

CHAPTER 1

INTRODUCTION

1.1 ABOUT PROJECT

The **Personalized Shopping Destination** project aims to create a dynamic, user-focused online marketplace that offers a broad range of **men's and women's dresses**, along with other textile-based products, combining traditional craftsmanship with modern technology. Built using **React.js** for the frontend and **MongoDB** for the backend, this platform will provide a seamless and personalized shopping experience. Users will have the ability to explore various categories such as ethnic wear, casual wear, accessories, and home textiles while also customizing their purchases based on fabric type, size, color, and design preferences. React's real-time rendering and dynamic components will ensure users receive instant updates on product availability and customization options. The platform's commitment to sustainability will be reflected in its offering of eco-friendly, handcrafted textiles, supporting local artisans and promoting ethical production practices. Users can securely make payments through integrated gateway such as **PayPal**, while the shopping experience will be optimized for all devices using responsive design principles. Additionally, the **admin dashboard** will allow administrators to manage product listings, track inventory, monitor orders, and manage customer data efficiently. By combining the power of modern web technologies with the rich cultural heritage of textiles, the **Personalized Shopping Destination** aims to create an immersive and personalized online shopping environment that empowers artisans and offers customers high-quality, sustainable products tailored to their individual tastes.

1.2 EXISTING SYSTEM

An existing system for **Personalized Shopping Destination** typically includes a platform where users can browse various textiles such as dress accessories, clothing, and digital accessories. Features include detailed product listings, search filters, a shopping cart, and multiple payment options. Customers can create accounts to manage orders and track shipments. The system also supports returns and refunds, integrates with inventory management, and provides customer support through live chat or email. Additionally, the platform may offer promotions, loyalty programs, and AI-based product recommendations to enhance the shopping experience.

1.3 DRAWBACK OF EXISTING SYSTEM

- **Limited Customization:** Many platforms don't offer enough options for customers to tailor products to their preferences, such as fabric types or specific measurements.
- **Poor Product Visualization:** Relying solely on images can leave customers uncertain about the true look and feel of the textile, leading to potential dissatisfaction.
- **Shipping Delays:** Unpredictable delivery times, especially during peak seasons, can lead to customer frustration and lost trust.
- **Inadequate Customer Support:** Slow or insufficient support can leave customers feeling unsupported when dealing with issues related to orders or returns.

1.4 PROPOSED SYSTEM

A **Personalized Shopping Destination** for online textile shopping would focus on tailoring the shopping experience to each individual customer's preferences, making it more engaging and efficient. The system would use data-driven recommendations based on a customer's browsing history, past purchases, and style preferences. Customers could personalize their shopping journey by setting preferences for fabric types, colors, patterns, or even occasions. Additionally, a virtual try-on feature or fabric preview would allow users to see how textiles will look in their home or on their clothes, enhancing the decision-making process. With a seamless, user-friendly interface and personalized suggestions, this platform would aim to create a unique and tailored shopping experience for every user.

1.5 ADVANTAGES OF PROPOSED SYSTEM

- **Enhanced Personalization:** The system tailors the shopping experience based on individual preferences, offering product recommendations, customizations, and a more relevant selection, leading to higher customer satisfaction.
- **Improved Visualization:** The use of 3D models and augmented reality (AR) allows customers to better visualize textiles, ensuring they make informed decisions and reducing the chances of dissatisfaction or returns.
- **Reliable and Fast Shipping:** Offering real-time tracking, multiple shipping options, and guaranteed delivery times enhances the customer experience by ensuring timely and predictable deliveries.
- **24/7 Customer Support:** Round-the-clock assistance via live chat, AI-powered bots, and phone support ensures quick resolution of issues, improving customer trust and retention.

1.6 DISADVANTAGES OF PROPOSED SYSTEM

- **High Development and Maintenance Costs:** Integrating features like 3D models, augmented reality (AR), and AI-driven personalization can require significant investment in technology development, implementation, and ongoing maintenance.
- **Complex User Interface:** Adding too many customization options or advanced features may overwhelm some users, particularly those not as tech-savvy, potentially complicating the shopping experience.
- **Dependence on Technology:** The system's reliance on advanced technologies like AR and AI may face challenges with bugs, glitches, or limited user access due to device compatibility, affecting the overall experience.
- **Privacy and Data Security Concerns:** Collecting personal data for personalized recommendations could raise privacy concerns, and ensuring robust data protection measures will be essential to avoid security breaches.

CHAPTER 2

SYSTEM ANALYSIS

2.1 IDENTIFICATION OF NEED

A personalized shopping destination for textiles and accessories aims to provide an immersive and tailored shopping experience that meets the specific needs of each customer. This platform would use advanced data analytics and machine learning to recommend fabrics, clothing, and accessories based on the user's personal preferences, such as style, color choices, fabric types, and previous purchases. By analyzing user behavior, including browsing history, product searches, and feedback, the platform would present a curated selection of textiles and accessories, ensuring customers find items that match their individual tastes. The system could also offer personalized styling tips and suggestions for matching accessories with selected clothing. Additionally, location-based recommendations could highlight nearby stores or available inventory for quick delivery, and users would be alerted to sales, discounts, or exclusive deals on items of interest. With seamless integration of online and offline store inventories, customers would enjoy a hassle-free shopping experience, with the convenience of exploring a variety of products tailored specifically to them.

2.2 FEASIBILITY STUDY

The feasibility study for the personalized shopping destination project suggests that it is technically, economically, and operationally viable with careful planning and execution. It helps to determine whether the proposed system is technically, economically, and operationally feasible. Here are the key components to include in the feasibility study:

- Technical Feasibility
- Economic Feasibility
- Operational Feasibility

2.2.1 TECHNICAL FEASIBILITY

- The project involves implementing IoT-enabled looms, digital design tools, and e-commerce platforms. These technologies are readily available and can be customized for the textile sector.
- Adequate infrastructure, including internet connectivity, power supply, and centralized workspaces, is critical. These can be developed or upgraded in rural areas where artisans operate.
- Training programs will be essential to familiarize artisans with modern tools and technologies. Local support teams can be established to provide ongoing assistance.
- The project can be phased, starting with pilot programs in selected regions and scaling up based on feedback and results.

2.2.2 ECONOMIC FEASIBILITY

- The project requires significant funding for setting up infrastructure, purchasing technology, and providing training.
- However, grants, subsidies, and partnerships with government agencies, NGOs, and private investors can offset costs.
- The e-commerce platform and global market access will generate significant revenue by increasing the demand for high-quality, unique handloom products.
- By reducing reliance on middlemen and streamlining production, artisans will achieve higher earnings. The long-term benefits outweigh the initial costs.

2.2.3 OPERATIONAL FEASIBILITY

- The project relies on the active participation of artisans, designers, technology providers, and marketers.
- Early engagement and clear communication will ensure stakeholder alignment.
- Efficient inventory management and logistics systems will enable the hub to meet production demands and fulfill orders promptly.

- The model is scalable, allowing for expansion to other regions or incorporating new technologies as the hub grows.

2.3 SOFTWARE REQUIREMENT SPECIFICATION

A Software Requirement Specification (SRS) outlines the functional and non-functional requirements for the software that will support the system. The personalized shopping destination project aims to modernize the handloom industry by integrating digital tools and e-commerce capabilities to empower artisans, enhance product design, improve market access, and support sustainable practices. It addresses the needs of artisans, customers, and administrators while focusing on usability, security, scalability, and sustainability.

2.3.1 HARDWARE REQUIREMENTS

The hardware requirements for the textile design hub project are essential to ensure smooth operation and optimal performance of the system. The system will need high-performance servers to host the e-commerce platform, inventory management system, design tools, and quality control systems. These servers should have sufficient processing power, memory, and storage to handle large amounts of data from multiple users and IoT devices, ensuring scalability as the system grows. For the IoT-enabled looms, specialized hardware will be needed to monitor and control production, including sensors, actuators, and microcontrollers to track and optimize weaving processes.

- Processor Type : i5 processor
- Processor Speed : 1.00 GHz
- RAM : 6GB
- Hard Disk Capacity : 1TB(SSD)
- Mouse : Logitech

2.3.2 SOFTWARE REQUIREMENTS

The software requirements for the personalized shopping destination project are crucial for ensuring the smooth operation, security, and functionality of the system. MongoDB, a NoSQL database, will serve as the core data storage, utilizing collections to manage data such as user profiles, product catalogs, orders, inventory, design patterns, and quality control information. The backend will be powered by Node.js and Express.js, enabling the development of RESTful APIs to interact with the MongoDB database. React.js or Vue.js will be used for the frontend, ensuring a responsive and dynamic user interface that communicates with the backend through these APIs.

- Operating System : Windows 10
- Front end : HTML, CSS, React JS
- Back end : Node JS
- Database : Mango DB
- Tools : Visual Studio, Node JS

React JS

React.js is a widely-used open-source JavaScript library developed by Facebook for building user interfaces, particularly for single-page applications (SPAs). It follows a component-based architecture, where the UI is divided into reusable, self-contained components that manage their own state. This modular approach makes React highly efficient and easy to maintain. One of React's key features is its Virtual DOM, which helps improve performance by updating only the necessary parts of the real DOM when there is a change in the application's state, rather than reloading the entire page. React components are typically written using JSX (JavaScript XML), a syntax extension that allows developers to write HTML-like structures within JavaScript, making the code more readable and intuitive.

KEY FEATURES OF REACT JS

- React is built around the concept of components, which are self-contained, reusable building blocks for the UI. A component can be as simple as a button or as complex as an entire form or page. This modularity allows developers to maintain and update sections of an application independently.
- In React, developers describe how the UI should look based on the application's state. React then takes care of updating the DOM (Document Object Model) when the state changes. This approach is more intuitive than imperative programming, where you have to manually manage DOM updates.
- React components are typically written using JSX, a syntax extension that allows HTML-like code to be written within JavaScript. JSX makes the code more readable and easier to write by allowing developers to define the UI in a syntax that closely resembles HTML.

SERVER

Node.js is a cross-platform, open-source JavaScript runtime environment that can run on Windows, Linux, Unix, macOS, and more. Node.js runs on the V8 JavaScript engine, and executes JavaScript code outside a web browser.

Node.js lets developers use JavaScript to write command line tools and for server-side scripting. The ability to run JavaScript code on the server is often used to generate dynamic web page content before the page is sent to the user's web browser. Consequently, Node.js represents a "JavaScript everywhere" paradigm, unifying web-application development around a single programming language, as opposed to using different languages for the server- versus client-side programming.

DATABASE

MongoDB will be used as the database management system, given its flexibility and scalability for handling the various types of data involved in the project. Below is an example of how MongoDB collections could be structured, along with the relevant fields for each collection to support the various functionalities of the textile design hub.

- **Flexible Schema:** MongoDB's document-based structure allows for flexibility in storing various types of data, such as product details, orders, and customer information, without the need for predefined schemas. This is especially beneficial for handling data that may evolve over time, like product catalogs and design patterns.
- **Scalability:** As the textile design hub grows, MongoDB can scale horizontally by distributing data across multiple servers. This ensures that the system can handle an increasing number of users, orders, and products efficiently.
- **Real-Time Updates:** MongoDB's support for real-time data updates makes it ideal for tracking changes in product inventory, order status, and artisan activities, as well as integrating with IoT devices for loom monitoring.
- **High Availability:** MongoDB's replica sets provide redundancy and high availability, ensuring that the system remains operational even in the event of hardware failures.
- **Aggregation and Analytics:** MongoDB's aggregation framework can be used for generating reports and analyzing trends, such as sales data, popular products, or artisan performance, helping the hub make data-driven business decisions.
- **Ease of Integration with Frontend:** MongoDB's flexible data format works well with modern frontend frameworks like React.js or Vue.js, enabling the application to interact seamlessly with the database.

MANGO DB

MongoDB is a popular open-source NoSQL database that stores data in a flexible, document-oriented format. Unlike traditional relational databases, which use tables and rows, MongoDB stores data in collections and documents, using BSON (binary JSON) format. Each document can contain complex, nested data structures, providing flexibility to model data in a way that aligns more naturally with modern application needs. MongoDB is schema-less, allowing documents within the same collection to have different structures, making it easy to

adapt as application requirements evolve. This is particularly useful for projects like a textile design hub, where product details, customer data, and designs can vary greatly.

Features of Mango DB

- MongoDB stores data in documents (BSON format), making it more flexible compared to traditional relational databases that store data in tables.
- It allows you to store data without a fixed schema, meaning documents in the same collection can have different structures, providing flexibility in data modeling.
- It can scale horizontally by distributing data across multiple servers (sharding), making it ideal for handling large datasets and high-traffic applications.
- MongoDB supports replica sets, where data is replicated across multiple servers to ensure high availability and fault tolerance.
- MongoDB's aggregation framework allows you to process and transform data directly in the database, enabling complex queries such as grouping, filtering, and sorting.

CHAPTER 3

SYSTEM DESIGN

3.1 MODULE DESCRIPTION

Modular programming is the process of subdividing a computer program into separate sub-programs. A module is a separate software component. It can often be used in variety of applications and functions with other components of the system. Module is a software component or part of a program that contains one or more routines. The project contains the following modules:

- User profile module
- Admin dashboard module
- Order management module
- Product management module
- Payment gateway module

USER PROFIE MODULE

The User Profile Module in a textile project is a key component designed to manage and store information related to the users, such as customers, suppliers, or employees. It includes essential features like user information management, where details such as name, contact info, roles, and preferences are stored. This module helps in defining user roles and permissions, ensuring they can only access relevant data or perform authorized actions within the system. The authentication system is integrated, offering secure login methods, including options like two-factor authentication for enhanced security. Additionally, the module tracks the user's history, including orders, production records, and any interactions within the platform. Communication tools such as notifications and alerts keep users updated on order statuses, product availability, and promotions. It also supports integration with other modules, such as inventory or billing systems, allowing for a seamless flow of information. Users can personalize their profiles based on their preferences, which

helps improve engagement and satisfaction. To ensure privacy and data security, the module adheres to privacy regulations and provides users with control over what data they share. Overall, the User Profile Module streamlines operations, improves user experience, and ensures the system is secure and compliant with necessary regulations.

ADMIN DASHBOARD MODULE

The Admin Dashboard Module in a textile project serves as the central hub for administrators to manage and monitor the entire system efficiently. It provides a comprehensive overview of key metrics and real-time data, including sales, production status, inventory levels, and user activity. The dashboard allows administrators to track orders, manage user profiles, and oversee the progress of textile production or product shipments. It typically includes tools for monitoring financial transactions, generating reports, and managing suppliers and customers. Additionally, the module offers advanced controls for managing system settings, user roles, and permissions, ensuring that only authorized individuals can perform sensitive tasks. The Admin Dashboard may also feature customizable widgets and data visualizations, making it easy for admins to quickly assess performance and make informed decisions. By centralizing these functions, the Admin Dashboard streamlines operations, improves efficiency, and enhances the overall management of the textile project.

ORDER MANAGEMENT MODULE

The Order Management Module in a textile project is a critical component that facilitates the efficient processing and tracking of customer orders from initiation to delivery. It enables users to place and customize orders, track their status, and manage various stages of the order lifecycle, such as payment, production, and shipment. This module typically includes features like real-time order tracking, inventory management, and order history, allowing both customers and administrators to view the status and details of each order. It helps streamline communication between different departments, such as sales, production, and logistics, ensuring that the necessary resources are available to fulfill orders on time. The system also allows for easy modification or cancellation of

orders, while automatically updating inventory levels to reflect changes. Additionally, it integrates with billing and payment systems, facilitating smooth transactions and providing customers with receipts and invoices. The Order Management Module ultimately enhances customer satisfaction by ensuring accurate order fulfillment, timely deliveries, and seamless communication throughout the entire process.

PAYMENT GATEWAY MODULE

The Payment Gateway Module using PayPal in a textile project allows customers to securely process payments for their orders through PayPal, one of the most widely used and trusted online payment systems. This module integrates PayPal's secure API into the project, enabling users to make payments using their PayPal accounts or by using credit/debit cards directly through PayPal's payment processing platform. It ensures secure transaction processing by encrypting payment data, offering protection against fraud and unauthorized access. The integration supports various PayPal payment methods, including PayPal balances, credit cards, and debit cards, providing customers with flexible payment options.

The module also supports automatic payment verification, instantly updating the system once the payment is successful, and notifying both customers and administrators about the transaction status. Additionally, PayPal's buyer protection ensures that customers have peace of mind, knowing their transactions are secure. The Payment Gateway using PayPal can also handle refunds and partial payments, making it convenient for users to manage any adjustments. By using PayPal, the system offers a smooth, reliable, and internationally recognized payment process, enhancing the user experience and ensuring that payments are processed efficiently and securely in the textile project.

3.2 DATAFLOW DIAGRAM

The Data Flow Diagram provides information about the input and output of each entity and process itself. Data Flow Diagram is a graphical representation of data flow in any system. It is capable of illustrating incoming data flow, outgoing data flow and store data.

LEVEL 0 DFD

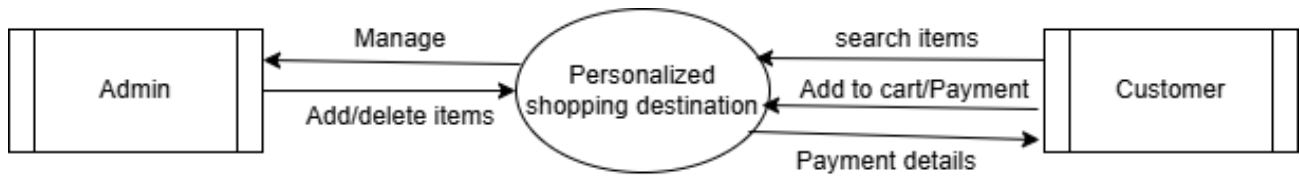


Figure 3.1 Data Flow Diagram Level 0

In Figure 3.1, Administrator and customer can login to the system and make use of their respective process in the system. This shows the overall view of the application where every customer need to register in this shop's website which will verify the credentials and navigate to the appropriate page.

LEVEL 1 DFD

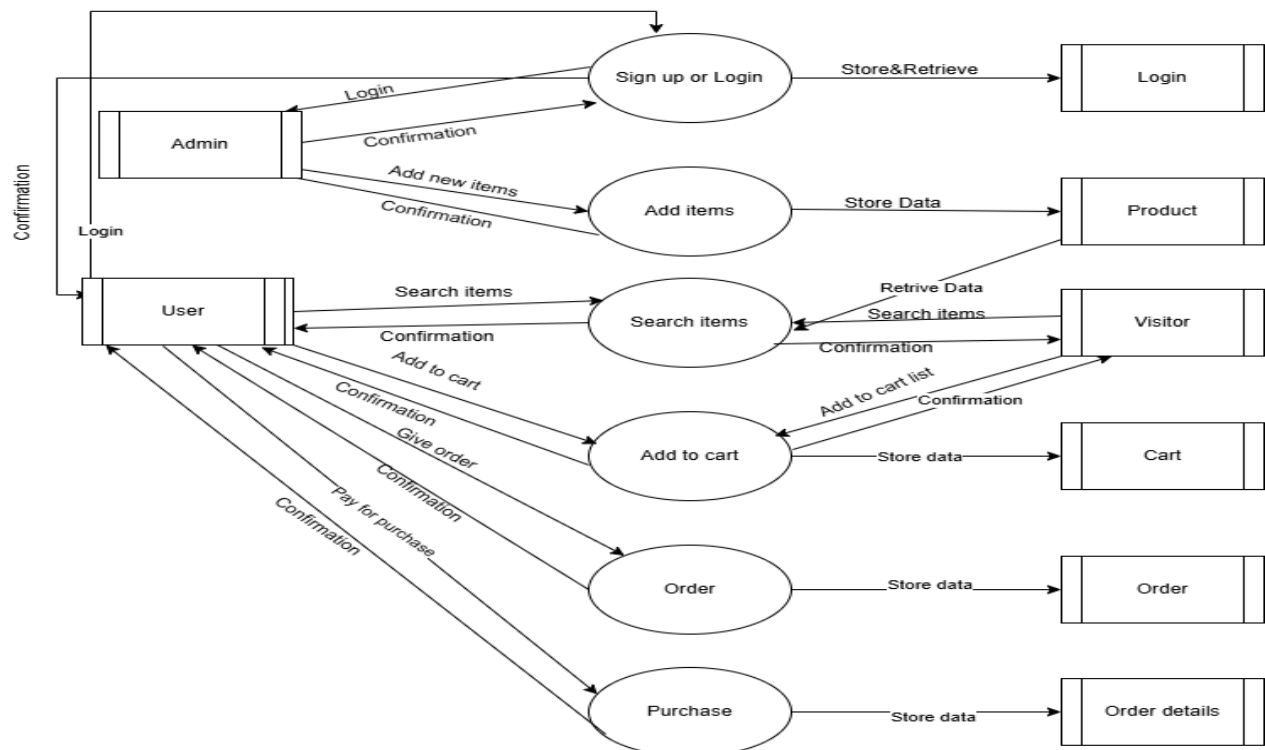


Figure 3.2 Data Flow Diagram Level 1

In Figure 3.2, Administrator can login to the portal and manages the user details, products and stock details. The user can also login to their account and order the product and view the product and manage the account. The above figure shows the user and administrator handling process. Data flows for each process from customer and administrator to the database is described.

3.3 USE CASE DIAGRAM

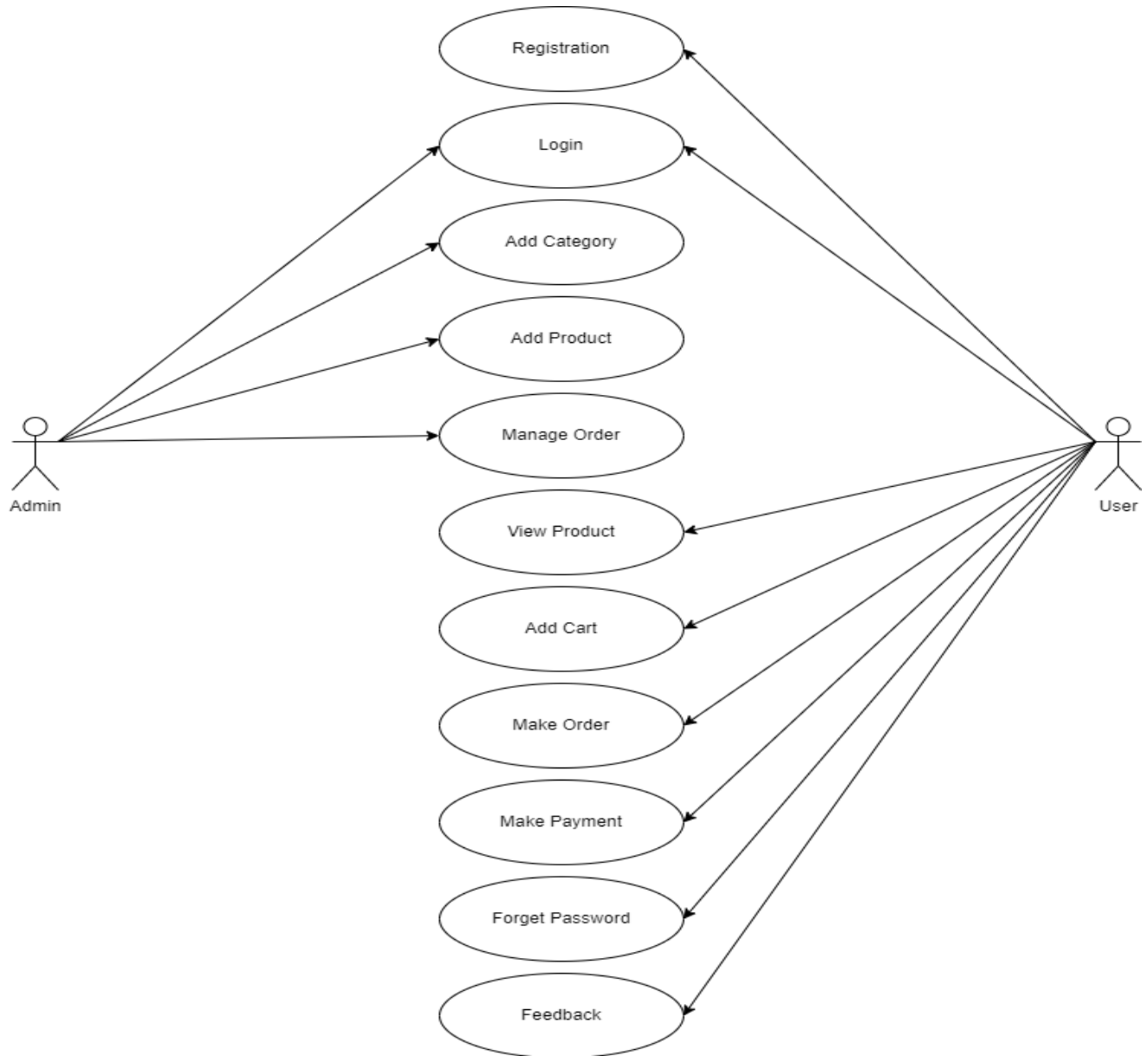


Figure 3.3 Use Case Diagram

In Figure 3.3, the interactions between various actors and the platform's functionalities. The diagram includes several key actors, such as Clients, Designers, Admins, and the Payment Gateway. The Client actor interacts with the platform by browsing available textile designs, customizing them, placing orders, making payments, and leaving feedback or reviews on purchased products.

3.4 CLASS DIAGRAM

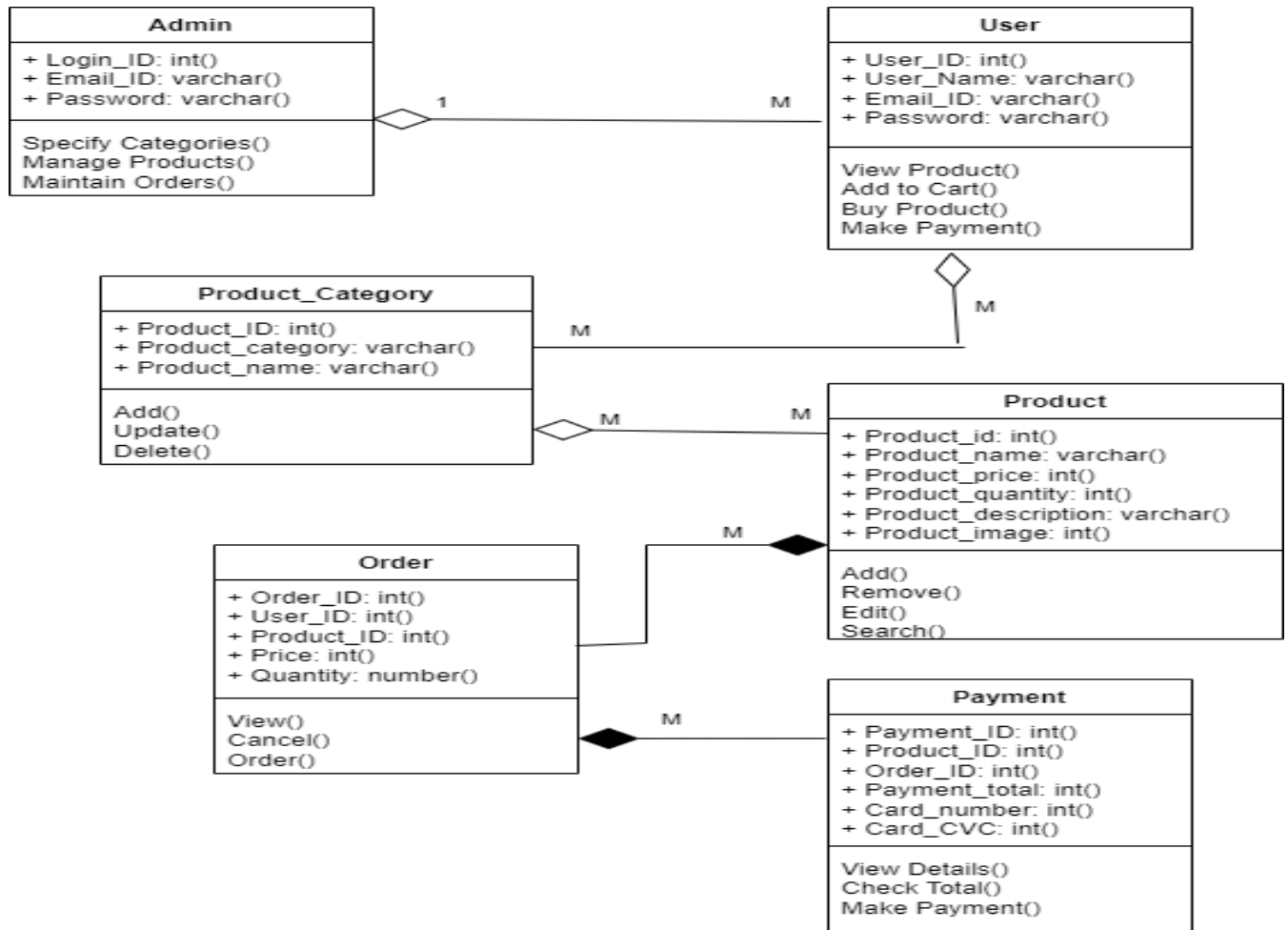


Figure 3.4 Class Diagram

In Figure 3.4 Class Diagram represents the core structure and relationships between key system components. It defines several primary classes, each responsible for different aspects of the system's functionality. The **Client** class captures essential client details such as client ID, name,

email, address, and payment information. It includes methods for browsing designs, customizing them, placing orders, making payments, and leaving feedback on purchased products.

3.5 INPUT DESIGN

The input design for **Personalized Shopping Destination** in a textile project aims to create a unique and tailored shopping experience by leveraging technology to cater to the individual preferences and needs of customers. The platform allows users to create personalized profiles where they can save details such as size, preferred colors, and fabric types. Using this information, the system offers product recommendations, curated fabric collections, and customized design options based on their previous purchases, browsing history, and style preferences. Advanced features like virtual try-ons or 3D previews further enhance the experience, allowing customers to visualize products before making a purchase. The platform also includes advanced search filters, enabling users to easily find fabrics based on material, texture, color, or sustainability criteria. Integration with secure payment gateways like PayPal ensures seamless transactions, while loyalty programs and personalized discounts encourage customer retention. With its user-friendly interface, mobile compatibility, and tailored approach, the Personalized Shopping Destination not only enhances the shopping experience but also fosters stronger connections between customers and the textile brand.

User Register Form

The screenshot displays a web browser window with the address bar showing 'localhost:3000/signup?redirect=/' and the page title 'Sign Up'. The website header includes 'RAGAVENTHRAA TEX' and a search bar. The main content area is titled 'Sign Up' and contains a registration form with the following fields:

- Name: prabhu s
- Email: Prabhus.23mca@kongu.edu
- Nic: s
- Address: 27, Surampatti.Erode.
- Phone Number: 9360837634
- Password: *****
- Confirm Password: *****

The Windows taskbar at the bottom shows the system clock as 08:57 AM on 04-03-2025.

Figure 3.5 User Register Form

Figure 3.5 describes the user register form the customer can register with their name and password. The customer should register themselves for further notification about their order details. If already have an account, just click the login option in the form. When a user registers on an online store, can create personal account that allows them to make purchases.

User Login Form

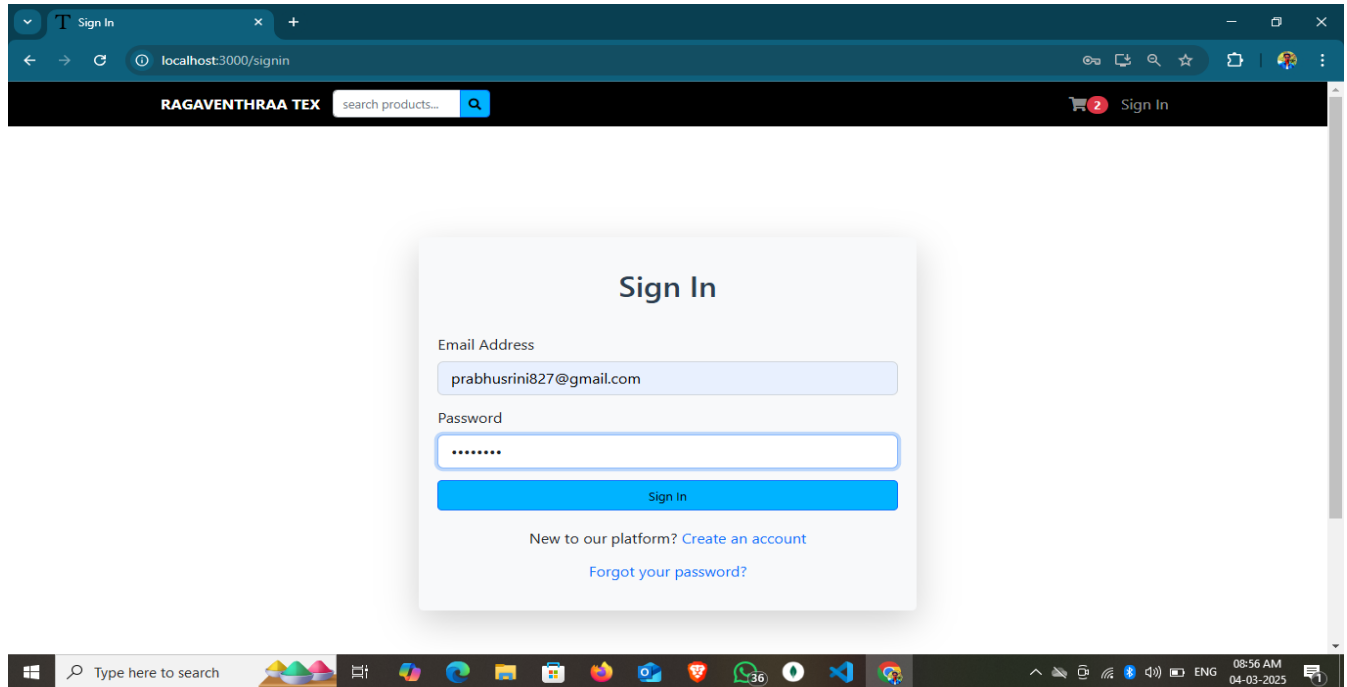
The image is a screenshot of a web browser window. The browser's address bar shows 'localhost:3000/signin'. The website's header is dark blue with the text 'RAGAVENTHRAA TEX' on the left, a search bar in the center, and a shopping cart icon with a '2' next to it and a 'Sign In' link on the right. The main content area is white and features a 'Sign In' form. The form has a title 'Sign In' in bold. Below the title are two input fields: 'Email Address' with the value 'prabhusrini827@gmail.com' and 'Password' with masked characters '.....'. A blue 'Sign In' button is positioned below the password field. At the bottom of the form, there are two links: 'New to our platform? Create an account' and 'Forgot your password?'. The Windows taskbar is visible at the bottom of the screen, showing various application icons and the system clock indicating 08:56 AM on 04-03-2025.

Figure 3.6 User Login Form

Figure 3.6 describes the user login form, the customer can login with their registered email id and password to access the portal. Finally click the login button to further details. When the user gets logged in, user can add the product to cart and purchase the products which are needed. If user doesn't have an account, click on sign up option.

3.6 OUTPUT DESIGN

In any system, the results of processing are conveyed to users and other systems through outputs. A high-quality output is one that satisfies the needs of the end user and provides the information in an understandable manner. How the information will be displaced for immediate

demand and the hard copy output are both decided during output design. It serves as the user's primary and most direct source of information. The system's ability to support user decision-making is improved through efficient and intelligent output design.

Computer output design should be done in an organized, well-thought-out manner, with the correct output developed while ensuring that each output component is designed so that users may utilize the system efficiently. When analysing computer generated output for design purposes, it is important to pinpoint the precise output that will satisfy the specifications. Create documents, reports, or other formats that contain information produced by the system.

The output form of an information system should accomplish the convey information about past activities, current status or projections of the Future and Signal important events, opportunities, problems, or warnings. Trigger and confirm an action.

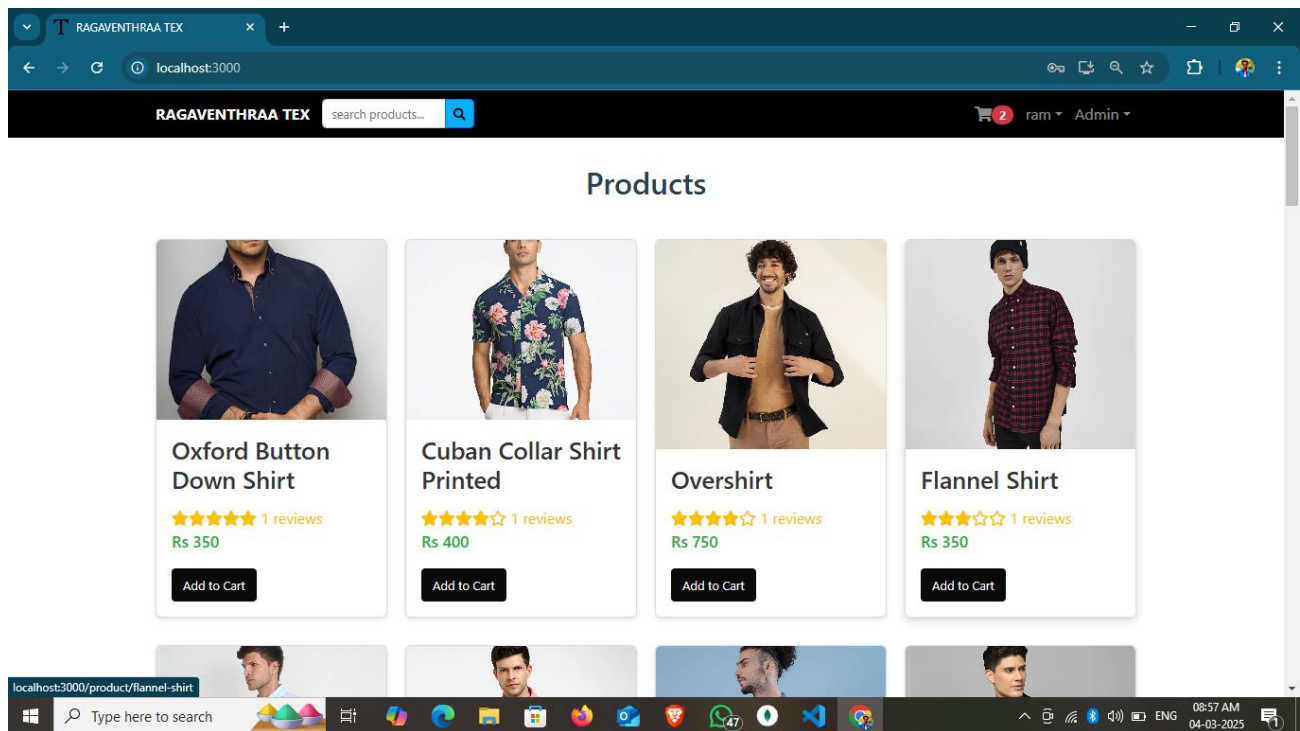


Figure 3.7 User Home Page

Figure 3.7 shows the user home page. The customer dashboard page on an online handloom products shop website offers users a centralized space to view and manage their account information

and preferences. It commonly includes sections for personal details allowing users to update and modify their information as necessary. Additionally, this page shows an order history section, display a comprehensive list of the customer's previous orders, including details such as order dates, items purchased and order statuses.

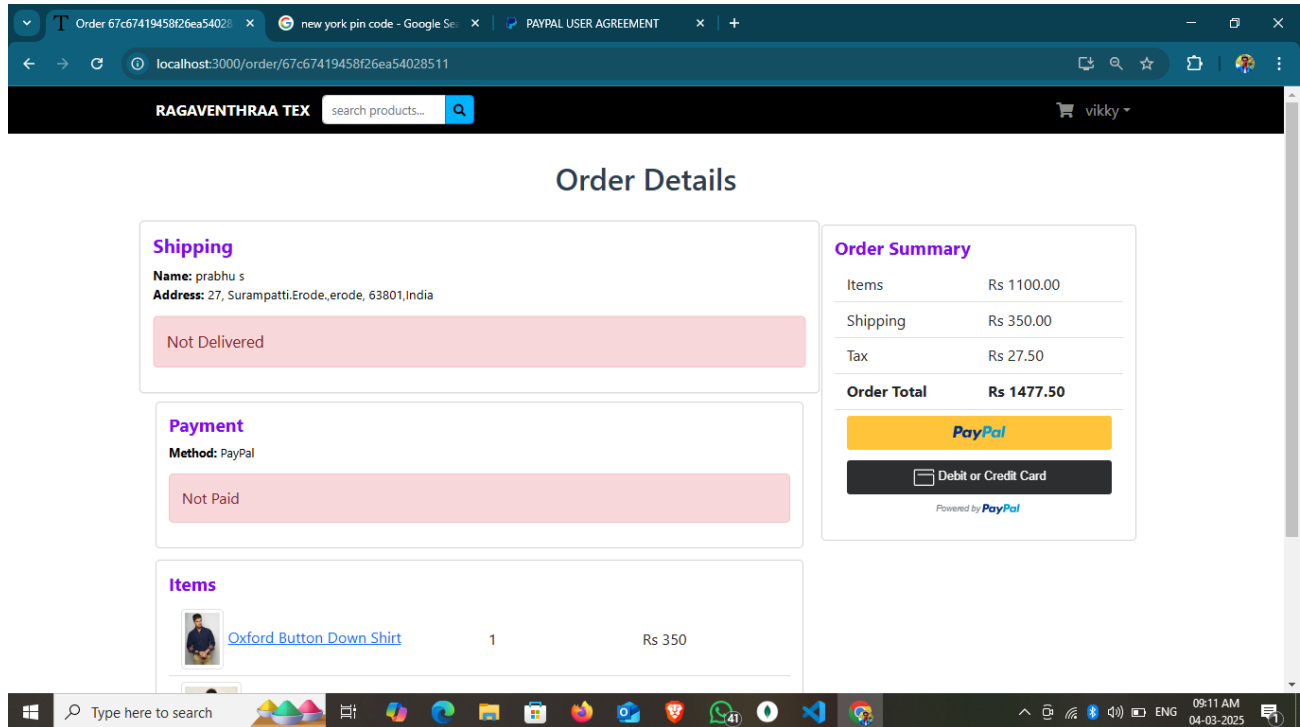


Figure 3.8 Payment page

Figure 3.7 shows product payment page to serves as the final step in the checkout process, where customers securely complete their transactions for orders. It provides an intuitive interface for users to review their purchase details, including the selected products, quantities, and total cost, before proceeding to payment. Customers can choose from various payment methods, such as credit/debit cards, PayPal, or other digital wallets. The page is designed to ensure security, featuring encryption and compliance with industry standards to protect sensitive financial information. It also typically includes options for applying discount codes or loyalty rewards, and users receive confirmation of their successful transaction and order summary upon payment completion.

CHAPTER 4

SYSTEM IMPLEMENTATION

System implementation is a set of procedures performed to complete the design contained in the approved system design document and to test, install, and begin to use the new or revised information system. Implementation allows the users to handle the system and plan for a smooth conversion. After the approval of the system by both end user and management the system was implemented.

4.1 IMPLEMENTATION

The implementation phase marks the transition from a conceptual system design to a functional operational system. It commences with meticulous planning, where decisions are made regarding the methods and timelines to be employed. A successful implementation process necessitates the conversion of existing manual records into digital formats, streamlining operations by utilizing computerized systems.

Central to the implementation process is the changeover stage, which entails assessing the developed tool against predefined criteria to ensure its suitability for user acceptance. This step is crucial in validating that the system meets the users' requirements effectively. Furthermore, users undergo comprehensive training on software utilization to ensure smooth adoption and integration into their workflows.

Implementation encompasses various activities, including individual programming tasks, rigorous system testing, user training sessions, and the full-scale operationalization of the proposed system's application subsystems. Key to this phase is the education of end-users, ideally initiated during the project's initial stages, to familiarize them with the forthcoming changes and prepare them for the transition.

4.2 CODE DESCRIPTION

Code description can be used to summarize code or to explain the programmer's intent. Good comments do not repeat the code or explain it. Programmer will clarify its intent. The back-end is implemented with Node.js and Express, with MongoDB as the database to store and manage data. The database schema includes key models such as User, Product, Order, and

Feedback. The User model handles client, designer, and admin roles, while the Product model stores design details like name, price, and stock. The Order model tracks client purchases and their statuses, and the Feedback model records product ratings and comments. REST API routes handle operations like user authentication, product management, order placement, and feedback submission. The application also ensures secure and efficient data handling, with MongoDB connected through Mongoose for a robust database interface.

4.3 STANDARDIZATION OF THE CODING

Coding standards define a programming style. A coding standard does not usually concern itself with wrong or right in a more abstract sense. It is simply a set of rules and guidelines for the formatting of source code. The other common type of coding standard is the one used in or between development teams. Professional code performs a job in such a way that is easy to maintain and debug. All the coding standards are followed while creating this project. Coding standards become easier, the earlier you start. It is better to do an eat job than cleaning up after all is done. Every coder will have a unique pattern than head here to. Such a style might include the convention coders uses to name variables and functions and how coder comments his work. When the pattern and style is standardized, it pays off the effort well in the long.

4.4 EXCEPTION HANDLING

Exception handling is a process or method used for handling the abnormal statements in the code and executing them. It also enables to handle the flow control of the code/program. For handling the code, various handlers are used that process the exception and execute the code. Mainly if-else block is used to handle errors using condition checking, if-else catch errors as a conditional statement. In many cases there are many which must be checking during an execution but if-else can only handle the defined conditions. In if-else, conditions are manually generated based on the task.

An error is a serious problem than an application does not usually get pass without incident. Errors cause an application to crash, and ideally send an error message offering some suggestion to resolve the problem and return to a normal operating state, there is no way to deal with errors live or in production the only solution is to detect them via error monitoring and bug tracking and dispatch a developer or two to sort out the code.

Unhandled exceptions, also known as errors, occur when an application encounters an exceptional condition for which there is no predefined solution or workaround. These exceptions typically lead to the termination or crashing of the program if not handled properly.

By implementing try...catch statements in programming languages, developers can anticipate and handle exceptions gracefully, allowing the application to continue running and providing a means to address the exceptional condition through custom logic or error handling routines. This proactive approach enhances the robustness and reliability of the software, ensuring a smoother user experience and minimizing unexpected crashes.

CHAPTER 5

SYSTEM TESTING

Software testing is the process of evaluating and verifying that a software product or application does what it is supposed to do. The goal of software testing is to identify and prevent bugs, reduce development costs, and improve performance. There are many different types of software tests, each with specific objectives and strategies which will be discussed below.

5.1 SYSTEM TESTING

Testing is an integral part of any system development life cycle. Insufficient and untested applications may trend to crash and result in loss of economic and manpower investment besides user's dissatisfaction and downfall of reputation. Software testing can be looked upon as one among the many processes, an organization performs, and that provides the last opportunity to correct any flaws in the developed system. Software testing includes selecting test data that have more probability of giving errors.

The first step in system testing is to develop a plan that tests all aspects of the system. Completeness, correctness, reliability and maintainability of the software are to be tested for the best quality assurance that the system meets the specification and requirements for its intended use and performance. System testing is the most useful practical process of executing a program with the implicit intention of finding errors that make the program fails.

System testing is done in three phases

- Unit testing
- Integration testing
- Validation testing

5.1.1 Unit Testing

The procedure level testing is made first. By giving improper inputs, the errors occurred are noted and eliminated. Then the web form level testing is made. For example, storage of data to the table in the correct manner.

Test Cases

Case 1

- Module : Administrator login module
- Test type : Loading of administrator
- Input : Username and password
- Expected Output : Administrator page opens

Test

- Input : Username and password
- Output : Administrator page opens
- Analysis : The expected output is same
- Result : Pass

Case 2

- Module : User module
- Test type : User login
- Input : User name
- Expected Output : Show Success Notification

Test

- Input : User name
- Output : Success Notification shown
- Analysis : The expected output is same
- Result : Pass

5.1.2 Integration Testing

Testing is done for each module. After testing all the modules, the modules are integrated and testing of the final system is done with the test data, specially designed to show that the system will operate successfully in all its aspects conditions. Thus, the system testing is a confirmation that all is correct and an opportunity to show the user that the system works.

Test Cases

Case 1

- Module : Administrator module
- Test type : Working of login, View Orders, View user details
- Input : On clicking respective buttons
- Expected Output : Navigation between modules is completed

Test

- Input : On clicking login, View Orders, View Product details
- Output : Respective forms open correctly
- Analysis : The expected output is same
- Result : Pass

Case 2

- Module : Administrator module
- Test type : Add, edit and delete products, category, manage orders
- Input : Navigation for getting reports
- Expected Output : Reports must be generated

Test

- Input : Add, update and delete products, Control order details
- Output : Reports are generated correctly
- Analysis : The expected output is same
- Result : Pass

5.1.3 Validation testing

The final step involves Validation testing, which determines whether the software function as the user expected. The end-user rather than the system developer conducts this test most software developers as a process called “Alpha and Beta Testing” to uncover that only the end user seems able to find.

Test Cases

Case 1

- Module : User Details module
- Test type : User registration details form
- Input : Input to all fields
- Expected Output : Data type for all the field should be validated

Test

- Input : Enter Contact Number
- Output : Must contain only 10 digits
- Analysis : The expected output is same
- Result : Pass

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

6.1 CONCLUSION

In conclusion, the *Personalized Shopping Destination* textile project provides a dynamic platform that connects designers and clients, offering a customized and seamless shopping experience for unique textile designs. By utilizing React JS for an engaging and responsive front-end and MongoDB for a scalable and efficient back-end, the system ensures smooth management of products, orders, and customer feedback. The project includes specialized modules for clients, designers, and administrators, allowing for easy navigation from browsing designs to processing payments and generating valuable insights. This project not only improves the accessibility and visibility of custom textile designs but also encourages collaboration and innovation within the industry, making it an essential tool for both creative professionals and consumers.

6.2 FUTURE ENHANCEMENT

The Personalized Shopping Destination textile project holds significant potential for future enhancements to further elevate its functionality and user experience. One key upgrade could be the integration of AI-driven design recommendations, offering clients personalized suggestions based on their preferences and browsing history. The inclusion of augmented reality (AR) for virtual try-ons would enable clients to see how designs look on garments or home decor items before making a purchase. Expanding the platform into a global marketplace with multi-currency, multilingual, and region-specific features would help broaden its reach and appeal. Implementing blockchain technology to protect the authenticity and intellectual property rights of exclusive textile designs would provide added value for designers. Eco-conscious options, like filtering designs based on sustainable materials, would attract environmentally conscious clients. Additionally, launching a mobile app for iOS and Android would provide clients with convenience on the go, while real-time collaboration tools would enhance client-designer interactions. Advanced analytics for tracking sales trends and design popularity, detailed order tracking, and community-driven features such as design

contests and forums would further enrich the platform. These improvements would transform the Personalized Shopping Destination textile project into a more innovative, inclusive, and globally impactful solution for the textile industry.

SUMMARY

The Personalized Shopping Destination textile project is a comprehensive and innovative platform designed to bridge the gap between designers and clients, offering a streamlined, efficient process for showcasing, customizing, and purchasing unique textile designs. The platform serves as a one-stop destination for textile design enthusiasts, providing both an engaging and user-friendly experience for clients seeking high-quality, custom-made textiles, while offering designers a space to showcase their creative work and connect with potential buyers.

At the core of the platform is its intuitive front-end, developed with React JS, ensuring that users have an easy-to-navigate and responsive interface. This front-end design is optimized for smooth interaction, allowing clients to effortlessly browse, filter, and explore a wide range of textile designs. Whether on desktop or mobile devices, users can enjoy a seamless experience, with quick load times and personalized recommendations tailored to their preferences. The user interface allows for both visual appeal and functionality, enhancing the shopping experience and encouraging users to interact with the platform for longer periods.

The platform is supported by a robust back-end built on MongoDB, ensuring efficient data management and scalability. This back-end system handles everything from product listings and customer orders to user accounts and feedback, enabling real-time updates and providing administrators with the tools to manage the platform effectively. By utilizing a flexible and scalable database, the system can handle increasing amounts of data and traffic, ensuring that the platform grows with the needs of the business and adapts to future demands without compromising performance.

APPENDICES

A. SAMPLE CODING

Homescreen.js

```
// eslint-disable-next-line no-unused-vars
import { useEffect, useReducer, useState } from "react";
import axios from "axios";
import Row from "react-bootstrap/Row";
import Col from "react-bootstrap/Col";
import Product from "../components/Product";
import { Helmet } from "react-helmet-async";
import LoadingBox from "../components/LoadingBox";
import MessageBox from "../components/MessageBox";
// import data from '../data';

const reducer = (state, action) => {
  switch (action.type) {
    case "FETCH_REQUEST":
      return { ...state, loading: true };
    case "FETCH_SUCCESS":
      return { ...state, products: action.payload, loading: false };
    case "FETCH_FAIL":
      return { ...state, loading: false, error: action.payload };
    default:
      return state;
  }
};
```

```

function HomeScreen() {
  const [{ loading, error, products }, dispatch] = useReducer(reducer, {
    products: [],
    loading: true,
    error: "",
  });
  // const [products, setProducts] = useState([]);
  useEffect(() => {
    const fetchData = async () => {
      dispatch({ type: "FETCH_REQUEST" });
      try {
        const result = await axios.get("/api/products");
        dispatch({ type: "FETCH_SUCCESS", payload: result.data });
      } catch (err) {
        dispatch({ type: "FETCH_FAIL", payload: err.message });
      }

      // setProducts(result.data);
    };
    fetchData();
  }, []);
  return (
    <div>
      <Helmet>
        <title>RAGAVENTHRAA TEX</title>
      </Helmet>
      <h1>Products</h1>
      <div className="products">

```

```

    {loading ? (
      <LoadingBox />
    ) : error ? (
      <MessageBox variant="danger">{error}</MessageBox>
    ) : (
      <Row>
        {products.map((product) => (
          <Col key={product.slug} sm={6} md={4} lg={3} className="mb-3">
            <Product product={product}></Product>
          </Col>
        ))}
      </Row>
    )}
  </div>
</div>
);
}

export default HomeScreen;

```

Cartscreen.js

```

import { useContext } from 'react';
import { Store } from '../Store';
import { Helmet } from 'react-helmet-async';
import Row from 'react-bootstrap/Row';
import Col from 'react-bootstrap/Col';
import MessageBox from '../components/MessageBox';
import ListGroup from 'react-bootstrap/ListGroup';
import Button from 'react-bootstrap/Button';

```



```
import Card from 'react-bootstrap/Card';
import { Link, useNavigate } from 'react-router-dom';
import axios from 'axios';

export default function CartScreen() {
  const navigate = useNavigate();
  const { state, dispatch: ctxDispatch } = useContext(Store);
  const {
    cart: { cartItems },
  } = state;

  const updateCartHandler = async (item, quantity) => {
    const { data } = await axios.get(`/api/products/${item._id}`);
    if (data.countInStock < quantity) {
      window.alert('Sorry. Product is out of stock');
      return;
    }
    ctxDispatch({
      type: 'CART_ADD_ITEM',
      payload: { ...item, quantity },
    });
  };

  const removeItemHandler = (item) => {
    ctxDispatch({ type: 'CART_REMOVE_ITEM', payload: item });
  };

  const checkoutHandler = () => {
    navigate('/signin?redirect=/shipping');
  };
}
```

```
};
```

```
return (
```

```
  <div>
```

```
    <Helmet>
```

```
      <title>Shopping Cart</title>
```

```
    </Helmet>
```

```
    <h1>Shopping Cart</h1>
```

```
    <Row>
```

```
      <Col md={8}>
```

```
        {cartItems.length === 0 ? (
```

```
          <MessageBox>
```

```
            Cart is empty. <Link to="/">Go Shopping</Link>
```

```
          </MessageBox>
```

```
        ) : (
```

```
          <ListGroup>
```

```
            {cartItems.map((item) => (
```

```
              <ListGroup.Item key={item._id}>
```

```
                <Row className="align-items-center">
```

```
                  <Col md={4}>
```

```
                    <img
```

```
                      src={item.image}
```

```
                      alt={item.name}
```

```
                      className="img-fluid rounded img-thumbnail"
```

```
                    ></img>{' '}
```

```
                    <Link to={` /product/${item.slug}`}>{item.name}</Link>
```

```
                  </Col>
```

```
                <Col md={3}>
```

```

<Button
  onClick={() =>
    updateCartHandler(item, item.quantity - 1)
  }
  variant="light"
  disabled={item.quantity === 1}
>
  <i className="fas fa-minus-circle"></i>
</Button>{' '}
<span>{item.quantity}</span>{' '}
<Button
  variant="light"
  onClick={() =>
    updateCartHandler(item, item.quantity + 1)
  }
  disabled={item.quantity === item.countInStock}
>
  <i className="fas fa-plus-circle"></i>
</Button>
</Col>
<Col md={3}>Rs {item.price}</Col>
<Col md={2}>
  <Button
    onClick={() => removeItemHandler(item)}
    variant="light"
  >
    <i className="fas fa-trash"></i>
  </Button>

```

```

        </Col>

    </Row>

    </ListGroup.Item>

    )})
</ListGroup>

)}
</Col>

<Col md={4}>
    <Card>
        <Card.Body>
            <ListGroup variant="flush">
                <ListGroup.Item>
                    <h3>
                        Subtotal ({cartItems.reduce((a, c) => a + c.quantity, 0)} items) : Rs{' '}
                        {cartItems.reduce((a, c) => a + c.price * c.quantity, 0)}
                    </h3>

                    </ListGroup.Item>
                    <ListGroup.Item>
                        <div className="d-grid">
                            <Button
                                type="button"
                                variant="primary"
                                onClick={checkoutHandler}
                                disabled={cartItems.length === 0}
                            >
                                <h5>Proceed to Checkout</h5>
                            </Button>

```

```

        </div>
      </ListGroup.Item>
    </ListGroup>
  </Card.Body>
</Card>
</Col>
</Row>
</div>
);
}

```

Paymentmethodscreen.js

```

import React, { useContext, useEffect, useState } from 'react';
import { Helmet } from 'react-helmet-async';
import { useNavigate } from 'react-router-dom';
import Form from 'react-bootstrap/Form';
import Button from 'react-bootstrap/Button';
import CheckoutSteps from '../components/CheckoutSteps';
import { Store } from '../Store';

export default function PaymentMethodScreen() {
  const navigate = useNavigate();
  const { state, dispatch: ctxDispatch } = useContext(Store);
  const {
    cart: { shippingAddress, paymentMethod },
  } = state;

  const [paymentMethodName, setPaymentMethod] = useState(
    paymentMethod || 'PayPal'
  );

```

```
);
```

```
useEffect(() => {  
  if (!shippingAddress.address) {  
    navigate('/shipping');  
  }  
}, [shippingAddress, navigate]);  
const submitHandler = (e) => {  
  e.preventDefault();  
  ctxDispatch({ type: 'SAVE_PAYMENT_METHOD', payload: paymentMethodName });  
  localStorage.setItem('paymentMethod', paymentMethodName);  
  navigate('/placeorder');  
};  
return (  
  <div>  
    <CheckoutSteps step1 step2 step3></CheckoutSteps>  
    <div className="container small-container">  
      <Helmet>  
        <title>Payment Method</title>  
      </Helmet>  
      <h1 className="my-3">Payment Method</h1>  
      <Form onSubmit={submitHandler}>  
        <div className="mb-3">  
          <Form.Check  
            type="radio"  
            id="PayPal"  
            label="PayPal"  
            value="PayPal"
```

```
        checked={paymentMethodName === 'PayPal'}
        onChange={(e) => setPaymentMethod(e.target.value)}
      />
    </div>
    <div className="mb-3">
      <Form.Check
        type="radio"
        id="Cash On Delivery"
        label="Cash On Delivery"
        value="Cash On Delivery"
        checked={paymentMethodName === 'Cash On Delivery'}

        onChange={(e) => setPaymentMethod(e.target.value)}
      />
    </div>
    <div className="mb-3">
      <Form.Check
        type="radio"
        id="Card"
        label="Card"
        value="Card"
        checked={paymentMethodName === 'Card'}
        onChange={(e) => setPaymentMethod(e.target.value)}
      />
    </div>
    <div className="mb-3">
      <Button type="submit">Continue</Button>
    </div>
```

</Form>

</div>

</div>

);

}

B. SCREENSHOTS

HOME PAGE

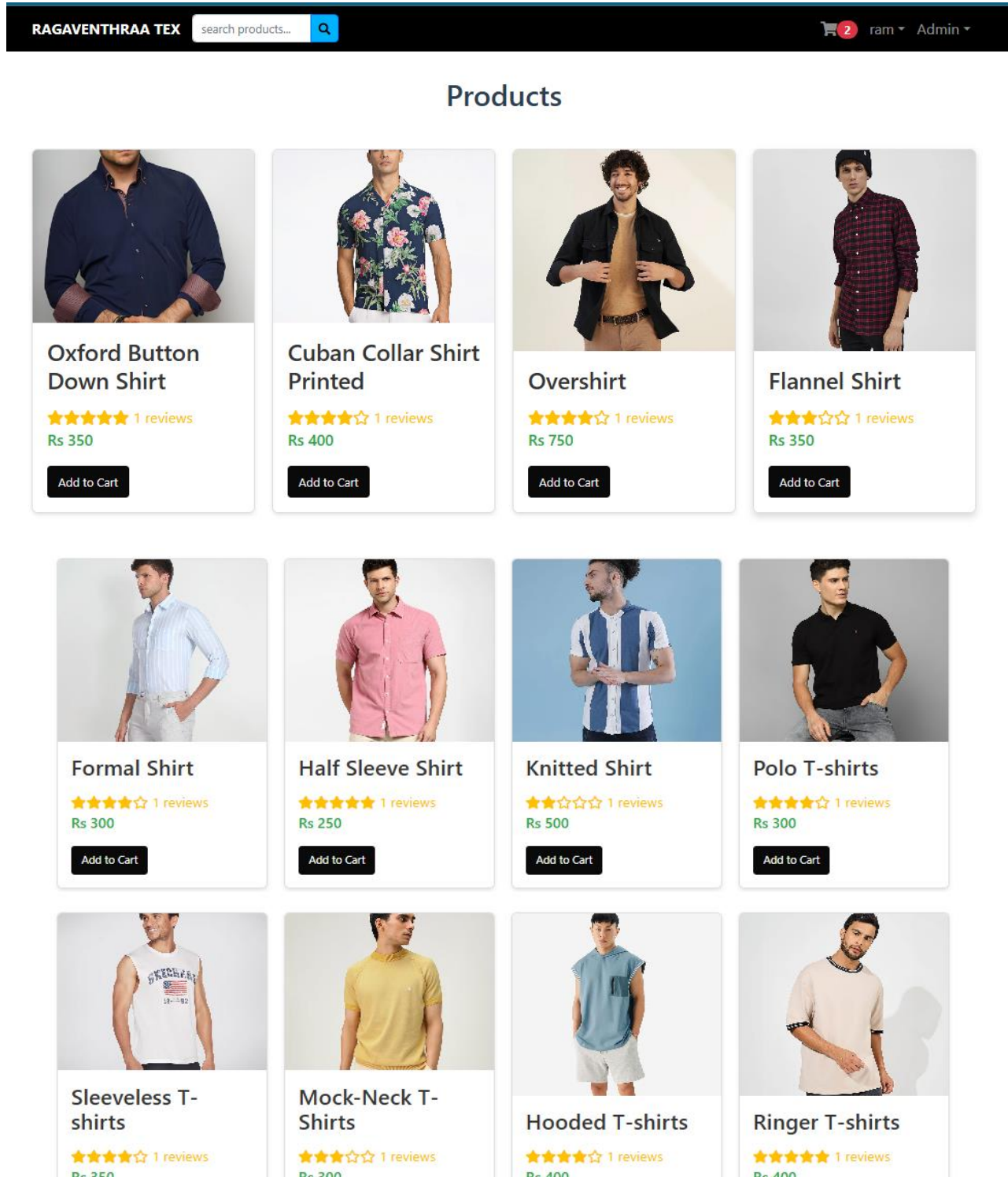


Figure B.1 Home Page

The home page features a captivating display of renowned brands and a wide array of products, providing users with a diverse selection. With seamless navigation, users can effortlessly explore the extensive range of offerings tailored to their preferences. Whether it's exploring trusted brands or discovering new products, the home page offers a dynamic and engaging experience. The products are shown in Figure B.1.

ADD TO CART PAGE

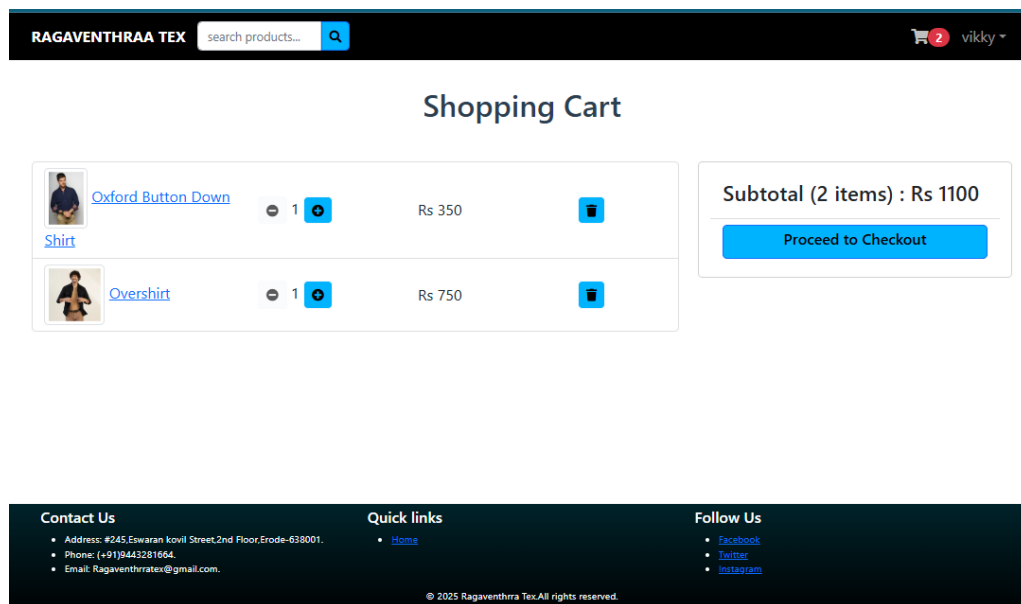


Figure B.2 Add to Cart Page

In Figure B.2, streamlines the shopping experience, allowing users to conveniently manage their selected items. It provides a clear overview of chosen products and enables users to adjust quantities or remove items as needed. With intuitive controls and a straightforward layout, the "Add to Cart" page enhances the checkout process for seamless transactions.

ADMIN DASHBOARD

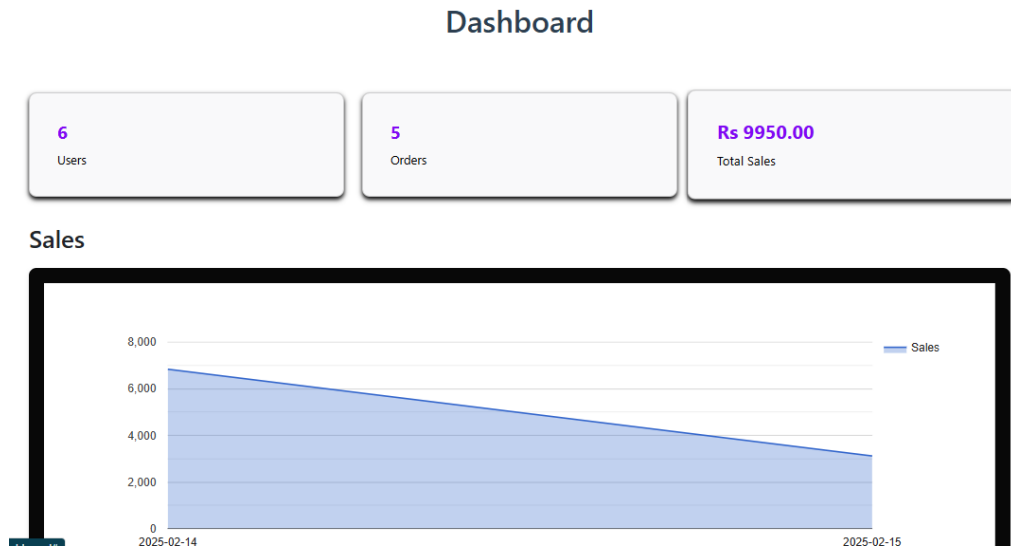


Figure B.3 Admin Dashboard

In Figure B.3, The Admin Dashboard for a Personalized Shopping Destination web application is designed to streamline the management of textile products, user personalization, orders, and analytics. The dashboard enables admins to manage users by adding, updating, and tracking their activity while also assigning roles such as Admin, Seller, and Customer.

PRODUCTS REPORT

The Products Report form for RAGAVENTHRAA TEX includes a search bar, a 'Download Report' button, and a 'Create Product' button. The form fields are: Product Name (shirt), Price (500), Category (male), Brand (levis), Stock Quantity (60), Description (Flannel shirts will probably never be out of style, we've seen them for years and can't get bored of the simple patterns, and the fuzzy soft texture makes them our favourite. We found), and Image URL (https://imgmediagumlet.lbb.in/media/2023/05/64745b1716cbc35fdad2c6e4_1685347095799.jpg).

RAGAVENTHRAA TEX

search products...

[Download Report](#)

RAGAVENTHRAA TEX

Products Report

Product Name:

Price:

Category:

Brand:

Stock Quantity:

Description:

Image URL:

[Create Product](#)

Figure B.4 Products Report

In Figure B.4, The Products Report in the admin dashboard of the Personalized Shopping Destination provides a detailed analysis of the product inventory, sales trends, and customer preferences. It includes key metrics such as the total number of products available, stock levels, top-selling fabrics, and products with low inventory. The report categorizes products based on fabric type, design, price range, and popularity, helping administrators make informed decisions about restocking and promotions.

ORDER REPORT



RAGAVENTHRAA TEX						
search products... 						
 2 ram Admin						
Download Order Summary						
RAGAVENTHRAA TEX						
Orders Report						
Order ID	USER	DATE	TOTAL	PAID	DELIVERED	ACTIONS
1	ram	2025-02-14	1170.00	No	No	Details Delete
2	vikky	2025-02-14	2707.50	No	No	Details Delete
3	ram	2025-02-14	2195.00	No	No	Details Delete
4	vikky	2025-02-14	760.00	No	No	Details Delete
5	vikky	2025-02-15	3117.50	No	No	Details Delete

Figure B.5 Orders Report

In Figure B.5, The Orders Report in the admin dashboard of the Textile Personalized Shopping Destination provides a comprehensive overview of order processing, sales performance, and customer purchasing behavior. It includes key details such as the total number of orders placed, completed, pending, canceled, and refunded, helping administrators track order fulfillment efficiency. The report categorizes orders based on payment status, delivery status, and order value, providing insights into revenue generation and operational effectiveness.

USERS REPORT

RAGAVENTHRAA TEX

search products...

2

ram

Admin

Download

Users

NAME	EMAIL	NIC	Address	Phone Number	IS ADMIN	ACTIONS
test	test@gmail.com	sgdgsgfdsgfc	test	9360837634	NO	<div>EditDelete</div>
test	test1@gmail.com	sjfjkdsjfhjd	test	1234567861	YES	<div>EditDelete</div>
SS 36 prabhu s	Prabhus.23mca@kongu.edu	sgdgsgfdsgfc	27, Surampatti.Erode.	9360837634	NO	<div>EditDelete</div>
ram	ram@gmail.com	sgdgsgfdsgfc	27, Surampatti.Erode.	9360837634	YES	<div>EditDelete</div>
1	eee@gmail.com	sxasassxasx	27, Surampatti.Erode.	9360837634	NO	<div>EditDelete</div>
vikky	vikky@gmail.com	senthilvik	1,nggo colony,erode.	1234567861	NO	<div>EditDelete</div>

Contact Us

Quick links

Follow Us

Figure B.6 Users Report

In Figure B.6, The Users Report in the admin dashboard of the Personalized Shopping Destination provides valuable insights into customer activity, user demographics, and engagement patterns. It includes key metrics such as the total number of registered users, active and inactive users, new sign-ups, and customer retention rates.

PRODUCT CATEGORIES

RAGAVENTHRAA TEX

search products...

vikky

Department

Any

Frocks- Women

Jeans- Men

Men Tshirts

Shirts- Men

Shoes

Suit- Men

Watch

Price

Any

Rs 1 to Rs 500

Rs 501 to Rs 1000

Rs 1001 to Rs 3000

Above Rs 4000

Avg. Customer Review

★★★★★ & up

★★★★☆ & up


★★★☆☆ & up

★★☆☆☆ & up

★☆☆☆☆ & up

29 Results

Sort by Newest Arrivals




frocks

★★★★★ 0 reviews

Rs 3500

Add to Cart




frock

★★★★★ 1 reviews

Rs 2000

Add to Cart





woman shoes

★★★★★ 1 reviews

Rs 3000

Add to Cart








Figure B.7 Product Categories

In Figure B.7, The Product Categories in the Personalized Shopping Destination web application help in organizing and classifying textile products based on various attributes, making it easier for customers to browse and shop efficiently

SHIPPING DETAILS

The screenshot shows the 'Enter Shipping Address' form in the Ragaventhraa Tex web application. The header bar is black with the brand name 'RAGAVENTHRAA TEX' on the left, a search bar with the placeholder 'search products...' and a magnifying glass icon in the center, and a shopping cart icon with a red '2' and the text 'vikky' on the right. Below the header is a progress bar with four steps: 'Sign-In', 'Shipping' (highlighted in orange), 'Payment', and 'Place Order'. The main heading 'Enter Shipping Address' is centered. The form fields are: 'Full Name' with the value 'prabhu s', 'Address' with the value '27, Surampatti.Erode.', 'District' with the value 'erode', 'Postal Code' with the value '63801', and 'Country' with a radio button selected for 'India'. A blue 'Continue' button is at the bottom.

Figure B.8 Shipping Address

In Figure B.8, The Shipping Address in the Personalized Shopping Destination web application is a crucial component of the order fulfillment process, ensuring accurate and timely delivery of products to customers. During checkout, users are required to enter their shipping details, including recipient name, street address, city, state, postal code, and country.

PAYMENT METHOD

The screenshot shows the 'Payment Method' form in the Ragaventhraa Tex web application. The header bar is black with the brand name 'RAGAVENTHRAA TEX' on the left, a search bar with the placeholder 'search products...' and a magnifying glass icon in the center, and a shopping cart icon with a red '2' and the text 'vikky' on the right. Below the header is a progress bar with four steps: 'Sign-In', 'Shipping', 'Payment' (highlighted in orange), and 'Place Order'. The main heading 'Payment Method' is centered. The form has three radio button options: 'PayPal' (selected), 'Cash On Delivery', and 'Card'. A blue 'Continue' button is at the bottom.

Figure B.9 Payment Method

In Figure B.9, The Payment Method in the Textile Personalized Shopping Destination web application offers customers a secure and convenient way to complete their purchases. The platform supports multiple payment options, including credit and debit cards, digital wallets such as PayPal, Google Pay, and Apple Pay, as well as net banking and UPI for seamless transactions.

PLACE ORDER

The screenshot displays the 'Order Details' section of the Ragaventhraa Tex web application. The header includes the brand name 'RAGAVENTHRAA TEX', a search bar, and a shopping cart icon labeled 'vikky'. The main content area is divided into three sections: 'Shipping', 'Payment', and 'Items'. The 'Shipping' section shows a status of 'Not Delivered'. The 'Payment' section shows a status of 'Not Paid'. The 'Items' section is partially visible. A floating notification box indicates 'Payment successful!'. To the right, an 'Order Summary' table provides a breakdown of costs.

Order Summary	
Items	Rs 1100.00
Shipping	Rs 350.00
Tax	Rs 27.50
Order Total	Rs 1477.50

Below the summary table, there are input fields for 'Card number' and 'MM / YY CVC', along with a 'Pay with Stripe' button.

Figure B.10 Order Details

In Figure B.10, The Order Details section in the Personalized Shopping Destination web application provides customers and administrators with a comprehensive summary of each purchase. When an order is placed, users can view key details such as the order ID, product names, quantity, price breakdown, payment method, shipping address, and expected delivery date.

REFERENCES

- [1] React: Up & Running – Stoyan Stefanov, O'Reilly Media, 2016.
- [2] Fullstack React: The Complete Guide to ReactJS and Friends – Anthony Accomazzo et al., Fullstack.io, 2017.
- [3] The Road to React – Robin Wieruch, Independently published, 2018.
- [4] React and React Native, 2nd Edition – Adam Boduch, Packt Publishing, 2019.
- [5] React Explained: Your Step-by-Step Guide to React – Zac Gordon, Independently published, 2019.
- [6] React Projects – Roy Derks, Packt Publishing, 2019.
- [7] Learning React, 2nd Edition – Alex Banks & Eve Porcello, O'Reilly Media, 2020.
- [8] React Cookbook: Recipes for Mastering the React Framework – David Griffiths & Dawn Griffiths, O'Reilly Media, 2020.
- [9] The React Workshop – Brandon Richey, Packt Publishing, 2020.
- [10] React 17 Design Patterns and Best Practices, 3rd Edition – Carlos Santana Roldán, Packt Publishing, 2021.