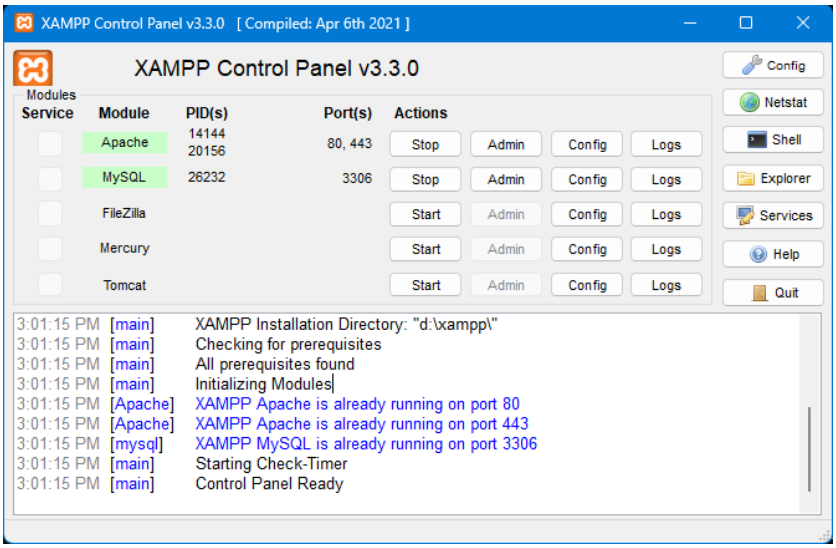
Sql:

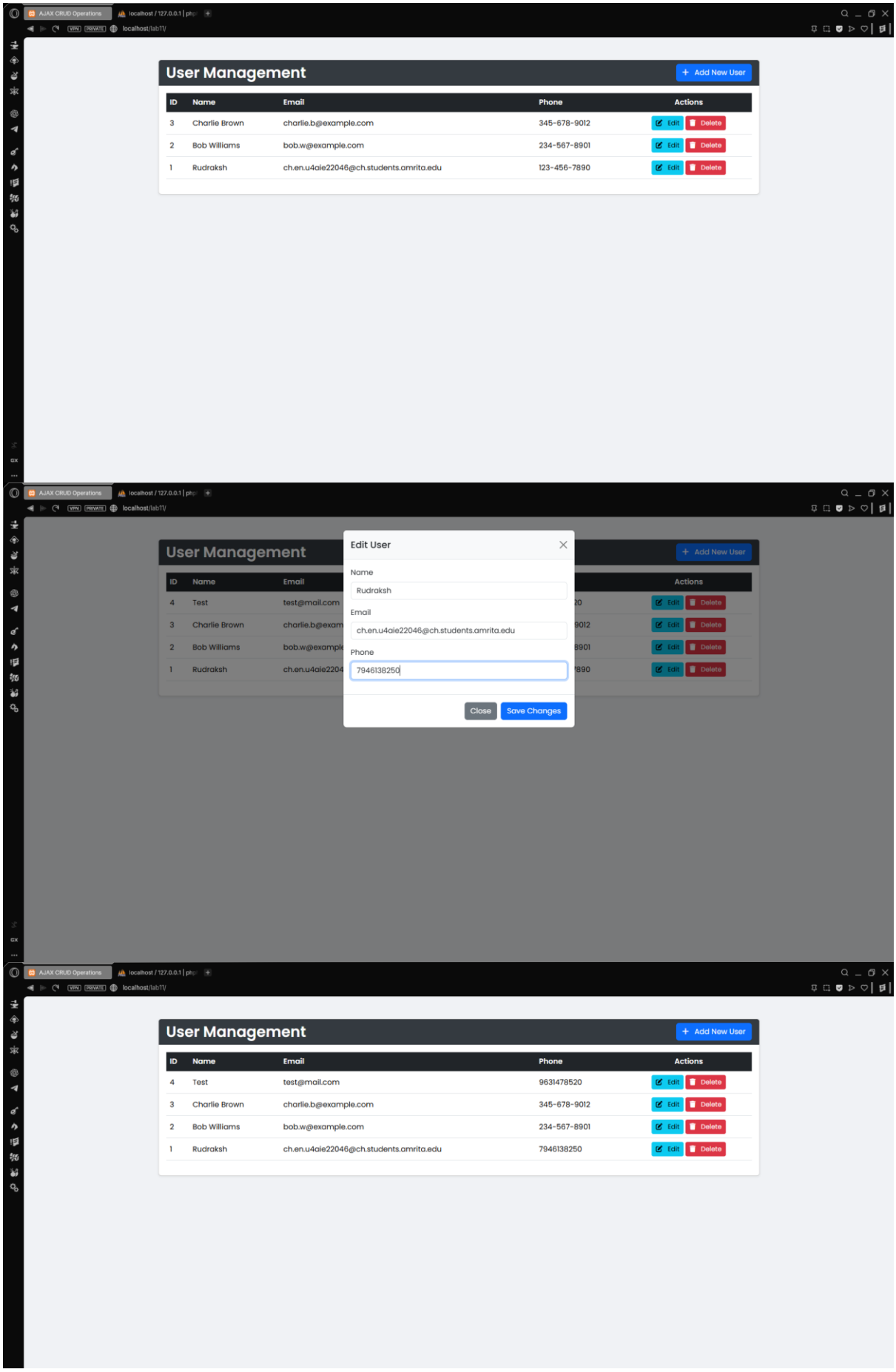
```
-- 1. Create a new database
CREATE DATABASE IF NOT EXISTS `lab11`;

-- 2. Use the new database
USE `lab11`;

-- 3. Create the 'users' table
CREATE TABLE IF NOT EXISTS `users` (
  `id` INT AUTO_INCREMENT PRIMARY KEY,
  `name` VARCHAR(100) NOT NULL,
  `email` VARCHAR(100) NOT NULL,
  `phone` VARCHAR(20) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- 4. Insert some sample data
INSERT INTO `users` (`name`, `email`, `phone`) VALUES
('Alice Johnson', 'alice.j@example.com', '123-456-7890'),
('Bob Williams', 'bob.w@example.com', '234-567-8901'),
('Charlie Brown', 'charlie.b@example.com', '345-678-9012');
```

How to Run This Project:

1. Set up a local server (XAMPP).
2. Place the entire folder inside your server's root directory (e.g., C:\xampp\htdocs\).
3. Create the database using the SQL script in XAMPP.
4. Open your browser and navigate to **http://localhost/lab11/**.

Output:




AJAX CRUD Operations
localhost / 127.0.0.1
localhost/lab11/
User Management
Add New User
ID
Name
Email
Phone
Actions
3
Charlie Brown
charlie.b@example.com
345-678-9012
2
Bob Williams
bob.w@example.com
234-567-8901
1
Rudraksh
ch.en.u4aie22046@ch.students.amrita.edu
123-456-7890
Edit
Delete
Edit User
Name
Rudraksh
Email
ch.en.u4aie22046@ch.students.amrita.edu
Phone
7946138250
Close
Save Changes
4
Test
test@mail.com
9631478520