**COLLEGE CODE : 9605**

**COLLEGE NAME : CAPE INSTITUTE OF TECHNOLOGY**

**DEPARTMENT: CSE 3Rd YEAR**

**STUDENT NM- ID :1F83228D01DE3E4B1E94965991A4B697**

**ROLL NO :960523104014**

**DATE : 27-10-2025**

**COMPLETED A PROJECT NAME AS PHASE -5**

**Technology Project Name : IBM – FE – E COMMERCE PRODUCT PAGE**

**SUBMITTED BY**

**NAME : Akash R**

**MOBILE NUMBER: 936155954**

**Project demonstration and**

**Documentation**

**Final demo walkthrough:**

* **Purpose:** Show how your project works from start to finish.
* **Format:** Can be a video recording, live demo, or slide walkthrough.
* **Key Points to Cover:**

* + Landing page and navigation
  + Product page layout (images, descriptions, price, ratings)
  + Add-to-cart / wishlist functionality (if implemented)
  + API integration (fetching product data)
  + Responsiveness on mobile & desktop
  + Any animations or interactive elements

**Final report:**

* **Introduction:**
  + Project title: IBM FE E-Commerce Product Page
  + Objective & scope of the project
  + HTML, CSS, JavaScript, React (if used)

**Tech Stack :**

* APIs (public/product API)
  + Any libraries (Bootstrap, Tailwind, Wiper’s, etc.)
* **Features Implemented:**
  + Product details display
  + Responsive design
  + Add to cart / quantity management
  + Reviews & ratings display (if any)
* **Learning Outcomes:**
  + Front-end development skills
  + API integration
  + UI/UX design principles
* **Conclusion & Future Scope**

**Screenshot API / documentation:**

* **Screenshots:**
  + Homepage
  + Product page
  + Cart / Checkout (if implemented)
  + Responsive views (mobile & desktop)

**API documentation:**

* Sample request & response
  + Integration method (fetch, axios, etc.)

**Challenges and solutions**

* **Example:**
  + **Challenge:** Product API was slow to respond

**Solution:** Implemented a loading spinner and error handling

* + **Challenge:** Responsive design breaking on mobile

**Solution:** Used CSS media queries & flexbox adjustments

* + **Challenge:** Cart functionality with dynamic data

**Solution:** Stored cart in localStorage and updated state dynamically

* **Screenshot API / documentation:**

<!DOCTYPE html>

<html Lang=”en”>

<head>

<meta charset=”UTF-8” />

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

<title>E-commerce Website</title>

<link href=<https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta3/dist/css/bootstrap.min.css>

<link href=<https://cdn.jsdelivr.net/npm/tailwindcss@2.2.19/dist/tailwind.min.css> rel=”stylesheet” />

<link rel=”stylesheet”

Href=<https://fonts.googleapis.com/css2?family=Material+Symbols+Outlined:opsz,wght,FILL,GRAD@24,400,0,0> />

<link

Href=<https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;600;700&family=Roboto:wght@300;400;700&display=swap>

Rel=”stylesheet” />

<style>

Body {

Font-family: ‘Poppins’, sans-serif;

Background-color: #f5f5f5;

}

Header {

Background-color: #fff;

Box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

Padding: 1rem;

Position: sticky;

Top: 0;

z-index: 999;

}

Nava a {

Color: #6b46c1;

Font-weight: 600;

Text-decoration: none;

Transition: color 0.3s ease;

}

Nava a:hover {

Color: #805ad5;

}

.hero {

Background-image: linear-gradient(to right, #6b46c1, #805ad5);

Color: #fff;

Padding: 4rem 2rem;

Text-align: center;

}

.hero h1 {

Font-size: 3rem;

Font-weight: 700;

Margin-bottom: 1rem;

}

.hero p {

Font-size: 1.25rem;

Margin-bottom: 2rem;

}

.btn-primary {

Background-color: #6b46c1;

Border-color: #6b46c1;

Font-weight: 600;

Transition: background-color 0.3s ease, border-color 0.3s ease;

}

.btn-primary:hover {

Background-color: #805ad5;

Border-color: #805ad5;

}

.categories, .featured-products {

Padding: 4rem 2rem;

}

.categories h2, .featured-products h2 {

Font-size: 2rem;

Font-weight: 700;

Margin-bottom: 2rem;

}

.category-item, .product-item {

Background-color: #fff;

Border-radius: 0.5rem;

Box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

Padding: 2rem;

Text-align: center;

Transition: transform 0.3s ease;

}

.category-item:hover, .product-item:hover {

Transform: translateY(-0.5rem);

}

.category-item img, .product-item img {

Object-fit: contain;

Margin-bottom: 1rem;

}

.category-item img {

Height: 100px;

Width: 100px;

}

.product-item img {

Height: 200px;

Width: 100%;

}

.category-item h3, .product-item h3 {

Font-size: 1.25rem;

Font-weight: 600;

Margin-bottom: 0.5rem;

}

.category-item p, .product-item p {

Color: #718096;

Font-size: 0.875rem;

}

.product-item .price {

Font-size: 1.25rem;

Font-weight: 600;

Color: #6b46c1;

}

.product-item div p {

Font-weight: bolder;

Text-align: left;

Margin-bottom: 0rem;

Color: #4F6F52;

}

Footer {

Background-color: #6b46c1;

Color: #fff;

Padding: 2rem;

Text-align: center;

}

.slider-container {

Display: flex;

Justify-content: space-between;

Border-radius: 0.5rem;

Box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

}

.slider-container img {

Width: 50%;

}

@media (max-width: 768px) {

.slider-container {

Flex-direction: column;

}

.slider-container img {

Width: 100%;

}

Nav {

Flex-direction: column;

Align-items: flex-start;

}

Nav a {

Margin-bottom: 0.5rem;

}

}

</style>

</head>

<body>

<header>

<nav class=”flex justify-between items-center flex-wrap”>

<a href=”#” class=”flex items-center space-x-2 mb-2 md:mb-0”>

<h3 class=”text-lg font-bold text-purple-800”>E-Commerce Website</h3>

</a>

<ul class=”flex space-x-4 text-purple-800 font-semibold flex-wrap”>

<li><a href=”/FAQ”>FAQ?</a></li>

<li><a href=”/myOrders”>Your orders</a></li>

<li><a href=”/cart\_item” id=”cartLink”>My Cart</a></li>

<li><a href=”/logout”>Logout</a></li>

</ul>

</nav>

</header>

<section class=”slider-container”>

<img loading=”lazy”

Src=<https://media.geeksforgeeks.org/wp-content/uploads/20250724181649377442/ecommerce-cart.webp>

Alt=”Slider 1”>

<img loading=”lazy” src=<https://media.geeksforgeeks.org/wp-content/uploads/20250724124044580202/kids_1.jpg>

Alt=”Slider 2”>

</section>

<section class=”hero”>

<h1>Welcome to our E-commerce Website</h1>

<p>Discover the best products and enjoy a seamless shopping experience.</p>

<button class=”btn btn-primary”>Shop Now</button>

</section>

<section class=”categories”>

<h2>Shop by Category</h2>

<div class=”grid grid-cols-1 sm:grid-cols-2 md:grid-cols-3 gap-4”>

<a href=”/men” class=”category-item”>

<img loading=”lazy”

Src=<https://media.geeksforgeeks.org/wp-content/uploads/20250724124356374694/male_1.jpg>

Alt=”Men’s Category”>

<h3>Men’s Wear</h3>

<p>Explore our wide range of men’s fashion.</p>

</a>

<a href=”/women” class=”category-item”>

<img loading=”lazy”

Src=<https://media.geeksforgeeks.org/wp-content/uploads/20250724125020503946/women_5.jpg>

Alt=”Women’s Category”>

<h3>Women’s Wear</h3>

<p>Discover the latest women’s fashion trends.</p>

</a>

<a href=”/kids” class=”category-item”>

<img loading=”lazy”

Src=<https://media.geeksforgeeks.org/wp-content/uploads/20250724125020976003/kids_2.jpg>

Alt=”Kids Category”>

<h3>Kids Wear</h3>

<p>Find stylish and comfortable options for your little ones.</p>

</a>

</div>

</section>

<section class=”featured-products”>

<h2>Featured Products</h2>

<div class=”grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-3 gap-4”>

<!—Example Product 🡪

<div class=”product-item” id=”Product\_1”>

<img loading=”lazy”

Src=<https://media.geeksforgeeks.org/wp-content/uploads/20250724124201503672/women_1.jpg>

Alt=”Product 1”>

<h3>Product 1</h3>

<p>Description: Dress 1</p>

<div>

<p class=”Discount”>Get 15% off</p>

<div class=”flex justify-between items-center”>

<span class=”price”>$200</span>

<button class=”btn btn-primary add-to-cart-btn”>Add to Cart</button>

</div>

</div>

</div>

<!—Repeat Products (Product\_2 to Product\_6)... same structure as above 🡪

</div>

</section>

<footer>

<p>&copy; 2024 E-commerce Website. All rights reserved.</p>

</footer>

<!—JavaScript 🡪

<script>

Let cart = JSON.parse(localStorage.getItem(‘cart’)) || [];

Document.querySelectorAll(‘.add-to-cart-btn’).forEach((btn) => {

Btn.addEventListener(‘click’, () => {

Const productCard = btn.closest(‘.product-item’);

Const product = productCard.id;

Const productName = productCard.querySelector(‘h3’).innerText;

Const productPrice = productCard.querySelector(‘.price’).innerText;

Const productImage = productCard.querySelector(‘img’).src;

Const product = {

Id: productId,

Name: productName,

Price: productPrice,

Image: productImage

};

Const exists = cart.find(item => item.id === product.id);

If (!exists) {

Cart.push(product);

localStorage.setItem(‘cart’, JSON.stringify(cart));

alert(`${product. Name} added to cart!`);

updateCartCount();

} else {

Alert(`${product.name} is already in your cart.`);

}

});

});

Function updateCartCount() {

Const cartCount = cart.length;

Let cartLink = Document.querySelectorAll(‘#cartLink’);

If (cartLink) {

cartLink. InnerText = `My Cart (${cartCount})`;

}

}

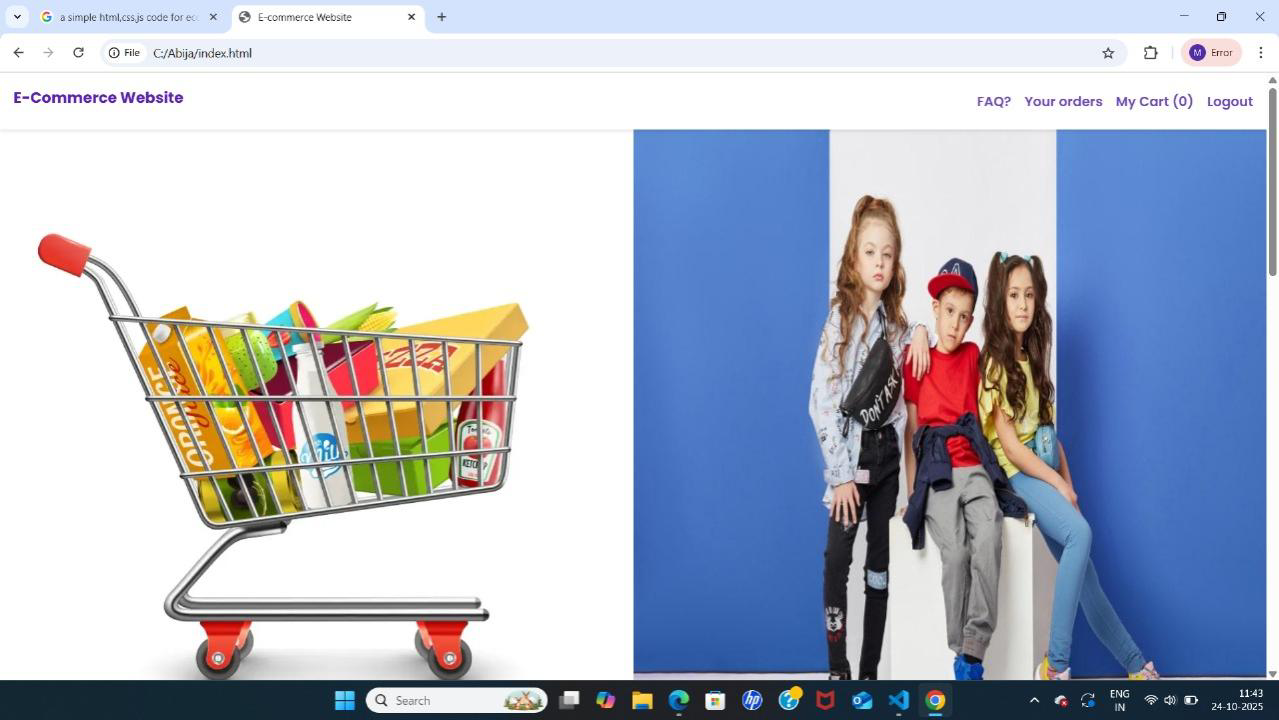
updateCartCount();

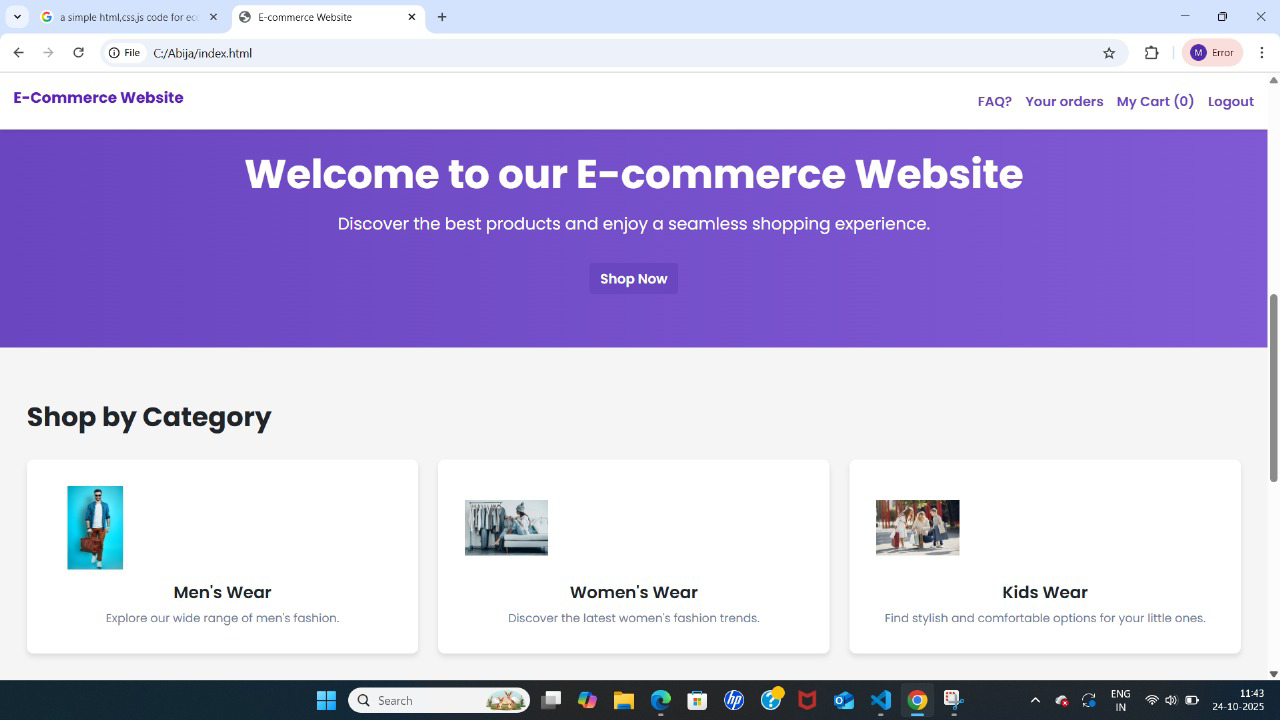
</script>

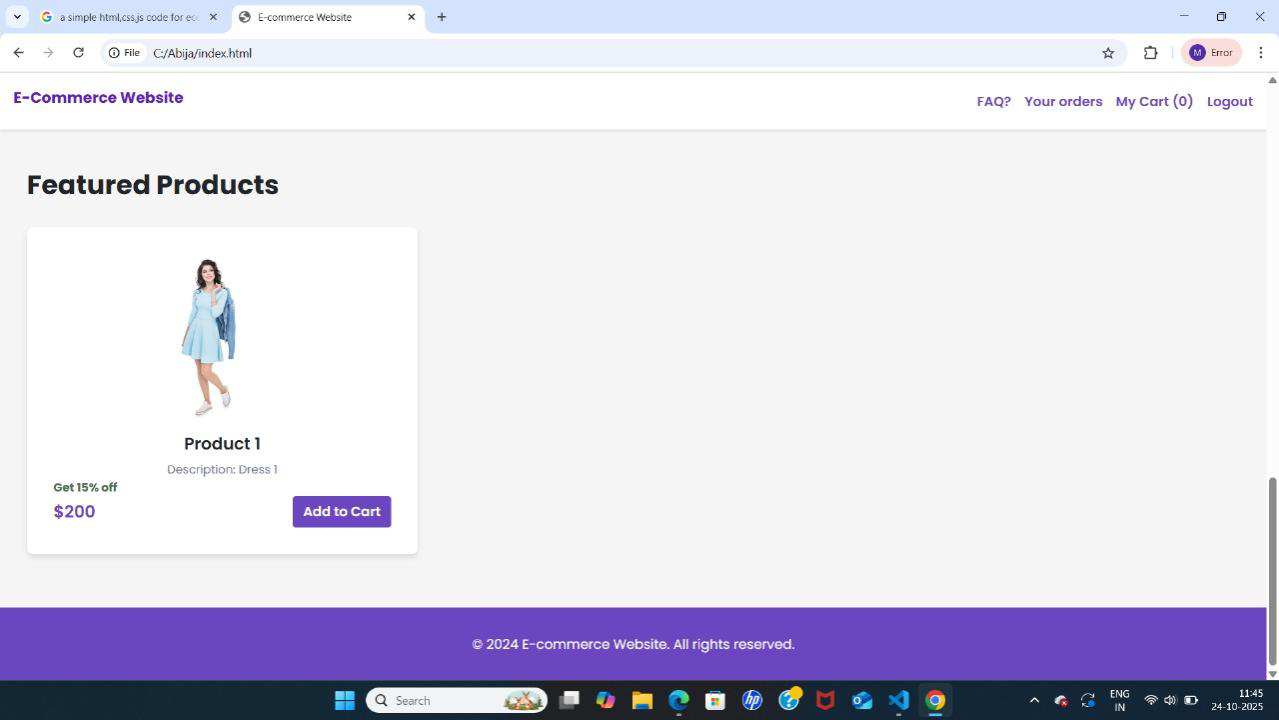
</body>

</html>

**OUTPUT :**







**Challenge 1: API Data Fetching Issues**

**Problem:**

**Slow API response times caused delays in displaying product data.**

**Sometimes API returned errors or incomplete data.**

**Solution:**

**Implemented a loading spinner to indicate data fetching.**

**Added error handling to display a friendly message if API fails.**

**Used synch /await with try-catch blocks for better control of API calls.**

**Challenge 2: Responsive Design Problems**

**Problem:**

**Product page layout broke on mobile or tablet devices.**

**Images and text overlapped or overflowed the container.**

**Solution:**

**Used CSS Flexbox and Grid for layout structure.**

**Added media queries for different screen widths (mobile, tablet, desktop).**

**Tested extensively on multiple devices and emulators to ensure proper responsiveness.**

**Challenge 3: Cart Functionality**

**Problem:**

**Managing cart state across pages was tricky.**

**Adding/removing items dynamically caused UI to freeze.**

**Solution:**

**Used React state and localStorage to persist cart data across sessions.**

**Implemented functions to add, remove, and update quantities efficiently.**

**Used conditional rendering to instantly update UI after cart changes.**

**Challenge 4: Image & Media Loading**

**Problem:**

**High-resolution product images caused slow loading.**

**Page felt heavy and unresponsive.**

**Solution:**

**Implemented lazy loading for images.**

**Optimized images by compressing without losing quality.**

**Used placeholder images while actual images loaded.**

**Challenge 5: UI/UX Design Consistency**

**Problem:**

**Different components had inconsistent spacing, fonts, and buttons.**

**Some elements looked misaligned or cluttered.**

**Solution:**

**Created a style guide for fonts, colors, and buttons.**

**Used CSS variables for consistent theming.**

**Regularly reviewed the design and aligned components using Flexbox/Grid.**

**Challenge 6: Navigation & Routing**

**Problem:**

**Navigating between pages (Home → Product → Cart) sometimes caused state loss.**

**Hard to maintain smooth transitions.**

**Solution:**

**Used React Router for smooth page navigation.**

**Preserved cart state using context API or localStorage.**

**Added back/forward navigation support for better UX.**

**Challenge 7: Testing & Debugging**

**Problem:**

**Bugs appeared intermittently, especially with dynamic data from APIs.**

**Hard to identify why certain features failed on specific devices.**

**Solution:**

**Used browser developer tools and console logging for debugging.**

**Conducted cross-browser testing on Chrome, Firefox, Edge, and mobile browsers.**

**Added unit testing for key functions (optional but useful).**

**GitHub Repository:**

<https://github.com/Ajiba441/E-commerce-product-page.git>

**Netify Deployment link :**

<https://app.netlify.com/teams/ajiba441/projects>