COLLEGE CODE : 8201

COLLEGE NAME: ARJ COLLEGE OF ENGINEERING AND TECHNOLOGY

DEPARTMENT : BE-COMPUTER SCIENCE AND ENGINEERING

STUDENT NM-ID: aut8201cse124

ROLL NO : 820123104032

DATE :21-10-2025

Completed the project named as PHASE-5

TECHNOLOGY PROJECT NAME : IBM-NJ-E-COMMERCE CART SYSTEM

SUBMITTED BY,

NAME : S.KANMANI

MOBILE NO: 9342119155

IBM-NJ-E-COMMERCE CART SYSTEM

PHASE 5- PROJECT DEMONSTRATION AND DOCUMENTATION

- *Final Walkthrough*
- *Project Report*
- *Screenshots/API Documentation*
- *Challenges and Solutions*
- *Github README and Setup guide*
- *Final Submission*

1) FINAL DEMO WALKTHROUGH:

Objective

The objective of the E-Commerce Cart System is to create a web-based platform that allows users to browse products, add them to a cart, and proceed to checkout with secure payment options. It streamlines the online shopping process for both users and administrators.

2) PROJECT REPORT:

1)Project overview:

- **Title:** E-Commerce Cart System
- **Developers:** [Your Name/Team Name]
- **Duration:** [Project Duration]

- **Supervisor:** [Mentor/Professor's Name]
- **Platform:** Web-based
- **Target Users:** Online shoppers and admin/store owners

2)Modules Covered:

- User Module
- Admin Module
- Product Catalog
- Cart and Checkout
- Payment Simulation
- Order Management

◆ 3. Key Features

🧒💻 User Side:

- User registration & login
- Browse products by category
- Add/remove items from the cart
- View cart summary
- Place orders and simulate payments
- View order history

🛠 Admin Side:

- Admin login
- Add/edit/delete product listings
- Manage categories
- View customer orders
- Update order status (Pending, Shipped, Delivered)
-

◆ 4. Technologies Used:

Layer	Technology
Frontend	HTML, CSS, JavaScript, Bootstrap
Backend	PHP / Node.js / Python (Django/Flask)
Database	MySQL / MongoDB / PostgreSQL
Tools Used	VS Code, XAMPP/Postman/GitHub

Layer	Technology
Optional APIs	Stripe/PayPal (Mock Payment), REST APIs
Hosting (Optional)	Heroku / Netlify / Firebase

◆ 5. Project Flow:

1. **Home Page** – Displays featured products and categories.
2. **User Authentication** – Register/Login functionality.
3. **Product Browsing** – View product details with "Add to Cart" option.
4. **Cart Page** – Shows selected products with quantity & total price.
5. **Checkout** – Enter address and simulate payment.
6. **Order Confirmation** – Displays order summary and tracking.

B. Admin Flow:

1. **Admin Login** – Secure login to admin dashboard.
2. **Product Management** – Add/edit/delete products and manage stock.
3. **Order Management** – View and update customer orders.
4. **Category Management** – Organize products by category.

◆ 6. Demo Walkthrough (Live Presentation Steps):

➢ Step 1: Launch Home Page

- Show the landing page with navigation bar and featured items.

➢ Step 2: User Sign Up / Login

- Demonstrate the registration and login process.

➢ Step 3: Browsing and Adding to Cart

- Browse through categories, open a product detail page, and add an item to the cart.

➢ Step 4: Cart and Checkout

- Open cart, adjust quantity, and proceed to checkout.
- Simulate payment (can use a mock interface or success page).

➢ Step 5: Order History

- Go to user profile and display past orders.

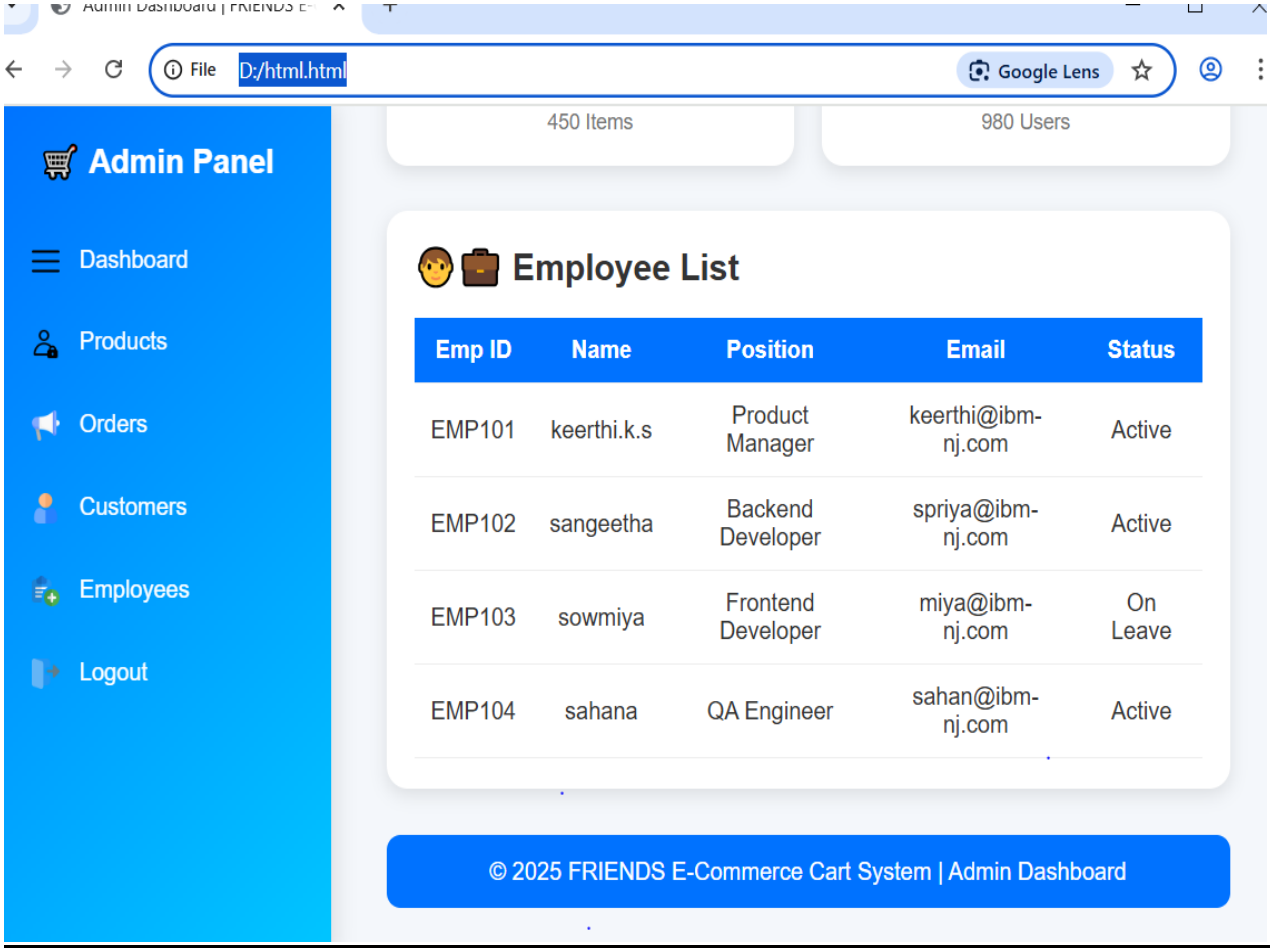
➢ Step 6: Admin Panel

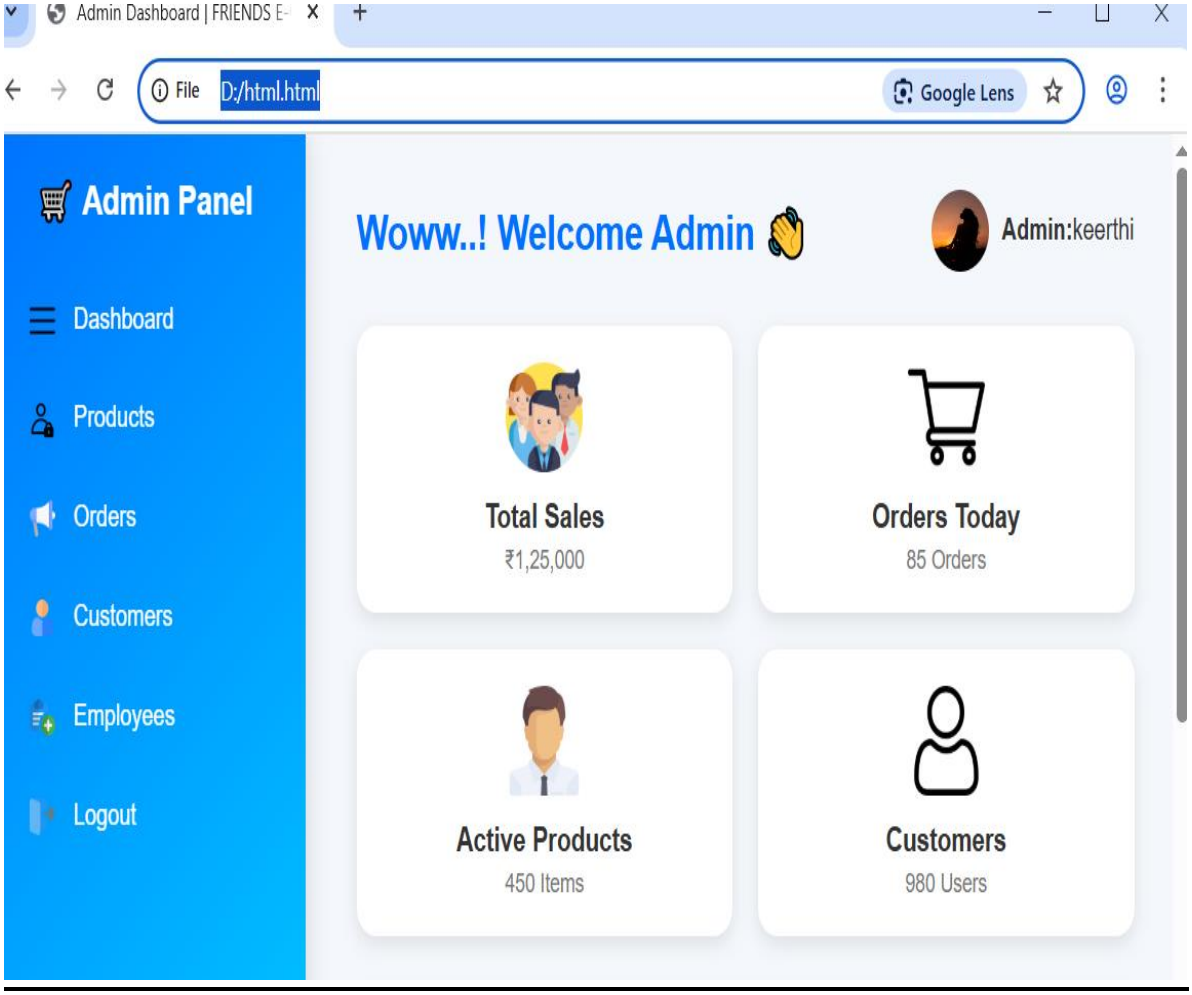
- Login as admin.
- Add a new product and show that it appears in the product catalog.
- Update order status for a placed order.

◆ 7. Future Enhancements (Optional):

- Integration with real payment gateways (Stripe, Razorpay)
- Mobile responsiveness/PWA
- Product search with filters and sorting
- User reviews and ratings
- Inventory alerts for low stock
- Email/SMS notification system

3) SCREENSHOTS/ API DOCUMENTATION:




Admin Dashboard | FRIENDS E-
File
D:/html.html
Google Lens
Admin Panel
Dashboard
Products
Orders
Customers
Employees
Logout
Woww..! Welcome Admin
Admin:keerthi
Total Sales
₹1,25,000
Orders Today
85 Orders
Active Products
450 Items
Customers
980 Users

PROGRAM:

```
<! html>
<html lang="en">
<head> DOCTYPE
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Admin Dashboard | FRIENDS E-Commerce</title>

  <style>
    /* === Basic Reset === */
    * {
      margin: 0;
      padding: 0;
      box-sizing: border-box;
      font-family: 'Poppins', sans-serif;
    }

    body {
      background-color: #f4f7fb;
      color: #333;
    }

    /* === Sidebar === */
    .sidebar {
      position: fixed;
      left: 0;
      top: 0;
      height: 100%;
      width: 250px;
      background: linear-gradient(135deg, #0072ff, #00c6ff);
      padding: 20px;
      color: white;
      box-shadow: 3px 0 15px rgba(0,0,0,0.1);
    }

    .sidebar h2 {
      text-align: center;
      margin-bottom: 30px;
      font-size: 22px;
    }

    .sidebar ul {
      list-style: none;
    }

    .sidebar ul li {
      padding: 12px 15px;
      margin: 10px 0;
      border-radius: 10px;
      transition: 0.3s;
    }

    .sidebar ul li:hover {
      background: rgba(255, 255, 255, 0.2);
      cursor: pointer;
```

```
}

.sidebar ul li img {
  width: 20px;
  vertical-align: middle;
  margin-right: 10px;
}

/* === Main Section === */
.main {
  margin-left: 260px;
  padding: 30px;
}

header {
  display: flex;
  justify-content: space-between;
  align-items: center;
  margin-bottom: 30px;
}

header h1 {
  font-size: 26px;
  color: #0072ff;
}

.admin-info {
  display: flex;
  align-items: center;
}

.admin-info img {
  width: 45px;
  height: 45px;
  border-radius: 50%;
  margin-right: 10px;
}

/* === Cards === */
.cards {
  display: grid;
  grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));
  gap: 20px;
  margin-bottom: 30px;
}

.card {
  background: white;
  border-radius: 15px;
  box-shadow: 0 4px 12px rgba(0,0,0,0.1);
  padding: 20px;
  text-align: center;
  transition: 0.3s;
}

.card:hover {
  transform: translateY(-8px);
}

.card img {
  width: 60px;
```

```
      margin-bottom: 10px;
    }

    .card h3 {
      color: #333;
      font-size: 18px;
      margin-bottom: 5px;
    }

    .card p {
      color: #777;
      font-size: 14px;
    }

    /* === Table Section === */
    .table-section {
      background: white;
      border-radius: 15px;
      box-shadow: 0 4px 12px rgba(0,0,0,0.1);
      padding: 20px;
    }

    table {
      width: 100%;
      border-collapse: collapse;
    }

    table thead {
      background: #0072ff;
      color: white;
    }

    table th, table td {
      padding: 12px;
      text-align: center;
      border-bottom: 1px solid #eee;
    }

    table tr:hover {
      background: #f1f8ff;
    }

    footer {
      text-align: center;
      padding: 15px;
      background: #0072ff;
      color: white;
      margin-top: 30px;
      border-radius: 10px;
    }
  </style>
</head>
<body>

  <!-- === Sidebar === -->
  <div class="sidebar">
    <h2>🛒 Admin Panel</h2>
    <ul>
      <li><img src="https://cdn-icons-
png.flaticon.com/512/1828/1828859.png"> Dashboard</li>
```

```
      <li><img src="https://cdn-icons-
png.flaticon.com/512/4298/4298885.png"> Products</li>
      <li><img src="https://cdn-icons-
png.flaticon.com/512/2630/2630620.png"> Orders</li>
      <li><img src="https://cdn-icons-
png.flaticon.com/512/1077/1077012.png"> Customers</li>
      <li><img src="https://cdn-icons-
png.flaticon.com/512/2921/2921226.png"> Employees</li>
      <li><img src="https://cdn-icons-
png.flaticon.com/512/1828/1828490.png"> Logout</li>
    </ul>
  </div>

  <!-- === Main Content === -->
  <div class="main">
    <header>
      <h1>Woww..! Welcome Admin 👋</h1>
      <div class="admin-info">
        <img src="https://encrypted-
tbn0.gstatic.com/images?q=tbn:ANd9GcTfnvISMw3_S5rzTpiYZGx2euGy_itSP0oKOw&s"
alt="Admin">
        <span><b>Admin:</b>keerthi</span>
      </div>
    </header>

    <!-- === Dashboard Cards === -->
    <div class="cards">
      <div class="card">
        <img src="https://cdn-icons-png.flaticon.com/512/1256/1256650.png">
        <h3>Total Sales</h3>
        <p>₹1,25,000</p>
      </div>

      <div class="card">
        <img src="https://cdn-icons-png.flaticon.com/512/1170/1170678.png">
        <h3>Orders Today</h3>
        <p>85 Orders</p>
      </div>

      <div class="card">
        <img src="https://cdn-icons-png.flaticon.com/512/265/265674.png">
        <h3>Active Products</h3>
        <p>450 Items</p>
      </div>

      <div class="card">
        <img src="https://cdn-icons-png.flaticon.com/512/1077/1077063.png">
        <h3>Customers</h3>
        <p>980 Users</p>
      </div>
    </div>

    <!-- === Employee / Product Table === -->
    <div class="table-section">
      <h2>👦💼 Employee List</h2>
      <br>
      <table>
        <thead>
          <tr>
            <th>Emp ID</th>
            <th>Name</th>
```

```
            <th>Position</th>
            <th>Email</th>
            <th>Status</th>
          </tr>
        </thead>
        <tbody>
          <tr>
            <td>EMP101</td>
            <td>keerthi.k.s</td>
            <td>Product Manager</td>
            <td>keerthi@ibm-nj.com</td>
            <td>Active</td>
          </tr>
          <tr>
            <td>EMP102</td>
            <td>sangeetha</td>
            <td>Backend Developer</td>
            <td>spriya@ibm-nj.com</td>
            <td>Active</td>
          </tr>
          <tr>
            <td>EMP103</td>
            <td>sowmiya</td>
            <td>Frontend Developer</td>
            <td>miya@ibm-nj.com</td>
            <td>On Leave</td>
          </tr>
          <tr>
            <td>EMP104</td>
            <td>sahana</td>
            <td>QA Engineer</td>
            <td>sahan@ibm-nj.com</td>
            <td>Active</td>
          </tr>
        </tbody>
      </table>
    </div>

    <footer>
      <p>© 2025 FRIENDS E-Commerce Cart System | Admin Dashboard</p>
    </footer>
  </div>

</body>
</html>
```

API Documentation (Demo):

HEADERS:

Content-Type: application/json

BODY:

```
{
  "email": "user@example.com",
```

```
  "password": "yourPassword123"
}
```

RESPONSE(Demo):

```
{
  "cart_id": "abc123",
  "user_id": "user001",
  "items": [
    {
      "product_id": "p001",
      "name": "Wireless Mouse",
      "quantity": 2,
      "price_per_unit": 25.99,
      "total_price": 51.98
    }
  ],
  "subtotal": 51.98,
  "currency": "USD"
```

4) CHALLENGES AND SOLUTIONS:

1. Cart Abandonment:

Challenge**:**

Many users add products to their carts but leave without completing the purchase. This can be due to unexpected costs, complicated checkout, or lack of trust.

Solutions**:**

- Show **total cost (including shipping/taxes)** upfront.
- Offer **guest checkout** (without mandatory account creation).
- Simplify the checkout process to **1–2 steps**.
- Send **abandoned cart emails** with reminders or discounts.
- Provide **clear return/refund policies** and trust signals (SSL, reviews, etc).

2. Inventory Sync Issues:

Challenge:

Cart systems may allow users to add out-of-stock items or fail to reserve items during checkout.

Solutions:

- Implement **real-time inventory checks** before checkout.
- Use a **reservation system** to temporarily hold items in cart for a short period.
- Alert users immediately when an item becomes **unavailable or limited**.

3.Complex Discount Rules:

Challenge:

Handling promotions like "Buy 1 Get 1", bulk discounts, or coupon stacking can become complex and buggy.

Solutions:

- Use a **rules engine** or a modular system to handle discounts.
- Create **comprehensive test cases** for edge cases.
- Show discount breakdown **transparently** in cart and checkout.

4. Multi-Currency and Localization:

Challenge:

Users from different countries expect prices, taxes, and formats in their local style.

Solutions:

- Detect user location via IP or allow manual selection.
- Convert prices with **live currency exchange rates**.
- Display amounts with **local formatting and tax inclusion/exclusion.**

5. Scalability and Performance:

Challenge**:**

During sales or high traffic, cart services can slow down or crash.

Solutions**:**

- Use **caching** for product data and common cart calculations.
- Implement **rate limiting** and **load balancing**.
- Use **microservices architecture** for modular scaling.
- Optimize database queries and use **asynchronous processes** where needed.

6.Security and Fraud Prevention:

Challenge**:**

Cart systems can be targeted for fraud (coupon abuse, fake orders, session hijacking).

Solutions**:**

- Use **HTTPS**, secure cookies, and **CSRF protection**.
- Validate and sanitize all user input.
- Implement **CAPTCHAs** and **bot detection** on sensitive operations.
- Monitor for **abnormal cart behavior** (e.g., repeated bulk coupon attempts).

7. Mobile Responsiveness:

Challenge**:**

A poorly optimized cart on mobile devices leads to higher abandonment rates.

Solutions**:**

- Design with **mobile-first UI** principles.
- Use **larger buttons**, simple forms, and easy quantity adjustments.
- Offer **mobile-friendly payment options** (Apple Pay, Google Pay).

8. Integration with Payment & Shipping:

Challenge**:**

Inaccurate or delayed shipping fee calculation and failed payment gateway responses affect conversion.

Solutions**:**

- Integrate with **real-time shipping APIs** (FedEx, UPS, etc.).
- Ensure **fallbacks** for payment gateway outages.
- Support **multiple payment gateways** for redundancy.

10. Analytics & Tracking

Challenge**:**

Lack of insight into how users interact with the cart prevents optimization.

Solutions**:**

- Track events like "Add to Cart," "Remove from Cart," "Begin Checkout."
- Use **funnels** in analytics tools (Google Analytics, Mixpanel, etc.).
- A/B test changes in cart UI/UX.

s.no	Challenge	Description	Solutions
1	Cart Abandonment	Users leave the site without completing purchase.	- Show full costs upfront - Simplify checkout - Send reminder emails - Enable guest checkout
2	Inventory Sync Issues	Users add out-of-stock items or oversell occurs.	- Real-time inventory checks - Temporary item reservation - Notify of low/out-of-stock

s.no	Challenge	Description	Solutions
			items
3	Session Management	Cart data lost if session expires or device is switched.	- Use local storage and server-side persistence - Enable login-based persistent carts
4	Complex Discount Rules	Promotions and coupons create bugs or confusion.	- Use rules engine - Modular discount logic - Transparent pricing breakdown
5	Multi-Currency & Localization	Prices, currencies, and taxes don’t match user expectations.	- Auto-detect location - Currency conversion - Local formats and tax handling
6	Scalability & Performance	Cart system slows or fails under high traffic.	- Use caching - Load balancing - Microservices and async processes
7	Security & Fraud Prevention	Vulnerable to hacks, fake orders, and abuse.	- HTTPS, CSRF, input validation - Bot detection - Monitor suspicious behavior
8	Mobile Responsiveness	Poor mobile UI leads to high drop-off rates.	- Mobile-first design - Simplified forms - Mobile payment options
9	Payment & Shipping Integration	Errors or delays in price/shipping calculation or payment processing.	- Real-time shipping APIs - Redundant payment gateways - Graceful fallback mechanisms

4)GITHUB REPOSITORY & SETUP GUIDE:

REPOSITORY NAME: IBM-NJ-E-COMMERCE CART SYSTEM

GITHUB LINK: https://github.com/sourabhDemo/ShoppingCartDemo

✓ GITHUB REPOSITORY

The complete project code has been uploaded to github for version control and easy accessibility. It includes all HTML,CCS, and JAVASCRIPT files used in the E-COMMERCE CART SYSTEM.

✓ SETUP GUIDE

1. **Choose a Platform**
   – Example: Shopify, WooCommerce, Wix, Magento
2. **Buy a Domain Name**
   – YourStoreName.com
3. **Get Web Hosting** *(for self-hosted platforms)*
   – Required for WordPress + WooCommerce
4. **Install eCommerce Software**
   – Example: WordPress + WooCommerce
5. **Choose & Customize a Theme**
   – Pick a professional, mobile-friendly design
6. **Add Your Products**
   – Product name, price, images, description, stock info
7. **Enable Cart Functionality**
   – Add "Add to Cart" and "View Cart" buttons
8. **Set Up Checkout Page**
   – Include billing, shipping, and payment options
9. **Configure Payment Gateways**
   – Example: PayPal, Stripe, Credit/Debit Cards
10. **Set Up Shipping Options**
    – Free shipping, flat rate, local pickup, etc.

11. **Apply Tax Rules**
    – Based on country or region
12. **Test the System**
    – Add items to cart, place a test order
13. **Enable Customer Features**
    – Login, guest checkout, order tracking
14. **Launch Your Store**
    – Make it live for customers
15. **Promote Your Store**
    – Use SEO, social media, ads, email marketing

5) FINAL SUBMISSION (REPO +DEPLOYED LINK)

Final Submission:

The completed project has been successfully uploaded and deployed for evaluation.

Github Repository Link:

https://github.com/sourabhDemo/ShoppingCartDemo

Deployed Link: AlegroCart — Demo:

https://www.alegrocart.com/demo/index.html

CONCLUSION:

An **E-Commerce cart system** is essential for online stores. It allows customers to select, review, and purchase products easily. Setting it up involves choosing a platform, adding products, enabling payments, and configuring checkout and shipping. Once tested and launched, it creates a smooth shopping experience and helps grow your online business efficiently.

ACKNOWLEDGEMENT:

I would like to express my sincere gratitude to everyone who supported the development of this eCommerce cart system. Special thanks to my mentors, team members, and all those who provided guidance, resources,

and encouragement throughout the process. Your support made this project possible and successful.

FINAL SUBMISSION:

- ✓ Project Title: E-Commerce Cart System
- ✓ Technology Used: HTML,CCS,JAVASCRIPT
- ✓ Organization: IBM Skillsbuild x Nan Mudhalvan
- ✓ Github Repository: https://github.com/sourabhDemo/ShoppingCartDemo

THANK YOU!!!

✓ Submitted by the Project TEAM
✓ Team Members:

1) ATCHAYASREE.J
2) KANMANI.S
3) KARTHIK.V
4) AUSTIN.R

✓ Under: IBM Skills BUILD – Nan Mudhalvan Node JS Technology