INDUSTRIAL TRAINING REPORT

ON

**Full Stack E-commerce Website Using MERN Stack | Project with Stripe**

**(PrimeWardrobe)**

Submitted in partial fulfillment of the requirements

For the award of the degree of

**BACHELOR OF TECHNOLOGY**

**IN**

**ARTIFICIAL INTELLIGENCE & MACHINE LEARNING**

Submitted By

**RISHABH GIRI KASHISH ARORA**

*(Roll No.02015611621) (Roll No.02915611621)*

**Under the guidance of**

Dr. Suman Bhatia

Professor

AIML Department



**Department of Artificial Intelligence & Machine Learning Dr. Akhilesh Das Gupta Institute of Professional Studies**

**(Guru Gobind Singh Indraprastha University, Dwarka, Delhi.) New Delhi -110053.**

**CERTIFICATE**

We hereby certify that the work that is being presented in the project report entitled **Full Stack E-commerce Website (PrimeWardrobe)**to the partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Artificial Intelligence & Machine Learning** from **Dr. Akhilesh Das Gupta Institute of Professional Studies**, New Delhi. This is an authentic record of our work carried out under the guidance of **Dr. Suman Bhatia, Professor in the AIML Department.**

The matter presented in this project has not been submitted by us for the award of any other degree elsewhere.

**RISHABH GIRI KASHISH ARORA**

*(Roll No.02015611621) (Roll No.02915611621)*

This is to certify that the above statement made by the candidate is correct to the best of my knowledge. He/She/They are permitted to appear in the Project External Examination.

**Dr. Suman Bhatia**

**Professor**

**Mr. Rishabh Jain Prof. (Dr.) Ankit Verma**

**Training Coordinator HOD (AIML)**

**ACKNOWLEDGEMENT**

We would like to acknowledge the contributions of the following persons, without whose help and guidance this report would not have been completed.

We acknowledge the counsel and support of our project guide **Dr.** **Suman Bhatia, Professor, AIML Department,** with respect and gratitude, whose expertise, guidance, support, encouragement, and enthusiasm has made this report possible. Their feedback vastly improved the quality of this report and provided an enthralling experience. We are indeed proud and fortunate to be supervised by him.

We are thankful to **Prof. (Dr.) Ankit Verma, HOD AIML Department, Dr. Akhilesh Das Gupta Institute of Professional Studies, New Delhi** for his constant encouragement, valuable suggestions and moral support and blessings.

We are immensely thankful to our esteemed **Director, Dr. Akhilesh Das Gupta Institute of Professional Studies, New Delhi** for his never-ending motivation and support.

We shall ever remain indebted to **Training Coordinator, AIML Department** and faculty and staff members of Dr. Akhilesh Das Gupta Institute of Professional Studies, New Delhi.

Finally, yet importantly, we would like to express our heartfelt thanks to God, our beloved parents for their blessings, our friends/classmates for their help and wishes for the successful completion of this project.

**RISHABH GIRI KASHISH ARORA**

*(Roll No.02015611621) (Roll No.02915611621)*

**ABSTRACT**

PrimeWardrobe is a fully-featured e-commerce application developed to provide a smooth and engaging online shopping experience, tailored to meet the demands of modern e-commerce users and administrators. Built with the MERN stack—MongoDB, Express, React, and Node.js—this application leverages cutting-edge technologies to create a platform that is both responsive and scalable. Key objectives of the project include delivering a user-friendly interface for customers, implementing efficient data management and security in the backend, and providing comprehensive control tools within an admin panel to manage products, users, and orders seamlessly. Additionally, the integration of Stripe enables secure, reliable online payment processing, ensuring user trust and convenience throughout the purchasing experience.

The PrimeWardrobe application is structured into three main components: frontend, backend, and an admin panel. The frontend prioritizes user experience, offering features such as product browsing, category filtering, detailed product views, a shopping cart, and checkout capabilities. Users can create accounts, log in securely, and manage their profiles, allowing for a personalized shopping journey. The backend utilizes Node.js and Express to support robust data handling, incorporating MongoDB for efficient storage of product, user, and order information. This component also manages user authentication via JWT, ensuring secure access and data protection. With RESTful API endpoints, the backend seamlessly connects the frontend to enable reliable interactions and real-time updates for the user. Finally, the admin panel provides vital management tools, allowing administrators to add, update, and delete products, monitor and process orders, update order statuses, and manage user data, making PrimeWardrobe an efficient and adaptable e-commerce management solution.

PrimeWardrobe’s development emphasizes modern design principles, high performance, and modular functionality. The application’s backend is equipped with strong security practices, including hashed password storage, encrypted sessions, and secure data exchanges, ensuring protection for both customers and administrators. Additionally, the Stripe integration facilitates quick and secure payment processing, accommodating multiple payment methods and enhancing transaction reliability. Deployed across platforms, PrimeWardrobe is designed to scale with business needs, providing flexibility for future feature expansion.

This paper presents a detailed examination of the project’s technical architecture, development methodologies, and deployment processes. PrimeWardrobe serves as a case study in leveraging the MERN stack and Stripe to create an accessible, secure, and scalable e-commerce platform. This research underscores the platform's ability to cater to diverse user needs and its potential for adaptation in a competitive e-commerce landscape.

**TABLE OF CONTENTS**

Certificate i

Acknowledgement ii

Abstract iii Table of Contents iv

List of Figure v

**CHAPTER 1: INTRODUCTION AND LITERATURE REVIEW**

* 1. Introduction 1
  2. Basic terms of project 1
  3. Literature Overview 1
  4. Motivation 2
  5. Organization of Project Report 2

**CHAPTER 2: METHODOLOGY ADOPTED**

2.1. Project Overview and Architecture 3

2.2 Backend Development 3

2.3 Project Overview and Architecture 3

2.4. Payment Integration with Stripe 4

**CHAPTER 3: DESIGNING AND RESULT ANALYSIS**

3.1. Application Design 5

3.2. User Interface Design and Responsiveness 6

3.3. Result Analysis and Testing 6

**CHAPTER 4: MERITS, DEMERITS AND APPLICATIONS**

4.1 Merits 7

4.2 Demerits 7

4.3 Applications 8

**CHAPTER 5: CONCLUSIONS AND FUTURE SCOPE**

5.1 Conclusion 9

5.2 Future Scope 9

**SCREENSHOTS OF THE WEB APPLICATION INTERFACE 10**

**CODES OF THE WEB APPLICATION INTERFACE 14**

**REFERENCES 41**

**RESEARCH PAPER 44**

**APPENDIX 59**

**List of Figures**

**Figure No. Title of Figure Page No.**

1.1 Landing Page 10

1.2 Landing Page 10

1.3 Landing Page 10

1.4 Landing Page 11

2 Collection Page 11

3 About Page 11

4 Contact Us Page 12

5 Login Page 12

6 Sign Up Page 12

7 Cart Page 13

**CHAPTER 1: INTRODUCTION AND LITERATURE REVIEW**

**1.1. Introduction**

The demand for online shopping has seen a remarkable increase in recent years, driven by convenience, variety, and accessibility. As businesses shift online, robust and scalable e-commerce platforms are essential for handling high volumes of transactions, secure payment processing, and seamless user experiences. This project, titled Create Full Stack E-commerce Website Using React JS | MERN Stack eCommerce Project with Stripe, aims to build an end-to-end e-commerce application using the MERN stack (MongoDB, Express, React, and Node.js) and integrates Stripe for secure payment handling. By utilizing the MERN stack, this project leverages a combination of powerful technologies for building both the frontend and backend, creating a seamless user experience and robust server-side functionality.

**1.2. Basic Terms of the Project**

* + 1. **MERN Stack:** The MERN stack is a JavaScript-based technology stack comprising MongoDB, Express, React, and Node.js, which collectively form a powerful toolset for full-stack development. MongoDB is used as a NoSQL database for storing user and product data, while Express and Node.js facilitate backend development and API creation. React, a JavaScript library, is used to develop the frontend, providing a dynamic and responsive user interface.
    2. **Stripe:** Stripe is a payment processing platform that enables secure and efficient online transactions. With features like card validation, fraud prevention, and easy integration, Stripe is commonly used in e-commerce projects to handle payments and manage customer transactions.
    3. **User Authentication and Authorization:** This is essential in e-commerce platforms to ensure that only registered users can access certain features. Authentication verifies a user's identity, while authorization controls the level of access to resources.
    4. **Shopping Cart and Order Management:** A shopping cart allows users to add, view, and manage products they wish to purchase, while order management handles the backend logic for processing and storing orders.

**1.3. Literature Overview**

In the e-commerce domain, numerous studies and projects have focused on developing full-stack applications with integrated payment processing. Recent advancements in JavaScript libraries and frameworks, like React and Node.js, have popularized the use of full-stack JavaScript for e-commerce applications. Research has highlighted the MERN stack's ability to provide a cohesive development experience and support high performance.

Additionally, studies on Stripe integration have demonstrated its reliability and security in handling financial transactions, making it an ideal choice for small to medium-sized e-commerce applications. Previous work in this area also underscores the importance of secure authentication, efficient database management, and responsive front-end design for a successful e-commerce experience.

**1.4. Motivation**

The primary motivation behind this project is to develop a comprehensive and user-friendly e-commerce platform that meets modern demands for security, scalability, and efficiency. E-commerce continues to transform the way people shop, and as such, it is essential to create systems that not only meet customer expectations but also provide developers with a streamlined and robust development process. Leveraging the MERN stack for both client-side and server-side functionality allows for consistent and seamless integration. Additionally, integrating Stripe provides a safe and straightforward solution for handling payments, enhancing user trust and providing a competitive edge.

**1.5. Organization of Project Report**

This report is organized into the following chapters:

* **Chapter 1:** Introduction and Literature Review, where we introduce the project, define basic terms, review related literature, outline our motivations, and summarize the organization of the report.
* **Chapter 2:** Methodology Adopted, detailing the theoretical background and approach taken in developing the project, covering the backend, frontend, and integration of payment functionalities.
* **Chapter 3:** Designing and Result Analysis, which describes the design choices, architecture, user interface, and outcomes of various components, along with testing results.
* **Chapter 4:** Merits, Demerits, and Applications, where we evaluate the strengths and limitations of the project and discuss its potential real-world applications.
* **Chapter 5:** Conclusion and Future Scope, summarizing the project, key learnings, and suggesting possible future enhancements.

**CHAPTER 2: METHODOLOGY ADOPTED**

**2.1. Project Overview and Architecture**

The methodology adopted for this project is centered around the MERN stack (MongoDB, Express, React, Node.js) to construct a full-stack e-commerce platform with Stripe integration for payment processing. This chapter explains each step of the development process, from backend setup to front-end design, as well as the integration of key features such as product catalog management, user authentication, shopping cart functionality, and secure payment processing.

**2.2 Backend Development**

The backend is constructed using Node.js and Express, which provide a powerful foundation for creating RESTful APIs to manage data and user requests. Key steps in backend development include:

**2.2.1 Database Design:** MongoDB is used as the NoSQL database, chosen for its scalability and flexibility. The database is structured to include collections for users, products, orders, and carts. Each collection has schemas that define the fields and data types.

**2.2.2 API Endpoints:** RESTful API endpoints are designed to handle core functionalities, such as user registration, login, product retrieval, order creation, and payment processing. The endpoints are secured to ensure data privacy and restrict unauthorized access.

**2.2.3 Authentication and Authorization:** Using JSON Web Tokens (JWT), the backend authenticates and authorizes users, ensuring that sensitive operations (like payment and order history) are only accessible to authenticated users.

**2.3 Frontend Development**

The frontend of the application is built with React, which enables the development of a dynamic, component-based user interface. Key components and features include:

**2.3.1 Product Catalog and Filtering:** The main page displays a list of products fetched from the backend. Users can view details for each product and filter based on categories, price range, or availability.

**2.3.2 Shopping Cart:** A cart component allows users to add, view, and remove items. The cart dynamically updates as users interact with it, showing the total price and a summary of items selected.

**2.3.3 User Registration and Login:** Users can register and log in through forms validated on both the client and server sides. Logged-in users have access to additional features like viewing order history and managing payment information.

**2.3.4 Responsive Design:** To ensure accessibility across devices, the application uses responsive design techniques with CSS and media queries, allowing it to function seamlessly on desktops, tablets, and mobile devices.

**2.4. Payment Integration with Stripe**

A critical part of the project involves integrating Stripe to handle secure payments. Stripe’s API enables seamless, PCI-compliant payment processing without storing sensitive card details on the server. Key aspects include:

**2.4.1 Checkout Flow:** When users proceed to checkout, they are redirected to a secure Stripe interface to enter payment details. Stripe processes the payment and sends a confirmation to the backend, which then updates the order status.

**2.4.2 Order and Payment Validation:** Once a payment is successful, the backend verifies the order details and updates the database to record the transaction. Users receive a confirmation message, and the order is stored for future reference in their account.

**2.4.3 Error Handling and Security:** Error handling is implemented to capture payment failures, allowing users to retry if necessary. Security measures, including HTTPS, CSRF protection, and Stripe’s embedded security protocols, ensure a secure transaction process.

**CHAPTER 3: DESIGNING AND RESULT ANALYSIS**

**3.1. Application Design**

The design of the e-commerce platform focuses on usability, responsiveness, and visual appeal. The user interface is developed with React to ensure a seamless, dynamic experience. Key design elements are structured to provide an intuitive flow, from product browsing to checkout, enhancing the overall user experience.

**3.1.1. Database Structure**

The database is designed using MongoDB and is divided into key collections:

* **Users Collection:** Stores user information, including login credentials, contact details, and order history.
* **Products Collection:** Contains product details such as name, description, price, images, and categories.
* **Orders Collection:** Manages order data, including purchased items, total amount, and payment status.
* **Cart Collection:** Holds data on items added to a user’s cart, allowing users to adjust quantities or remove products before checkout.

**3.1.2. Frontend Components and Layout**

The front end is composed of reusable React components:

* **Header and Navigation:** Provides easy access to product categories, the cart, and user account options.
* **Product Display:** A responsive grid that displays product images, names, prices, and add-to-cart buttons. Each product card links to a detailed view.
* **Shopping Cart:** A dedicated component to manage and review items before purchase, dynamically updating the cart total as items are added or removed.
* **Checkout Page:** Handles user details and payment options, leading to a secure payment flow through Stripe integration.

**3.1.3. Payment Flow and Security**

The checkout process is secured with Stripe, ensuring PCI compliance and a seamless user experience:

* **Stripe Integration:** When users proceed to payment, they are redirected to Stripe's secure interface to enter their payment details. This method protects sensitive information and ensures compliance with industry standards.
* **Order Validation and Confirmation:** Upon payment, the backend validates and updates the order status, storing transaction details for future reference. Users are presented with a confirmation screen, reinforcing a successful transaction.

**3.2. User Interface Design and Responsiveness**

The platform’s design prioritizes responsive design principles, ensuring functionality across devices. CSS and media queries are used extensively to adapt layouts for desktops, tablets, and mobile screens. Key aspects include:

* + 1. **Fluid Layout:** A grid-based layout allows for smooth resizing of product displays, ensuring readability and usability on smaller screens.
    2. **Adaptive Navigation:** The header and navigation bar adapt to device screens, transforming into a collapsible menu on mobile devices for easy access.
    3. **Consistent Styling:** Colors, fonts, and buttons maintain a cohesive look throughout, enhancing the visual appeal and professional look of the platform.

**3.3. Result Analysis and Testing**

The platform’s components were tested individually and in integration to ensure a smooth, error-free experience. Testing covered:

* + 1. **Functional Testing:** Ensured that each component, from user authentication to cart management and Stripe payment processing, operated as expected.
    2. **Load Testing:** Simulated multiple users to assess the platform’s performance and responsiveness under heavy load, focusing on database queries and API calls.
    3. **Security Testing:** Verified that sensitive operations, particularly user login and payment processing, were secure. Emphasis was placed on data encryption, CSRF protection, and secure token handling.

**CHAPTER 4: MERITS, DEMERITS, AND APPLICATIONS**

**4.1 Merits**

This e-commerce platform offers several advantages due to its design and technology stack:

* + 1. **Full-Stack Integration with MERN:** The MERN stack (MongoDB, Express, React, Node.js) ensures efficient handling of both frontend and backend processes, enabling a seamless user experience from data storage to user interaction.
    2. **Real-Time Updates and Responsiveness:** With React, the front end provides real-time updates, allowing changes in the cart or product list to be instantly reflected without reloading the page. This results in faster and more responsive interactions, which is essential in e-commerce.
    3. **Secure Payment Processing with Stripe:** Integrating Stripe ensures that payments are processed securely, protecting user data while maintaining PCI compliance. Stripe’s secure API provides robust fraud detection, encryption, and safe handling of sensitive payment information.
    4. **Scalability:** MongoDB’s NoSQL structure and Node.js’s event-driven nature make the application easily scalable to handle increasing data and user load. The modular design allows for the addition of new features, like more payment gateways or enhanced product categories, with minimal restructuring.
    5. **User Authentication and Authorization:** Secure login and restricted access enhance user security, protecting sensitive data and preventing unauthorized access to personal information and payment details.

**4.2 Demerits**

Despite its advantages, the platform has some limitations:

* + 1. **Complex Setup and Maintenance:** The MERN stack requires a working knowledge of various technologies, which may increase complexity in setup and maintenance. Maintaining database integrity and synchronizing backend and frontend can be challenging for less experienced developers.
    2. **Scalability Constraints in High-Traffic Scenarios:** Although scalable, the platform may face bottlenecks with very high traffic unless hosted on a dedicated server or with additional load-balancing mechanisms. This could impact performance during peak shopping times or promotional events.
    3. **Reliance on Third-Party Payment Processor (Stripe):** While Stripe provides excellent security, it may limit payment options in regions where Stripe is unavailable. Adding alternative payment processors would require additional development and security compliance checks.

**4.3 Applications**

This platform can serve a variety of e-commerce needs and can be applied in different contexts:

* + 1. **Small to Medium-Sized Online Retailers:** Businesses looking to sell products online can use this platform to offer a complete shopping experience, from product listing to checkout.
    2. **Educational Projects:** This project serves as a learning tool for students and developers to understand full-stack development and e-commerce functionalities, including secure payment integration.
    3. **Customizable Business Solutions:** Given its modular design, the platform can be adapted for niche markets, allowing businesses to customize it for specific product types or additional features like loyalty programs, digital downloads, or multi-vendor marketplaces.

**CHAPTER 5: CONCLUSIONS AND FUTURE SCOPE**

**5.1 Conclusion**

This project demonstrates the successful development of a full-stack e-commerce platform using the MERN stack and Stripe integration for secure payments. Through a combination of MongoDB for data storage, Express and Node.js for server-side operations, and React for a responsive frontend, the platform provides a seamless and efficient user experience. The integration of Stripe adds a layer of security for payment processing, ensuring that user transactions are handled with industry-standard protections.

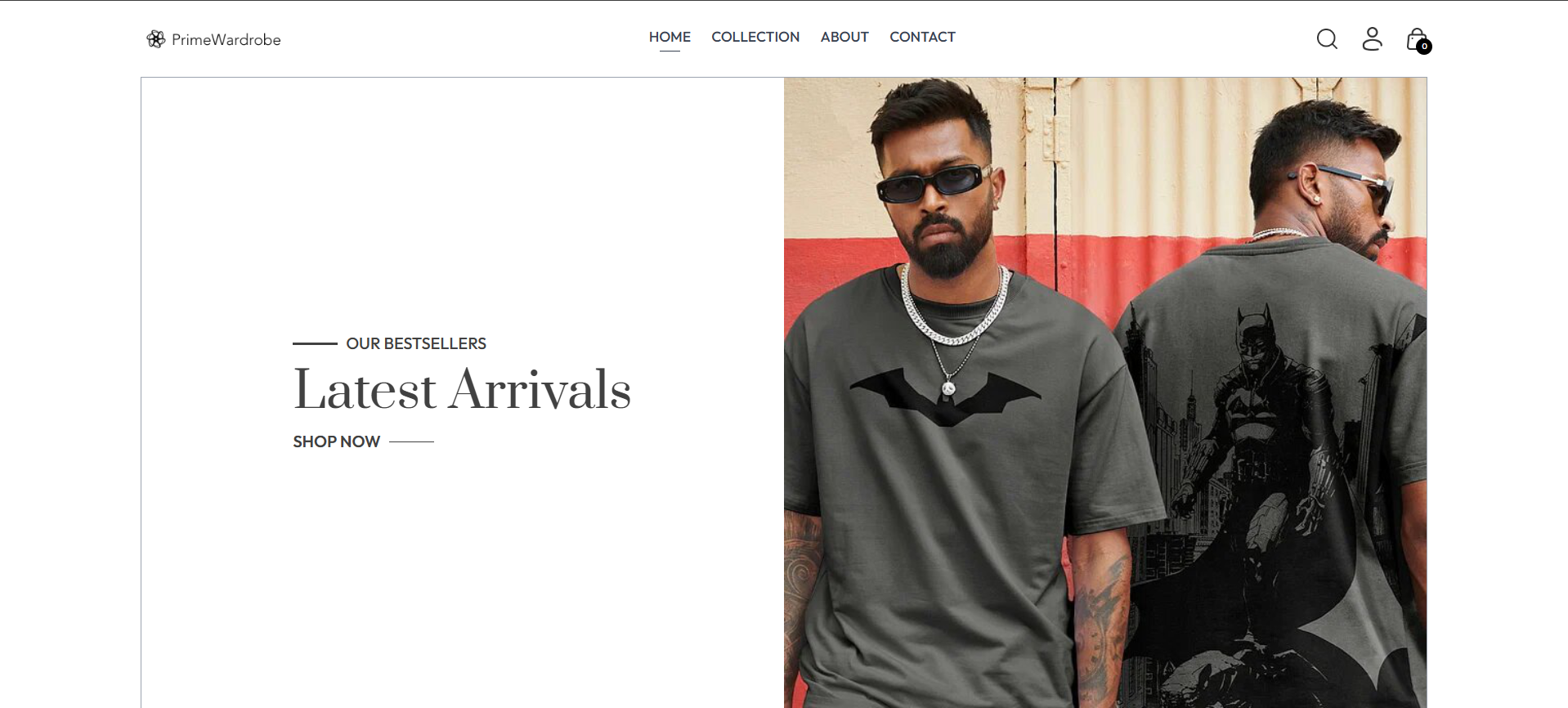
The platform fulfills its objective of delivering a user-friendly interface for browsing products, managing a shopping cart, and completing secure transactions. By leveraging the capabilities of the MERN stack, this project offers a robust foundation that can handle the basic requirements of an online store while remaining scalable and adaptable. This project has been a comprehensive exercise in full-stack development, covering database management, API creation, frontend design, and secure payment handling, contributing to a well-rounded e-commerce solution.

**5.2 Future Scope**

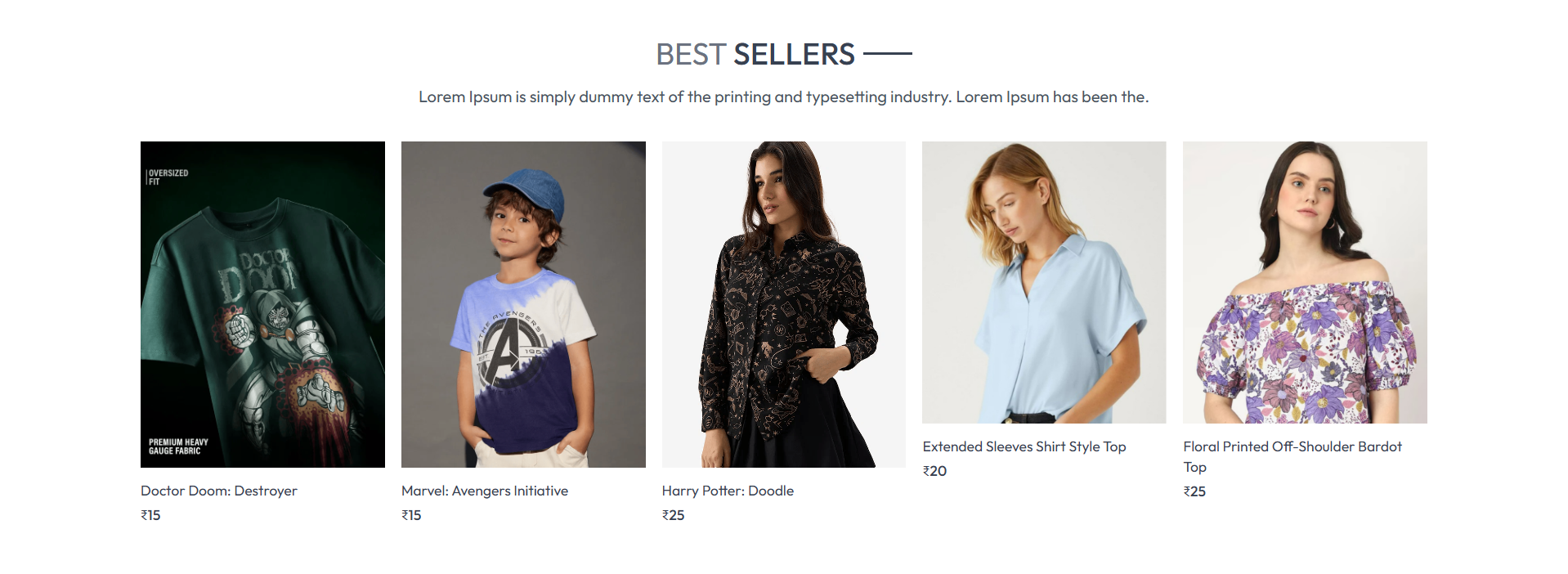
While the project accomplishes its primary objectives, there are several potential areas for enhancement:

* + 1. **Expanded Payment Options:** Adding additional payment gateways like PayPal, Razorpay, or regional options would provide greater flexibility and accessibility for users in different regions.
    2. **Scalability and Load Balancing:** For larger-scale deployment, incorporating load balancing and database sharding could improve performance and handle higher traffic loads during peak times.
    3. **Advanced Security Measures:** While Stripe handles payment security, additional features such as two-factor authentication (2FA) and activity logging for user accounts could provide extra security.
    4. **Analytics and Reporting:** Adding an analytics dashboard would allow administrators to track sales, monitor product popularity, and understand user behavior, which could inform data-driven decisions and targeted marketing.
    5. **Mobile Application:** Developing a mobile app version of the platform, using frameworks like React Native, could expand accessibility and offer a dedicated shopping experience for mobile users.

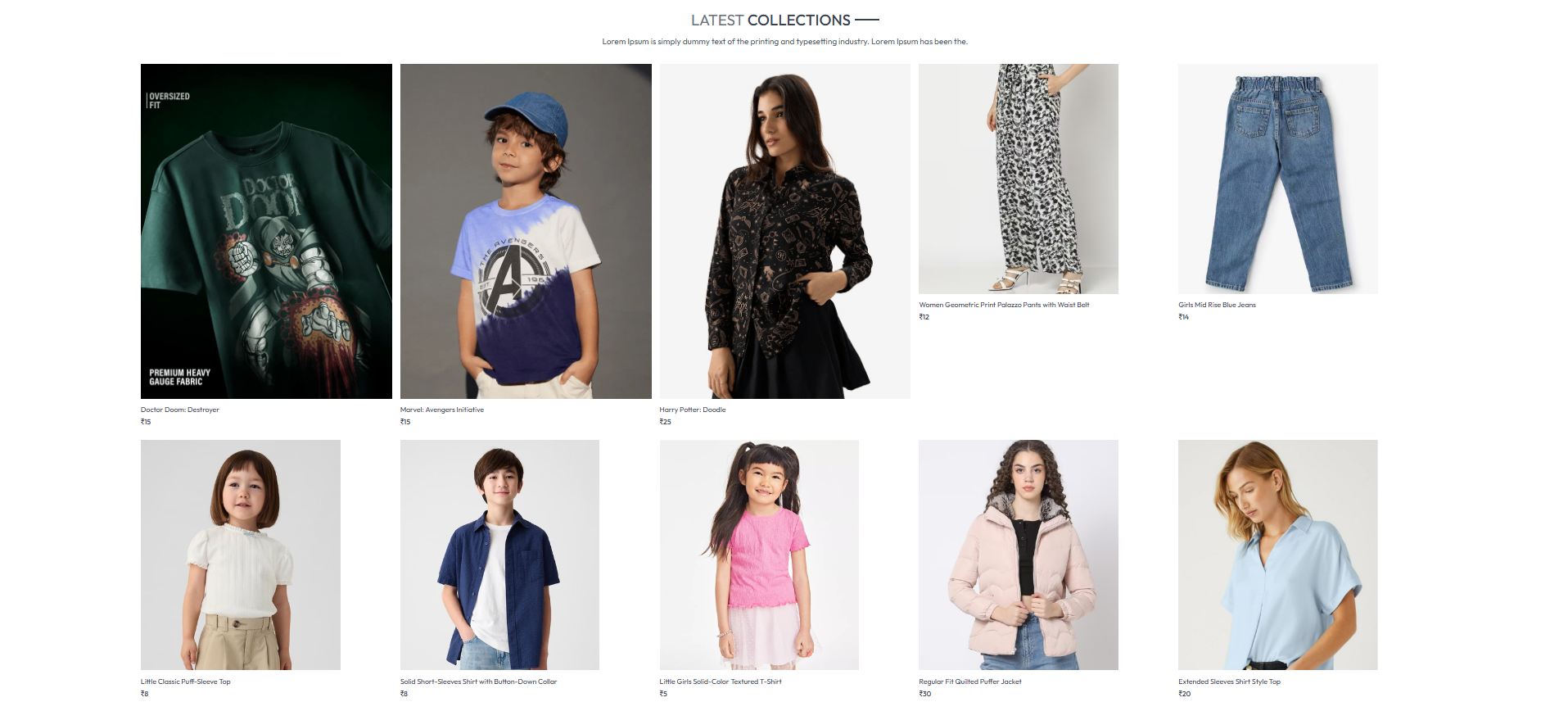
**Screenshots of the Web Application Interface**

****

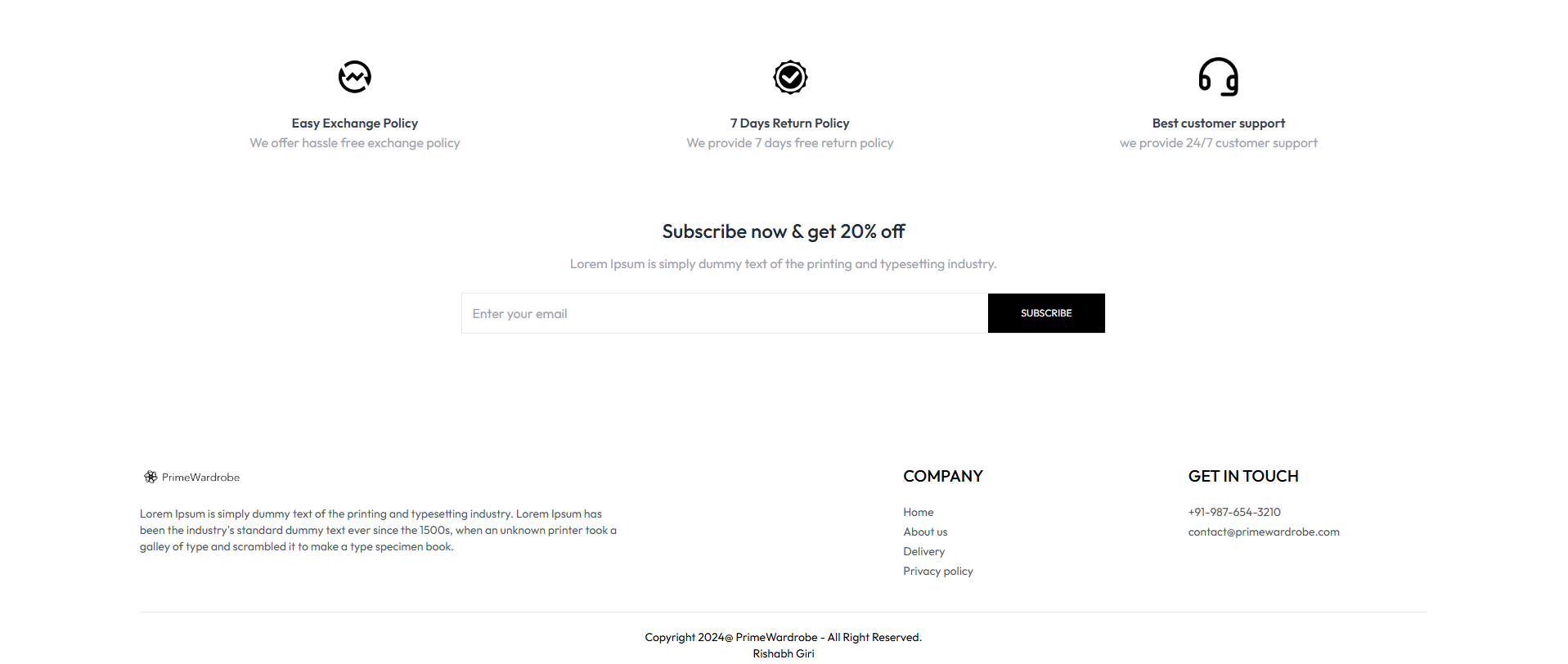
**Fig.no.1.1- Landing Page**

****

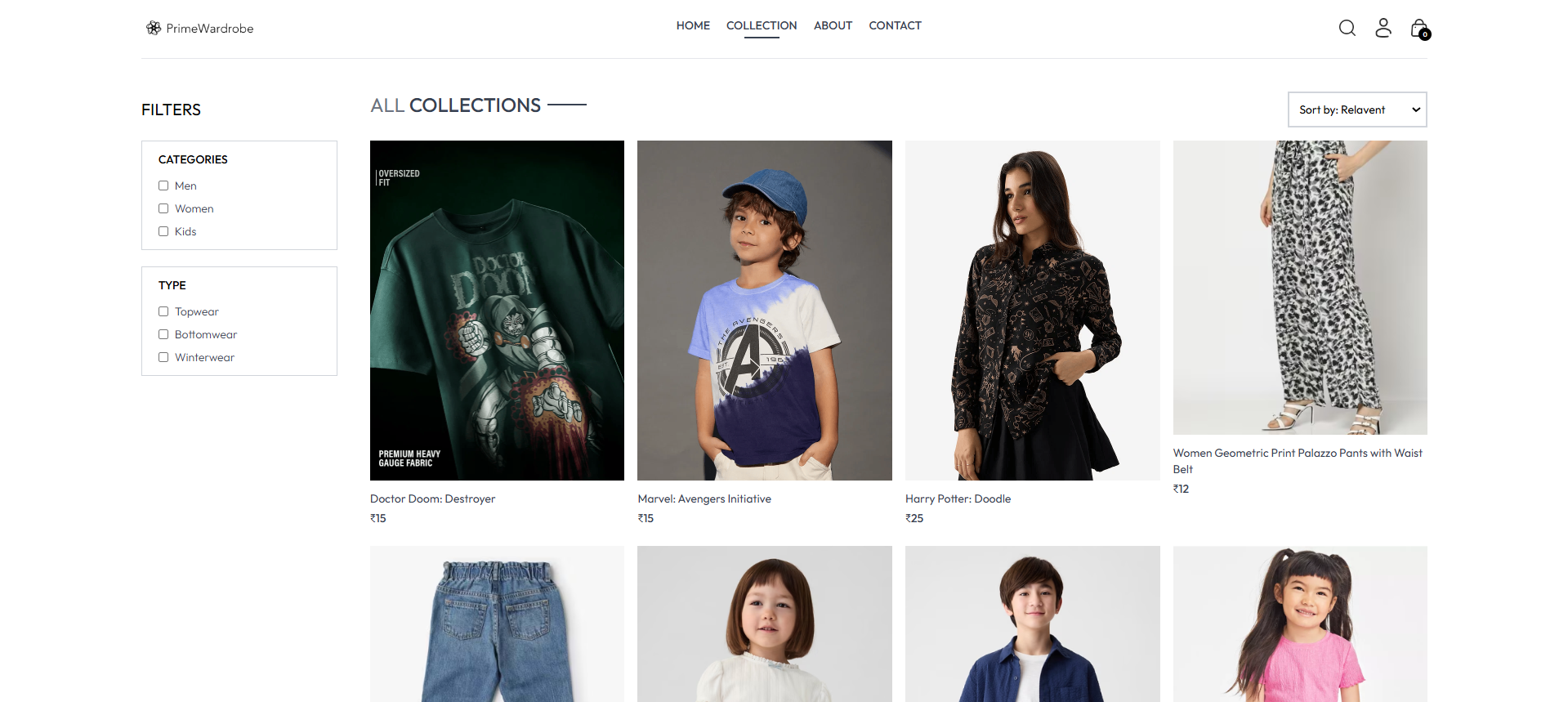
**Fig.no.1.2- Landing Page**

****

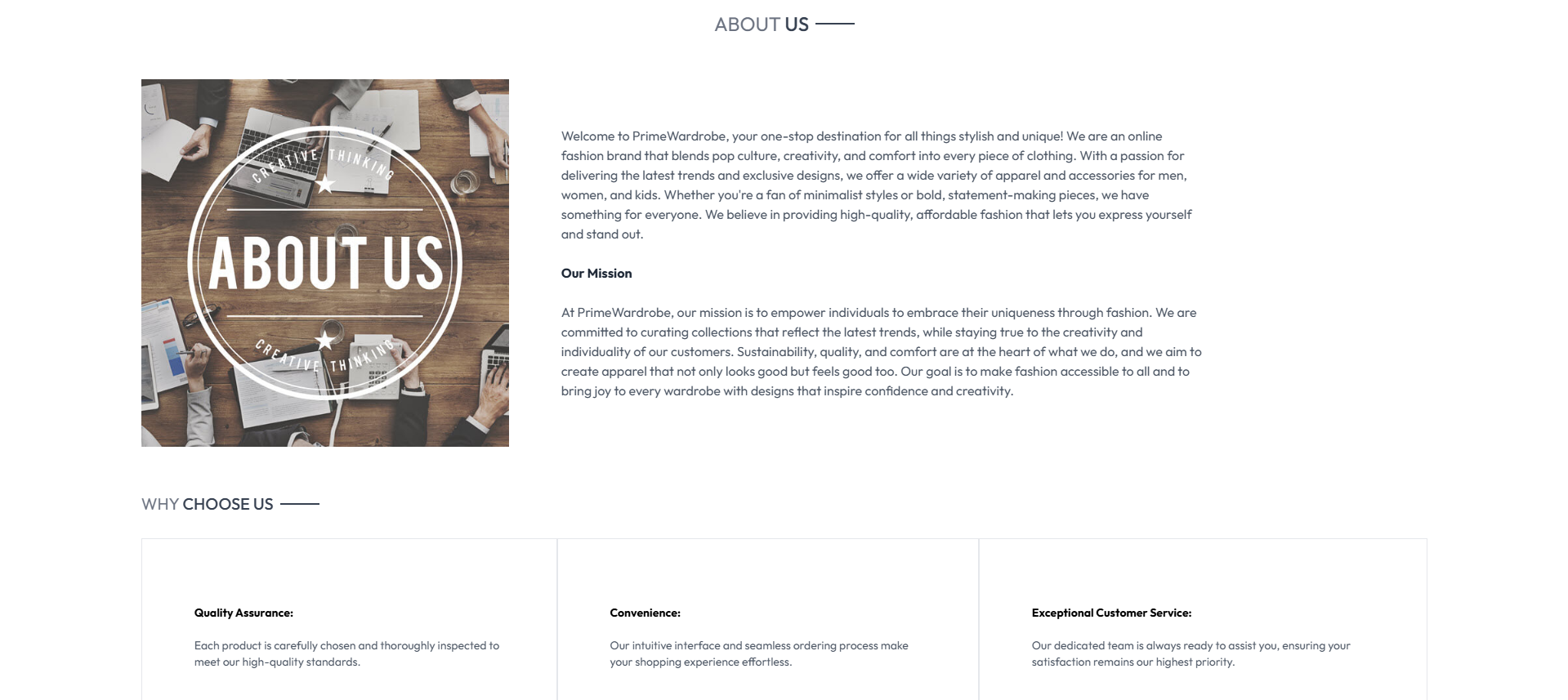
**Fig.no.1.3- Landing Page**

****

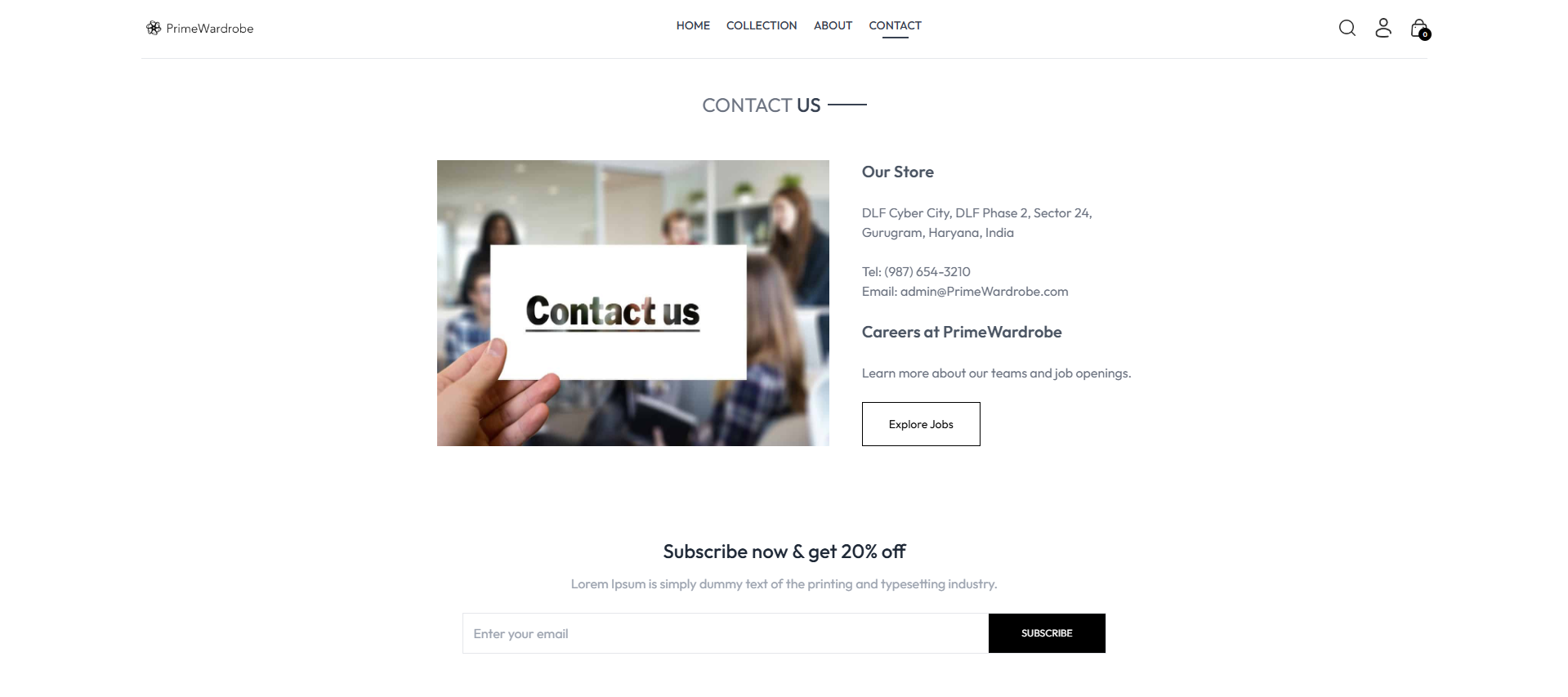
**Fig.no.1.4- Landing Page**

****

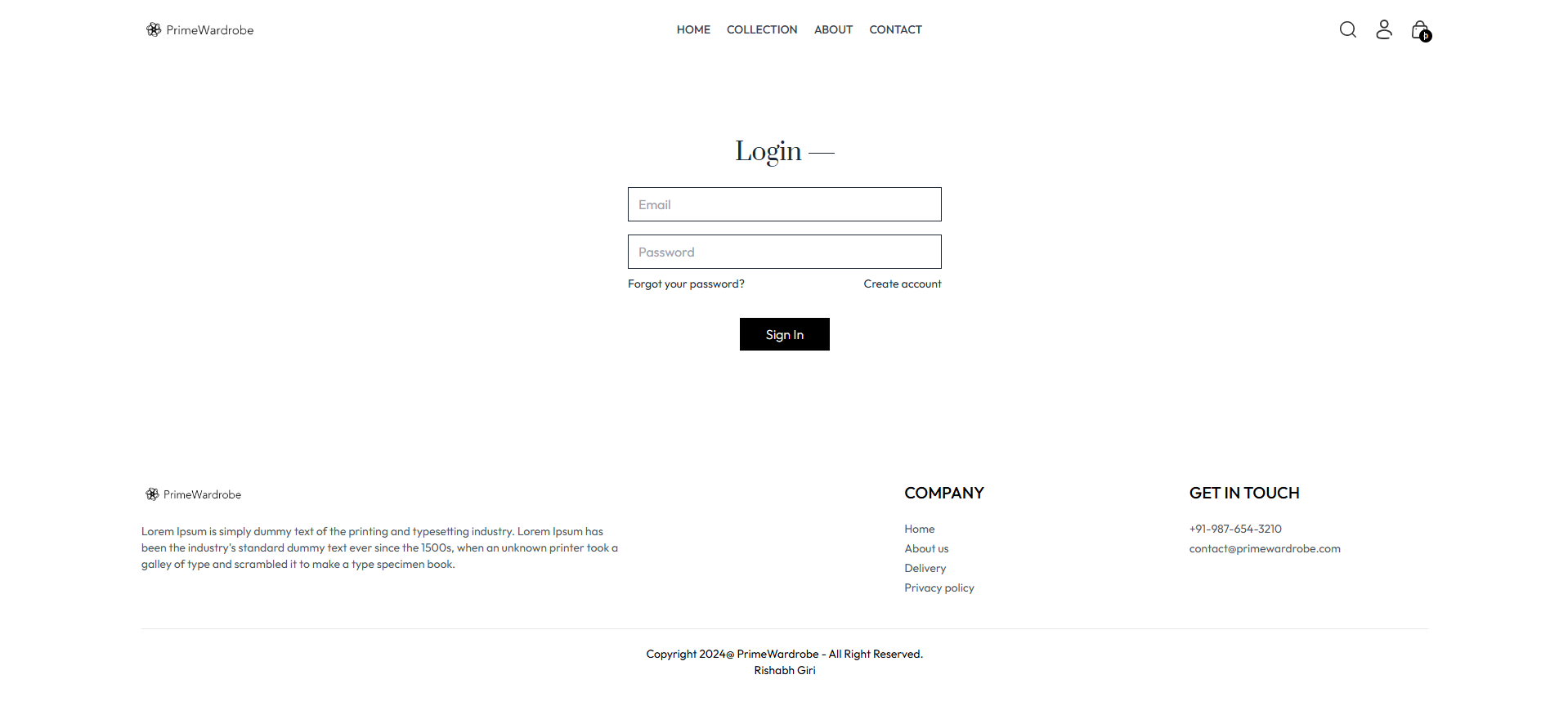
**Fig.no.2- Collection Page**

****

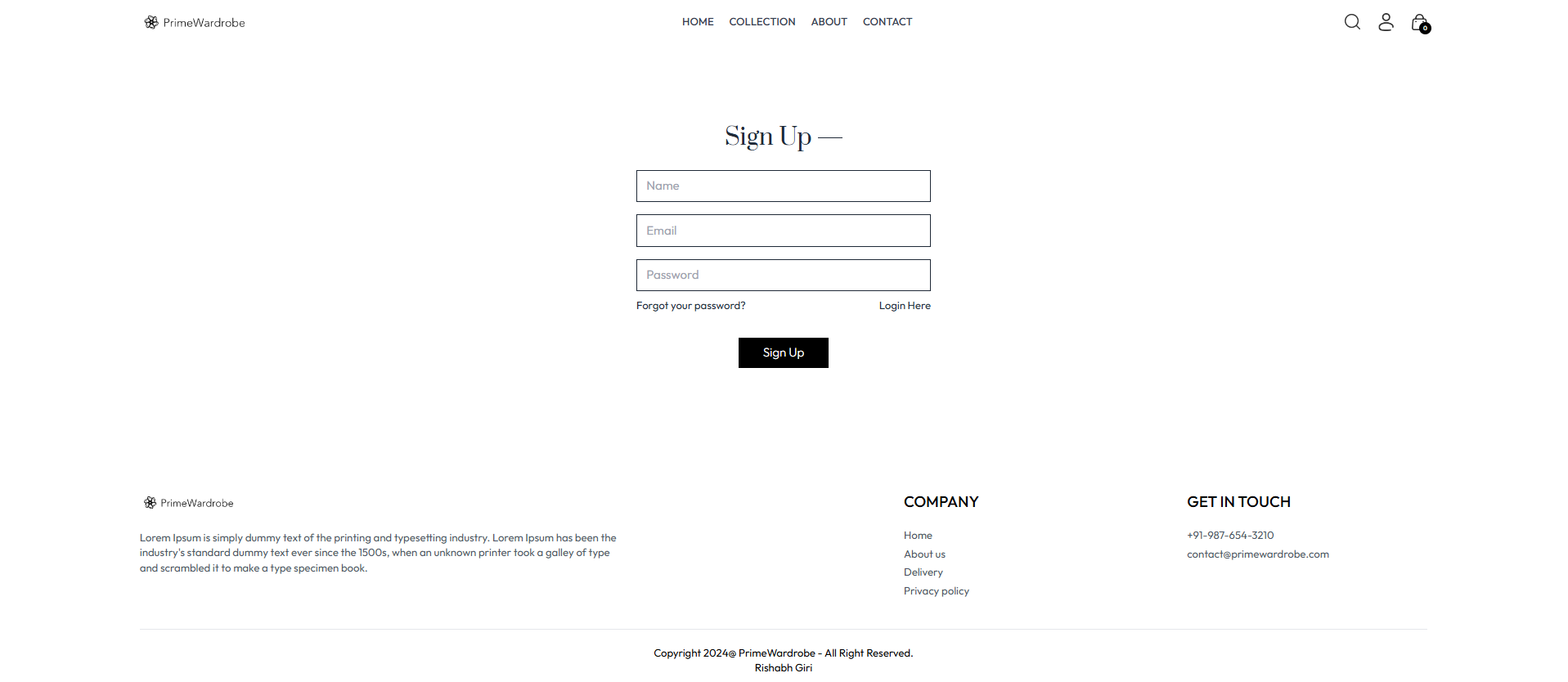
**Fig.no.3- About Page**

****

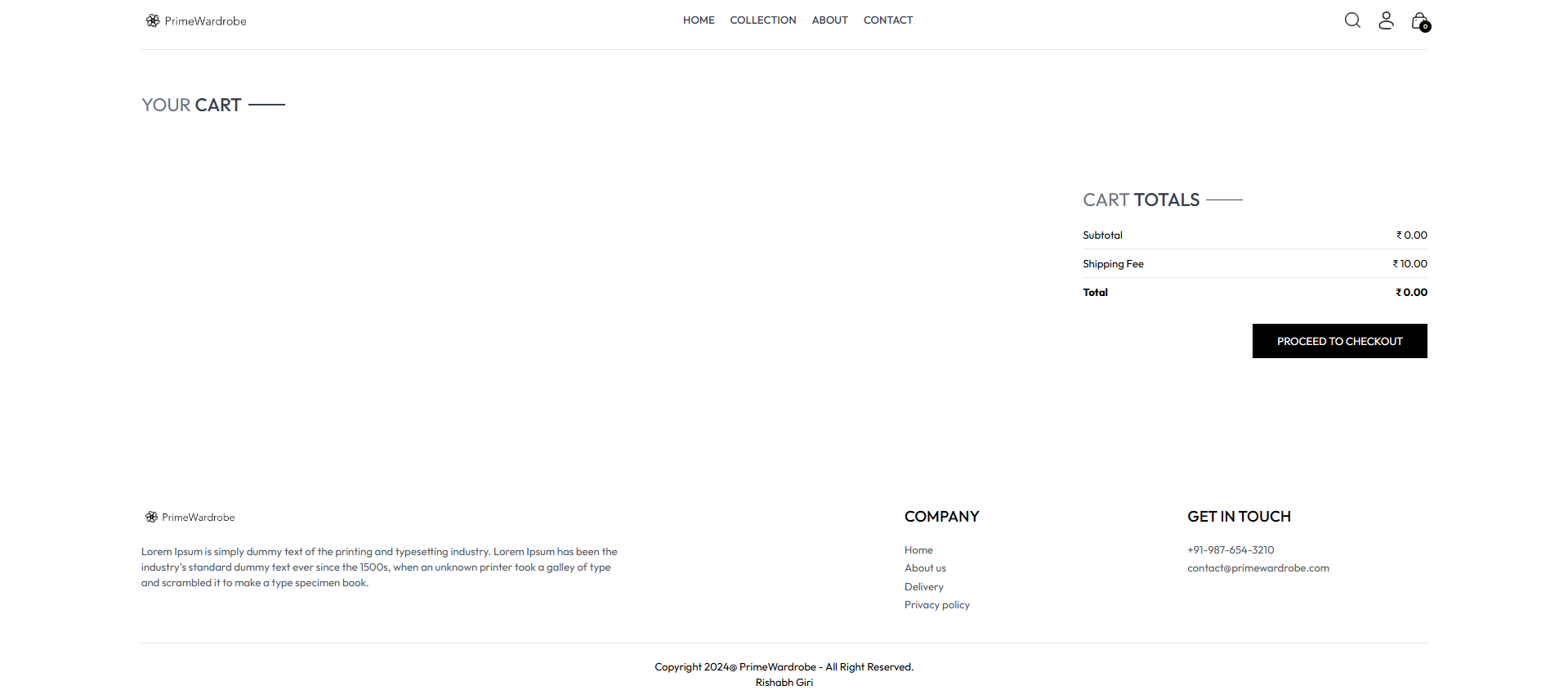
**Fig.no.4-Contact Us Page**

****

**Fig.no.5-Login Page**

****

**Fig.no.6-Sign Up Page**

****

**Fig.no.7-Cart Page**

**Codes of the Web Application Interface**

* **Navbar.jsx**

|  |
| --- |
| import React, { useContext, useState } from 'react'  import {assets} from '../assets/assets'  import { Link, NavLink } from 'react-router-dom'  import { ShopContext } from '../context/ShopContext';  const Navbar = () => {      const [visible,setVisible] = useState(false);      const {setShowSearch , getCartCount , navigate, token, setToken, setCartItems} = useContext(ShopContext);      const logout = () => {          navigate('/login')          localStorage.removeItem('token')          setToken('')          setCartItems({})      }    return (      <div className='flex items-center justify-between py-5 font-medium'>          <Link to='/'><img src={assets.logo} className='w-36' alt="" /></Link>        <ul className='hidden sm:flex gap-5 text-sm text-gray-700'>            <NavLink to='/' className='flex flex-col items-center gap-1'>              <p>HOME</p>              <hr className='w-2/4 border-none h-[1.5px] bg-gray-700 hidden' />          </NavLink>          <NavLink to='/collection' className='flex flex-col items-center gap-1'>              <p>COLLECTION</p>              <hr className='w-2/4 border-none h-[1.5px] bg-gray-700 hidden' />          </NavLink>          <NavLink to='/about' className='flex flex-col items-center gap-1'>              <p>ABOUT</p>              <hr className='w-2/4 border-none h-[1.5px] bg-gray-700 hidden' />          </NavLink>          <NavLink to='/contact' className='flex flex-col items-center gap-1'>              <p>CONTACT</p>              <hr className='w-2/4 border-none h-[1.5px] bg-gray-700 hidden' />          </NavLink>        </ul>        <div className='flex items-center gap-6'>              <img onClick={()=> { setShowSearch(true); navigate('/collection') }} src={assets.search\_icon} className='w-5 cursor-pointer' alt="" />                <div className='group relative'>                  <img onClick={()=> token ? null : navigate('/login') } className='w-5 cursor-pointer' src={assets.profile\_icon} alt="" />                  {/\* Dropdown Menu \*/}                  {token &&                  <div className='group-hover:block hidden absolute dropdown-menu right-0 pt-4'>                      <div className='flex flex-col gap-2 w-36 py-3 px-5  bg-slate-100 text-gray-500 rounded'>                          <p className='cursor-pointer hover:text-black'>My Profile</p>                          <p onClick={()=>navigate('/orders')} className='cursor-pointer hover:text-black'>Orders</p>                          <p onClick={logout} className='cursor-pointer hover:text-black'>Logout</p>                      </div>                  </div>}              </div>              <Link to='/cart' className='relative'>                  <img src={assets.cart\_icon} className='w-5 min-w-5' alt="" />                  <p className='absolute right-[-5px] bottom-[-5px] w-4 text-center leading-4 bg-black text-white aspect-square rounded-full text-[8px]'>{getCartCount()}</p>              </Link>              <img onClick={()=>setVisible(true)} src={assets.menu\_icon} className='w-5 cursor-pointer sm:hidden' alt="" />        </div>          {/\* Sidebar menu for small screens \*/}          <div className={`absolute top-0 right-0 bottom-0 overflow-hidden bg-white transition-all ${visible ? 'w-full' : 'w-0'}`}>                  <div className='flex flex-col text-gray-600'>                      <div onClick={()=>setVisible(false)} className='flex items-center gap-4 p-3 cursor-pointer'>                          <img className='h-4 rotate-180' src={assets.dropdown\_icon} alt="" />                          <p>Back</p>                      </div>                      <NavLink onClick={()=>setVisible(false)} className='py-2 pl-6 border' to='/'>HOME</NavLink>                      <NavLink onClick={()=>setVisible(false)} className='py-2 pl-6 border' to='/collection'>COLLECTION</NavLink>                      <NavLink onClick={()=>setVisible(false)} className='py-2 pl-6 border' to='/about'>ABOUT</NavLink>                      <NavLink onClick={()=>setVisible(false)} className='py-2 pl-6 border' to='/contact'>CONTACT</NavLink>                  </div>          </div>      </div>    )  }  export default Navbar |

* **Hero.jsx**

|  |
| --- |
| import React from 'react'  import { assets } from '../assets/assets'  const Hero = () => {    return (      <div className='flex flex-col sm:flex-row border border-gray-400'>        {/\* Hero Left Side \*/}        <div className='w-full sm:w-1/2 flex items-center justify-center py-10 sm:py-0'>              <div className='text-[#414141]'>                  <div className='flex items-center gap-2'>                      <p className='w-8 md:w-11 h-[2px] bg-[#414141]'></p>                      <p className=' font-medium text-sm md:text-base'>OUR BESTSELLERS</p>                  </div>                  <h1 className='prata-regular text-3xl sm:py-3 lg:text-5xl leading-relaxed'>Latest Arrivals</h1>                  <div className='flex items-center gap-2'>                      <p className='font-semibold text-sm md:text-base'>SHOP NOW</p>                      <p className='w-8 md:w-11 h-[1px] bg-[#414141]'></p>                  </div>              </div>        </div>        {/\* Hero Right Side \*/}        <img className='w-full sm:w-1/2' src={assets.hero\_img} alt="" />      </div>    )  }  export default Hero |

* **BestSeller.jsx**

|  |
| --- |
| import React, { useContext, useEffect, useState } from 'react'  import { ShopContext } from '../context/ShopContext'  import Title from './Title';  import ProductItem from './ProductItem';  const BestSeller = () => {      const {products} = useContext(ShopContext);      const [bestSeller,setBestSeller] = useState([]);      useEffect(()=>{          const bestProduct = products.filter((item)=>(item.bestseller));          setBestSeller(bestProduct.slice(0,5))      },[products])    return (      <div className='my-10'>        <div className='text-center text-3xl py-8'>          <Title text1={'BEST'} text2={'SELLERS'}/>          <p className='w-3/4 m-auto text-xs sm:text-sm md:text-base text-gray-600'>          Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the.          </p>        </div>        <div className='grid grid-cols-2 sm:grid-cols-3 md:grid-cols-4 lg:grid-cols-5 gap-4 gap-y-6'>          {              bestSeller.map((item,index)=>(                  <ProductItem key={index} id={item.\_id} name={item.name} image={item.image} price={item.price} />              ))          }        </div>      </div>    )  }  export default BestSeller |

* **LatestCollection.jsx**

|  |
| --- |
| import React, { useContext, useEffect, useState } from 'react'  import { ShopContext } from '../context/ShopContext'  import Title from './Title';  import ProductItem from './ProductItem';  const LatestCollection = () => {      const { products } = useContext(ShopContext);      const [latestProducts,setLatestProducts] = useState([]);      useEffect(()=>{          setLatestProducts(products.slice(0,10));      },[products])    return (      <div className='my-10'>        <div className='text-center py-8 text-3xl'>            <Title text1={'LATEST'} text2={'COLLECTIONS'} />            <p className='w-3/4 m-auto text-xs sm:text-sm md:text-base text-gray-600'>            Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the.            </p>        </div>        {/\* Rendering Products \*/}        <div className='grid grid-cols-2 sm:grid-cols-3 md:grid-cols-4 lg:grid-cols-5 gap-4 gap-y-6'>          {            latestProducts.map((item,index)=>(              <ProductItem key={index} id={item.\_id} image={item.image} name={item.name} price={item.price} />            ))          }        </div>      </div>    )  }  export default LatestCollection |

* **OurPolicy.jsx**

|  |
| --- |
| import React from 'react'  import { assets } from '../assets/assets'  const OurPolicy = () => {    return (      <div className='flex flex-col sm:flex-row justify-around gap-12 sm:gap-2 text-center py-20 text-xs sm:text-sm md:text-base text-gray-700'>          <div>          <img src={assets.exchange\_icon} className='w-12 m-auto mb-5' alt="" />          <p className=' font-semibold'>Easy Exchange Policy</p>          <p className=' text-gray-400'>We offer hassle free  exchange policy</p>        </div>        <div>          <img src={assets.quality\_icon} className='w-12 m-auto mb-5' alt="" />          <p className=' font-semibold'>7 Days Return Policy</p>          <p className=' text-gray-400'>We provide 7 days free return policy</p>        </div>        <div>          <img src={assets.support\_img} className='w-12 m-auto mb-5' alt="" />          <p className=' font-semibold'>Best customer support</p>          <p className=' text-gray-400'>we provide 24/7 customer support</p>        </div>      </div>    )  }  export default OurPolicy |

* **NewsletterBox.jsx**

|  |
| --- |
| import React from 'react'  const NewsletterBox = () => {      const onSubmitHandler = (event) => {          event.preventDefault();      }    return (      <div className=' text-center'>        <p className='text-2xl font-medium text-gray-800'>Subscribe now & get 20% off</p>        <p className='text-gray-400 mt-3'>        Lorem Ipsum is simply dummy text of the printing and typesetting industry.        </p>        <form onSubmit={onSubmitHandler} className='w-full sm:w-1/2 flex items-center gap-3 mx-auto my-6 border pl-3'>          <input className='w-full sm:flex-1 outline-none' type="email" placeholder='Enter your email' required/>          <button type='submit' className='bg-black text-white text-xs px-10 py-4'>SUBSCRIBE</button>        </form>      </div>    )  }  export default NewsletterBox |

* **Footer.jsx**

|  |
| --- |
| import React from 'react'  import { assets } from '../assets/assets'  const Footer = () => {    return (      <div>        <div className='flex flex-col sm:grid grid-cols-[3fr\_1fr\_1fr] gap-14 my-10 mt-40 text-sm'>          <div>              <img src={assets.logo} className='mb-5 w-32' alt="" />              <p className='w-full md:w-2/3 text-gray-600'>              Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.              </p>          </div>          <div>              <p className='text-xl font-medium mb-5'>COMPANY</p>              <ul className='flex flex-col gap-1 text-gray-600'>                  <li>Home</li>                  <li>About us</li>                  <li>Delivery</li>                  <li>Privacy policy</li>              </ul>          </div>          <div>              <p className='text-xl font-medium mb-5'>GET IN TOUCH</p>              <ul className='flex flex-col gap-1 text-gray-600'>                  <li>+91-987-654-3210</li>                  <li>contact@primewardrobe.com</li>              </ul>          </div>        </div>          <div>              <hr />              <p className='py-5 text-sm text-center'>Copyright 2024@ PrimeWardrobe - All Right Reserved. <br />                Rishabh Giri              </p>          </div>      </div>    )  }  export default Footer |

* **SearchBar.jsx**

|  |
| --- |
| import React, { useContext, useEffect, useState } from 'react'  import { ShopContext } from '../context/ShopContext'  import { assets } from '../assets/assets';  import { useLocation } from 'react-router-dom';  const SearchBar = () => {      const { search, setSearch, showSearch, setShowSearch} = useContext(ShopContext);      const [visible,setVisible] = useState(false)      const location = useLocation();      useEffect(()=>{          if (location.pathname.includes('collection')) {              setVisible(true);          }          else {              setVisible(false)          }      },[location])   return showSearch && visible ? (      <div className='border-t border-b bg-gray-50 text-center'>        <div className='inline-flex items-center justify-center border border-gray-400 px-5 py-2 my-5 mx-3 rounded-full w-3/4 sm:w-1/2'>          <input value={search} onChange={(e)=>setSearch(e.target.value)} className='flex-1 outline-none bg-inherit text-sm' type="text" placeholder='Search'/>          <img className='w-4' src={assets.search\_icon} alt="" />        </div>        <img onClick={()=>setShowSearch(false)} className='inline w-3 cursor-pointer' src={assets.cross\_icon} alt="" />      </div>    ) : null  }  export default SearchBar |

* **Home.jsx**

|  |
| --- |
| import React from 'react'  import Hero from '../components/Hero'  import LatestCollection from '../components/LatestCollection'  import BestSeller from '../components/BestSeller'  import OurPolicy from '../components/OurPolicy'  import NewsletterBox from '../components/NewsletterBox'  const Home = () => {    return (      <div>        <Hero />        <BestSeller/>        <LatestCollection/>        <OurPolicy/>        <NewsletterBox/>      </div>    )  }  export default Home |

* **Collection.jsx**

|  |
| --- |
| import React, { useContext, useEffect, useState } from 'react'  import { ShopContext } from '../context/ShopContext'  import { assets } from '../assets/assets';  import Title from '../components/Title';  import ProductItem from '../components/ProductItem';  const Collection = () => {    const { products , search , showSearch } = useContext(ShopContext);    const [showFilter,setShowFilter] = useState(false);    const [filterProducts,setFilterProducts] = useState([]);    const [category,setCategory] = useState([]);    const [subCategory,setSubCategory] = useState([]);    const [sortType,setSortType] = useState('relavent')    const toggleCategory = (e) => {      if (category.includes(e.target.value)) {          setCategory(prev=> prev.filter(item => item !== e.target.value))      }      else{        setCategory(prev => [...prev,e.target.value])      }    }    const toggleSubCategory = (e) => {      if (subCategory.includes(e.target.value)) {        setSubCategory(prev=> prev.filter(item => item !== e.target.value))      }      else{        setSubCategory(prev => [...prev,e.target.value])      }    }    const applyFilter = () => {      let productsCopy = products.slice();      if (showSearch && search) {        productsCopy = productsCopy.filter(item => item.name.toLowerCase().includes(search.toLowerCase()))      }      if (category.length > 0) {        productsCopy = productsCopy.filter(item => category.includes(item.category));      }      if (subCategory.length > 0 ) {        productsCopy = productsCopy.filter(item => subCategory.includes(item.subCategory))      }      setFilterProducts(productsCopy)    }    const sortProduct = () => {      let fpCopy = filterProducts.slice();      switch (sortType) {        case 'low-high':          setFilterProducts(fpCopy.sort((a,b)=>(a.price - b.price)));          break;        case 'high-low':          setFilterProducts(fpCopy.sort((a,b)=>(b.price - a.price)));          break;        default:          applyFilter();          break;      }    }    useEffect(()=>{        applyFilter();    },[category,subCategory,search,showSearch,products])    useEffect(()=>{      sortProduct();    },[sortType])    return (      <div className='flex flex-col sm:flex-row gap-1 sm:gap-10 pt-10 border-t'>          {/\* Filter Options \*/}        <div className='min-w-60'>          <p onClick={()=>setShowFilter(!showFilter)} className='my-2 text-xl flex items-center cursor-pointer gap-2'>FILTERS            <img className={`h-3 sm:hidden ${showFilter ? 'rotate-90' : ''}`} src={assets.dropdown\_icon} alt="" />          </p>          {/\* Category Filter \*/}          <div className={`border border-gray-300 pl-5 py-3 mt-6 ${showFilter ? '' :'hidden'} sm:block`}>            <p className='mb-3 text-sm font-medium'>CATEGORIES</p>            <div className='flex flex-col gap-2 text-sm font-light text-gray-700'>              <p className='flex gap-2'>                <input className='w-3' type="checkbox" value={'Men'} onChange={toggleCategory}/> Men              </p>              <p className='flex gap-2'>                <input className='w-3' type="checkbox" value={'Women'} onChange={toggleCategory}/> Women              </p>              <p className='flex gap-2'>                <input className='w-3' type="checkbox" value={'Kids'} onChange={toggleCategory}/> Kids              </p>            </div>          </div>          {/\* SubCategory Filter \*/}          <div className={`border border-gray-300 pl-5 py-3 my-5 ${showFilter ? '' :'hidden'} sm:block`}>            <p className='mb-3 text-sm font-medium'>TYPE</p>            <div className='flex flex-col gap-2 text-sm font-light text-gray-700'>              <p className='flex gap-2'>                <input className='w-3' type="checkbox" value={'Topwear'} onChange={toggleSubCategory}/> Topwear              </p>              <p className='flex gap-2'>                <input className='w-3' type="checkbox" value={'Bottomwear'} onChange={toggleSubCategory}/> Bottomwear              </p>              <p className='flex gap-2'>                <input className='w-3' type="checkbox" value={'Winterwear'} onChange={toggleSubCategory}/> Winterwear              </p>            </div>          </div>        </div>        {/\* Right Side \*/}        <div className='flex-1'>          <div className='flex justify-between text-base sm:text-2xl mb-4'>              <Title text1={'ALL'} text2={'COLLECTIONS'} />              {/\* Porduct Sort \*/}              <select onChange={(e)=>setSortType(e.target.value)} className='border-2 border-gray-300 text-sm px-2'>                <option value="relavent">Sort by: Relavent</option>                <option value="low-high">Sort by: Low to High</option>                <option value="high-low">Sort by: High to Low</option>              </select>          </div>          {/\* Map Products \*/}          <div className='grid grid-cols-2 md:grid-cols-3 lg:grid-cols-4 gap-4 gap-y-6'>            {              filterProducts.map((item,index)=>(                <ProductItem key={index} name={item.name} id={item.\_id} price={item.price} image={item.image} />              ))            }          </div>        </div>      </div>    )  }  export default Collection |

* **About.jsx**

|  |
| --- |
| import React from 'react'  import Title from '../components/Title'  import { assets } from '../assets/assets'  import NewsletterBox from '../components/NewsletterBox'  const About = () => {    return (      <div>        <div className='text-2xl text-center pt-8 border-t'>            <Title text1={'ABOUT'} text2={'US'} />        </div>        <div className='my-10 flex flex-col md:flex-row gap-16'>            <img className='w-full md:max-w-[450px]' src={assets.about\_img} alt="" />            <div className='flex flex-col justify-center gap-6 md:w-2/4 text-gray-600'>                <p>Welcome to PrimeWardrobe, your one-stop destination for all things stylish and unique! We are an online fashion brand that blends pop culture, creativity, and comfort into every piece of clothing. With a passion for delivering the latest trends and exclusive designs, we offer a wide variety of apparel and accessories for men, women, and kids. Whether you're a fan of minimalist styles or bold, statement-making pieces, we have something for everyone. We believe in providing high-quality, affordable fashion that lets you express yourself and stand out.</p>                <b className='text-gray-800'>Our Mission</b>                <p>At PrimeWardrobe, our mission is to empower individuals to embrace their uniqueness through fashion. We are committed to curating collections that reflect the latest trends, while staying true to the creativity and individuality of our customers. Sustainability, quality, and comfort are at the heart of what we do, and we aim to create apparel that not only looks good but feels good too. Our goal is to make fashion accessible to all and to bring joy to every wardrobe with designs that inspire confidence and creativity.</p>            </div>        </div>        <div className=' text-xl py-4'>            <Title text1={'WHY'} text2={'CHOOSE US'} />        </div>        <div className='flex flex-col md:flex-row text-sm mb-20'>            <div className='border px-10 md:px-16 py-8 sm:py-20 flex flex-col gap-5'>              <b>Quality Assurance:</b>              <p className=' text-gray-600'>Each product is carefully chosen and thoroughly inspected to meet our high-quality standards.</p>            </div>            <div className='border px-10 md:px-16 py-8 sm:py-20 flex flex-col gap-5'>              <b>Convenience:</b>              <p className=' text-gray-600'>Our intuitive interface and seamless ordering process make your shopping experience effortless.</p>            </div>            <div className='border px-10 md:px-16 py-8 sm:py-20 flex flex-col gap-5'>              <b>Exceptional Customer Service:</b>              <p className=' text-gray-600'>Our dedicated team is always ready to assist you, ensuring your satisfaction remains our highest priority.</p>            </div>        </div>        <NewsletterBox/>        </div>    )  }  export default About |

* **Contact.jsx**

|  |
| --- |
| import React from 'react'  import Title from '../components/Title'  import { assets } from '../assets/assets'  import NewsletterBox from '../components/NewsletterBox'  const Contact = () => {    return (      <div>          <div className='text-center text-2xl pt-10 border-t'>            <Title text1={'CONTACT'} text2={'US'} />        </div>        <div className='my-10 flex flex-col justify-center md:flex-row gap-10 mb-28'>          <img className='w-full md:max-w-[480px]' src={assets.contact\_img} alt="" />          <div className='flex flex-col justify-center items-start gap-6'>            <p className='font-semibold text-xl text-gray-600'>Our Store</p>            <p className=' text-gray-500'>DLF Cyber City, DLF Phase 2, Sector 24,<br />  Gurugram, Haryana, India</p>            <p className=' text-gray-500'>Tel: (987) 654-3210 <br /> Email: admin@PrimeWardrobe.com</p>            <p className='font-semibold text-xl text-gray-600'>Careers at PrimeWardrobe</p>            <p className=' text-gray-500'>Learn more about our teams and job openings.</p>            <button className='border border-black px-8 py-4 text-sm hover:bg-black hover:text-white transition-all duration-500'>Explore Jobs</button>          </div>        </div>        <NewsletterBox/>      </div>    )  }  export default Contact |

* **Product.jsx**

|  |
| --- |
| import React, { useContext, useEffect, useState } from 'react'  import { useParams } from 'react-router-dom'  import { ShopContext } from '../context/ShopContext';  import { assets } from '../assets/assets';  import RelatedProducts from '../components/RelatedProducts';  const Product = () => {    const { productId } = useParams();    const { products, currency ,addToCart } = useContext(ShopContext);    const [productData, setProductData] = useState(false);    const [image, setImage] = useState('')    const [size,setSize] = useState('')    const fetchProductData = async () => {      products.map((item) => {        if (item.\_id === productId) {          setProductData(item)          setImage(item.image[0])          return null;        }      })    }    useEffect(() => {      fetchProductData();    }, [productId,products])    return productData ? (      <div className='border-t-2 pt-10 transition-opacity ease-in duration-500 opacity-100'>        {/\*----------- Product Data-------------- \*/}        <div className='flex gap-12 sm:gap-12 flex-col sm:flex-row'>          {/\*---------- Product Images------------- \*/}          <div className='flex-1 flex flex-col-reverse gap-3 sm:flex-row'>            <div className='flex sm:flex-col overflow-x-auto sm:overflow-y-scroll justify-between sm:justify-normal sm:w-[18.7%] w-full'>                {                  productData.image.map((item,index)=>(                    <img onClick={()=>setImage(item)} src={item} key={index} className='w-[24%] sm:w-full sm:mb-3 flex-shrink-0 cursor-pointer' alt="" />                  ))                }            </div>            <div className='w-full sm:w-[80%]'>                <img className='w-full h-auto' src={image} alt="" />            </div>          </div>          {/\* -------- Product Info ---------- \*/}          <div className='flex-1'>            <h1 className='font-medium text-2xl mt-2'>{productData.name}</h1>            <div className=' flex items-center gap-1 mt-2'>                <img src={assets.star\_icon} alt="" className="w-3 5" />                <img src={assets.star\_icon} alt="" className="w-3 5" />                <img src={assets.star\_icon} alt="" className="w-3 5" />                <img src={assets.star\_icon} alt="" className="w-3 5" />                <img src={assets.star\_dull\_icon} alt="" className="w-3 5" />                <p className='pl-2'>(122)</p>            </div>            <p className='mt-5 text-3xl font-medium'>{currency}{productData.price}</p>            <p className='mt-5 text-gray-500 md:w-4/5'>{productData.description}</p>            <div className='flex flex-col gap-4 my-8'>                <p>Select Size</p>                <div className='flex gap-2'>                  {productData.sizes.map((item,index)=>(                    <button onClick={()=>setSize(item)} className={`border py-2 px-4 bg-gray-100 ${item === size ? 'border-orange-500' : ''}`} key={index}>{item}</button>                  ))}                </div>            </div>            <button onClick={()=>addToCart(productData.\_id,size)} className='bg-black text-white px-8 py-3 text-sm active:bg-gray-700'>ADD TO CART</button>            <hr className='mt-8 sm:w-4/5' />            <div className='text-sm text-gray-500 mt-5 flex flex-col gap-1'>                <p>100% Original product.</p>                <p>Cash on delivery is available on this product.</p>    <p>Easy return and exchange policy within 7 days.</p>            </div>          </div>        </div>        {/\* ---------- Description & Review Section ------------- \*/}        <div className='mt-20'>          <div className='flex'>            <b className='border px-5 py-3 text-sm'>Description</b>            <p className='border px-5 py-3 text-sm'>Reviews (122)</p>          </div>          <div className='flex flex-col gap-4 border px-6 py-6 text-sm text-gray-500'>            <p>An e-commerce website is an online platform that facilitates the buying and selling of products or services over the internet. It serves as a virtual marketplace where businesses and individuals can showcase their products, interact with customers, and conduct transactions without the need for a physical presence. E-commerce websites have gained immense popularity due to their convenience, accessibility, and the global reach they offer.</p>            <p>E-commerce websites typically display products or services along with detailed descriptions, images, prices, and any available variations (e.g., sizes, colors). Each product usually has its own dedicated page with relevant information.</p>          </div>        </div>        {/\* --------- display related products ---------- \*/}        <RelatedProducts category={productData.category} subCategory={productData.subCategory} />      </div>    ) : <div className=' opacity-0'></div>  }  export default Product |

* **Cart.jsx**

|  |
| --- |
| import React, { useContext, useEffect, useState } from 'react'  import { ShopContext } from '../context/ShopContext'  import Title from '../components/Title';  import { assets } from '../assets/assets';  import CartTotal from '../components/CartTotal';  const Cart = () => {    const { products, currency, cartItems, updateQuantity, navigate } = useContext(ShopContext);    const [cartData, setCartData] = useState([]);    useEffect(() => {      if (products.length > 0) {        const tempData = [];        for (const items in cartItems) {          for (const item in cartItems[items]) {            if (cartItems[items][item] > 0) {              tempData.push({                \_id: items,                size: item,                quantity: cartItems[items][item]              })            }          }        }        setCartData(tempData);      }    }, [cartItems, products])    return (      <div className='border-t pt-14'>        <div className=' text-2xl mb-3'>          <Title text1={'YOUR'} text2={'CART'} />        </div>        <div>          {            cartData.map((item, index) => {              const productData = products.find((product) => product.\_id === item.\_id);              return (                <div key={index} className='py-4 border-t border-b text-gray-700 grid grid-cols-[4fr\_0.5fr\_0.5fr] sm:grid-cols-[4fr\_2fr\_0.5fr] items-center gap-4'>                  <div className=' flex items-start gap-6'>                    <img className='w-16 sm:w-20' src={productData.image[0]} alt="" />                    <div>                      <p className='text-xs sm:text-lg font-medium'>{productData.name}</p>                      <div className='flex items-center gap-5 mt-2'>                        <p>{currency}{productData.price}</p>                        <p className='px-2 sm:px-3 sm:py-1 border bg-slate-50'>{item.size}</p>                      </div>                    </div>                  </div>                  <input onChange={(e) => e.target.value === '' || e.target.value === '0' ? null : updateQuantity(item.\_id, item.size, Number(e.target.value))} className='border max-w-10 sm:max-w-20 px-1 sm:px-2 py-1' type="number" min={1} defaultValue={item.quantity} />                  <img onClick={() => updateQuantity(item.\_id, item.size, 0)}  className='w-4 mr-4 sm:w-5 cursor-pointer' src={assets.bin\_icon} alt="" />                </div>              )            })          }        </div>        <div className='flex justify-end my-20'>          <div className='w-full sm:w-[450px]'>            <CartTotal />            <div className=' w-full text-end'>              <button onClick={() => navigate('/place-order')} className='bg-black text-white text-sm my-8 px-8 py-3'>PROCEED TO CHECKOUT</button>            </div>          </div>        </div>      </div>    )  }  export default Cart |

* **Orders.jsx**

|  |
| --- |
| import React, { useContext, useEffect, useState } from 'react'  import { ShopContext } from '../context/ShopContext'  import Title from '../components/Title';  import axios from 'axios';  const Orders = () => {    const { backendUrl, token , currency} = useContext(ShopContext);    const [orderData,setorderData] = useState([])    const loadOrderData = async () => {      try {        if (!token) {          return null        }        const response = await axios.post(backendUrl + '/api/order/userorders',{},{headers:{token}})        if (response.data.success) {          let allOrdersItem = []          response.data.orders.map((order)=>{            order.items.map((item)=>{              item['status'] = order.status              item['payment'] = order.payment              item['paymentMethod'] = order.paymentMethod              item['date'] = order.date              allOrdersItem.push(item)            })          })          setorderData(allOrdersItem.reverse())        }        } catch (error) {        }    }    useEffect(()=>{      loadOrderData()    },[token])    return (      <div className='border-t pt-16'>          <div className='text-2xl'>              <Title text1={'MY'} text2={'ORDERS'}/>          </div>          <div>              {                orderData.map((item,index) => (                  <div key={index} className='py-4 border-t border-b text-gray-700 flex flex-col md:flex-row md:items-center md:justify-between gap-4'>                      <div className='flex items-start gap-6 text-sm'>                          <img className='w-16 sm:w-20' src={item.image[0]} alt="" />                          <div>                            <p className='sm:text-base font-medium'>{item.name}</p>                            <div className='flex items-center gap-3 mt-1 text-base text-gray-700'>                              <p>{currency}{item.price}</p>                              <p>Quantity: {item.quantity}</p>                              <p>Size: {item.size}</p>                            </div>                            <p className='mt-1'>Date: <span className=' text-gray-400'>{new Date(item.date).toDateString()}</span></p>                            <p className='mt-1'>Payment: <span className=' text-gray-400'>{item.paymentMethod}</span></p>                          </div>                      </div>                      <div className='md:w-1/2 flex justify-between'>                          <div className='flex items-center gap-2'>                              <p className='min-w-2 h-2 rounded-full bg-green-500'></p>                              <p className='text-sm md:text-base'>{item.status}</p>                          </div>                          <button onClick={loadOrderData} className='border px-4 py-2 text-sm font-medium rounded-sm'>Track Order</button>                      </div>                  </div>                ))              }          </div>      </div>    )  }  export default Orders |

* **PlaceOrder.jsx**

|  |
| --- |
| import React, { useContext, useState } from 'react'  import Title from '../components/Title'  import CartTotal from '../components/CartTotal'  import { assets } from '../assets/assets'  import { ShopContext } from '../context/ShopContext'  import axios from 'axios'  import { toast } from 'react-toastify'  const PlaceOrder = () => {      const [method, setMethod] = useState('cod');      const { navigate, backendUrl, token, cartItems, setCartItems, getCartAmount, delivery\_fee, products } = useContext(ShopContext);      const [formData, setFormData] = useState({          firstName: '',          lastName: '',          email: '',          street: '',          city: '',          state: '',          zipcode: '',          country: '',          phone: ''      })      const onChangeHandler = (event) => {          const name = event.target.name          const value = event.target.value          setFormData(data => ({ ...data, [name]: value }))      }      const initPay = (order) => {          const options = {              key: import.meta.env.VITE\_RAZORPAY\_KEY\_ID,              amount: order.amount,              currency: order.currency,              name:'Order Payment',              description:'Order Payment',              order\_id: order.id,              receipt: order.receipt,              handler: async (response) => {                  console.log(response)                  try {                        const { data } = await axios.post(backendUrl + '/api/order/verifyRazorpay',response,{headers:{token}})                      if (data.success) {                          navigate('/orders')                          setCartItems({})                      }                  } catch (error) {                      console.log(error)                      toast.error(error)                  }              }          }          const rzp = new window.Razorpay(options)          rzp.open()      }      const onSubmitHandler = async (event) => {          event.preventDefault()          try {              let orderItems = []              for (const items in cartItems) {                  for (const item in cartItems[items]) {                      if (cartItems[items][item] > 0) {                          const itemInfo = structuredClone(products.find(product => product.\_id === items))                          if (itemInfo) {                              itemInfo.size = item                              itemInfo.quantity = cartItems[items][item]                              orderItems.push(itemInfo)                          }                      }                  }              }              let orderData = {                  address: formData,                  items: orderItems,                  amount: getCartAmount() + delivery\_fee              }                switch (method) {                  // API Calls for COD                  case 'cod':                      const response = await axios.post(backendUrl + '/api/order/place',orderData,{headers:{token}})                      if (response.data.success) {                          setCartItems({})                          navigate('/orders')                      } else {                          toast.error(response.data.message)                      }                      break;                  case 'stripe':                      const responseStripe = await axios.post(backendUrl + '/api/order/stripe',orderData,{headers:{token}})                      if (responseStripe.data.success) {                          const {session\_url} = responseStripe.data                          window.location.replace(session\_url)                      } else {                          toast.error(responseStripe.data.message)                      }                      break;                  case 'razorpay':                      const responseRazorpay = await axios.post(backendUrl + '/api/order/razorpay', orderData, {headers:{token}})                      if (responseRazorpay.data.success) {                          initPay(responseRazorpay.data.order)                      }                      break;                  default:                      break;              }          } catch (error) {              console.log(error)              toast.error(error.message)          }      }      return (          <form onSubmit={onSubmitHandler} className='flex flex-col sm:flex-row justify-between gap-4 pt-5 sm:pt-14 min-h-[80vh] border-t'>              {/\* ------------- Left Side ---------------- \*/}              <div className='flex flex-col gap-4 w-full sm:max-w-[480px]'>                  <div className='text-xl sm:text-2xl my-3'>                      <Title text1={'DELIVERY'} text2={'INFORMATION'} />                  </div>                  <div className='flex gap-3'>                      <input required onChange={onChangeHandler} name='firstName' value={formData.firstName} className='border border-gray-300 rounded py-1.5 px-3.5 w-full' type="text" placeholder='First name' />                      <input required onChange={onChangeHandler} name='lastName' value={formData.lastName} className='border border-gray-300 rounded py-1.5 px-3.5 w-full' type="text" placeholder='Last name' />                  </div>                  <input required onChange={onChangeHandler} name='email' value={formData.email} className='border border-gray-300 rounded py-1.5 px-3.5 w-full' type="email" placeholder='Email address' />                  <input required onChange={onChangeHandler} name='street' value={formData.street} className='border border-gray-300 rounded py-1.5 px-3.5 w-full' type="text" placeholder='Street' />                  <div className='flex gap-3'>                      <input required onChange={onChangeHandler} name='city' value={formData.city} className='border border-gray-300 rounded py-1.5 px-3.5 w-full' type="text" placeholder='City' />                      <input onChange={onChangeHandler} name='state' value={formData.state} className='border border-gray-300 rounded py-1.5 px-3.5 w-full' type="text" placeholder='State' />                  </div>                  <div className='flex gap-3'>                      <input required onChange={onChangeHandler} name='zipcode' value={formData.zipcode} className='border border-gray-300 rounded py-1.5 px-3.5 w-full' type="number" placeholder='Zipcode' />                      <input required onChange={onChangeHandler} name='country' value={formData.country} className='border border-gray-300 rounded py-1.5 px-3.5 w-full' type="text" placeholder='Country' />                  </div>                  <input required onChange={onChangeHandler} name='phone' value={formData.phone} className='border border-gray-300 rounded py-1.5 px-3.5 w-full' type="number" placeholder='Phone' />              </div>              {/\* ------------- Right Side ------------------ \*/}              <div className='mt-8'>                  <div className='mt-8 min-w-80'>                      <CartTotal />                  </div>                  <div className='mt-12'>                      <Title text1={'PAYMENT'} text2={'METHOD'} />                      {/\* --------------- Payment Method Selection ------------- \*/}                      <div className='flex gap-3 flex-col lg:flex-row'>                          <div onClick={() => setMethod('stripe')} className='flex items-center gap-3 border p-2 px-3 cursor-pointer'>                              <p className={`min-w-3.5 h-3.5 border rounded-full ${method === 'stripe' ? 'bg-green-400' : ''}`}></p>                              <img className='h-5 mx-4' src={assets.stripe\_logo} alt="" />                          </div>                          <div onClick={() => setMethod('razorpay')} className='flex items-center gap-3 border p-2 px-3 cursor-pointer'>                              <p className={`min-w-3.5 h-3.5 border rounded-full ${method === 'razorpay' ? 'bg-green-400' : ''}`}></p>                              <img className='h-5 mx-4' src={assets.razorpay\_logo} alt="" />                          </div>                          <div onClick={() => setMethod('cod')} className='flex items-center gap-3 border p-2 px-3 cursor-pointer'>                              <p className={`min-w-3.5 h-3.5 border rounded-full ${method === 'cod' ? 'bg-green-400' : ''}`}></p>                              <p className='text-gray-500 text-sm font-medium mx-4'>CASH ON DELIVERY</p>                          </div>                      </div>                      <div className='w-full text-end mt-8'>                          <button type='submit' className='bg-black text-white px-16 py-3 text-sm'>PLACE ORDER</button>                      </div>                  </div>              </div>          </form>      )  }  export default PlaceOrder |

* **Login.jsx**

|  |
| --- |
| import React, { useContext, useEffect, useState } from 'react'  import { ShopContext } from '../context/ShopContext';  import axios from 'axios';  import { toast } from 'react-toastify';  const Login = () => {    const [currentState, setCurrentState] = useState('Login');    const { token, setToken, navigate, backendUrl } = useContext(ShopContext)    const [name,setName] = useState('')    const [password,setPasword] = useState('')    const [email,setEmail] = useState('')    const onSubmitHandler = async (event) => {        event.preventDefault();        try {          if (currentState === 'Sign Up') {              const response = await axios.post(backendUrl + '/api/user/register',{name,email,password})            if (response.data.success) {              setToken(response.data.token)              localStorage.setItem('token',response.data.token)            } else {              toast.error(response.data.message)            }          } else {            const response = await axios.post(backendUrl + '/api/user/login', {email,password})            if (response.data.success) {              setToken(response.data.token)              localStorage.setItem('token',response.data.token)            } else {              toast.error(response.data.message)            }          }        } catch (error) {          console.log(error)          toast.error(error.message)        }    }    useEffect(()=>{      if (token) {        navigate('/')      }    },[token])    return (      <form onSubmit={onSubmitHandler} className='flex flex-col items-center w-[90%] sm:max-w-96 m-auto mt-14 gap-4 text-gray-800'>          <div className='inline-flex items-center gap-2 mb-2 mt-10'>              <p className='prata-regular text-3xl'>{currentState}</p>              <hr className='border-none h-[1.5px] w-8 bg-gray-800' />          </div>          {currentState === 'Login' ? '' : <input onChange={(e)=>setName(e.target.value)} value={name} type="text" className='w-full px-3 py-2 border border-gray-800' placeholder='Name' required/>}          <input onChange={(e)=>setEmail(e.target.value)} value={email} type="email" className='w-full px-3 py-2 border border-gray-800' placeholder='Email' required/>          <input onChange={(e)=>setPasword(e.target.value)} value={password} type="password" className='w-full px-3 py-2 border border-gray-800' placeholder='Password' required/>          <div className='w-full flex justify-between text-sm mt-[-8px]'>              <p className=' cursor-pointer'>Forgot your password?</p>              {                currentState === 'Login'                ? <p onClick={()=>setCurrentState('Sign Up')} className=' cursor-pointer'>Create account</p>                : <p onClick={()=>setCurrentState('Login')} className=' cursor-pointer'>Login Here</p>              }          </div>          <button className='bg-black text-white font-light px-8 py-2 mt-4'>{currentState === 'Login' ? 'Sign In' : 'Sign Up'}</button>      </form>    )  }  export default Login |

* **Verify.jsx**

|  |
| --- |
| import React from 'react'  import { useContext } from 'react'  import { ShopContext } from '../context/ShopContext'  import { useSearchParams } from 'react-router-dom'  import { useEffect } from 'react'  import {toast} from 'react-toastify'  import axios from 'axios'  const Verify = () => {      const { navigate, token, setCartItems, backendUrl } = useContext(ShopContext)      const [searchParams, setSearchParams] = useSearchParams()        const success = searchParams.get('success')      const orderId = searchParams.get('orderId')      const verifyPayment = async () => {          try {              if (!token) {                  return null              }              const response = await axios.post(backendUrl + '/api/order/verifyStripe', { success, orderId }, { headers: { token } })              if (response.data.success) {                  setCartItems({})                  navigate('/orders')              } else {                  navigate('/cart')              }          } catch (error) {              console.log(error)              toast.error(error.message)          }      }      useEffect(() => {          verifyPayment()      }, [token])      return (          <div>          </div>      )  }  export default Verify |

**APPENDIX**

**Project Setup Details:**

* **Development tools:** Bun as the runtime environment, MongoDB for database management, Express.js for backend API development, React.js for frontend, and Stripe API for payment processing.
* **Issues Encountered:**
  + Hydration mismatches and integration issues with Radix UI components in Bun.
  + Addressed with specific configurations and troubleshooting steps to enhance compatibility and eliminate mismatches.

**Technology Stack:**

* **Frontend**: React.js for interactive UI, with Radix UI for accessible components.
* **Backend**: Node.js and Express.js for server and API management.
* **Database**: MongoDB for flexible, schema-less data handling.
* **Payment Integration**: Stripe API for secure, PCI-compliant transaction processing.
* **Tools**: Visual Studio Code for coding, Git for version control, and Bun for efficient runtime handling.

**Modules and Libraries:**

* **Authentication**: JSON Web Tokens (JWT) for secure login and session management.
* **UI Components**: Radix UI, providing accessible and customizable frontend elements.
* **State Management**: Context API for managing global state across React components.

**Testing and Validation:**

* Functional testing of individual modules for cart functionality, product catalog, and payment flow.
* Load testing using simulated user interactions to ensure the platform can handle increased traffic.
* Security testing focused on encryption, data integrity, and secure authentication protocols.

**Documentation and Reference Material:**

* MongoDB, React, Node.js, and Express.js official documentation were referred for best practices and troubleshooting.
* Stripe API documentation was extensively used to ensure proper implementation of secure payment features.

**Future Enhancements:**

* Expand payment options to include providers such as PayPal and Razorpay.
* Develop a mobile application version using React Native for enhanced mobile accessibility.
* Implement an analytics dashboard for tracking user behavior and sales trends.