

GODAVARI INSTITUTE OF MANAGEMENT **AND RESEARCH**

Subject : Advanced software development

Project Topic : E-commerce Website

Name: 1) Bhagyashri Shivdas Pawar Roll No: 36

2) Sayali Udhav Phegade Roll No : 38

Class – MCA -I sem -II

Year - 2023 - 24

- Subject Teacher - Charushila Chaudhari

Introduction:

In today's digital age, e-commerce has revolutionized the way people buy and sell goods and services. With the rise of online shopping, businesses are increasingly leveraging digital platforms to reach a global audience, streamline transactions, and provide personalized shopping experiences. Our e-commerce website, built using the MERN stack, is at the forefront of this digital transformation, offering a seamless and user-friendly platform for buying and selling a wide range of products.

Overview of the Project:

Our e-commerce website is a comprehensive online marketplace where users can browse, search, and purchase products from various categories. Leveraging the power of the MERN stack – MongoDB, Express.js, React.js, and Node.js – we have developed a robust and scalable platform that provides a rich user experience and supports efficient management of product listings, orders, and payments.

Purpose and Objectives:

The primary purpose of our e-commerce website is to provide a convenient and secure online shopping experience for users, allowing them to discover and purchase products with ease. Our objectives include:



Search...

SEARCH

Hi, admin



Fisrt Product

★★★★★ 4 reviews

\$25.26



Fisrt Product

★★★★★ 4 reviews

\$25.26



Fisrt Product

★★★★★ 4 reviews

\$25.26



Fisrt Product

★★★★★ 4 reviews

\$25.26

Fisrt Product

Fisrt Product

★★★★★ 4 reviews

\$150.26



Fisrt Product

★★★★★ 4 reviews

\$25.26



Fisrt Product

★★★★★ 4 reviews

\$25.26



Fisrt Product

★★★★★ 4 reviews

\$25.26



Fisrt Product

★★★★★ 4 reviews

\$120

1

2

3

4



Call us 24/7
12121211231



Headquater
Bigtime



fax
113212311213

Project Overview:

Background Information:

The e-commerce website project aims to create a comprehensive online platform for buying and selling various products. With the increasing trend of online shopping, there is a growing demand for robust e-commerce platforms that provide a seamless shopping experience for users. The project is developed using the MERN stack, which includes MongoDB, Express.js, React.js, and Node.js, leveraging their capabilities to build a scalable and efficient web application.

Project Scope and Boundaries:

The scope of the project includes:

- User authentication and authorization system
- Product browsing and searching functionality
- Shopping cart management
- Order processing and payment gateway integration
- Admin dashboard for managing products, orders, and users
- Responsive design for optimal viewing across devices

The project boundaries are defined by:

- Limitations of the chosen technologies and frameworks
- Defined project requirements and objectives
- Constraints such as time, budget, and resources

Technologies Used

For an e-commerce website built using the MERN (MongoDB, Express.js, React, Node.js) stack, here's a list of technologies, frameworks, and libraries typically used along with justifications for their selection:

1. MongoDB:

- MongoDB is a NoSQL database that provides flexibility and scalability, making it suitable for e-commerce applications with dynamic and evolving data structures.
- Justification: MongoDB's document-based model allows for easy storage and retrieval of product data, user information, orders, and other relevant data.

2. Express.js:

- Express.js is a minimal and flexible Node.js web application framework that simplifies the process of building server-side logic and APIs.
- Justification: Express.js provides a lightweight and efficient way to handle HTTP requests, define routes, and interact with the MongoDB database, making it ideal for building the backend of an e-commerce website.

3. React:

- React is a JavaScript library for building user interfaces, providing a component-based architecture and efficient rendering.
- Justification: React's declarative and component-based approach allows for the creation of interactive and dynamic user interfaces, facilitating tasks such as product browsing, search, and checkout processes.

4. Node.js:

- Node.js is a JavaScript runtime environment that enables server-side scripting, allowing developers to build scalable and efficient web applications.
- Justification: Node.js provides event-driven and non-blocking I/O capabilities, making it well-suited for handling concurrent requests and performing server-side operations in an e-commerce application.

5. Express.js Middleware:

- Various middleware packages such as body-parser, cors, and morgan are commonly used with Express.js to enhance functionality and handle cross-origin resource sharing (CORS), request parsing, and logging.
- Justification: Middleware helps streamline server-side operations, improve security, and enhance the performance of the application.

6. Redux:

- Redux is a predictable state container for JavaScript applications, commonly used with React to manage application state in a more organized and scalable manner.
- Justification: For complex e-commerce applications with multiple components and shared state, Redux helps centralize state management, improve data flow, and simplify debugging.

7. JSON Web Tokens (JWT) for Authentication:

- JWT is a standard for securely transmitting information between parties as a JSON object, commonly used for implementing authentication and authorization in web applications.
- Justification: JWT-based authentication provides a stateless and scalable solution for user authentication, enabling secure access to protected routes and resources in the e-commerce website.

8. Stripe or PayPal SDK (for Payment Processing):

- Stripe and PayPal SDKs provide APIs and tools for integrating payment processing functionalities into e-commerce applications, including support for credit/debit card payments, digital wallets, and subscriptions.
- Justification: These payment gateways offer secure and reliable payment processing, ensuring smooth transactions and a seamless checkout experience for customers.

Installation and Setup:

System Requirements:

- Operating System: Windows, macOS, or Linux
- Node.js: Version 12.x or higher
- MongoDB: Version 4.x or higher
- Git: Version control system (optional, but recommended)

Dependencies and Prerequisites:

1. Node.js and npm:

- Install Node.js and npm from the official website: [Node.js Downloads](<https://nodejs.org/en/download/>)

- Verify the installation by running the following commands in your terminal or command prompt:

```
...
```

```
node -v
```

```
npm -v
```

```
...
```

2. MongoDB:

- Install MongoDB Community Edition from the official website: [MongoDB Downloads](<https://www.mongodb.com/try/download/community>)

- Follow the installation instructions for your operating system.

- Start the MongoDB service.

3. Install Dependencies:

- Navigate into the project directory:

...

```
cd <project-directory>
```

...

- Run npm install to install the project dependencies:

...

```
npm install
```

...

4. Start the Development Server:

- Once the dependencies are installed and environment variables are configured, you can start the development server by running:

...

```
npm start
```

...

- This will start the server and your e-commerce website will be accessible at ``http://localhost:5000`` (or the specified port).

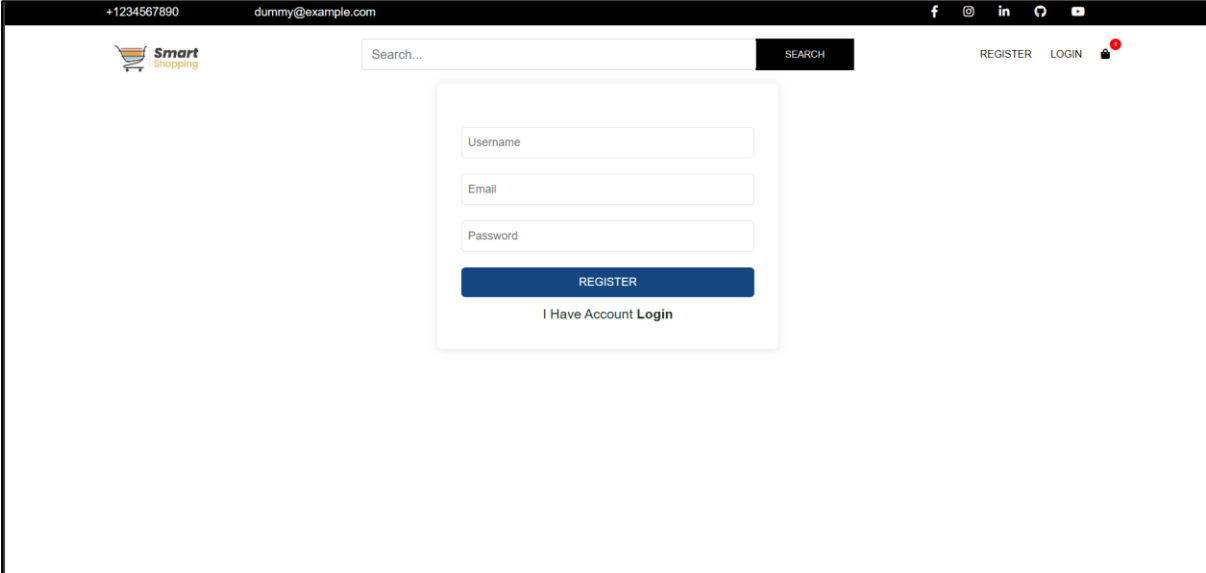
Pages Information

Register Page:

The register page enables new users to create an account on the e-commerce platform. It collects necessary information from users and stores it securely in the database. Here's what you'll typically find on a registration page:

Input Fields:

- Username: Unique identifier for the user's account.
- Email: Valid email address for account verification and communication.
- Password: Secure password to protect the user's account.

A screenshot of a web browser displaying a registration page for 'Smart Shopping'. The browser's address bar shows '+1234567890' and 'dummy@example.com'. The page header includes a shopping cart icon, the 'Smart Shopping' logo, a search bar with 'Search...' text and a 'SEARCH' button, and links for 'REGISTER', 'LOGIN', and a notification bell. The main content area features a registration form with three input fields labeled 'Username', 'Email', and 'Password'. Below these fields is a blue 'REGISTER' button and a link that says 'I Have Account Login'.

Login Page:

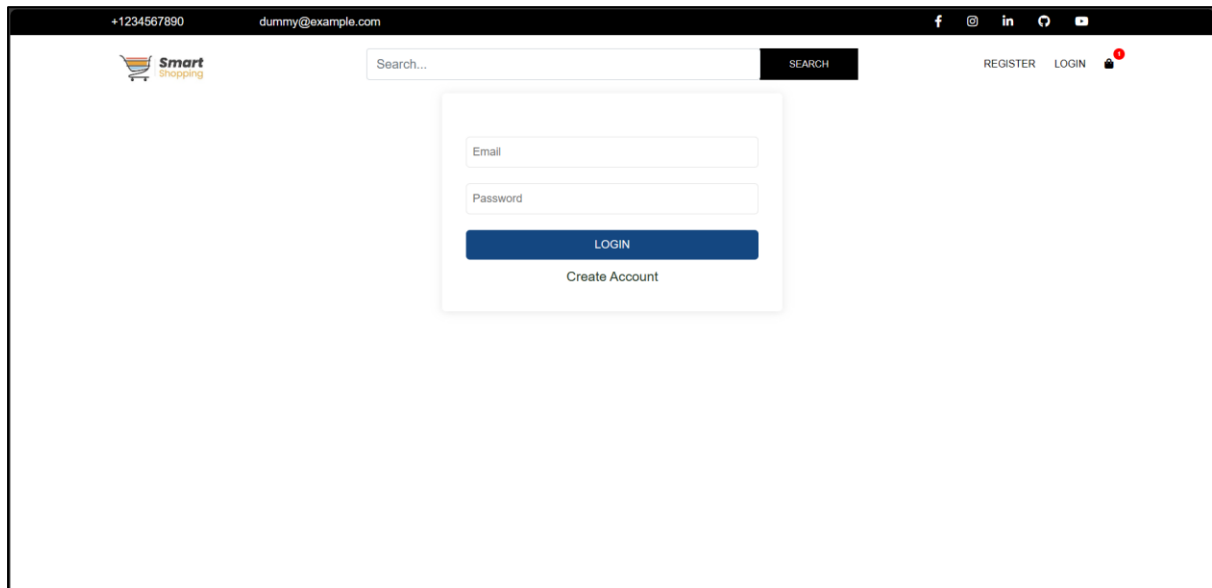
The login page allows registered users to authenticate themselves by entering their credentials, typically a username/email and password. Here are the key features and components of a login page:

1. Input Fields:

- Username/Email: Allows users to enter their registered username or email address.
- Password: Secure input field for entering the user's password.

2. Submit Button:

- "Login" or "Sign In" button triggers the authentication process when clicked.



The screenshot shows a web application interface for 'Smart Shopping'. At the top, there is a header bar with a phone number '+1234567890', an email 'dummy@example.com', and social media icons for Facebook, Instagram, LinkedIn, and YouTube. Below the header, the 'Smart Shopping' logo is on the left, followed by a search bar with the placeholder text 'Search...' and a 'SEARCH' button. To the right of the search bar are links for 'REGISTER' and 'LOGIN', along with a shopping cart icon that has a red notification bubble. In the center of the page, there is a login form with two input fields labeled 'Email' and 'Password'. Below these fields is a blue 'LOGIN' button and a link that says 'Create Account'.

Profile Page

1. Personal Information:

- Displayed name
- Email address
- Contact information (phone number, address)

2. Account Settings:

- Change password
- Update email address
- Manage communication preferences (newsletter subscriptions, notifications)

3. Order History:

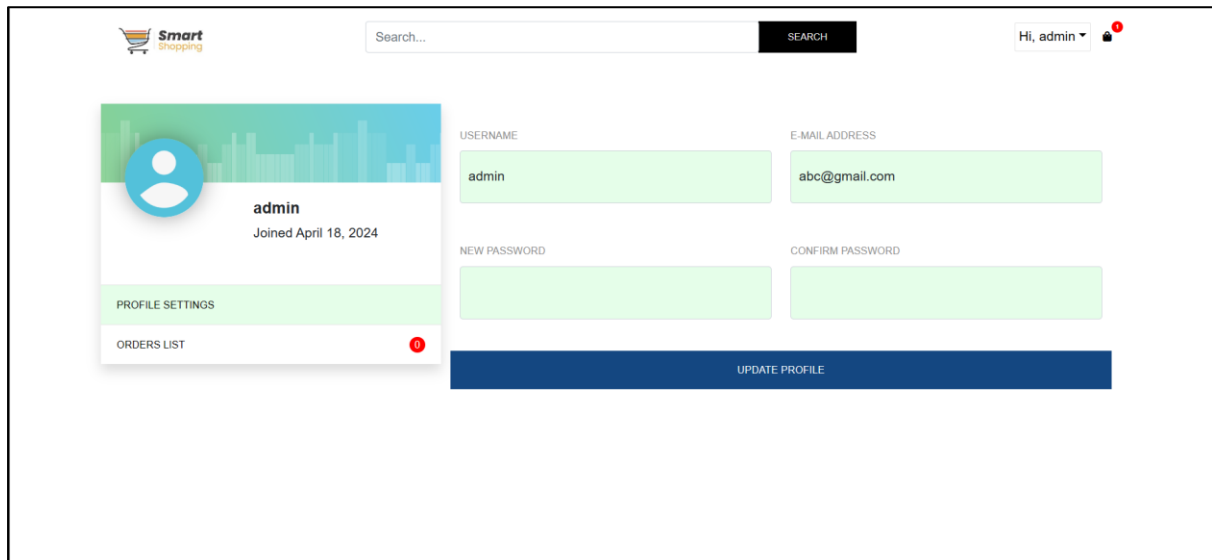
- List of past orders
- Order details (items purchased, order status, tracking information)
- Ability to view/print order invoices

4. Wishlist:

- Items saved for future purchase
- Option to add or remove items from the wishlist

5. Logout Button:

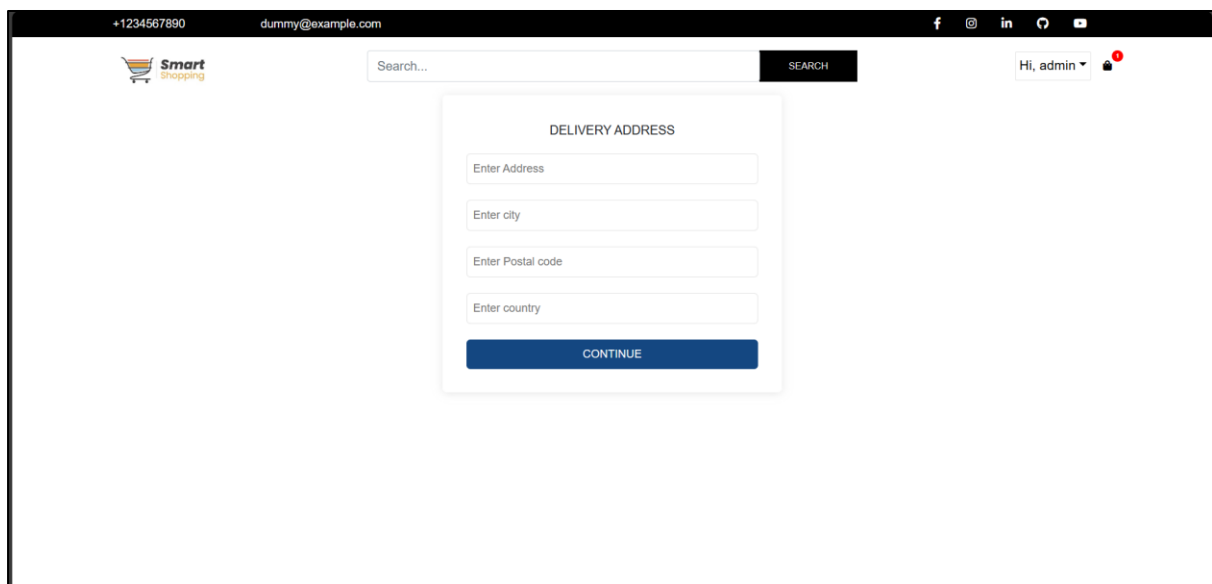
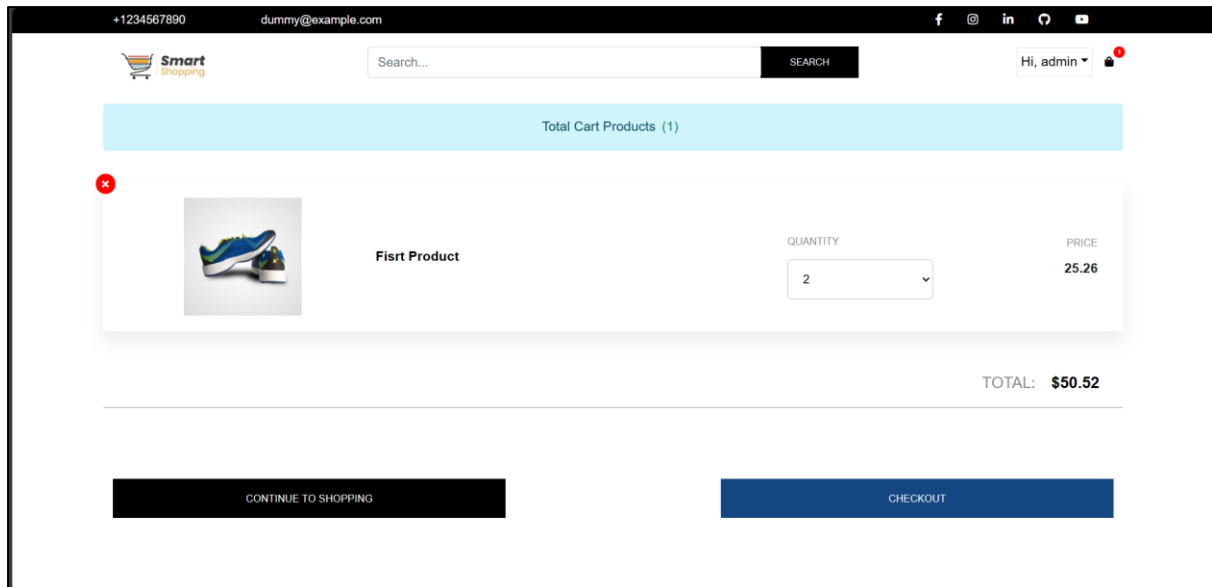
- Option to securely log out of the account



The image shows a user profile and password update form. At the top left is the 'Smart Shopping' logo. To its right is a search bar with the text 'Search...' and a 'SEARCH' button. Further right is a user greeting 'Hi, admin' with a dropdown arrow and a notification icon. Below the logo is a user profile card for 'admin' with a blue circular profile picture and the text 'Joined April 18, 2024'. The card has two links: 'PROFILE SETTINGS' and 'ORDERS LIST'. To the right of the card are four input fields: 'USERNAME' (containing 'admin'), 'E-MAIL ADDRESS' (containing 'abc@gmail.com'), 'NEW PASSWORD', and 'CONFIRM PASSWORD'. At the bottom right is a blue 'UPDATE PROFILE' button.

Cart Page

- Adding Items: Users can add items to the cart from product pages by clicking on "Add to Cart" buttons.
- Adjusting Quantities: Users can increase or decrease the quantity of items directly on the cart page.
- Removing Items: Users can remove items from the cart using a "Remove" button or icon associated with each item.
- Promo Code Application: Users can enter promo codes in a designated field to apply discounts to their cart total.
- Checkout Process: Users can proceed to the checkout process from the cart page by clicking on the "Checkout" button.



Payment Page:

The payment screen is a crucial component of any e-commerce platform, serving as the final step in the purchasing process. It provides users with a secure and convenient way to complete their transactions by entering payment details and confirming their orders. This screen is designed to instill confidence in users by ensuring that their sensitive information is protected throughout the payment process.

Key Components:

1. Payment Methods Selection:

2. Billing Information:

3. Payment Details:

4. Order Summary

5. Security Measures

6. Confirmation and Completion.

7. Error Handling

The screenshot shows a shopping cart interface for 'Smart shopping'. At the top, there's a header with a phone number '+1234567890', an email 'dummy@example.com', and social media icons. Below the header, there's a search bar and a user profile 'Hi, admin'. The main content area is divided into three sections: 'Customer' (admin, abc@gmail.com), 'Order Info' (Shipping: India, Pay method: PayPal), and 'Deliver to' (Address: bha, nv, 52). Below these, there's a product list with one item: 'Fisrt Product' (note the typo) with a quantity of 2 and a subtotal of \$50.52. To the right, there's a summary table showing 'Products' (\$50.52), 'Shipping' (\$100.00), 'Tax' (\$7.58), and a 'Total' of \$158.10. A 'PLACE ORDER' button is at the bottom right.

Products	Quantity	Subtotal
Fisrt Product	2	\$50.52

Products	\$50.52
Shipping	\$100.00
Tax	\$7.58
Total	\$158.10

PLACE ORDER

Database Schema Model

1. User Table:

- user_id (Primary Key)
- username
- email
- password (hashed)

2. Product Table:

- product_id (Primary Key)
- name
- image
- description
- price
- numReviews
- countInStock
- reviews
- rating

3. Order Table:

- order_id (Primary Key)
- user_id (Foreign Key to User Table)
- total_amount
- status (e.g., pending, processing, completed)
- created_at
- updated_at

5. Order Item Table:

```
{  
  user,  
  orderItems: [  
    {  
      name,  
      image,  
      qty,  
      price
```

```
    },  
    product: {  
      type,  
      required,  
      ref  
    },  
  },  
],  
shippingAddress:  
{  
  address,  
  city,  
  postalCode,  
  country,  
},  
  
paymentMethod: {  
  type,  
  required,  
  ref,  
},  
paymentResult: {  
  id,  
  status,  
  update_time,  
  email_address,  
},  
taxPrice,
```

```
    shippingPrice,  
    totalPrice,  
    isPaid,  
    paidAt ,  
    isDelivered,  
    deliveredAt  
  },  
  {  
    timestamps: true  
  }
```

An Entity-Relationship Diagram (ERD) visualizes the relationships between different entities in the database. While the above schema provides a textual representation, an ERD would provide a graphical representation of these relationships, making it easier to understand the database structure at a glance.