LOVELY PROFESSIONAL UNIVERSITY

E-commerce Today's Fashion website

Software Requirements Specification

INT-222

Name:- Abhishek Kumar
Reg. No.:- 12106579
Section:- K21DA
Roll No.:- 15

Table of Contents

1. Introduction

Welcome to the Software Requirement Specification (SRS) document for ecommerce website shopper. In the digital age, ecommerce platforms play a crucial role in facilitating online transactions, connecting businesses with consumers worldwide. This document serves as a comprehensive guide outlining the requirements, functionalities, and specifications necessary for the development of a robust and scalable ecommerce platform tailored to meet the needs of modern online businesses.

1.1 Purpose

The SRS serves as a comprehensive guide that ensures everyone involved in the project understands the ecommerce website's scope, objectives, and constraints. It facilitates effective communication between technical and non-technical stakeholders, guiding the development process from inception to completion. Ultimately, it aims to deliver a high-quality, feature-rich ecommerce platform that meets the client's needs and expectations, while also being maintainable and scalable for future enhancements.

1.2 Scope

1. Identification of the Software Product:

Name: Shopper Ecommerce Website Platform

2. Description of the Software Product:

The ecommerce website platform will enable users to browse, search, purchase, and manage products and services online. It will include features such as user authentication, product catalog management, shopping cart functionality, checkout process, payment gateway integration, order management, and user account management.
It will not handle physical product shipping logistics or provide customer support services beyond basic account management.

3. Application of the Software:

Benefits, Objectives, and Goals:
The primary goal is to provide a seamless and intuitive online shopping experience for customers, enhancing user satisfaction and driving sales.

Objectives include:
Facilitating easy product discovery through intuitive navigation and search functionalities.
Streamlining the checkout process to minimize cart abandonment rates.
Ensuring secure and reliable payment processing through integration with trusted payment gateways.

Scope:
The software will focus on providing a user-friendly interface for both customers and administrators to manage products, orders, and user accounts.
It will support parameter-driven product searches, user-definable product categories, and customizable user profiles.
Consistency with higher-level specifications, such as System Requirement Specification, will be ensured to maintain coherence across all project documents.

1.4 References

This subsection should:
(1) Provide a complete list of all documents referenced elsewhere in the SRS, or in a separate, specified document.
(2) Identify each document by title, report number - if applicable - date, and publishing organization.
(3) Specify the sources from which the references can be obtained.
This information may be provided by reference to an appendix or to another document.

1.5 Overview

T In this subsection, we'll provide an overview of the content and organization of the Software Requirement Specification (SRS) document for the ecommerce website:

Description of SRS Content:

The rest of the SRS document contains detailed specifications and requirements essential for the development of the ecommerce website. It covers both functional and non-functional aspects of the system, including user interfaces, system features, external interfaces, performance requirements, security measures, and more.
Specific sections address requirements related to user authentication, product management, shopping cart functionality, checkout process, payment integration, order management, and user account management. Additionally, non-functional requirements such as performance, scalability, reliability, and usability are also detailed.
Organization of the SRS:

The SRS document is organized into several sections, each addressing specific aspects of the ecommerce website development:
Introduction: Provides an overview of the SRS document, its purpose, and intended audience.
Overall Description: Offers a high-level description of the ecommerce website, including objectives, features, and constraints.
Specific Requirements: Details functional and non-functional requirements, categorized into distinct subsections for clarity.
External Interface Requirements: Describes interfaces with external systems or users, including hardware, software, and communication interfaces.

System Features: Lists and describes the core features and functionalities of the ecommerce website, often organized into logical groupings.
Other Nonfunctional Requirements: Addresses performance, reliability, security, usability, and other non-functional aspects of the system.
Appendices: Includes supplementary information such as glossary of terms, references, and supporting documentation.

2. General Description

In the General Description section of the SRS for the ecommerce website, we provide an overview of the broad factors influencing the product and its requirements. This section offers context without specifying detailed requirements, outlining key elements such as market analysis, business goals, user profiles, technology stack, regulatory considerations, project constraints, and risks/assumptions. Its purpose is to facilitate understanding and guide the development process toward meeting project objectives.

2.1 Product Perspective

In the Product Perspective subsection of the SRS for the ecommerce website, we provide context by comparing and contrasting our product with related products or projects. This helps stakeholders understand how our ecommerce platform fits into the broader market landscape and ecosystem. We may discuss how our platform differs in terms of features, target audience, technology stack, integration capabilities, or market positioning. This comparison aids in identifying unique selling points and potential areas for improvement or differentiation.

2.2 Product Functions

In the Product Functions subsection of the SRS for the ecommerce website, we provide a concise summary of the core functions that the software will perform. This includes essential features such as product browsing, searching, purchasing, and account management. Additionally, it may encompass functionalities related to user authentication, product catalog management, shopping cart management, checkout process, payment integration, order management, and user profile customization. This summary serves as a high-level overview of the software's capabilities, guiding further detailed specification and development efforts.

2.3 User Characteristics

In the User Characteristics subsection of the SRS for the ecommerce website, we describe the general traits and behaviors of the eventual users that will impact specific requirements. This includes demographics such as age, gender, location, education level, and technical proficiency. Additionally, we consider user preferences, needs, and expectations regarding the ecommerce platform, such as preferred payment methods, device usage (desktop, mobile), and shopping habits (frequency, average order value). Understanding these characteristics helps tailor the user experience and feature set to meet the diverse needs of the target audience effectively.

2.4 General Constraints

In the General Constraints subsection of the SRS for the ecommerce website, we outline various factors that will limit the developer's options when designing the system. These constraints may include technical limitations such as hardware or software compatibility requirements, as well as dependencies on external systems or services. Additionally, constraints related to budget, time, and resource availability may impact the development process. Regulatory compliance requirements, such as data protection laws or industry standards for online payment security, may also impose constraints on system design. By identifying and documenting these constraints, we ensure that the development team is aware of limitations from the outset and can plan accordingly to address them effectively.

2.5 Assumptions and Dependencies

In the Assumptions and Dependencies subsection of the SRS for the ecommerce website, we list all factors that could affect the requirements outlined in the document. These factors are not design constraints but rather potential changes that may impact the stated requirements. Assumptions might include specific hardware or software availability, regulatory compliance of third-party services, or user behavior patterns. Dependencies could involve external systems, APIs, or services that the ecommerce platform relies on for functionality. By documenting these assumptions and dependencies, we ensure that any changes affecting the requirements are identified and addressed promptly, maintaining the accuracy and relevance of the SRS throughout the development process.

3. Specific Requirements

In this section of the SRS for the ecommerce website, we will delineate the Design Requirements (D-requirements) crucial for guiding the software design, implementation, and testing phases of the project. Each requirement outlined here will meticulously adhere to the following criteria:

3.1 External Interface Requirements

3.1.1 User Interfaces

Description: The user interface is the primary interaction point for users accessing the ecommerce website. It includes web pages, forms, menus, and other elements that users interact with to browse products, make purchases, and manage their accounts.

Requirements:
Intuitive and responsive design to ensure usability across devices (desktop, mobile, tablet).
Visually appealing layout with clear navigation menus and search functionality.
Consistent design elements and branding throughout the website.
Interactive features such as product images, zoom functionality, and hover effects.
User-friendly checkout process with step-by-step guidance and error validation.
Accessibility features to accommodate users with disabilities, such as screen readers and keyboard navigation.

3.1.2 Hardware Interfaces

Description: Hardware interfaces define how the ecommerce website interacts with physical devices or components.
Requirements:

Compatibility with common hardware devices such as computers, smartphones, and tablets.
Support for various screen sizes and resolutions to ensure optimal display on different devices.
Integration with barcode scanners, printers, or other hardware devices for order fulfillment and inventory management (if applicable).

3.1.3 Software Interfaces

Description: Software interfaces specify how the ecommerce website interacts with other software systems or components.
Requirements:
Integration with payment gateways for processing transactions securely.
Compatibility with customer relationship management (CRM) systems for managing customer data and interactions.
Integration with inventory management systems for tracking product availability and stock levels.
Compatibility with content management systems (CMS) for managing website content and updates.
Support for third-party plugins or extensions for additional functionality, such as social media integration or analytics tracking.

3.1.4 Communications Interfaces

Description: Communications interfaces define how the ecommerce website communicates with external systems or services.
Requirements:
Secure communication protocols (HTTPS) to protect user data during transmission.
Integration with email services for order confirmation, shipping notifications, and customer support communications.
Support for real-time chat or messaging systems for customer support inquiries.
Integration with SMS or push notification services for order updates and promotions.

3.2 Functional Requirements

This section describes specific features of the software project. If desired, some requirements may be specified in the use-case format and listed in the Use Cases Section.

Correctness: This ensures that each requirement accurately reflects the intended functionality or behavior of the software. It prevents misunderstandings or misinterpretations during development, leading to a software product that meets the client's expectations.

Traceability: Each requirement will be linked backward and forward to prior and future artifacts within the project documentation, such as user stories, use cases, and test cases. This traceability ensures that every aspect of the software is accounted for and aligns with the overall project objectives.

Unambiguity: Requirements will be articulated clearly and precisely, leaving no room for confusion or multiple interpretations. Clear requirements reduce the risk of misunderstandings among stakeholders and developers, leading to a more efficient development process.

Verifiability: All requirements will be testable, meaning that their implementation can be verified through rigorous testing procedures. This ensures that the software meets the specified criteria and functions as intended, enhancing its reliability and quality.

Prioritization: Requirements will be prioritized based on their importance and stability. Critical functionalities are addressed first, allowing for the delivery of essential features early in the development process. Prioritization enables incremental development, allowing for flexibility and adaptation to changing project needs.

Completeness: Each requirement will comprehensively cover all essential aspects of the software's functionality. This ensures that no critical features are overlooked or omitted, preventing gaps or ambiguity in the software's specification.

Consistency: Requirements will maintain consistency with each other and with higher-level specifications. This eliminates contradictions or conflicts within the documentation, ensuring a coherent and unified understanding of the software's requirements.

Uniqueness: Each requirement will have a unique identifier, such as a numbering or labeling scheme. This facilitates easy reference and tracking throughout the development lifecycle, enabling efficient communication and management of project requirements

3.2.1 Functional Requirement 1: User Registration

3.2.1.1 Introduction

- *This feature allows users to create accounts on the ecommerce website, enabling personalized shopping experiences and order tracking.*

3.2.1.2 Inputs

- *User-provided information: Name, email address, password, etc.*

3.2.1.3 Processing

- *Validate input data for completeness and correctness.*
- *Check for duplicate email addresses to ensure uniqueness.*
- *Encrypt and securely store user passwords in the database.*

3.2.1.4 Outputs

- *Successful registration confirmation message.*
- *User account creation with unique identifier.*

3.2.1.5 Error Handling

- *Notify users of missing or invalid input data.*

- *Alert users if the email address is already associated with an existing account.*
- *Provide error messages for password strength requirements.*

3.2.2 Functional Requirement 2: Product Search and Filtering

3.2.2.1 Introduction

- *This feature allows users to search for products based on keywords and apply filters to narrow down search results.*

3.2.2.2 Inputs

- *User-provided search query: Keywords, product categories, filters (price range, size, color, etc.).*

3.2.2.3 Processing

- *Retrieve relevant products from the database based on search criteria.*
- *Apply selected filters to refine search results.*
- *Display search results based on relevance or other sorting criteria.*

3.2.2.4 Outputs

- *List of products matching the search query and selected filters.*
- *Visual representation of search results with product images and descriptions.*

3.2.2.5 Error Handling

- *Notify users if no products match the search criteria.*
- *Handle errors gracefully if the search functionality is temporarily unavailable.*
- *Provide suggestions for alternative search terms if the search yields no results.*

3.2.3 Functional Requirement 3: Product Detail Page

3.2.3.1 Introduction

- *This feature displays detailed information about individual products, including descriptions, specifications, pricing, and availability.*

3.2.3.2 Inputs

- *Selection of a specific product from search results or product categories.*

3.2.3.3 Processing

- *Retrieve product details from the database based on the selected product identifier.*
- *Calculate pricing based on any applicable discounts or promotions.*

- *Check product availability and display relevant information (in stock, out of stock, pre-order, etc.).*
-

3.2.3.4 Outputs

- *Comprehensive product information, including description, specifications, pricing, and availability.*
- *Option to add the product to the shopping cart for purchase.*

3.2.3.5 Error Handling

- *Notify users if the selected product is no longer available.*
- *Display error messages if there are issues retrieving product details from the database.*
- *Provide a message if the product page cannot be accessed due to technical difficulties.*

3.2.4 Functional Requirement 4: Shopping Cart Management

3.2.4.1 Introduction

- *This feature enables users to add, remove, and manage items in their shopping carts before proceeding to checkout.*

3.2.4.2 Inputs

- *User actions: Adding items to the cart, updating quantities, removing items..*

3.2.4.3 Processing

- *Store selected items and quantities in the user's session or database.*
- *Calculate the total price of items in the cart, including any applicable taxes or shipping fees.*
- *Update cart contents in real-time as users make changes.*

3.2.4.4 Outputs

- *Visual representation of the shopping cart with item details, quantities, and prices.*
- *Subtotal, taxes, shipping fees, and total order amount.*

3.2.4.5 Error Handling

- *Notify users if the cart is empty when attempting to proceed to checkout.*
- *Handle errors gracefully if the cart contents cannot be retrieved or updated.*
- *Display error messages if there are issues calculating the total price or applying discounts.*

3.2.5 Functional Requirement 5: User Authentication and Login

3.2.5.1 Introduction

- *This feature allows registered users to securely log in to their accounts to access personalized features and order history.*

3.2.5.2 Inputs

- *Verify user credentials against stored data in the database.*
- *Implement secure authentication mechanisms (e.g., hashing and salting passwords).*
- *Create and manage user sessions upon successful login.*

3.2.5.3 Processing

- *Store selected items and quantities in the user's session or database.*
- *Calculate the total price of items in the cart, including any applicable taxes or shipping fees.*
- *Update cart contents in real-time as users make changes.*

3.2.5.4 Outputs

- *Confirmation of successful authentication and login.*
- *Access to user-specific features and account information.*

3.2.5.5 Error Handling

- *Notify users if login credentials are incorrect or if the account does not exist.*
- *Implement measures to prevent brute force attacks and unauthorized access attempts.*
- *Display error messages if there are issues with the authentication process or session management.*

3.2.6 Functional Requirement 6: Checkout Process

3.2.6.1 Introduction

- *This feature guides users through the process of completing their purchases and provides options for payment and shipping.*

3.2.6.2 Inputs

- *User-selected items from the shopping cart.*
- *Shipping address, payment method, and any applicable discount codes.*

3.2.6.3 Processing

- *Collect and validate user-provided information for shipping and payment.*
- *Calculate order totals, including taxes and shipping fees.*
- *Generate order confirmation and invoice for the user and the system.*
-

3.2.6.4 Outputs

- *Summary of the order, including item details, shipping address, payment method, and total amount.*
- *Order confirmation number and receipt.*

3.2.6.5 Error Handling

- *Notify users of any incomplete or incorrect information during the checkout process.*
- *Handle errors related to payment processing or address validation.*
- *Provide assistance or alternative payment methods if there are issues completing the transaction.*

3.2.7 Functional Requirement 7: Order Management

3.2.7.1 Introduction

- *This feature allows administrators to manage orders, process payments, and update order statuses.*

3.2.8.2 Inputs

- *User-provided order details, payment information, and shipping preferences.*
- *Administrator actions: Reviewing, approving, or canceling orders.*

3.2.7.3 Processing

- *Store order details and payment information securely in the database.*
- *Update order status based on payment confirmation, shipping updates, or user actions.*
- *Generate order invoices and receipts for users and administrators.*

3.2.7.4 Outputs

- *Visual representation of order details, including items, quantities, prices, and shipping information.*
- *Options for administrators to update order status, print packing slips, or generate shipping labels.*

3.2.7.5 Error Handling

- *Notify administrators of any payment processing errors or incomplete orders.*

- *Handle errors related to updating order status or generating order documents.*
- *Provide options for manual intervention or assistance in resolving order-related issues.*

3.2.8 Functional Requirement 8: Product Inventory Management

3.2.8.1 Introduction

- *This feature allows administrators to manage product inventory, including adding new products, updating quantities, and marking items as out of stock.*

3.2.8.2 Inputs

- *Administrator actions: Adding new products, updating product details, adjusting inventory levels.*

3.2.8.3 Processing

- *Store product details, including descriptions, prices, and inventory levels, in the database.*
- *Update inventory counts based on user purchases, returns, or restocking activities.*
- *Generate alerts for low inventory levels or out-of-stock products.*

3.2.8.4 Outputs

- *Visual representation of product inventory, including current quantities and availability status.*
- *Options for administrators to update product details, images, and pricing.*

3.2.8.5 Error Handling

- *Notify administrators of any errors or issues encountered during inventory management tasks.*
- *Implement measures to prevent overselling or underselling of products.*
- *Provide options for manual intervention or adjustments in case of inventory discrepancies.*

3.2.9 Functional Requirement 9: Customer Support and Feedback

3.2.9.1 Introduction

- *This feature allows users to contact customer support for assistance, report issues, or provide feedback on their shopping experience.*

3.2.9.2 Inputs

- *User-provided information: Contact details, order numbers, descriptions of issues or feedback.*

3.2.9.3 Processing

- *Route user inquiries to appropriate customer support channels, such as email, chat, or ticketing systems.*
- *Assign and track support tickets to ensure timely resolution.*
- *Collect and analyze user feedback to identify areas for improvement.*

3.2.9.4 Outputs

- *Confirmation of user inquiry submission and receipt of feedback.*
- *Resolution of user issues and follow-up communication from customer support.*

3.2.9.5 Error Handling

- *Notify users if there are issues submitting inquiries or feedback.*
- *Handle errors related to tracking and resolving support tickets.*
- *Provide timely updates and communication to users throughout the support process.*

3.2.9 Functional Requirement 10: Website Administration and Content Management

3.2.10.1 Introduction

- *This feature enables administrators to manage website content, including product listings, promotional banners, and site settings.*

3.2.10.2 Inputs

- *Administrator actions: Adding, updating, or removing products, creating promotional campaigns, configuring site settings.*

3.2.10.3 Processing

- *Store website content, including product images, descriptions, and metadata, in the database.*
- *Implement content management tools for administrators to easily update and maintain website content.*
- *Track changes and revisions to website content for auditing and version control purposes.*

3.2.10.4 Outputs

- *Visual representation of website content management interface, including options for adding, editing, and deleting content.*
- *Updates to website content reflected in real-time for users.*

3.2.10.5 Error Handling

- *Notify administrators of any errors encountered during content management tasks, such as uploading images or updating product details.*
- *Implement rollback mechanisms to revert changes in case of accidental deletions or modifications.*
- *Provide options for manual intervention or assistance in resolving content-related issues.*

3.5 Non-Functional Requirements

Non-functional requirements may exist for the following attributes. Often these requirements must be achieved at a system-wide level rather than at a unit level. State the requirements in the following sections in measurable terms (e.g., 95% of transaction shall be processed in less than a second, system downtime may not exceed 1 minute per day, > 30 day MTBF value, etc).

3.5.1 Performance

Requirement: The ecommerce website shall process 95% of transactions in less than 2 seconds.

Justification: Ensures a responsive user experience and minimizes wait times during browsing and checkout.

3.5.2 Reliability

Requirement: The ecommerce website shall have a Mean Time Between Failures (MTBF) value of at least 30 days.

Justification: Ensures that the website operates reliably over an extended period, minimizing disruptions to user activities.

3.5.3 Availability

Requirement: The ecommerce website shall maintain at least 99.9% uptime, allowing for no more than 43.2 minutes of downtime per month.

Justification: Ensures that the website is consistently accessible to users, maximizing customer satisfaction and minimizing revenue loss due to downtime.

3.5.4 Security

Requirement: The ecommerce website shall comply with industry-standard security protocols, including HTTPS encryption for all data transmission and adherence to PCI DSS standards for handling payment information.

Justification: Ensures the protection of sensitive user information and prevents unauthorized access or data breaches.

3.5.5 Maintainability

Requirement: The ecommerce website shall be built using modular and well-documented code, with a code coverage of at least 80% for unit tests.

+Justification: Facilitates ongoing maintenance and updates to the website, reducing the risk of errors or regressions during development.

3.5.6 Portability

Requirement: The ecommerce website shall be compatible with major web browsers (Chrome, Firefox, Safari, Edge) with responsive design.

Justification: Ensures that the website functions correctly across different devices and platforms, maximizing accessibility for users.

3.7 Design Constraints

Specify design constrains imposed by other standards, company policies, hardware limitation, etc. that will impact this software project.

3.9 Other Requirements

Catchall section for any additional requirements.

4. Analysis Models

List all analysis models used in developing specific requirements previously given in this SRS. Each model should include an introduction and a narrative description. Furthermore, each model should be traceable the SRS's requirements.

4.1 Data Flow Diagrams (DFD)

Level 0 DFD:

Processes:

Manage Products

User Authentication

Manage Orders

Generate Reports

Admin Dashboard

Data Stores:

MongoDB Database

File Storage System for Images

External Entities:

Users (Customers)

Admin Users

Data Flows:

User Registration Details

Product Information

Order Details

Image Uploads

Authentication Data

Report Data

Interfaces:

Website Interface for Users

Admin Panel Interface

Level 1 DFD:

Processes:

Manage Products

Receive Product Details

Update Product Information

Delete Products

User Authentication

Verify User Credentials

Provide Access Tokens

Manage Orders

Process Order Details

Update Order Status

Generate Reports

Compile Sales Data

Create Analytics Reports

Admin Dashboard

Display Sales Metrics

Manage Product Listings

Data Stores:

MongoDB Database

File Storage System for Images

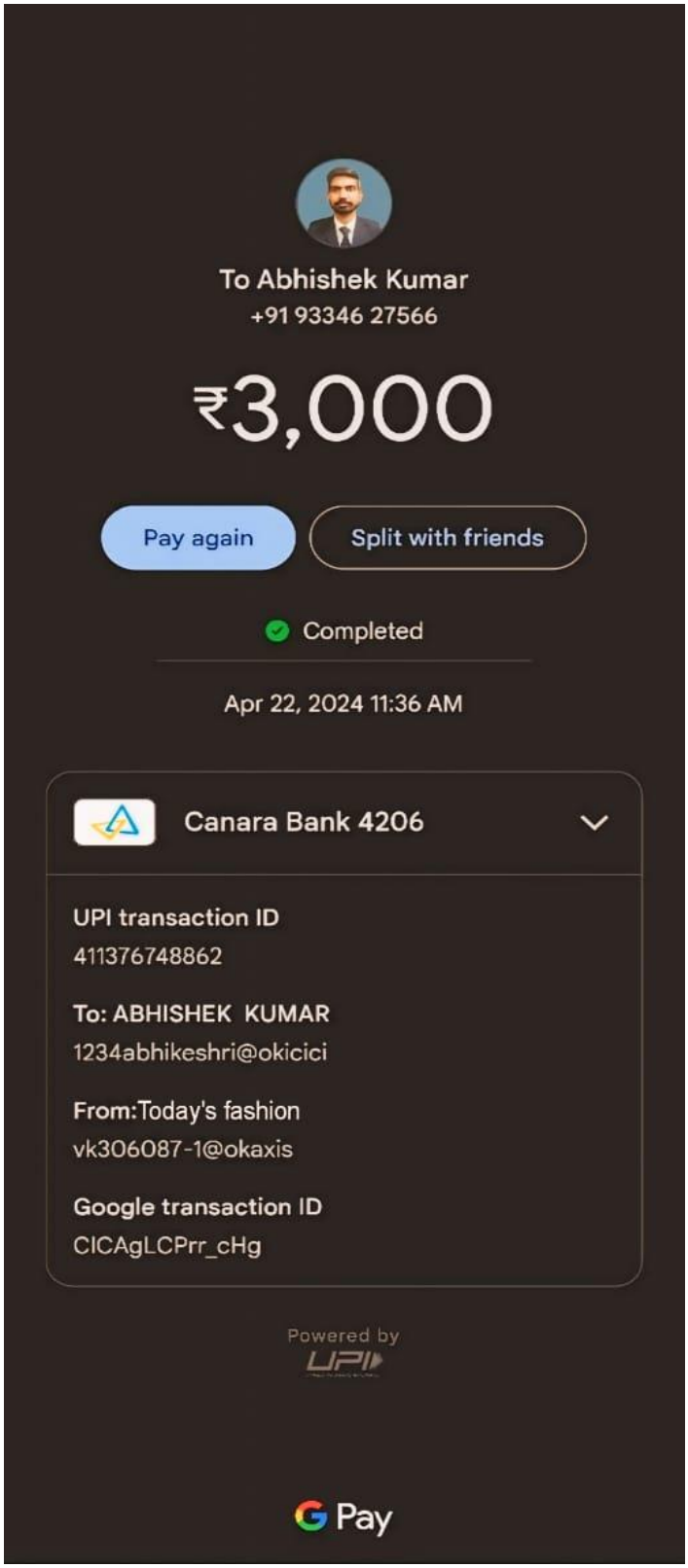
CLIENT APPROVAL FORM

PROJECT NAME	TODAY'S FASHION		
JOB LOCATION			
EST. START DATE		EST. FINISH DATE	30.04.24
PROJECT LEADER	Abhishek kumar	COMPANY	Student
CONTACT NAME	9334627566	ADDRESS	Lovely Professional University, Phagwara,Punjab
PHONE	9334627566		
EMAIL	1234abhikeshri@gmail.c		

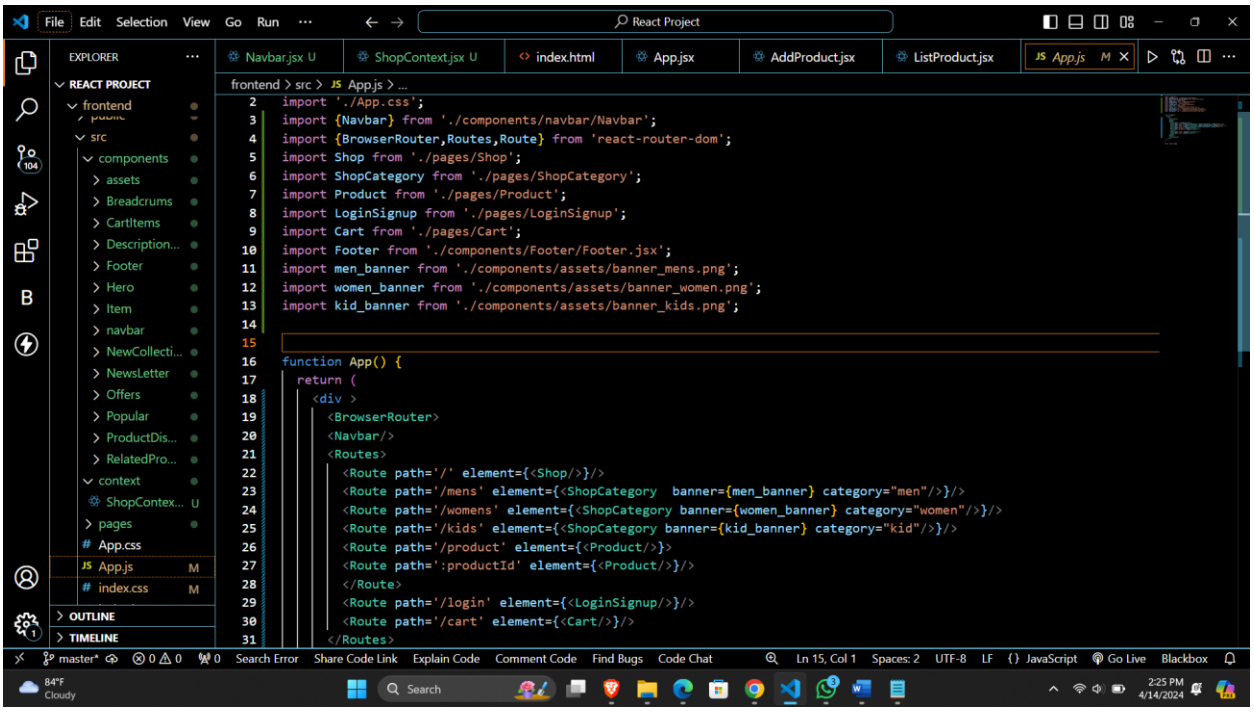
SUMMARY	Creating a complete e-commerce website using the MERN stack, including an overview of the website structure and navigation elements.
DESIRED OUTCOME	Desired outcome of the project is to deliver a fully operational e-commerce Today's Fashion website with a user-friendly interface,
ACTION TO COMPLETION	This includes frontend development, backend setup, admin panel creation, user authentication, and thorough testing. The desired outcome is a fully
BENEFITS OF PROJECT	The project's benefits lie in delivering an enhanced user experience, ensuring secure data management, enabling efficient admin functionality.
PROJECTED SCHEDULE	The project schedule for the Today's Fashion website covers 10 weeks, focusing on planning, design, development, testing, and content
PROJECTED BUDGET	3000
PROJECTED TEAM AND RESOURCE REQUIREMENTS	Resource requirements for developing a full-stack e-commerce website using the MERN stack would typically include roles such as Full-Stack
PROPOSAL MAY BE WITHDRAWN IF NOT ACCEPTED BY DATE OF	30.04.24

ACCEPTANCE OF PROPOSAL			
AUTHORIZED CLIENT SIGNATURE		DATE OF ACCEPTANCE	30.04.24

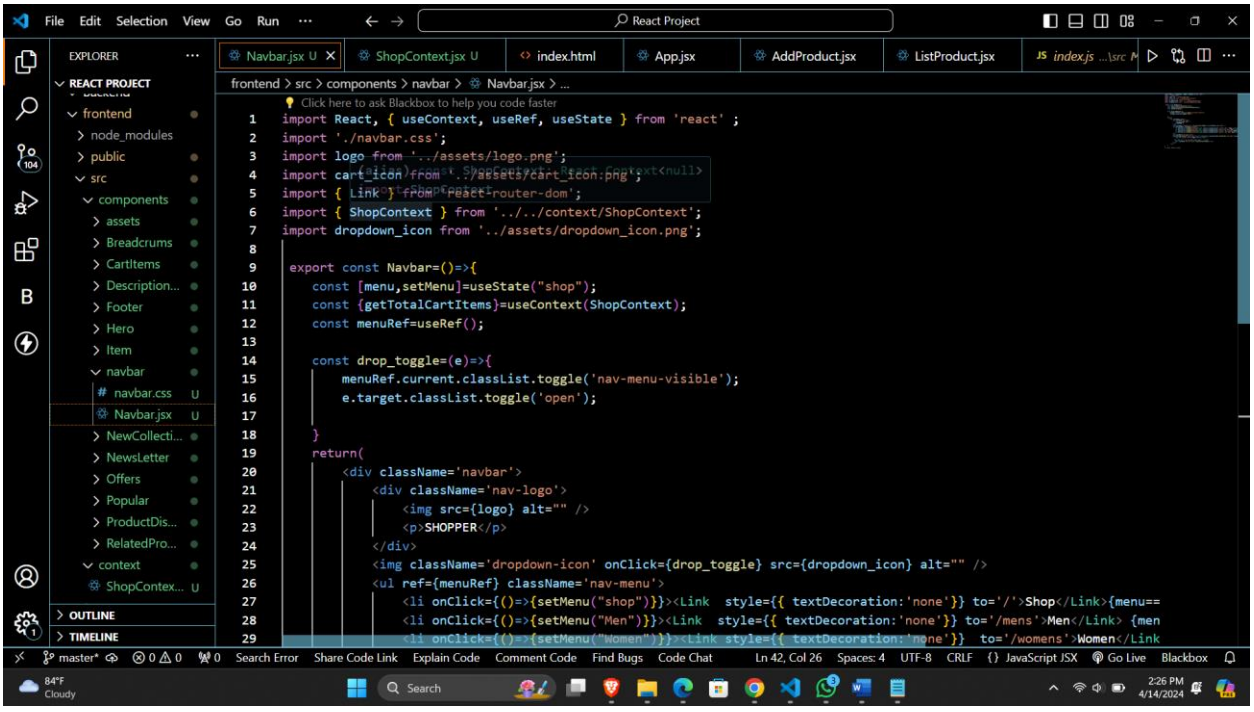
6. PAYMENT RECEIPT :-



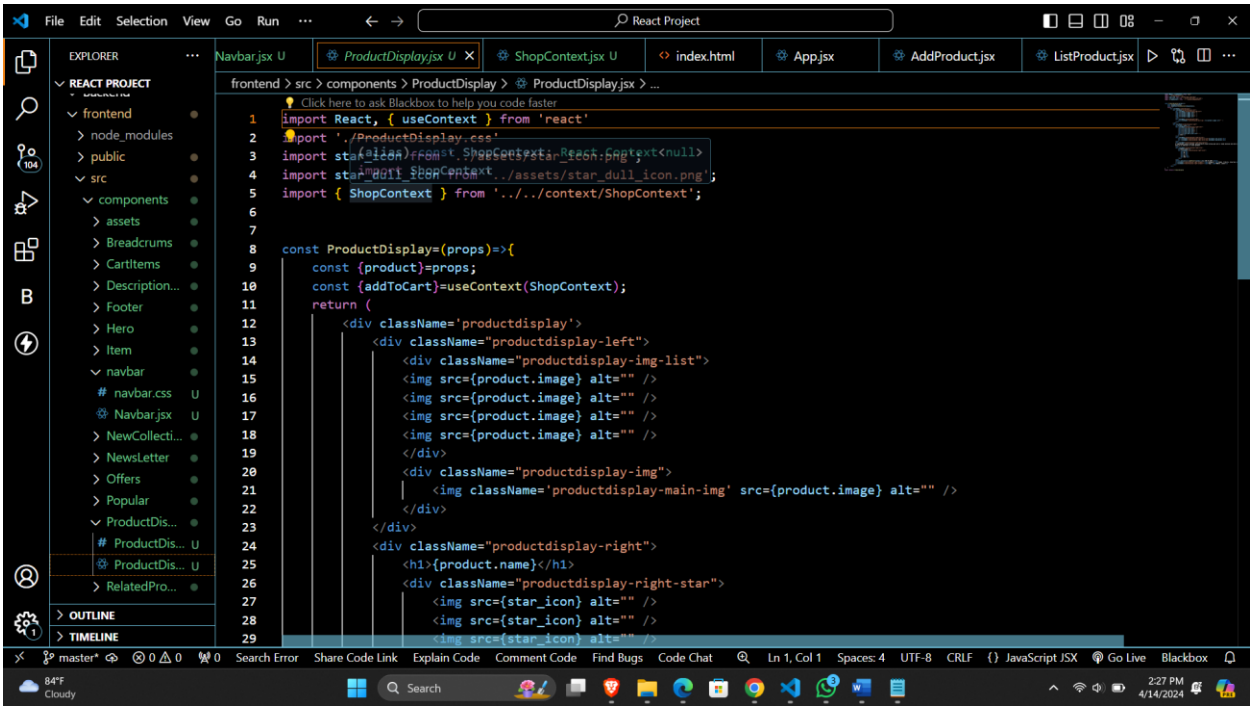
7. SOURCE CODE:-



```
import './App.css';
import {Navbar} from './components/navbar/Navbar';
import {BrowserRouter,Routes,Route} from 'react-router-dom';
import Shop from './pages/Shop';
import ShopCategory from './pages/ShopCategory';
import Product from './pages/Product';
import LoginSignup from './pages/LoginSignup';
import Cart from './pages/Cart';
import Footer from './components/Footer/Footer.jsx';
import men_banner from './components/assets/banner_mens.png';
import women_banner from './components/assets/banner_women.png';
import kid_banner from './components/assets/banner_kids.png';

function App() {
  return (
    <div >
      <BrowserRouter>
      <Navbar/>
      <Routes>
        <Route path='/' element={<Shop/>}/>
        <Route path='/mens' element={<ShopCategory  banner={men_banner} category="men"/>}/>
        <Route path='/womens' element={<ShopCategory banner={women_banner} category="women"/>}/>
        <Route path='/kids' element={<ShopCategory banner={kid_banner} category="kid"/>}/>
        <Route path='/product' element={<Product/>}>
        <Route path=':productId' element={<Product/>}/>
        </Route>
        <Route path='/login' element={<LoginSignup/>}/>
        <Route path='/cart' element={<Cart/>}/>
      </Routes>
```

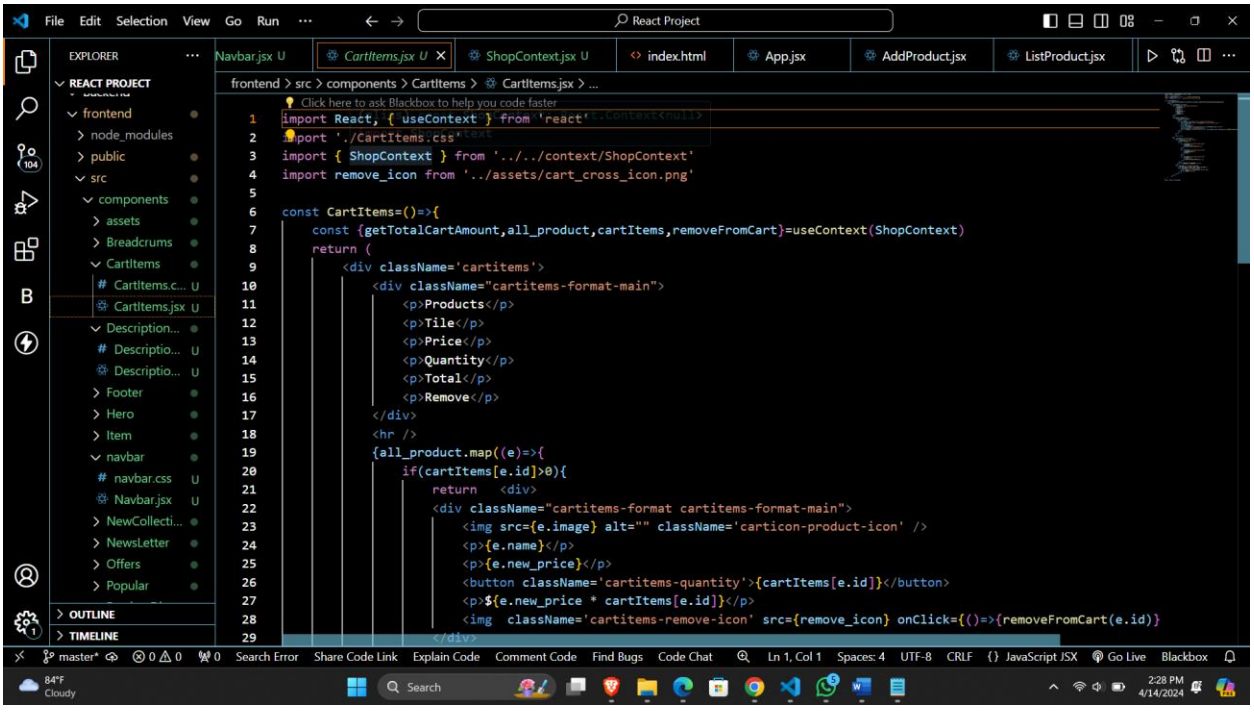


```
import React, { useContext, useRef, useState } from 'react' ;
import './navbar.css';
import logo from '../assets/logo.png';
import cart_icon from '../assets/cart_icon.png';
import { Link } from 'react-router-dom';
import { ShopContext } from '../../context/ShopContext';
import dropdown_icon from '../assets/dropdown_icon.png';

export const Navbar=()=>{
   const [menu,setMenu]=useState("shop");
   const {getTotalCartItems}=useContext(ShopContext);
   const menuRef=useRef();

   const drop_toggle=(e)=>{
      menuRef.current.classList.toggle('nav-menu-visible');
      e.target.classList.toggle('open');

   }
   return(
      <div className='navbar'>
         <div className='nav-logo'>
            <img src={logo} alt="" />
            <p>SHOPPER</p>
         </div>
         <img className='dropdown-icon' onClick={drop_toggle} src={dropdown_icon} alt="" />
         <ul ref={menuRef} className='nav-menu'>
            <li onClick={()=>{setMenu("shop")}}><Link  style={{ textDecoration:'none'}} to='/'>Shop</Link>{menu==
            <li onClick={()=>{setMenu("Men")}}><Link  style={{ textDecoration:'none'}} to='/mens'>Men</Link> {men
            <li onClick={()=>{setMenu("Women")}}><Link style={{ textDecoration:'none'}}  to='/womens'>Women</Link
```


File Edit Selection View Go Run
React Project
Navbar.jsx U
ProductDisplay.jsx U
ShopContext.jsx U
index.html
App.jsx
AddProduct.jsx
ListProduct.jsx
frontend > src > components > ProductDisplay > ProductDisplay.jsx > ...
Click here to ask Blackbox to help you code faster
import React, { useContext } from 'react'
import './ProductDisplay.css'
import star_icon from '../assets/star_icon.png'
import star_dull_icon from '../assets/star_dull_icon.png'
import { ShopContext } from '../../context/ShopContext';
const ProductDisplay=(props)=>{
const {product}=props;
const {addToCart}=useContext(ShopContext);
return (

<h1>{product.name}</h1>

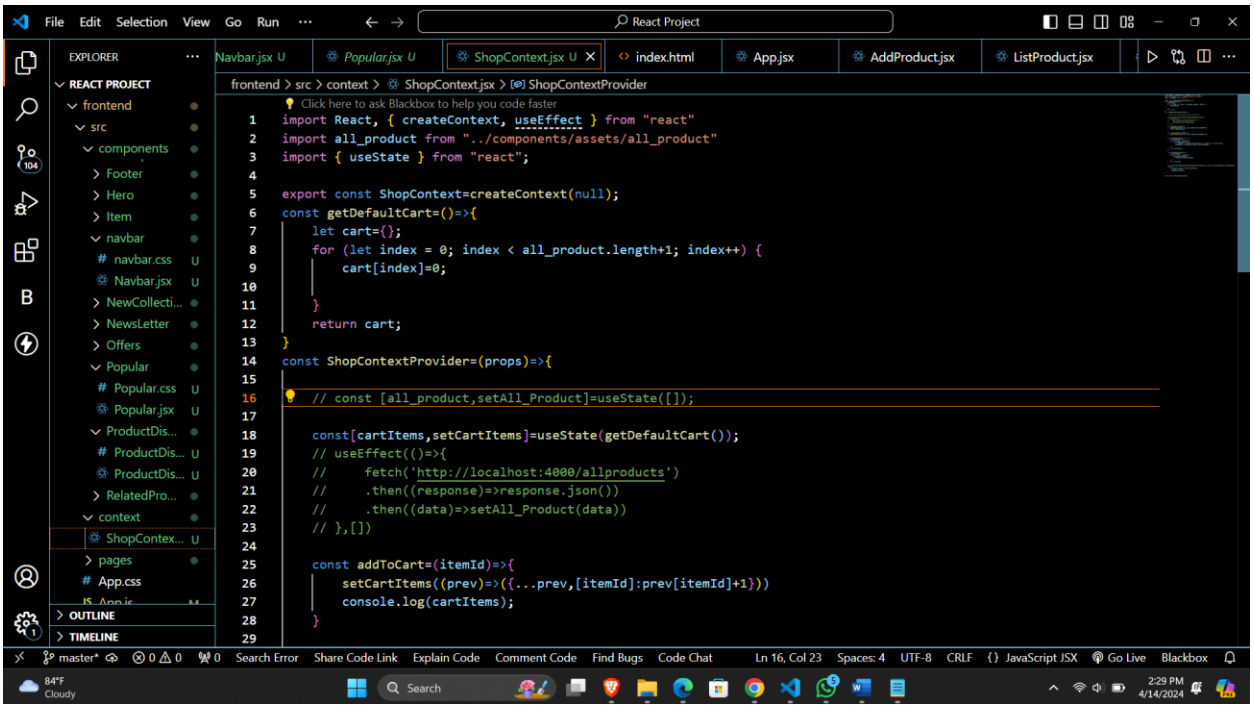
EXPLORER
REACT PROJECT
frontend
node_modules
public
src
components
assets
Breadcrums
CartItems
Description...
Footer
Hero
Item
navbar
navbar.css
Navbar.jsx
NewCollecti...
NewsLetter
Offers
Popular
ProductDis...
RelatedPro...
OUTLINE
TIMELINE
master*
Search Error
Share Code Link
Explain Code
Comment Code
Find Bugs
Code Chat
Ln 1, Col 1
Spaces: 4
UTF-8
CRLF
JavaScript JSX
Go Live
Blackbox
84°F
Cloudy
Search
2:27 PM
4/14/2024

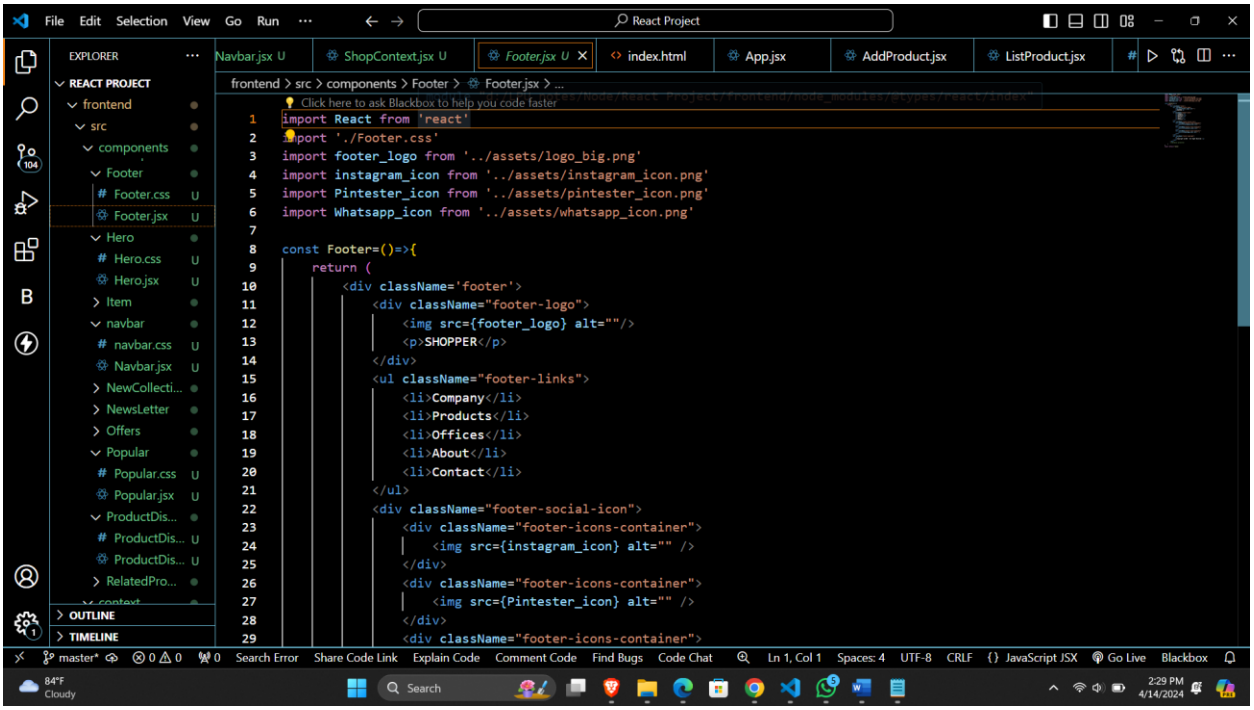

File Edit Selection View Go Run
React Project
Navbar.jsx U
CartItems.jsx U
ShopContext.jsx U
index.html
App.jsx
AddProduct.jsx
ListProduct.jsx
frontend > src > components > CartItems > CartItems.jsx > ...
Click here to ask Blackbox to help you code faster
import React, { useContext } from 'react'
import './CartItems.css'
import { ShopContext } from '../../context/ShopContext'
import remove_icon from '../assets/cart_cross_icon.png'
const CartItems=()=>{
const {getTotalCartAmount,all_product,cartItems,removeFromCart}=useContext(ShopContext)
return (

Products
Tile
Price
Quantity
Total
Remove

<hr />
{all_product.map((e)=>{
if(cartItems[e.id]>0){
return

{e.name}
{e.new_price}
<button classname='cartitems-quantity'>{cartItems[e.id]}</button>
${e.new_price * cartItems[e.id]}
<img classname='cartitems-remove-icon' src={remove_icon} onClick={()=>{removeFromCart(e.id)}
EXPLORER
REACT PROJECT
frontend
node_modules
public
src
components
assets
Breadcrums
CartItems
CartItems.c...
CartItems.jsx
Description...
Footer
Hero
Item
navbar
navbar.css
Navbar.jsx
NewCollecti...
NewsLetter
Offers
Popular
OUTLINE
TIMELINE
master*
Search Error
Share Code Link
Explain Code
Comment Code
Find Bugs
Code Chat
Ln 1, Col 1
Spaces: 4
UTF-8
CRLF
JavaScript JSX
Go Live
Blackbox
84°F
Cloudy
Search
2:28 PM
4/14/2024

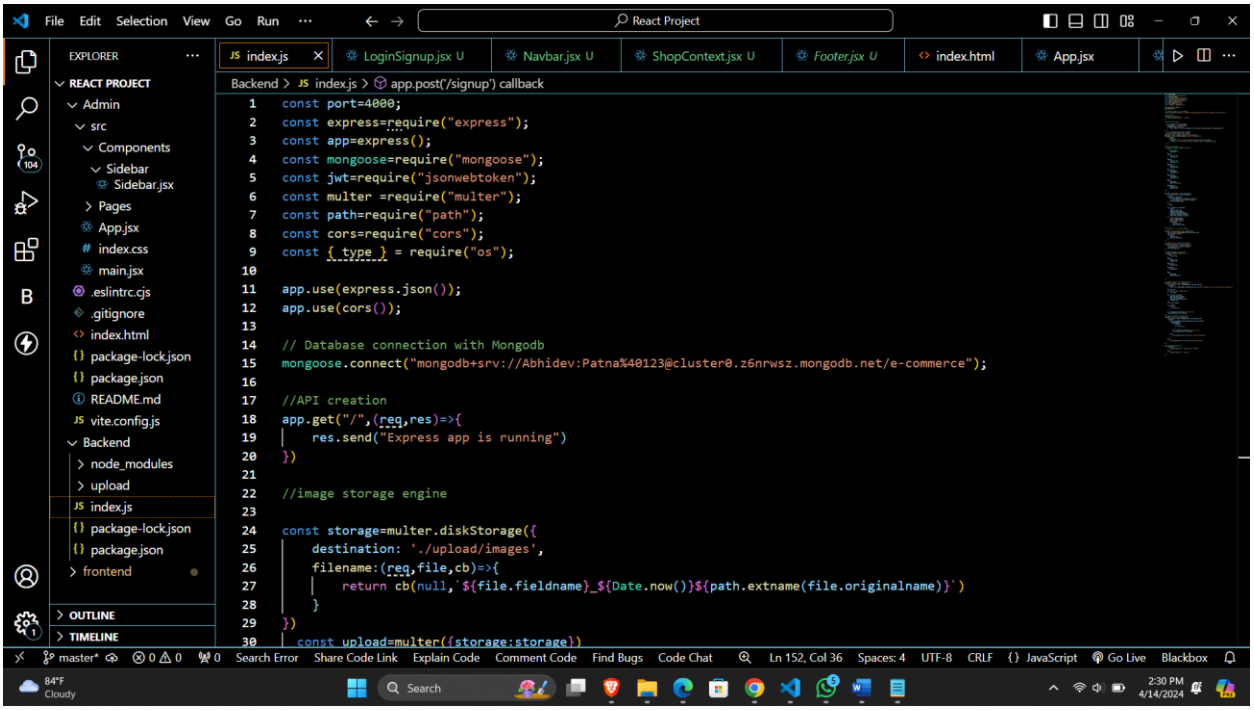

File Edit Selection View Go Run
React Project
EXPLORER
REACT PROJECT
frontend
src
components
Footer
Hero
Item
navbar
NewsLetter
Offers
Popular
pages
context
OUTLINE
TIMELINE
Navbar.jsx U
Popular.jsx U
ShopContext.jsx U
index.html
App.jsx
AddProduct.jsx
ListProduct.jsx
frontend > src > context > ShopContext.jsx > ShopContextProvider
Click here to ask Blackbox to help you code faster
import React, { createContext, useEffect } from "react"
import all_product from "../components/assets/all_product"
import { useState } from "react";
export const ShopContext=createContext(null);
const getDefaultCart=()=>{
let cart={};
for (let index = 0; index < all_product.length+1; index++) {
cart[index]=0;
}
return cart;
}
const ShopContextProvider=(props)=>{
// const [all_product,setAll_Product]=useState([]);
const[cartItems,setCartItems]=useState(getDefaultCart());
// useEffect(()=>{
// fetch('http://localhost:4000/allproducts')
// .then((response)=>response.json())
// .then((data)=>setAll_Product(data))
// },[])
const addToCart=(itemId)=>{
setCartItems((prev)=>({...prev,[itemId]:prev[itemId]+1}))
console.log(cartItems);
}
Search Error Share Code Link Explain Code Comment Code Find Bugs Code Chat
Ln 16, Col 23 Spaces: 4 UTF-8 CRLF JavaScript JSX Go Live Blackbox
84°F Cloudy
Search
2:29 PM 4/14/2024

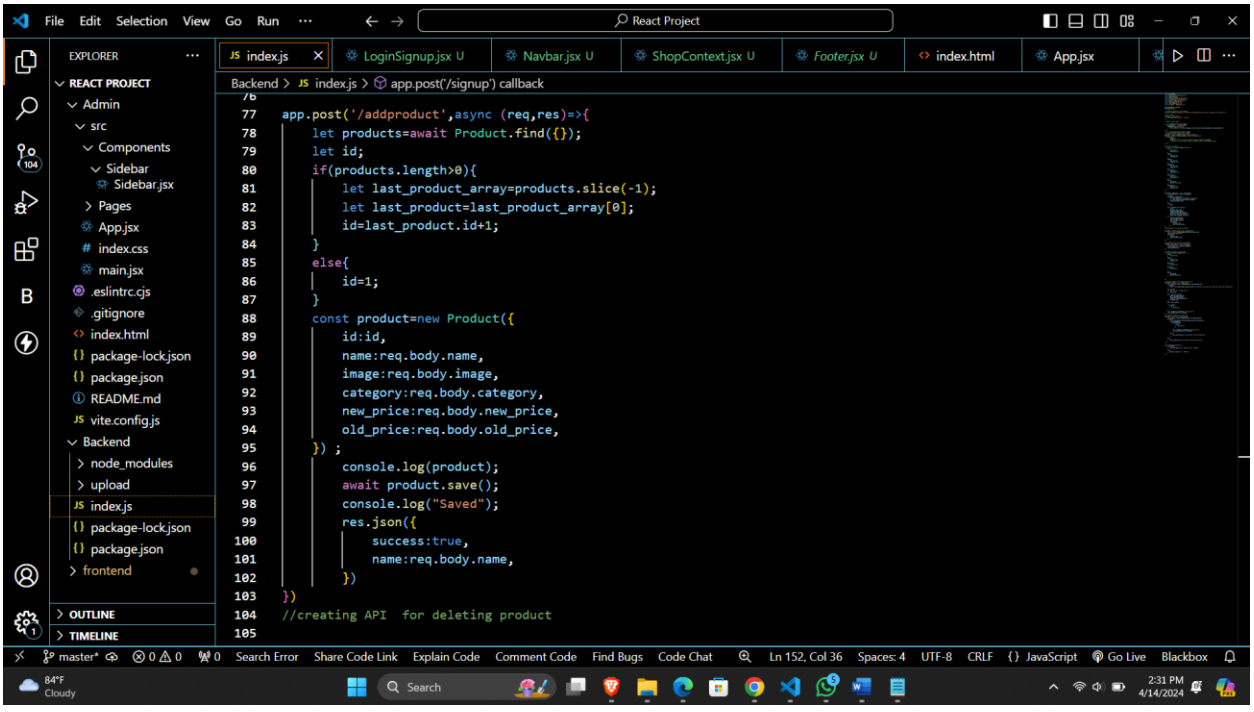

File Edit Selection View Go Run
React Project
EXPLORER
REACT PROJECT
frontend
src
components
Footer
Footer.css
Footer.jsx
Hero
Item
navbar
NewsLetter
Offers
Popular
ProductDis...
RelatedPro...
OUTLINE
TIMELINE
Navbar.jsx U
ShopContext.jsx U
Footer.jsx U
index.html
App.jsx
AddProduct.jsx
ListProduct.jsx
frontend > src > components > Footer > Footer.jsx > ...
Click here to ask Blackbox to help you code faster
import React from 'react'
import './Footer.css'
import footer_logo from '../assets/logo_big.png'
import instagram_icon from '../assets/instagram_icon.png'
import Pintester_icon from '../assets/pintester_icon.png'
import Whatsapp_icon from '../assets/whatsapp_icon.png'
const Footer=()=>{
return (

SHOPPER

<ul className="footer-links">
Company
Products
Offices
About
Contact

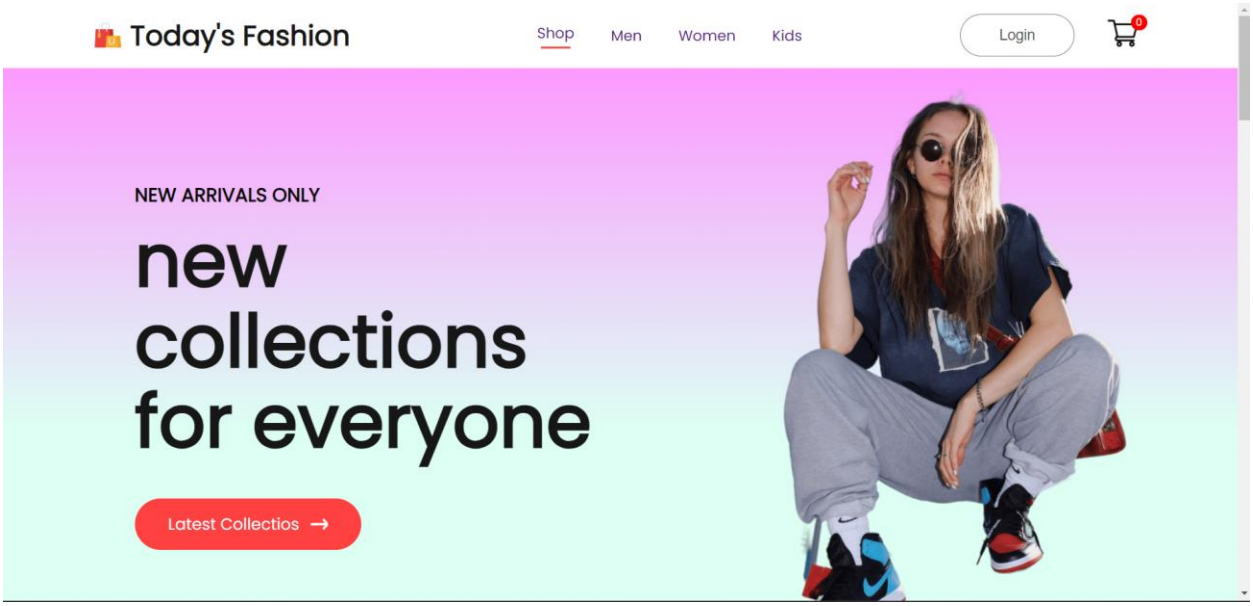
Search Error Share Code Link Explain Code Comment Code Find Bugs Code Chat
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF JavaScript JSX Go Live Blackbox
84°F Cloudy
Search
2:29 PM 4/14/2024

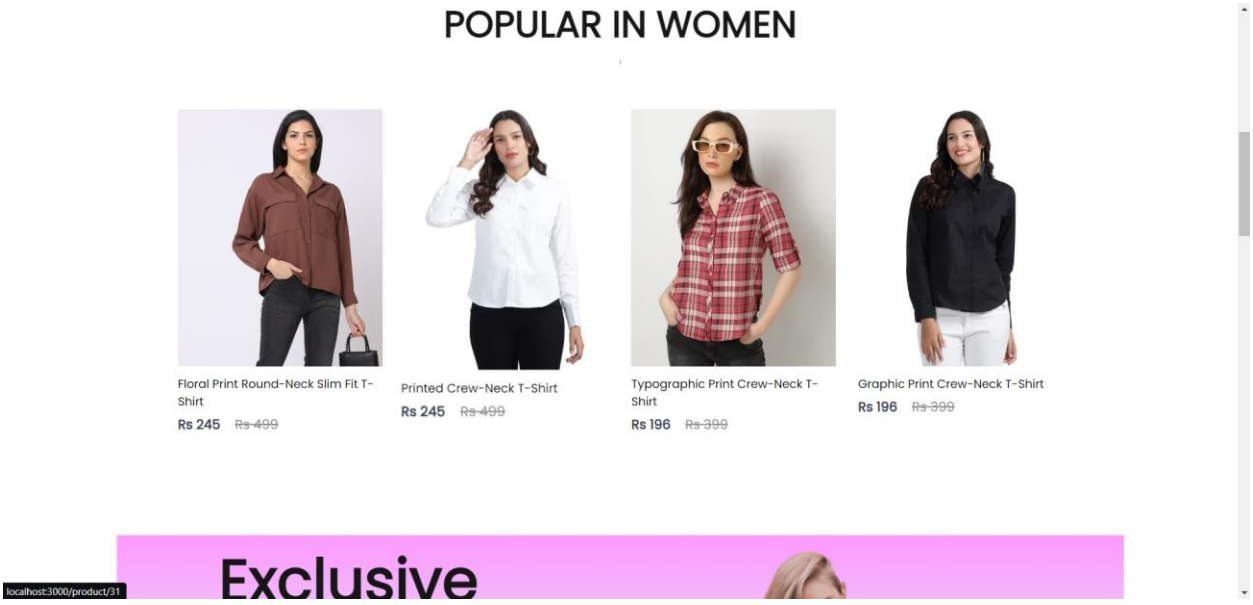

File Edit Selection View Go Run
React Project
EXPLORER
REACT PROJECT
Admin
src
Components
Sidebar
Sidebar.jsx
Pages
App.jsx
index.css
main.jsx
.eslintrc.cjs
.gitignore
index.html
package-lock.json
package.json
README.md
vite.config.js
Backend
node_modules
upload
index.js
package-lock.json
package.json
frontend
OUTLINE
TIMELINE
index.js
LoginSignup.jsx U
Navbar.jsx U
ShopContext.jsx U
Footer.jsx U
index.html
App.jsx
Backend > index.js > app.post('/signup') callback
1 const port=4000;
2 const express=require("express");
3 const app=express();
4 const mongoose=require("mongoose");
5 const jwt=require("jsonwebtoken");
6 const multer =require("multer");
7 const path=require("path");
8 const cors=require("cors");
9 const { type } = require("os");
10
11 app.use(express.json());
12 app.use(cors());
13
14 // Database connection with Mongodb
15 mongoose.connect("mongodb+srv://Abhidev:Patna%40123@cluster0.z6nrwsz.mongodb.net/e-commerce");
16
17 //API creation
18 app.get("/",(req,res)=>{
19 res.send("Express app is running")
20 })
21
22 //image storage engine
23
24 const storage=multer.diskStorage({
25 destination: './upload/images',
26 filename:(req,file,cb)=>{
27 return cb(null,`${file.fieldname}_${Date.now()}${path.extname(file.originalname)}`)
28 }
29 })
30 const upload=multer({storage:storage})
master* Search Error Share Code Link Explain Code Comment Code Find Bugs Code Chat Ln 152, Col 36 Spaces: 4 UTF-8 CRLF JavaScript Go Live Blackbox
84°F Cloudy
Search
2:30 PM 4/14/2024

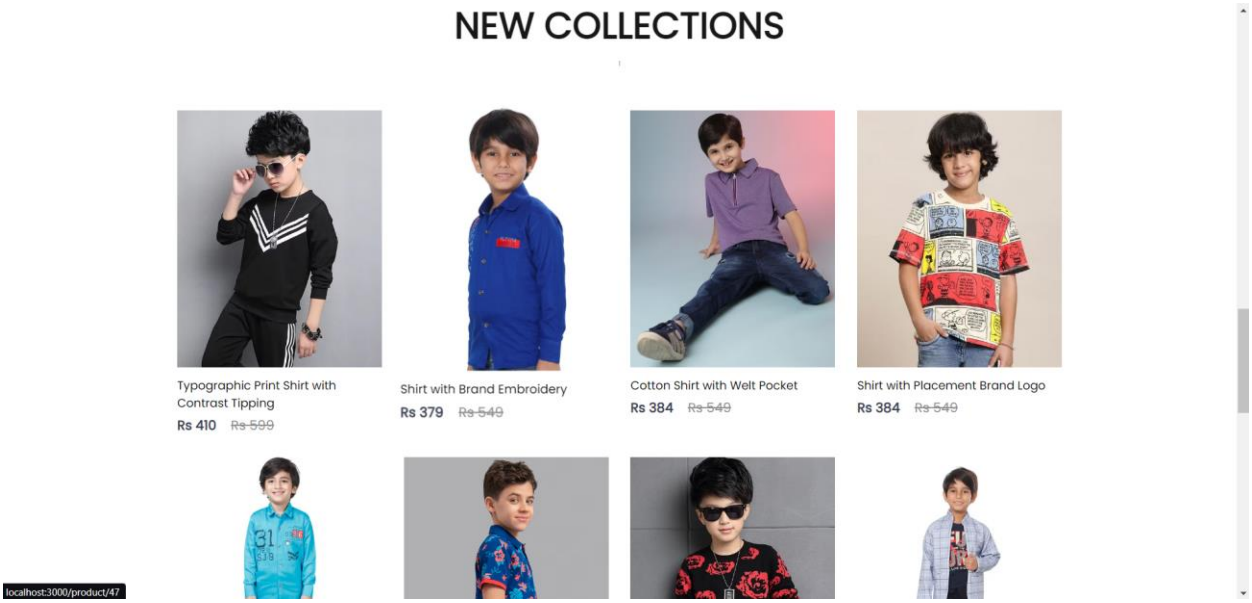

File Edit Selection View Go Run
React Project
EXPLORER
REACT PROJECT
Admin
src
Components
Sidebar
Sidebar.jsx
Pages
App.jsx
index.css
main.jsx
.eslintrc.cjs
.gitignore
index.html
package-lock.json
package.json
README.md
vite.config.js
Backend
node_modules
upload
index.js
package-lock.json
package.json
frontend
OUTLINE
TIMELINE
index.js
LoginSignup.jsx U
Navbar.jsx U
ShopContext.jsx U
Footer.jsx U
index.html
App.jsx
Backend > index.js > app.post('/signup') callback
77 app.post('/addproduct',async (req,res)=>{
78 let products=await Product.find({});
79 let id;
80 if(products.length>0){
81 let last_product_array=products.slice(-1);
82 let last_product=last_product_array[0];
83 id=last_product.id+1;
84 }
85 else{
86 id=1;
87 }
88 const product=new Product({
89 id:id,
90 name:req.body.name,
91 image:req.body.image,
92 category:req.body.category,
93 new_price:req.body.new_price,
94 old_price:req.body.old_price,
95 }) ;
96 console.log(product);
97 await product.save();
98 console.log("Saved");
99 res.json({
100 success:true,
101 name:req.body.name,
102 })
103 })
104 //creating API for deleting product
105
master* Search Error Share Code Link Explain Code Comment Code Find Bugs Code Chat Ln 152, Col 36 Spaces: 4 UTF-8 CRLF JavaScript Go Live Blackbox
84°F Cloudy
Search
2:31 PM 4/14/2024

8. OUTPUT OF THE PROJECT :-

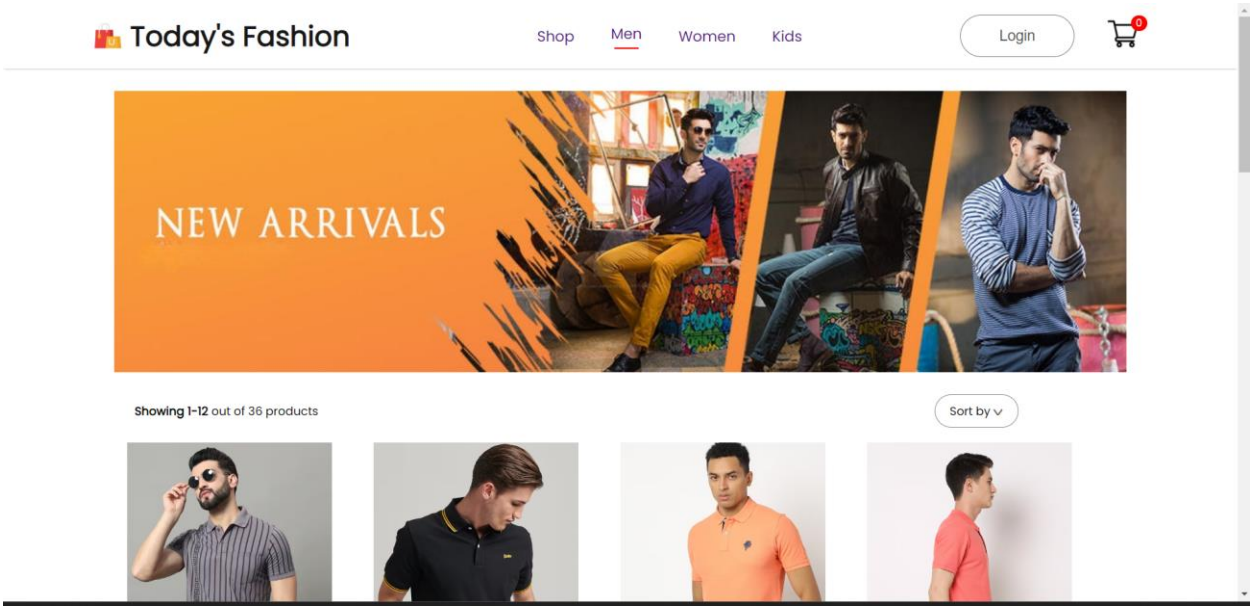
HOME PAGE :-

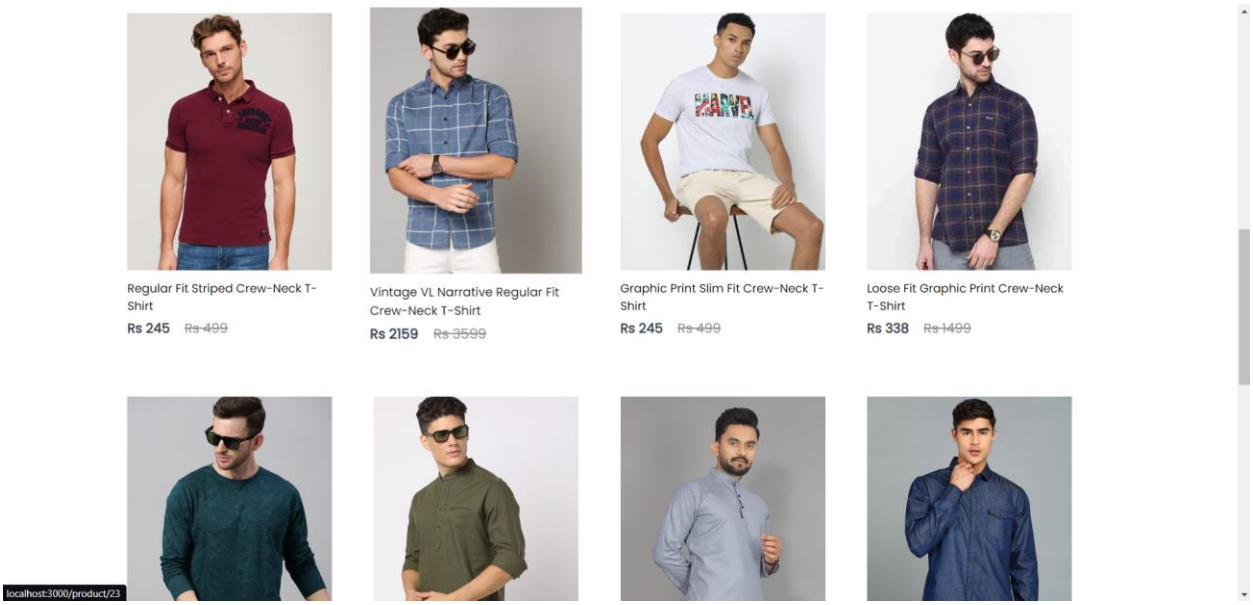




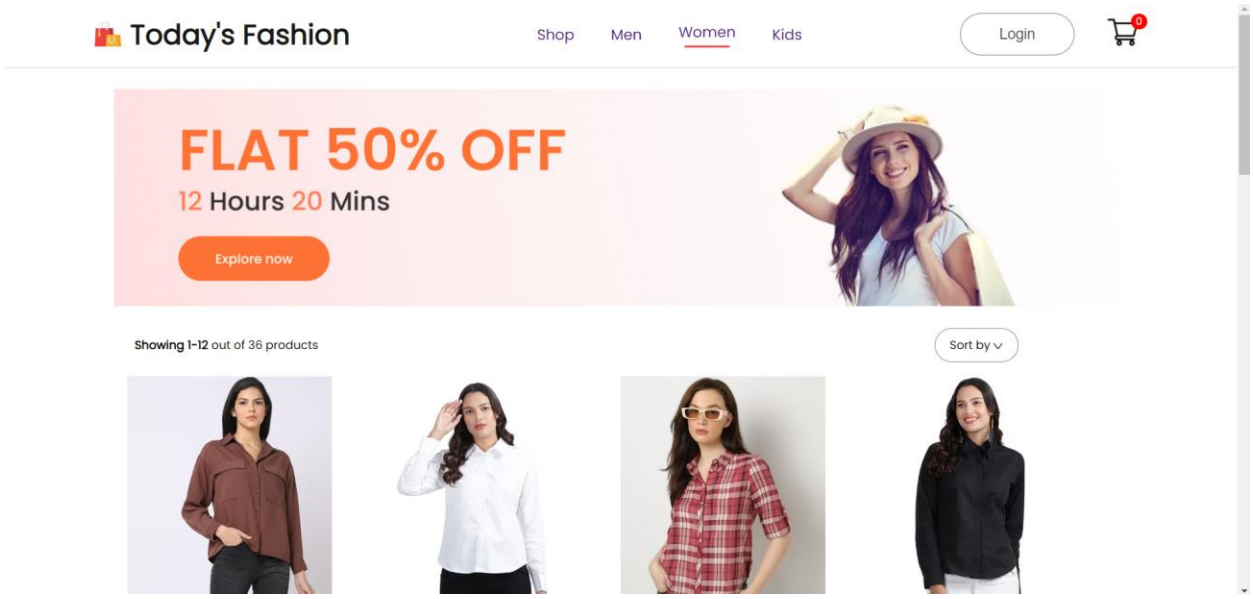

NEW COLLECTIONS
Typographic Print Shirt with Contrast Tipping
Rs 410 Rs 599
Shirt with Brand Embroidery
Rs 379 Rs 549
Cotton Shirt with Welt Pocket
Rs 384 Rs 549
Shirt with Placement Brand Logo
Rs 384 Rs 549
localhost:3000/product/47

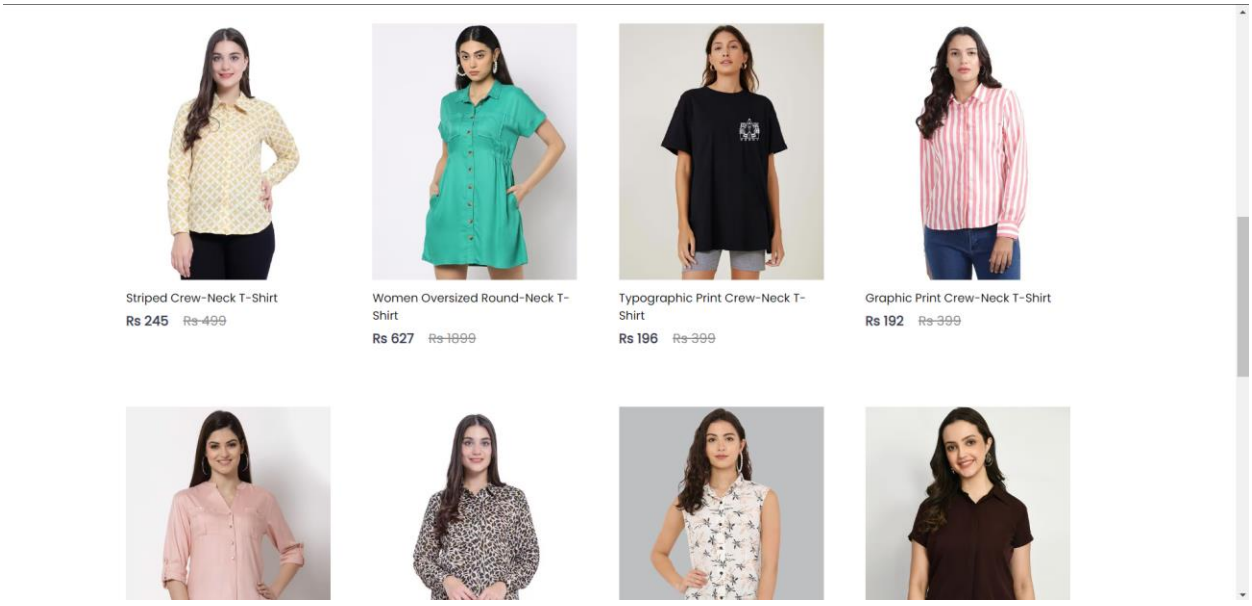
MENS PAGE :-


Today's Fashion
Shop
Men
Women
Kids
Login
NEW ARRIVALS
Showing 1-12 out of 36 products
Sort by

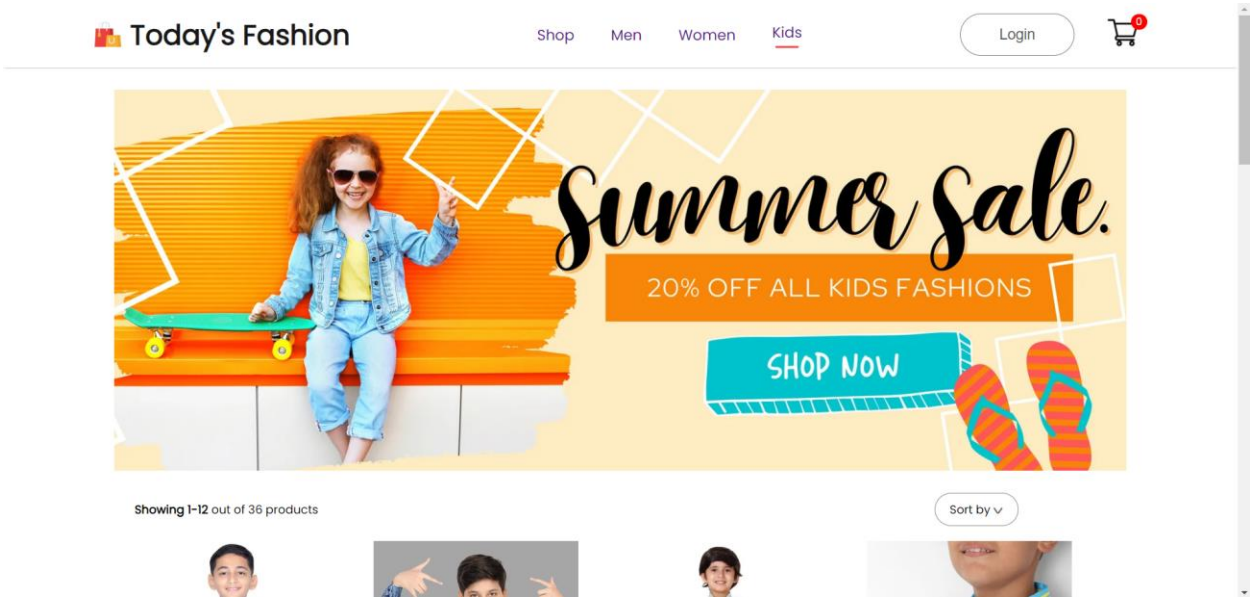

Regular Fit Striped Crew-Neck T-Shirt
Rs 245 Rs 499
Vintage VL Narrative Regular Fit Crew-Neck T-Shirt
Rs 2159 Rs 3599
Graphic Print Slim Fit Crew-Neck T-Shirt
Rs 245 Rs 499
Loose Fit Graphic Print Crew-Neck T-Shirt
Rs 338 Rs 1499
localhost:3000/product/23

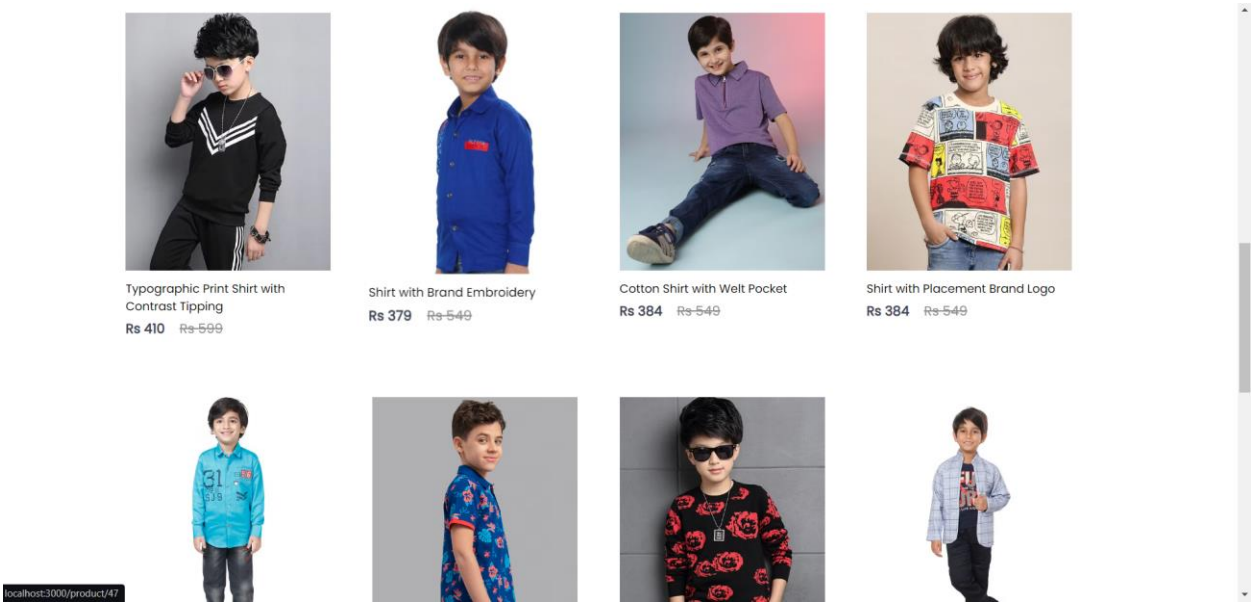
WOMENS PAGE :-


Today's Fashion
Shop
Men
Women
Kids
Login
FLAT 50% OFF
12 Hours 20 Mins
Explore now
Showing 1-12 out of 36 products
Sort by

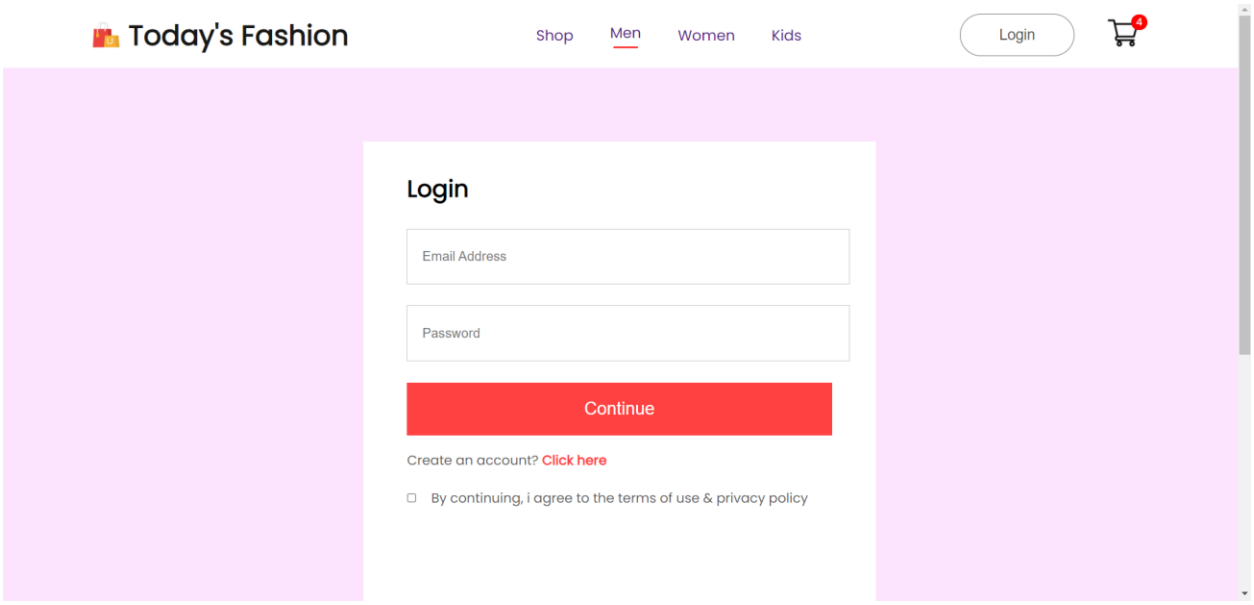

Striped Crew-Neck T-Shirt
Rs 245 Rs 499
Women Oversized Round-Neck T-Shirt
Rs 627 Rs 1899
Typographic Print Crew-Neck T-Shirt
Rs 196 Rs 399
Graphic Print Crew-Neck T-Shirt
Rs 192 Rs 399

KIDS PAGE :-

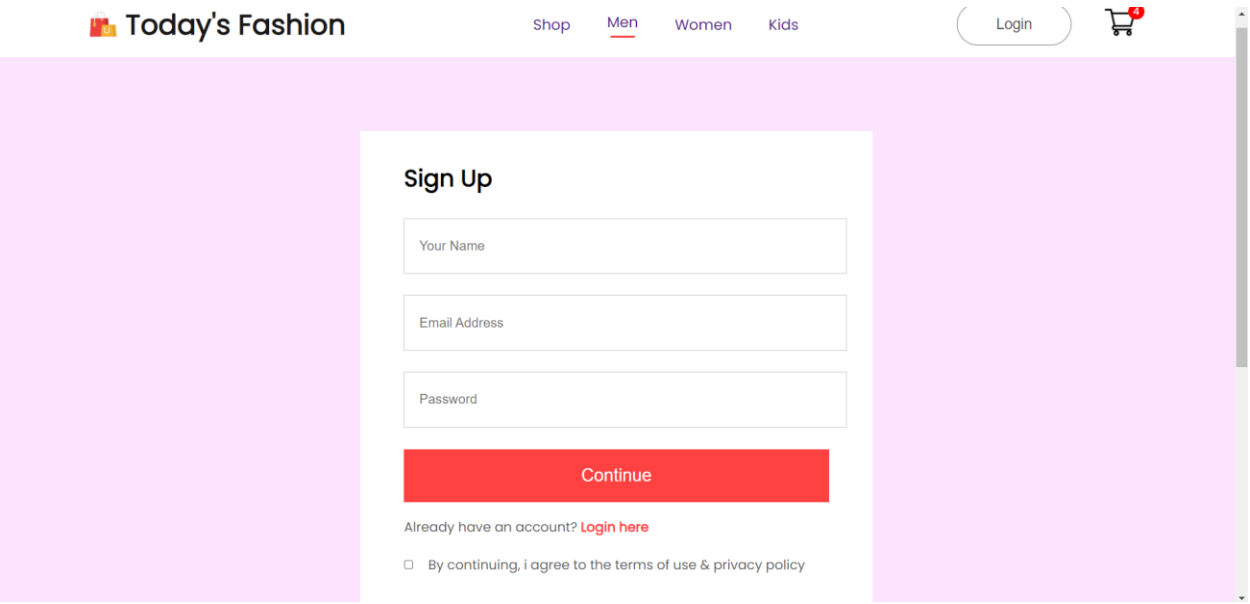

Today's Fashion
Shop
Men
Women
Kids
Login
Summer Sale.
20% OFF ALL KIDS FASHIONS
SHOP NOW
Showing 1-12 out of 36 products
Sort by


Typographic Print Shirt with Contrast Tipping
Rs 410 Rs 599
Shirt with Brand Embroidery
Rs 379 Rs 549
Cotton Shirt with Welt Pocket
Rs 384 Rs 549
Shirt with Placement Brand Logo
Rs 384 Rs 549
localhost:3000/product/47

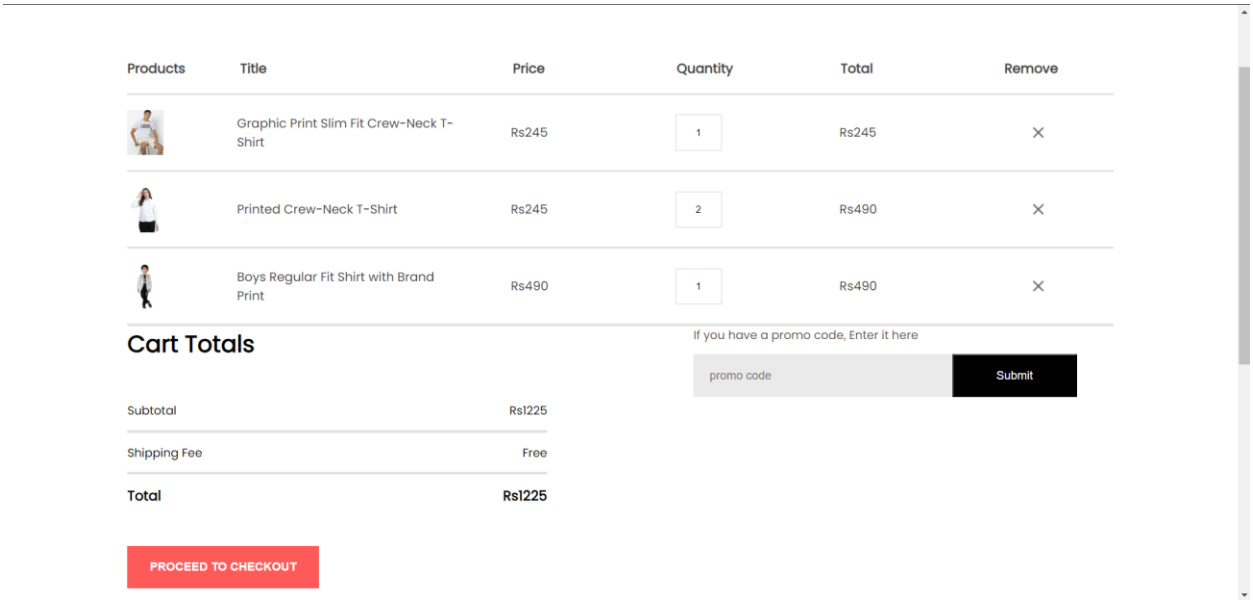
LOGIN PAGE :-


Today's Fashion
Shop
Men
Women
Kids
Login
Login
Email Address
Password
Continue
Create an account? Click here
By continuing, i agree to the terms of use & privacy policy

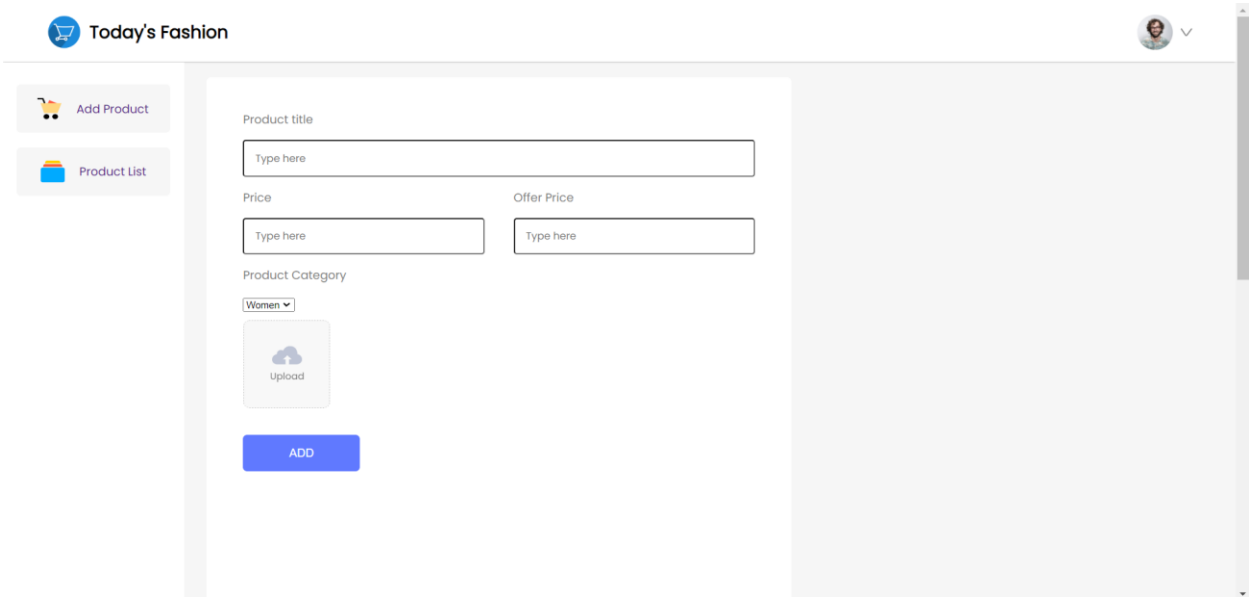
SIGNUP PAGE :-

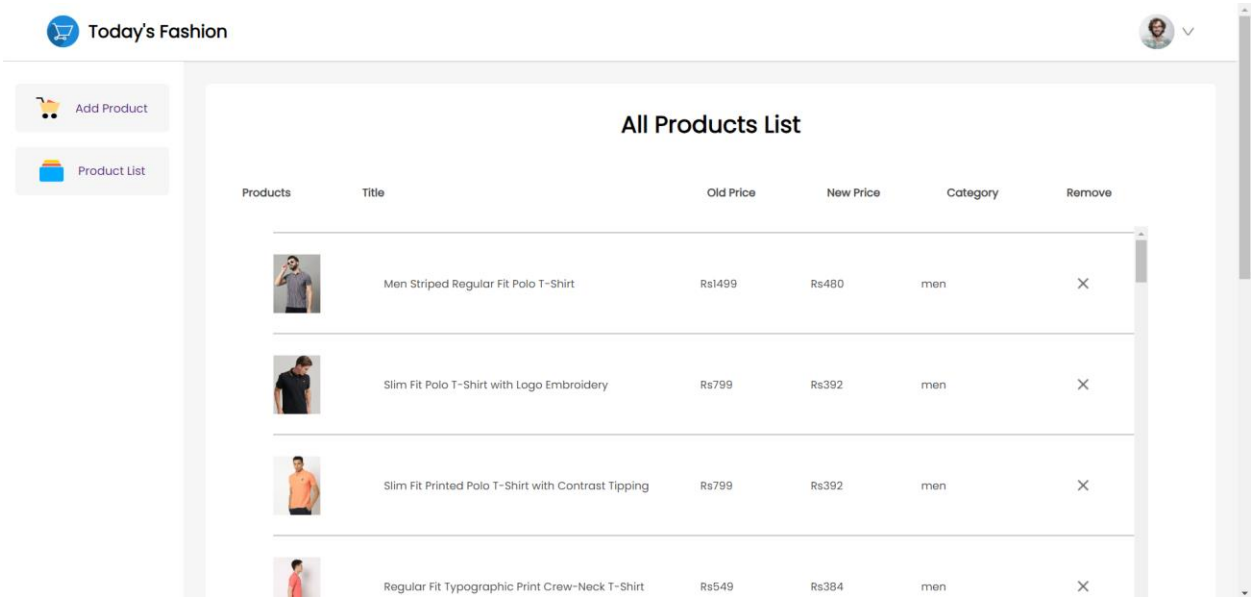


CART PAGE :-

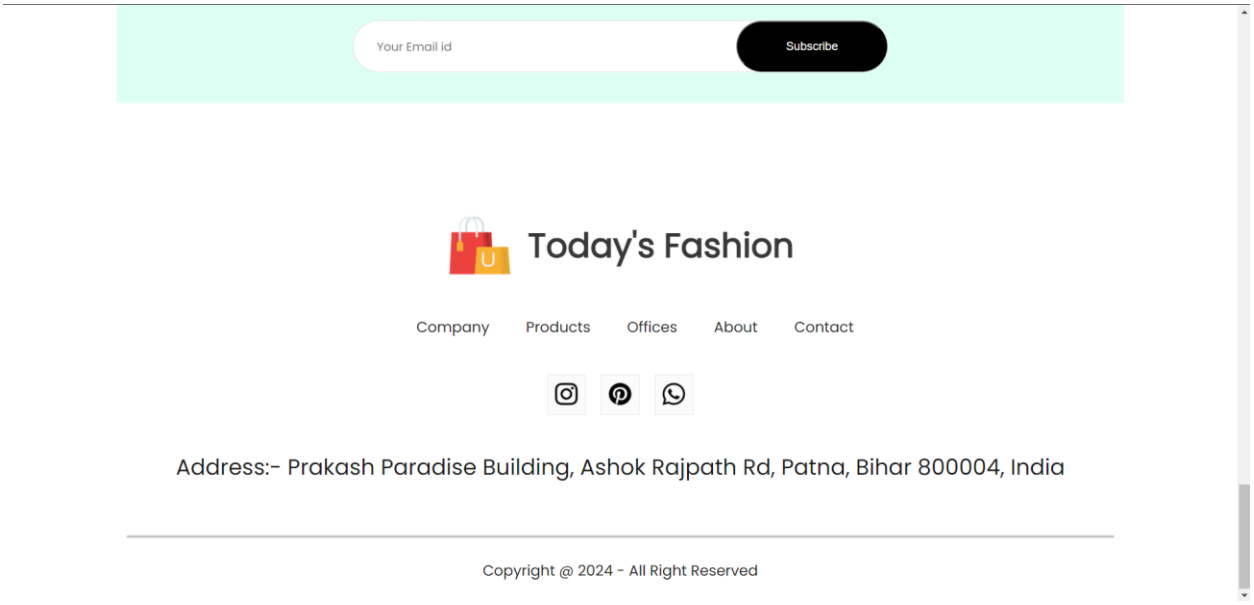


ADMIN PANEL :-


Today's Fashion
Add Product
Product List
Product title
Type here
Price
Offer Price
Type here
Type here
Product Category
Women
Upload
ADD


Today's Fashion
Add Product
Product List
All Products List
Products
Title
Old Price
New Price
Category
Remove
Men Striped Regular Fit Polo T-Shirt
Rs1499
Rs480
men
Slim Fit Polo T-Shirt with Logo Embroidery
Rs799
Rs392
men
Slim Fit Printed Polo T-Shirt with Contrast Tipping
Rs799
Rs392
men
Regular Fit Typographic Print Crew-Neck T-Shirt
Rs549
Rs384
men

FOOTER :-



THANK YOU!!!