ON STREAM PORTAL FOR PAINT SHOP

A MINI PROJECT REPORT

Submitted by

BALACHANDRAN S

(Reg. No: 24MCR007)

GHAUTHAM M

(Reg. No: 24MCR025)

JEGADEEP P

(Reg. No: 24MCR045)

in partial fulfillment of the

requirements for the award of the

degree of

MASTER OF COMPUTER APPLICATIONS
IN

DEPARTMENT OF COMPUTER APPLICATIONS

KONGU ENGINEERING COLLEGE

(Autonomous)

PERUNDURAI ERODE – 638 060

DECEMBER 2024

DEPARTMENT OF COMPUTER APPLICATIONS

KONGU ENGINEERING COLLEGE

BONAFIDE CERTIFICATE

This is to certify that the project report titled **ON STREAM PORTAL FOR PAINT SHOP** is the approved record of work done by **BALACHANDRAN S (REG.NO:24MCR007), GHAUTHAM M (REG.NO:24MCR025), JEGADEEP P (REG.NO:24MCR045)** in partial fulfilment for the award of Master of Computer Applications of Anna University Chennai during the year 2024-2025.

SUPERVISOR **HEAD OF THE DEPARTMENT**

(Signature with seal)

Date:

Submitted for the end semester viva-voce examination held on _____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION

We affirm that the project report titled **ON STREAM PORTAL FOR PAINT STORE** being submitted in partial fulfilment for the award of **MCA Degree in Master of Computer Applications** is the original work carried out by us. It has not formed the part of any other project submitted for award of any degree, either in this or any other university.

BALACHANDRAN S (REG.NO:24MCR007)

GHAUTHAM M (REG.NO:24MCR025)

JEGADEEP P(REG.NO:24MCR045)

I certify that the declaration made by the above candidates is true to the best of my knowledge.

Signature of the Supervisor

(Name & Designation)

ACKNOWLEDGEMENT

We express our sincere thanks to our beloved correspondent **Thiru.A.K.ILANGO B.Com., M.B.A., LLB.,** and other philanthropic trust members of Kongu Vellalar Institute of Technology Trust for having provided with necessary resources to complete this project.

We are always grateful and heartfelt thanks to our beloved visionary Principal **Dr.V.BALUSAMY B.E.(Hons)., M.Tech., Ph.D.,** for his motivation and moral support.

We express our deep sense of gratitude and profound thanks to **Dr.A.TAMILARASI M.SC.,M.Phi.,Ph.D.,M.Tech.,** Head of the Department, Master of Computer Applications for her invaluable commitment and guidance for this project.

We are in immense pleasure to express our hearty thanks to our beloved Coordinators **S.P.VIJAYAKUMAR** MCA,., M.Phil., Ph.d.,MBA, and our project guide **Dr.M.PYINGKODI** MCA, M.Phil., PhD for providing valuable guidance and constant support throughout the courseour project. We also thank the teaching, non-teaching staff members, fellow students andour parents who stood with us to complete our project successfully.

ABSTRACT

In response to the growing demand for enhanced digital experiences in the retail sector, businesses must adapt to meet the evolving expectations of their customers. This project focuses on the development of a specialized website for a paint store. The primary goal of this project is to leverage website development to enhance the customer experience for a paint store. Through a comprehensive exploration of web technologies, design principles, and e-commerce functionalities, an online platform will be crafted, to showcase the store's extensive range of paint products and also to provide a seamless and engaging shopping experience.

The website is designed to be visually appealing and simple to use. It showcases a wide range of paint colours, finishes, and brands. Each type of paint is presented with a picture and a description, helping customers make informed choices. Understanding the paint selection process, the colour matching, and product information requirements will be crucial in designing a user-friendly interface. This will involve creating navigation structure, optimizing search functionality, and presenting product details effectively.

The project will integrate e-commerce capabilities to facilitate online paint purchases. Secure payment gateways and a streamlined checkout process will be implemented to enhance user convenience and trust. The website is designed to work well on computers, tablets, and smartphones.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABBREVIATION	EXPANSION
HTML	HyperText Markup Language
CSS	Cascading Style Sheets
JS	Java Script
JSX	Java Script XML
JSON	JavaScript Object Notation

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW THE PROJECT

The **ON STREAM PORTAL FOR PAINT SHOP** is a web-based application that uses HTML, CSS, Bootstrap and React.js as front-end development and Firebase, Node.js as back-end development tools**.** The main goal of the project is to provide products to customers in an intelligent way and increase sales effectively. This website facilitates online sales by presenting our wide range of paint products with detailed descriptions, prices.

The website is designed to be easy to navigate and user friendly. Customers can explore, select and purchase products online. We provide a visually appealing and informative platform to showcase our extensive inventory of high-quality paints. Here, the login page is created to protect your inventory from product threats and misuse. To enable e-commerce functionality, allowing customers to shop online, place orders, and experience the convenience of doorstep delivery. The website will provide directions to our physical store for in-person shopping. The website will feature details about the store, the products we offer, and an option to virtually communicate with the shop owner.

The home page providing an overview of the store and a navigation bar to access different sections. The Products page showcases a different variety of paints in the store. Customers can add items to their shopping cart and purchase as needed. Cash on delivery is also available.

In additional features can be added that automatically create and store inventory information after purchase. Now the project is being developed as an online store. It contains a number of tasks such as customer information, product information, product price, purchase information, etc. This application contains a general organization profile and purchasing information. Each new stock is added and initialized with the stored name and price. To ensure that your website is responsive and works on a variety of devices such as desktops, tablets, and mobile phones.

1.2 PROBLEM DEFINITION

The Sri Venkateshwara Paint Store, a well-known retailer specializing in quality paints, needs to establish a strong online presence. The key issue is the lack of an online platform, causing challenges like limited reach and difficulty for customers to find product information. This puts the store at a disadvantage compared to online competitors. The project's main goals are to create an easy-to-use website for online sales and provide detailed product information. It involves using HTML, CSS, Bootstrap, React.js for design, and Firebase and Node.js for functionality. Customers will be able to browse, buy, and receive products conveniently.

CHAPTER 2

SYSTEM ANALYSIS

2.1 EXISTING SYSTEM

The existing system are rooted in traditional in-store processes, lacking online functionalities. In-store operations involve manual tracking of paint products, monitoring stock levels, and processing sales transactions through conventional point-of-sale methods. Customer interactions are limited to in-person experiences, with information conveyed verbally or through physical labels, and transactions completed using cash or cards at the physical checkout counter. Inventory management relies on periodic manual checks and restocking based on observed product depletion. The absence of an online platform poses challenges in market reach and limits accessibility to detailed product information outside the physical store. This analysis sets the groundwork for future enhancements, focusing on integrating online features to address these limitations and propel the paint store into the digital landscape.

2.1.1 Drawbacks of Existing System

- The current system confines the paint store's customer base to local visitors, missing out on the potential to reach a wider audience through online channels.
- Lack of online shopping options forces customers to visit the store physically for every transaction, leading to inconvenience and restricting sales opportunities.
- The store's diverse paint products are not effectively showcased due to the lack of an online platform, which hampers customer awareness and promotional opportunities.
- Reliance on manual inventory management poses the risk of errors, delays, and inaccuracies, impacting stock management and product availability.

2.2 PROPOSED SYSTEM

The focus of the proposed paint store website is to create a specialized online platform that enhances the customer experience and meets the changing expectations of the retail store. Utilizing web technologies and e-commerce features, the website will act as an attractive and easy-to-use display for the store's wide variety of paint products. We will focus on creating a smooth and captivating shopping experience by carefully applying design principles such as intuitive navigation, optimized search, and effective product presentation. By incorporating secure e-commerce functionality, customers will be able to purchase paint online with ease and confidence, ensuring a convenient and reliable transaction experience. This will not only meet the immediate needs of customers but also establish a lasting online presence for the paint store in the competitive retail store.

2.2.1 Advantages of Proposed System

The proposed system has following advantages:

- Reduces the human effort compared to other conventional methods.
- Accuracy and error free transaction process.
- Reports are easily prepared and efficient.
- User Data is maintained by the database and reduces data loss.
- Help the user to reduce the workload and mental conflict.
- Long distance people can also buy paints easily.

2.3 FEASIBILITY STUDY

The feasibility study shows that developing the paint store website is economically viable and strategically aligned with the changing customer demands in the retail industry. The initiative aims to meet the increasing need for improved digital experiences through the use of web technologies and e-commerce capabilities. The cost-benefit analysis shows that the upfront costs for technology infrastructure, skilled personal, and design elements are justified by the expected profits from online sales and a larger customer pool. By implementing e-commerce capabilities, we can enhance the overall customer experience,

increase sales, and build brand loyalty. The long-term cost savings from process automation and targeted digital marketing outweigh the operating costs of website maintenance. The feasibility study confirms that developing the paint store website is not only financially viable but also strategically positioned to succeed in a competitive retail environment.

2.3.1 Economic Feasibility

The development of the paint store website is economically viable due to a through cost-benefit analysis. It is anticipated that the initial investment in technology infrastructure, skilled resources, design elements, and e-commerce integration will result insignificant returns. The potential revenue from online sales and a larger customer base is greater than the costs of maintaining the website, managing content, and providing customer support. By incorporating e-commerce features, while also boosting their brand's presence and building customer loyalty. The expected return on investment comes from higher sales, lower marketing expenses due to targeted digital strategies, and more efficientoperations through process automation. The project's economic viability is confirmed by the long-term cost savings, as well as its positive impact on brand image and market presence, in meeting the changing demands of retail customers.

2.3.2 Operational Feasibility

The paint store website is operationally feasible as it aligns strategically with the operational needs of the retail sector. The main goal is to improve customer experience by smoothly blending web technologies, design principles, and e-commerce functionalities. The focus on creating a user-friendly interface with a streamlined navigation structure and powerful search capabilities is intended to meet the unique demands of customers when choosing paint. Adding e-commerce capabilities for online paint purchases not only fulfils market needs but also simplifies operational processes through a user-friendly checkout process. The website's ability to work well on different devices means it is efficient and accessible to a wide range of customers.

2.3.3 Technical Feasibility

The paint store website technical feasibility lies in the use of advanced web technologies and design principles to build a sophisticated online platform. The website is able to efficiently facilitate online paint purchases. By strategically implementing responsive design techniques, we ensure that our website functions optimally on all devices, such as computers, tablets, and smartphones, making it more accessible to a wide range of users. The primary focus of the project is on database design and management, which will help to create smooth navigation, improve search functionality, and enhance product presentation. By embracing these technical considerations, the proposed solution iswell positioned to achieve the project's goals and shows a dedication to staying ahead of technological advancements in the retail store.

CHAPTER 3

SYSTEM REQUIREMENTS

3.1 Hardware Requirements

Processor	:	Intel Core i3
RAM	:	4 GB RAM
Hard disk	:	500 GB
Keyboard	:	Standard 102 keys
Mouse	:	Optical Mouse

3.2 Software Specification

Operating System	: Windows 11 and below
Environment	: Chrome, Visual Studio Code
Frontend	: HTML, CSS, Bootstrap, React.JS
Scripting language	: JavaScript
Backend	: Node.JS
Database	: Firebase

3.2.1 Front End

HTML

HTML (HyperText Markup Language) stands as the cornerstone tool in shaping the user interface for the website. Operating as a markup language, HTML utilizes a systematic arrangement of tags to define and structure content elements across web pages. The system requirements for HTML are comprehensive, encompassing the creation of

well-organized documents that facilitate seamless communication between web browsers and users. The latest iteration, HTML5, introduces a suite of semantic elements, including `<nav>`, `<article>`, and `<section>`, elevating the document's structural clarity and enhancing accessibility. HTML serves as the framework for constructing various components such as product descriptions, pricing details, and interactive forms, fostering an engaging user experience. The incorporation of HTML5's advanced features supports multimedia integration, enabling the presentation of paint products with rich visual content. The commitment to adhering to HTML standards ensures not only a robustfoundation but also a responsive, user-centric, and visually captivating interface to delivering a top-notch online shopping experience for the customers.

CSS

CSS (Cascading Style Sheets) emerges as a pivotal component in shaping the visual allure and structural integrity of the online platform. Operating as the styling language, CSS meticulously dictates the presentation of HTML elements, contributing to a seamless and aesthetically pleasing user experience. The system requirements for CSS encompass the creation of designs that are not only responsive but also adaptive, leveraging features like media queries to ensure optimal display across an array of devices. CSS extends its versatility to the customization of colors, fonts, spacing, and other stylistic attributes, fostering a coherent and engaging design language. The incorporation of CSS frameworks, potentially utilizing Bootstrap, streamlines the styling process, offering a systematic and efficient approach to crafting design elements. Furthermore, CSS introduces the capacity for implementing animations and transitions, elevating the user experience by introducing dynamic and visually captivating effects. Prioritizing adherence to CSS standards ensures not only a unified design language but also promotes a harmonious balance between functionality and aesthetics. The result is a sophisticated, user-friendly interface that seamlessly integrates design and functionality for an immersive and enjoyable online interaction. CSS offers selectors that enable precise targeting of HTML elements, allowing developers to apply styles selectively. It also supports the concept of classes and IDs, offering a modular and reusable approach to styling.

BOOTSTRAP

Bootstrap, a versatile front-end framework, assumes a central role in defining the visual and structural aspects of the online platform. Renowned for its responsiveness, Bootstrap significantly expedites the development process by providing a comprehensive toolkit of pre-designed components and a flexible grid system. The system requirements for Bootstrap involve seamless integration with HTML and CSS, fostering the creation of adaptive and mobile-first designs. Capitalizing on Bootstrap's extensive library of reusable components, including navigation bars, cards, and forms, ensures a cohesive and standardized appearance, fostering a visually consistent and user-friendly interface. Moreover, Bootstrap facilitates the efficient incorporation of CSS and JavaScript functionalities, enabling the implementation of dynamic features without compromising flexibility. Its emphasis on scalability and modularity empowers developers to create a streamlined and cohesive design language, fostering an immersive and modern user experience. Embracing Bootstrap aligns with the objective of optimizing development efficiency while ensuring a polished, contemporary, and user-centric design ethos for the online platform.

JAVA SCRIPT

JavaScript, a dynamic scripting language, assumes a crucial role in elevating the interactivity and functionality of the online platform. Operating as a client-sideprogramming language, JavaScript imparts real-time updates and enables dynamic content manipulation, fostering a seamless and captivating user experience devoid of page reloads. The system requirements for JavaScript entail seamless integration with HTML and CSS, empowering the development of interactive features ranging from form validations to dynamic image sliders and asynchronous data loading. Its versatility extends to the integration of JavaScript frameworks such as React.js, augmenting its capabilities and facilitating the creation of intricate, responsive, and interactive user interfaces. JavaScript's prowess in handling user inputs, interacting with APIs, and dynamically updating content establishes it as the linchpin for cultivating a dynamic and user-centric online environment.

REACT.JS

React.js, a dynamic and powerful JavaScript library, assumes a pivotal role in shaping the online platform's sophisticated user interface. Renowned for its declarative and component-based paradigm, React.js simplifies the development of interactive and responsive interfaces, fostering a modular and maintainable code structure. In terms of system requirements, React.js seamlessly integrates with HTML and CSS, enabling the creation of encapsulated components that enhance code readability and reusability. Leveraging React's virtual DOM optimizes rendering performance, ensuring efficient updates and a smooth user experience. The unidirectional data flow simplifies state management, contributing to the overall predictability and reliability of the application's behaviour. Beyond these foundational features, React.js integrates seamlessly with various tools and libraries, such as Prop Types for type-checking and Hooks for functional component state management, providing developers with a versatile toolkit for crafting intricate user interfaces. The library's thriving ecosystem extends to state management solutions like Redux, allowing for centralized data control, and React Router for efficient client-side navigation. React.js commitment to reusability, coupled with its expansive ecosystem, empowers developers to implement advanced features, ensuring the creation of a scalable, feature-rich, and user-friendly online platform.

3.2.2 Back End

NODE.JS

Node.js, a dynamic and versatile JavaScript runtime, assumes a paramount role in the comprehensive back-end development of the online platform. Renowned for its non-blocking, event-driven architecture, Node.js ensures optimal concurrency, making it adept at handling numerous simultaneous connections and scalable for high-traffic scenarios. As a critical system requirement, Node.js seamlessly integrates with a myriad of modules and packages, fostering the creation of scalable and performant server-side applications. Its asynchronous I/O operations are instrumental in enhancing the platform's responsiveness, particularly crucial for managing real-time data interactions and delivering

swift server responses. Node.js, employing the Common JS module system, promotes a modular and well-organized codebase, enhancing maintainability and facilitating collaboration among developers. The rich ecosystem surrounding Node.js, exemplified by the npm package manager, grants developers access to an extensive repository of pre-built modules and tools, expediting the development process and promoting code reusability. Beyond its foundational features, Node.js plays a versatile role, facilitating the seamless integration of Web Sockets for real-time bidirectional communication, essential for interactive and dynamic user experiences. Additionally, Node.js excels in supporting the development of RESTful APIs, laying the groundwork for robust and efficient communication between the front-end and back-end components. Embracing Node.js empowers developers to construct a resilient, scalable, and efficient back-end infrastructure for the online platform, ensuring optimal performance, responsiveness, and adaptability to evolving user needs.

FIREBASE

Firebase, a comprehensive mobile and web application development platform, serves as a pivotal component in the development stack of the online platform. Offering a suite of services, Firebase simplifies and accelerates both front-end and back-end development processes. As a real-time NoSQL database, Firebase's Firestore facilitates seamless data storage and retrieval, crucial for managing dynamic content and user interactions. The Authentication service ensures secure user management with various authentication providers. Cloud Functions enable serverless computing, allowing the execution of backend logic without managing servers. Firebase Hosting provides reliable and scalable hosting solutions, ensuring the efficient delivery of content to users. The integration of Firebase into the system requirements enables effortless implementation of features like real-time updates, user authentication, and serverless computing, contributing to the creation of a dynamic, secure, and scalable online platform. The versatility and integration capabilities of Firebase make it a valuable asset in crafting a feature-rich and efficient user experience for the platform's visitors.

CHAPTER 4

SYSTEM DESIGN

4.1 MODULE DESCRIPTION

4.1.1 Home Module

The Home Module serves as the entry point and foundational interface of the online platform. This module encapsulates the primary components and functionalities that users encounter upon accessing the website. The key features of the Home Module include an interactive and visually appealing landing page that provides an overview of the paint store's offerings. It incorporates a navigation bar to seamlessly guide users to different sections of the platform, such as Products, About Us, and Account. The module also features dynamic content areas, showcasing promotional information, featured products, and any ongoing sales or events. Additionally, its responsive design ensures an optimal viewing experience across various devices, contributing to a user-friendly and engaging introduction to the online platform.

4.1.2 Product Module

The Product Module forms the core of the online platform, offering an extensive and organized display of the paint store's product inventory. This module encompasses a dynamic and visually appealing product catalog, categorized for easy navigation. Each product listing includes detailed information such as product images, descriptions, prices. The Product Module also integrates an intuitive shopping cart system, allowing users to add desired items for seamless online transactions. The responsive design ensures aconsistent and enjoyable browsing experience across devices, reinforcing the online platform's commitment to providing a comprehensive and user-centric product showcase.

4.1.3 Authentication Module

The Authentication Module serves as the secure gateway for user interaction with the online platform, encompassing essential functionalities to manage user identities and access. This module includes a seamless user registration process, allowing individuals to create accounts by providing necessary information. The login mechanism ensures secure access to user accounts. Users can manage their accounts through account settings, enabling profile customization and password recovery options for a user-friendly experience.

4.1.4 Checkout Module

The Checkout Module serves as the concluding phase in the user's online shopping journey, streamlining the process from product selection to order completion. It encompasses a user-friendly interface for reviewing and confirming selected items, entering shipping details, and selecting secure payment options through integrated paymentgateways. The module employs address validation and error-checking mechanisms toenhance the accuracy of user inputs and generates detailed order summaries for final confirmation. With a responsive design, the Checkout Module ensures a seamless
experience across various devices, aiming to provide users with a secure, efficient, and satisfying conclusion to their transactions on the online platform.

4.1.5 Admin Module

The Admin Module acts as the command centre for platform administrators, consolidating essential tools for streamlined management of the online store. Administrators can effortlessly oversee product-related activities, manage orders, and monitor inventory. User management features enable efficient handling of customer accounts, while analytics and reporting tools offer valuable insights for data-driven decision-making. With a focus on security, the module ensures that sensitive information remains accessible only to authorized personal, contributing to the creation of a well- organized, responsive, and secure online store environment.

4.2 USE CASE DIAGRAM

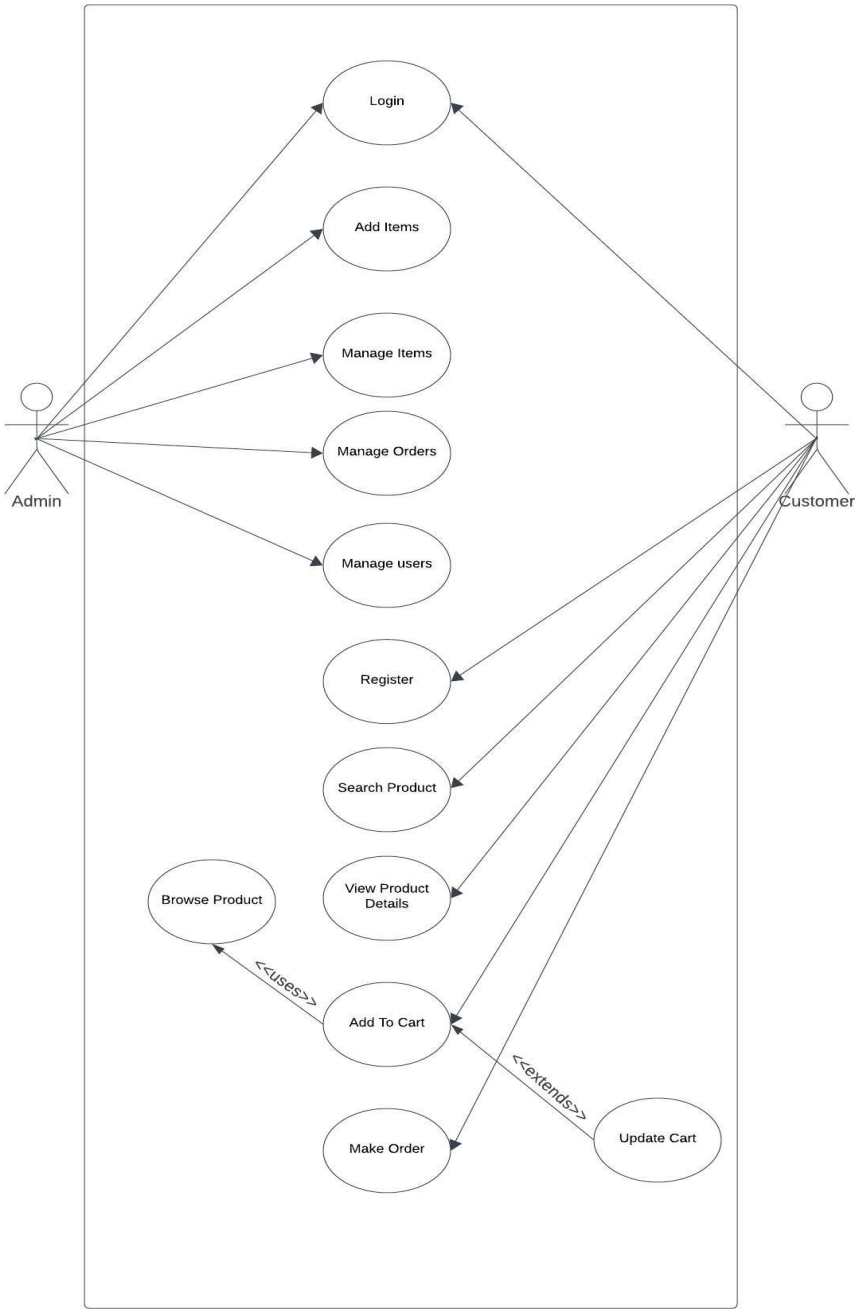

Figure 4.1 Use Case Diagram

4.3 SYSTEM FLOW DIAGRAM

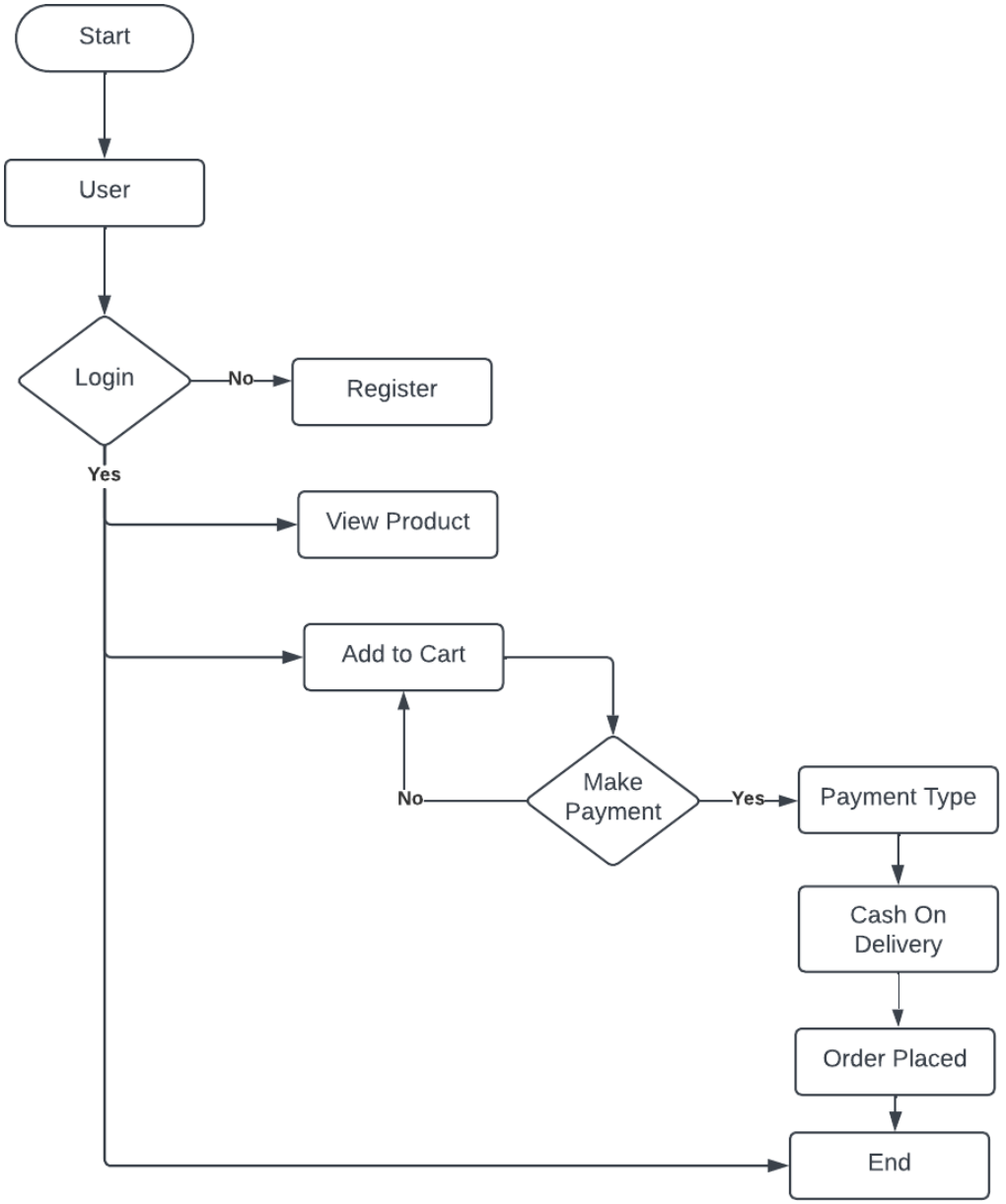

Figure 4.2 User System Flow Diagram

4.4 ADMIN FLOW DIAGRAM

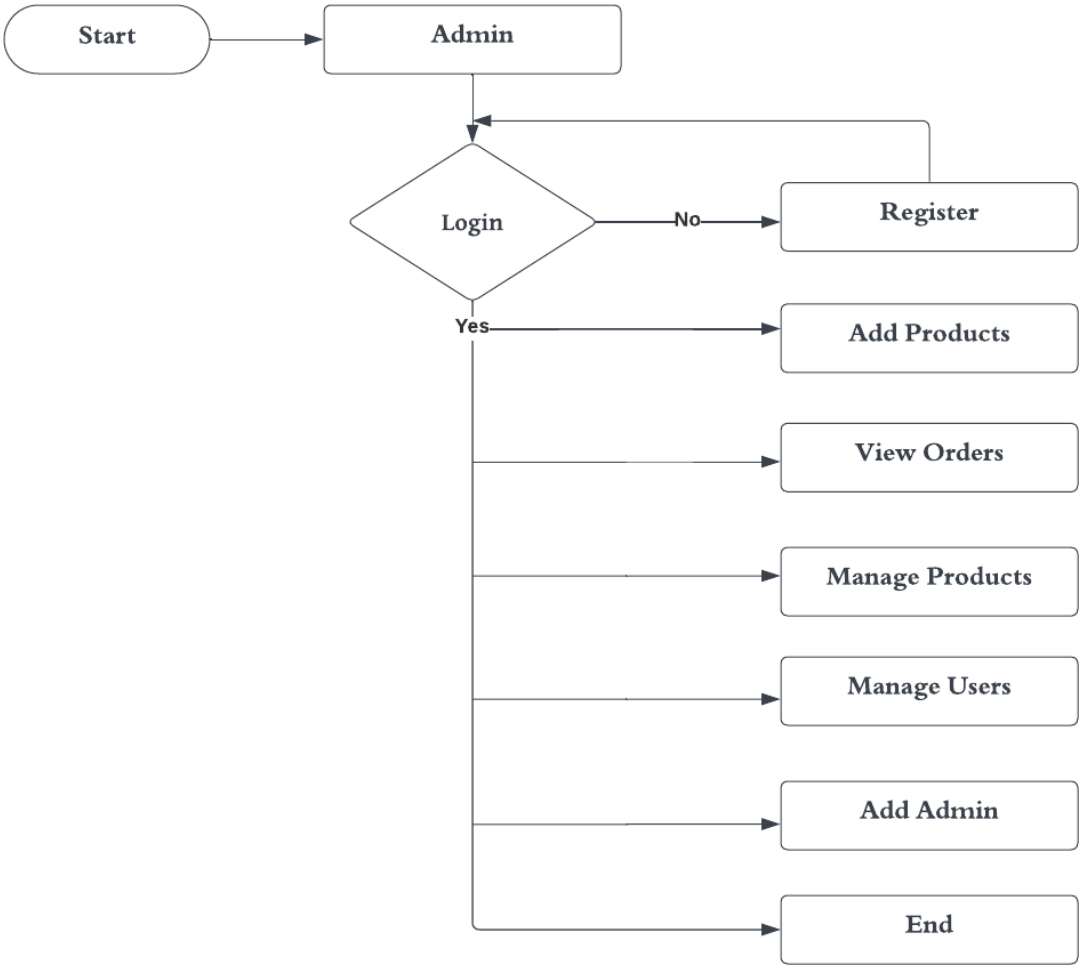

Figure 4.3 Admin System Flow Diagram

4.5 DATA FLOW DIAGRAM

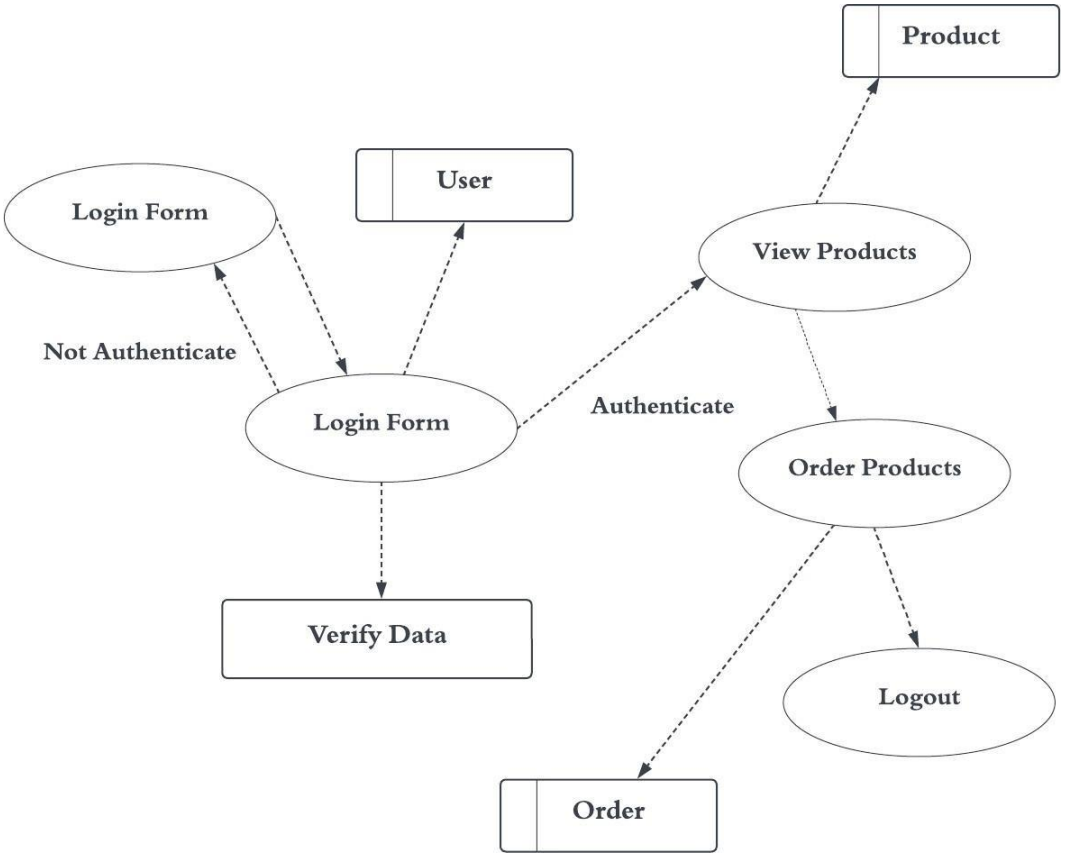

Figure 4.4 DFD for User Activities

4.6 E-R DIAGRAM

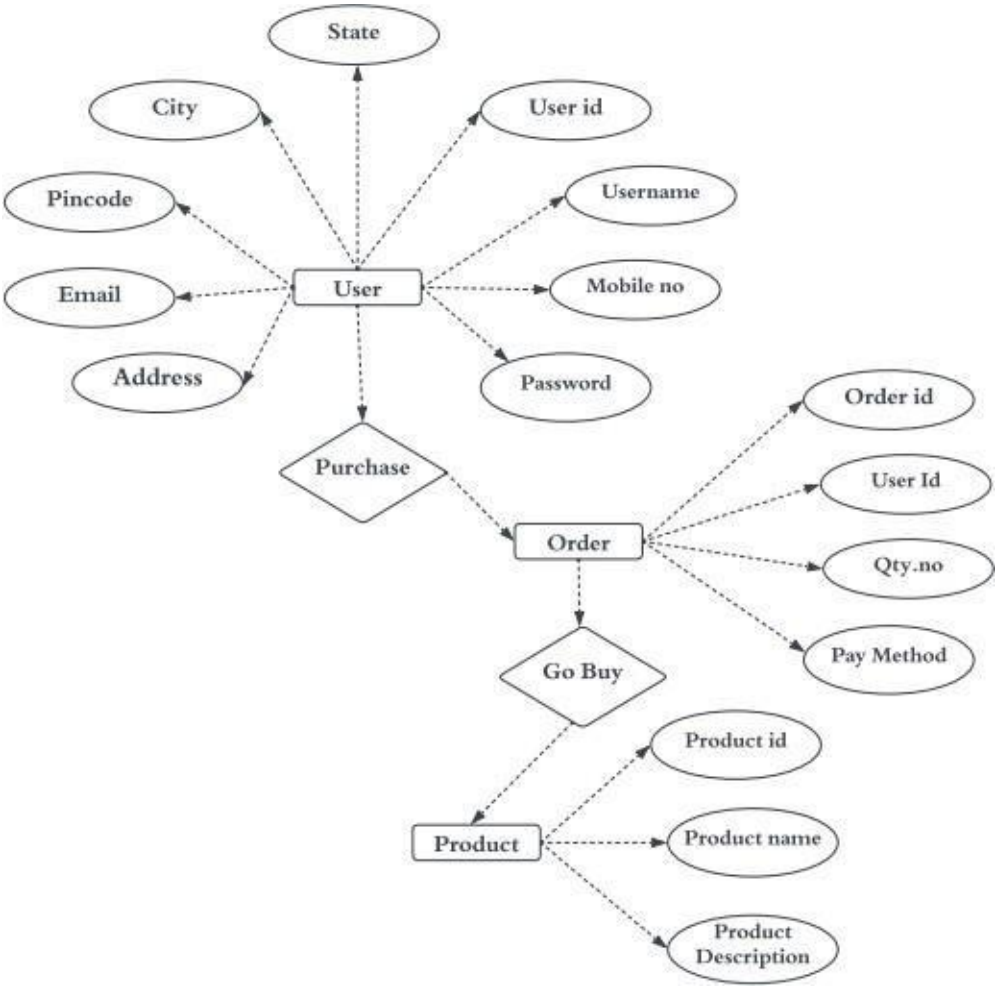

Figure 4.5 E-R Diagram

4.7 DATABASE DESIGN

A database serves as an organized repository capable of efficiently storing and retrieving information. In the context of Firebase Realtime Database, data is stored as JSON objects, forming a cloud-hosted JSON tree. Unlike traditional SQL databases with tables and records, the Realtime Database employs a structure where added data becomes anode in the JSON tree, each with an associated key.

While the Firebase Realtime Database allows nesting data up to 32 levels deep, it is advisable to avoid excessive nesting due to the nature of data retrieval. Fetching data at a specific location in the database entails retrieving all its child nodes, and when providing read or write access at a node, access is granted to all data beneath that node. Therefore, a flat data structure is recommended for practicality and efficiency.

To address the challenges associated with nested data, a strategy involves flattening the data structure. By avoiding deep nesting and opting for a more streamlined design, data can be separated into distinct paths. This approach facilitates efficient downloading of specific data subsets as needed, overcoming the issues posed by nested structures, especially when iterating through the data.

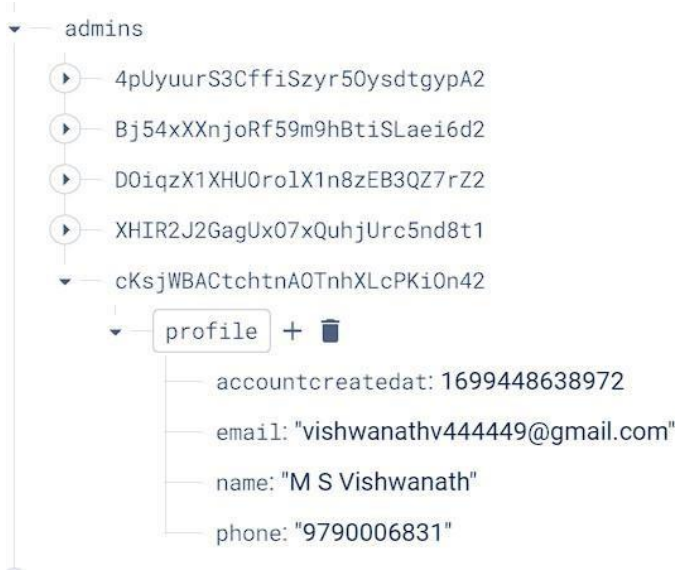

Figure 4.6 Admin Database

Figure 4.7 Order Database

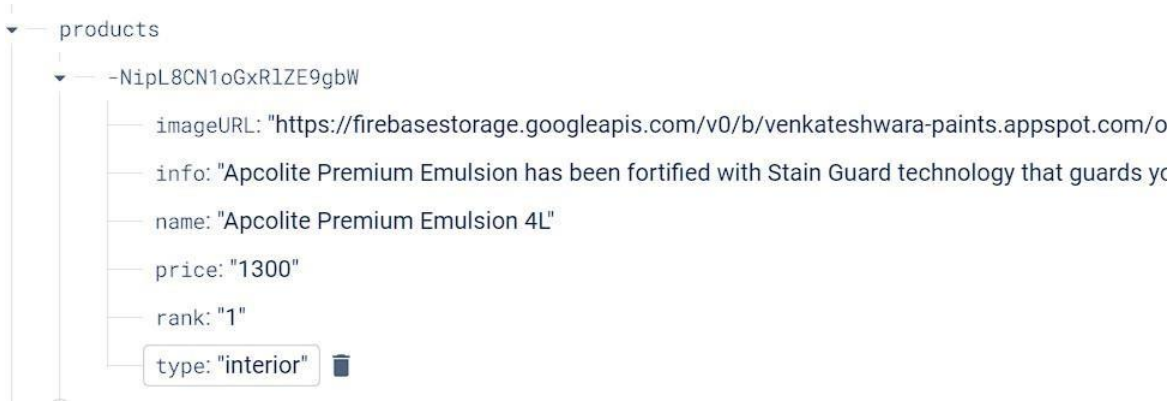

Figure 4.8 Product Database

4.8 INPUT DESIGN

Input design is the critical process of transforming user-originated inputs into a computer-readable format. It stands as one of the most resource-intensive phases in the operation of a computerized system and is often a focal point for system challenges. Many issues within a system can be traced back to faulty input design and methodology. Each aspect of input design should undergo meticulous analysis and construction. The design should facilitate the smooth passage of inputs across various networks in a reliable manner, particularly for installations in remote networks. The input database is subject to specific constraints, including the acquisition of all files from the disk by data, suitability for data clearance, and presentation in an understandable and correct format.

The system's functionality relies on taking inputs from users, processing them, and generating outputs. Input design serves as the bridge connecting the information system with its users and should prioritize user-friendliness to provide relevant information effectively. The decisions made during input design aim to minimize time consumption, simplifying application sensitivity. The project's implementation aims for low manpower attrition with reasonable output. Considering budget constraints, the system development adheres to financial limits, utilizing freely available technologies and only purchasing customized products. Input data for the system can originate not only as raw data captured from scratch but also as the output of another system or subsystem. Input design encompasses all phases, from creating initial data to entering it into the system, ensuring accuracy and correctness.

The input design goes beyond conventional practices, introducing innovative elements to elevate user interaction and system performance. Smart defaults and auto-population features optimize data entry efficiency, remembering frequently used information. Robust data validation algorithms ensure the integrity of inputs, while user-friendly error recovery mechanisms guide users through corrections seamlessly. Emphasis on data privacy includes end-to-end encryption and strict access controls, complying with legal standards. Intuitive feedback and progress indicators enhance user confidence, and cross-platform consistency guarantees a uniform experience.

4.9 OUTPUT DESIGN

Output design encompasses the results and information generated by the system, tailored for comprehension and usability by end-users. It plays a pivotal role in the evaluation of the application's usefulness, forming the basis upon which the system's effectiveness is assessed. The software's output serves as a critical input for remote installations, triggering immediate alerts to enhance system functionality. The effectiveness of output design is pivotal, directly impacting the user's experience and satisfaction with the system.

In the context of computer output, which serves as a primary source of information for users, output design particularly focuses on form design. An efficient output design aims to enhance the user interface, ensuring that the information presented isnot only clear and comprehensible but also visually appealing. The term "output" extends to any information displayed by the information system.

When analysts undertake output design, they identify specific outputs necessary to meet end-user requirements. Previewing output reports by users is of utmost importance, as users are the ultimate judges of output quality and, consequently, the success of the system. The process of designing output involves a comprehensive analysis to determine which applications, websites, or documents are allowed or blocked. Various options for allowing are explored, enhancing the adaptability of the output to user preferences.The design of the output prioritizes attractiveness, convenience, and informativeness.In essence, output design is a strategic process that goes beyond functionality, focusing on creating outputs that are not only technically sound but also user-centric and aesthetically pleasing.

CHAPTER 5

SYSTEM TESTING

Once the source code has been finalized, it undergoes documentation, including associated data structures. The next crucial phase involves testing and validation, where thorough examinations are conducted to identify and rectify errors. During this process, the developer meticulously designs and executes tests, aiming to showcase the program's functionality while actively seeking and addressing potential errors. Despite the developer's careful efforts, it is acknowledged that errors may persist, and if not detected by the developer, end-users will inevitably discover them.

The developer holds continuous responsibility for testing individual components, such as modules, within the program. Additionally, the developer often conducts integration testing, a crucial step in verifying the seamless integration of all components into the complete application structure. This comprehensive approach ensures that not only the individual modules but also the entire application functions cohesively and reliably.

5.1 UNIT TESTING

During the unit testing phase, it is imperative to scrutinize the programs constituting the system. Consequently, Unit check-out is sometimes synonymous with Program check-out. The software program units within a system comprise modules and routines that collaborate to fulfil specific functions. Unit testing involves assessing these modules individually to identify errors proactively. This approach enables the identification of both code and logic errors that might go unnoticed if the module were tested in isolation. This testing process is conducted during the programming stage to ensure the robustness and accuracy of the individual components before their integration into the broader system.

Test Case 1

Module **:** Admin Login

Test Type **:** Loading of appropriate form for administrator

Input **:** Username and Password

Expected Output **:** Display admin products

Sample Test

Output **:** Redirect to Main Page and display the admin products

Analysis **:** In this form, the username and password has been tested by correct format. If the username and password is mismatched to data in the database, here the data mismatch error can occur.

5.2 INTEGRATION TESTING

Integration testing is done to test if the individual modules work together as one single unit. In integration testing, the individual modules that are to be integrated are available for testing. Thus, the manual test data that used to test the interfaces was replaced by that which is generated automatically from the various modules. It can be used for testing how the modules would actually interact with the proposed system. The modulesare integrated and tested to reveal the problem interfaces.

Test Case 1

Module **:** Admin products

Test Type **:** Order Management and Product Management

Input **:** Navigation between Admin options Expected

Expected Output **:** Navigation between modules is completed

Sample Test

Input **:** By login page, Add Product items.

Output **:** Respective products open correctly and display the type of product.

Analysis **:** Each category of the product will be open.

5.3 VALIDATION TESTING

Verification and validation checking out are critical tests, which might be achieved earlier than the product has been surpassed over to the customer. This makes sure that the software program checking out lifestyles cycle begins off and evolves early. The intention of each verification and validation is to make certain that the product is made in step with the necessities of the customer and does certainly fulfil the supposed purpose.

Test Case 1

Module **:** Register

Test Type **:** Register new user

Input **:** Input to all fields Expected

Expected Output **:** No Required field should not be empty

Sample Test

Input **:** Input for a required field is not provided

Output **:** Provide all the required fields.

Analysis **:** The expected output is same, so the form passed the validation test

CHAPTER 6

SYSTEM IMPLEMENTATION

Upon completion of the initial system design, the customer was consulted for their approval, ensuring that further enhancements to the system could proceed in alignment with their preferences. Following the development of the system, a demonstration was provided to showcase its operation to the customer. The primary goal ofthis demonstration was to identify and rectify any potential malfunctions within the system. Once the system's operation was approved, it was integrated into the organization. Initially, the system operated in parallel with the existing manual system to facilitate a smooth transition. Rigorous testing using real-time data validated that the system operated seamlessly and was user-friendly.System implementation is the process of transforming a new or revisedsystem design into an operational one. During the initial design phase, a demonstration is conducted for end-users to illustrate how the system functions. This method is employed toverify and identify any logical issues within the system by inputting various combinations or examining data. After receiving approval from both end-users and management, the system is officially implemented. System implementation encompasses several activities, and the six key steps are as follows.

6.1 CODING

Coding is a pivotal stage where the tangible design specifications, formed through collaboration with the evaluation team, are meticulously translated into computer code by the programming team. This transformative process involves the implementation of algorithms and logic outlined in the design phase. The coding phase serves as the bridge between conceptualization and execution, with the programming team working to convert high-level design concepts into a functional and executable form. This intricate process demands precision, attention to detail, and adherence to coding standards to ensure the creation of robust and efficient software. Regular collaboration and communication between the evaluation and programming teams are essential to address any challenges, refine the code, and ensure alignment with the intended functionality of the software.

6.2 TESTING

The testing phase commences simultaneously with the coding procedure, allowing for the examination of each software module as it is developed. This parallel progression ensures an iterative and collaborative approach, where coding and testing activities occur concurrently. As each module is coded, it undergoes thorough testing to identify and rectify any potential issues early in the development process. This integrated approach facilitates real-time feedback, fostering a more efficient and streamlined development lifecycle.

6.3 INSTALLATION

Installation refers to the procedure of substituting an existing device with a new one. This involves the conversion of existing data, software, documentation, and painting techniques to formats compatible with the new device. Additionally, installation encompasses the configuration of hardware components and the integration of software necessary for the seamless operation of the new device. It is a critical phase that demands meticulous attention to detail to ensure a smooth transition, minimizing disruptions in workflow. Regular testing during the installation process is imperative to validate the compatibility of the existing data and software with the new device, and any necessary adjustments are made to optimize performance.

6.4 DOCUMENTATION

Documentation comprises individual publications that offer comprehensive information on effectively utilizing the system and navigating its workflow following the setup procedure. These materials serve as valuable resources, providing users with step-by-step instructions, guidelines, and insights into the system's functionalities. They aim to empower users with the knowledge required for seamless interaction with the system, covering aspects such as system features, troubleshooting, and best practices.

6.5 TRAINING

A training plan serves as a structured approach to rapidly educate users on the effective utilization of a new system. It is a systematic method designed to impart the necessary skills and knowledge, enabling consumers to efficiently navigate and leverage the features of the newly implemented system. The training plan outlines the curriculum, instructional methods, and resources required to facilitate a smooth learning process for end- users.

6.6 SUPPORT

The educational approach underwent a likely enhancement before the commencement of the project, signifying a proactive measure to ensure optimal preparation. This improvement might have involved refining instructional strategies, updating learning materials, and incorporating the latest pedagogical techniques. A well-prepared educational foundation is crucial for project success, fostering a conducive environment for effective learning and skill development. This proactive stance reflects a commitment to continuous improvement and aligns the educational approach with the evolving needs and advancements in the respective field. This upgraded education process sets the stage for a project that is not only well-informed but also agile and responsive to the dynamic educational landscape.

CHAPTER 7

CONCLUSION AND FUTURE DEVELOPMENT

7.1 CONCLUSION

The project titled **ON STREAM PORTAL FOR PAINT SHOP** hasbeen successfully completed. This initiative aimed to create a user-friendly website, facilitating seamless searching, viewing, and selection of paint products. Customers can easily access comprehensive specifications for each product. The system has undergone meticulous development, ensuring it is error-free, efficient, and time-effective. Security measures have been integrated throughout the entire system. The design is scalable, allowing for easy implementation in nearby or branded paint stores with minimal modifications. The system introduces a moderation feature, providing moderators with enhanced control over products within their domain. This adaptable design serves as a blueprint for similar ventures, enabling other shops to offer various products with straightforward adjustments. For online transactions, customers need internet access and a valid payment method.

7.2 FUTURE ENHANCEMENTS

This chapter includes the Conclusion reached after creating the current version of the website to meet the system objectives.

- Explore the incorporation of augmented reality technology, allowing customers to virtually "try out" paint colours in their real-world spaces through their mobile devices. This interactive feature can significantly impact the purchasing decision.
- Diversify the range of offerings beyond paints to include related products like brushes, rollers, and other painting accessories.
- The system can be extended to allow the users to create accounts and save products into a wish list.

APPENDIX 1

SAMPLE CODING

Engine.jsx

```
import React, { useEffect, useState } from 'react'
import { onAuthStateChanged } from 'firebase/auth';
import { auth, db } from './firebase';
import { get, onValue, ref } from 'firebase/database';
import Navbar from './components/Navbar'

import Footer from './components/Footer'
import Home from  './pages/Home'
import About from './pages/About'
import Contact from './pages/Contact'
import Account from './pages/Account'
import Login from './pages/auth/Login'
import Register from './pages/auth/Register'
import Cart from './pages/Cart'
import Products from './pages/Products'
import ForgotPassword from './pages/auth/ForgotPassword'
const Engine = () => {
 useEffect(() => {
      window.scrollTo(0, 0); // Scroll to the top of the page
   })
   const [loggedindetails, setloggedindetails] = useState(null);
   const [loggedinuid, setloggedinuid] = useState(null);
```

```
useEffect(() => {
    const unsubscribe = onAuthStateChanged(auth, (user) => {
        if (user) {
            const uid = user.uid;
            setloggedinuid(uid)
            // Reference to the user's data within the Realtime Database
            const userRef = ref(db, 'users/' + uid);// Check if the user
            data exists in the Realtime Database get(userRef)
                .then((snapshot) => {
                    if (snapshot.exists()) {
                        setloggedinuid(uid);
                        const userData = snapshot.val();
                        console.log()
                        setloggedindetails(userData);
                        setcomponent("Home");
                    } else {
                        setloggedindetails(null);
                        setcomponent("Home");
                    }
                })
                .catch((error) => {
                    console.error('Error fetching Realtime Database data:', error);
                });
        } else {
            setloggedindetails(null);
            setcomponent("Home");
```

```
            }
        });
        return () => {
            unsubscribe();};

    }, []);
    const [component, setcomponent] = useState("Home")
    const render = () => {
        switch (component) {
            case "Home":
                return <Home fitcomponent={fitcomponent} />;
            case "About":
                return <About />;
            case "Contact":
                return <Contact />;
            case "Products":
                return <Products productref={currentproduct}
 setcurrentproduct={setcurrentproduct} />;
            case  "Account":
                return <Account />;
            case "Login":
                return <Login componentrender={componentrender} />;
            case "Register":
                return <Register componentrender={componentrender} />;
            case "Cart":
                return <Cart componentrender={componentrender}
userData={loggedindetails}/>;
```

```
            case "ForgotPassword":return <ForgotPassword
            componentrender={componentrender} />;

        }

    };

    const [currentproduct, setcurrentproduct] = useState("Interior")

    const fitcomponent = (value1, value2) => {

        setcomponent(value1)

        setcurrentproduct(value2)

    }

    const componentrender = (componentName) => {

        setcomponent(componentName);

    }

    return (

        <div className='d-flex flex-column vh-100'>

<Navbar componentrender={componentrender} component={component}
userData={loggedindetails} />

            <div style={{ paddingTop: "100px" }}>

                {render()}

            </div>

            <Footer />

        </div>

    )

}

export default Engine
```

Home.jsx

```
import React from 'react'
import ExteriorFrontpage from './Products/ExteriorFrontPage';
import WoodpaintFrontPage from './Products/WoodpaintFrontPage';
import InteriorFrontPage from './Products/InteriorFrontPage';
const Home = ({ fitcomponent }) => {
  const GoToExterior = () => {
  fitcomponent("Products", "Exterior");
  };
  const GoToInterior = () => {
    fitcomponent("Products", "Interior");
  };
  const GoToWoods = () => {
    fitcomponent("Products", "WoodPaints");
  };
  return (
    <div>
      <>
        <section id="home">
          <div class="container">
            <h2> <span>Welcome to Sri Venkateshwara Paints</span> <br /> Your Online
Paint Store</h2>
            <p>Get your favorite paint delivered to your doorstep.</p>
            <button onClick={GoToInterior}>Explore Now</button>
```

```
        </div>
      </section>
      <section id="new" className="w-100 container-fluid mt-4">
        <div className="row">
        <div className="one col-lg-4 col-md-12 col-12">
          <div className="image-container position-relative">
            <img className="img-fluid same-height" src="Images/interior.png" alt="" />
            <div className="centered">
              <h2>Interior Paints</h2>
              <button className="btn btn-dark" onClick={GoToInterior}>Buy
Now</button>
            </div>
          </div>
        </div>
        <div className="one col-lg-4 col-md-12 col-12">
          <div className="image-container position-relative">
            <img className="img-fluid same-height" src="Images/exterior.png" alt="" />
            <div className="centered">
              <h2>Exterior Paints</h2>
              <button className="btn btn-dark" onClick={GoToExterior}>Buy
Now</button>
            </div>
          </div>
        </div>
```

```
          <div className="one col-lg-4 col-md-12 col-12">
            <div className="image-container position-relative">
              <img className="img-fluid same-height" src="Images/Wood.png" alt="" />
              <div className="centered">
                <h2>Wood Paints</h2>
                <button className="btn btn-dark" onClick={GoToWoods}>Buy
Now</button>
              </div>
            </div>
          </div>
        </div>
      </section>
      <section id="Featured" className="my-5 pb-5">
        <div className="container text-center mt-5 py-5">
          <h3>Featured Interior Paints</h3>
          <hr className="mx-auto" />
          <p>Best Selling Interior Paints</p>
        </div>
        <InteriorFrontPage />
        <div className="container text-center mt-5 py-5">
          <h3>Featured Exterior Paints</h3>
          <hr className="mx-auto" />
          <p>Best Selling Exterior Paints</p>
        </div>
```

```
        <ExteriorFrontpage />
        <div className="container text-center mt-5 py-5">
          <h3>Featured Wood Paints</h3>
          <hr className="mx-auto" />
          <p>Best Selling Wood Paints</p>
        </div>
        <WoodpaintFrontPage />
      </section>
    </>
  </div>
 )
}
export default Home
```

Product.jsx

```
import React, { useEffect, useState } from 'react'
import Interior from './Products/Interior';
import Exterior from './Products/Exterior';
import Woodpaints from './Products/Woodpaints';
const Products = ({ productref, componentrender }) => {
   const [component, setcomponent] = useState("Interior")
   const render = () => {
      switch (component) {
         case "Interior":
            return <Interior componentrender={componentrender}/>;
```

```
            case "Exterior":
                return <Exterior componentrender={componentrender}/>;
            case "WoodPaints":
                return <Woodpaints componentrender={componentrender}/>;
        }
    }
    useEffect(() => {
        // Use the productref prop to set the initial component once, when the component mounts
        if (productref === "Interior") {
            setcomponent("Interior");
        } else if (productref === "Exterior") {
            setcomponent("Exterior");
        } else if (productref === "WoodPaints") {
            setcomponent("WoodPaints");
        }
    }, [productref]);
    return (
        <div className="">
            <div className="container text-center py-2">
                <h3>Our Featured Products</h3>
                <hr className="mx-auto" />
                <p>Here you can check our new products with fair price</p>
            </div>
```

```
        <div className='d-flex flex-row justify-content-center container '>
            <div className='d-flex flex-row justify-content-center border rounded pt-2'>
            <p className={` p-2 fw-regular mx-2 ${component === "Interior" ? "text-coral" : ""}`} onClick={() => setcomponent("Interior")}>Interior Paints</p> <p className={` p-2 fw-regular mx-2 ${component === "Exterior" ? "text-coral" : ""}`} onClick={() => setcomponent("Exterior")}>Exterior Paints</p>       <p className={` p-2 fw-regular mx-2 ${component === "WoodPaints" ? "text-coral" : ""}`} onClick={() => setcomponent("WoodPaints")}>Wood Paints</p>
            </div> </div>
        <div className='container mt-3'>
            {render()}
        </div></div>
    )}
```

APPENDIX-2

SCREENSHOTS

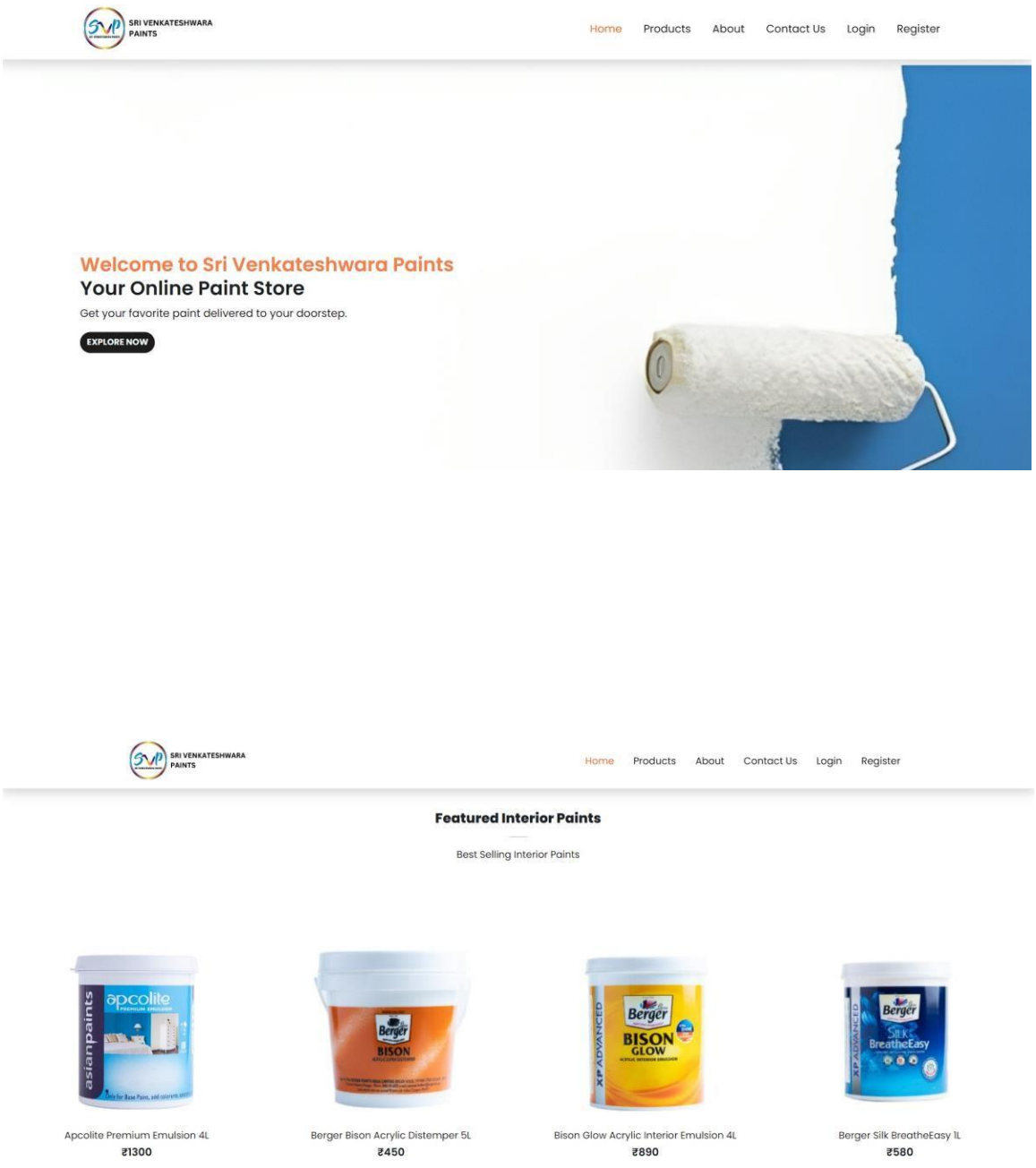

Figure A 2.1 Home Page

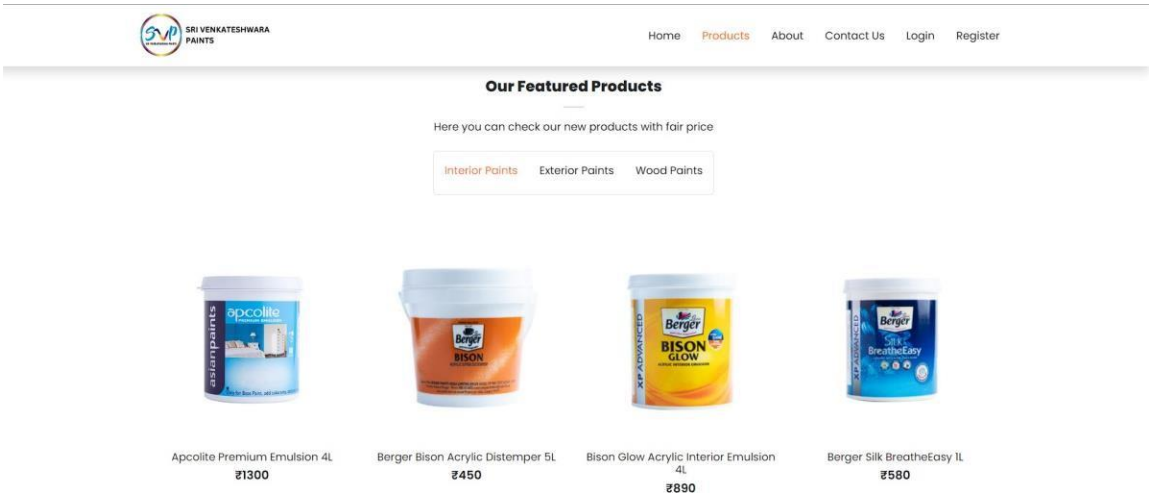

Figure A 2.2 Interior Products

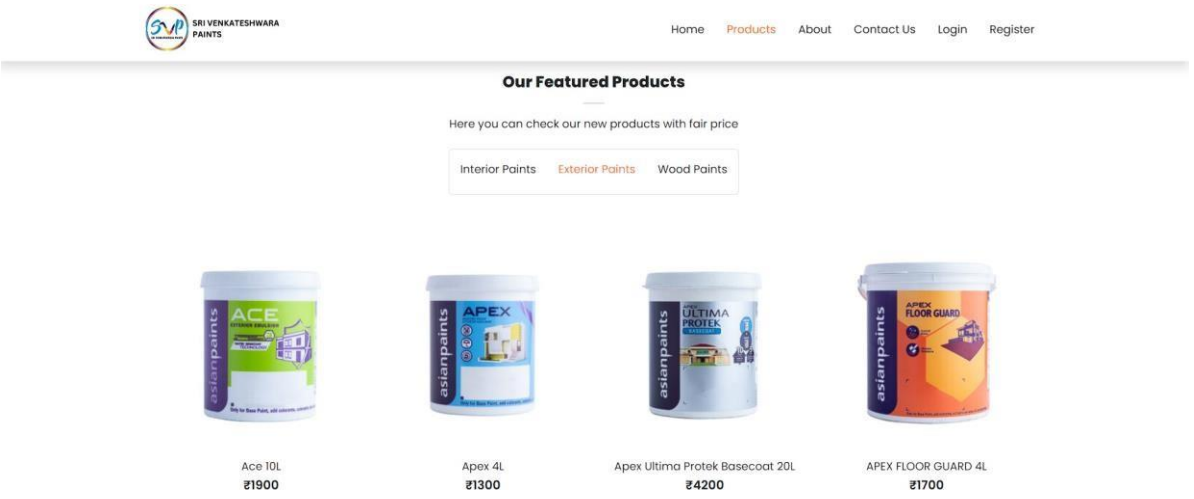

Figure A 2.3 Exterior Products

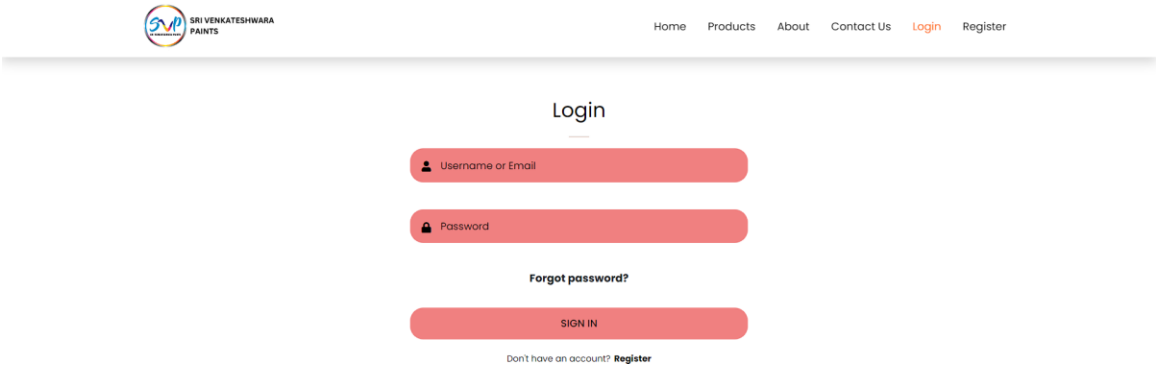

Figure A 2.4 User Login Form

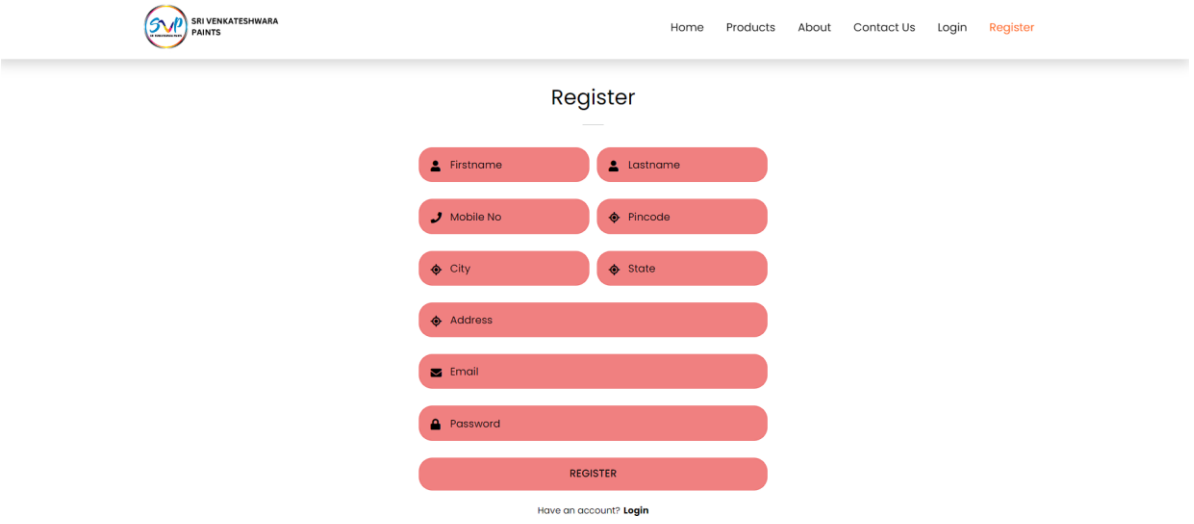

Figure A 2.5 User Registration Form

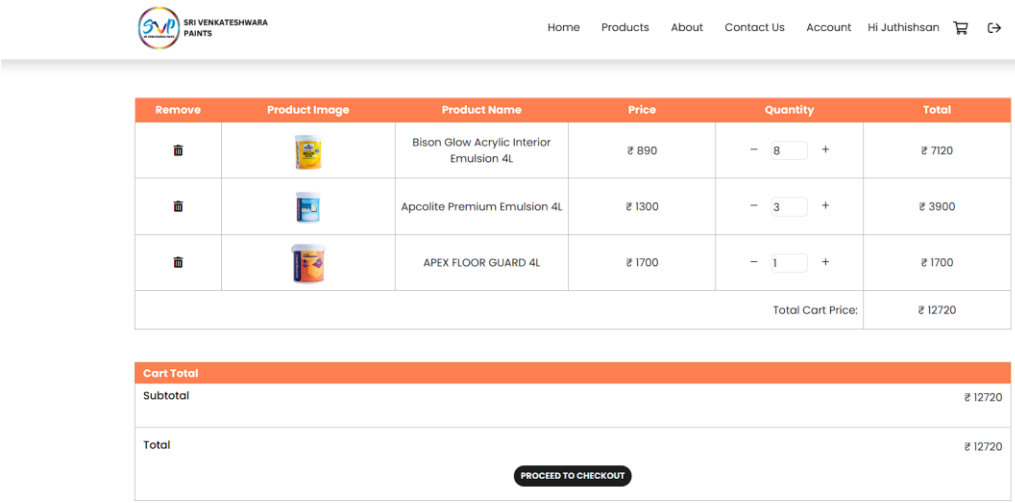

Figure A 2.6 Cart Page

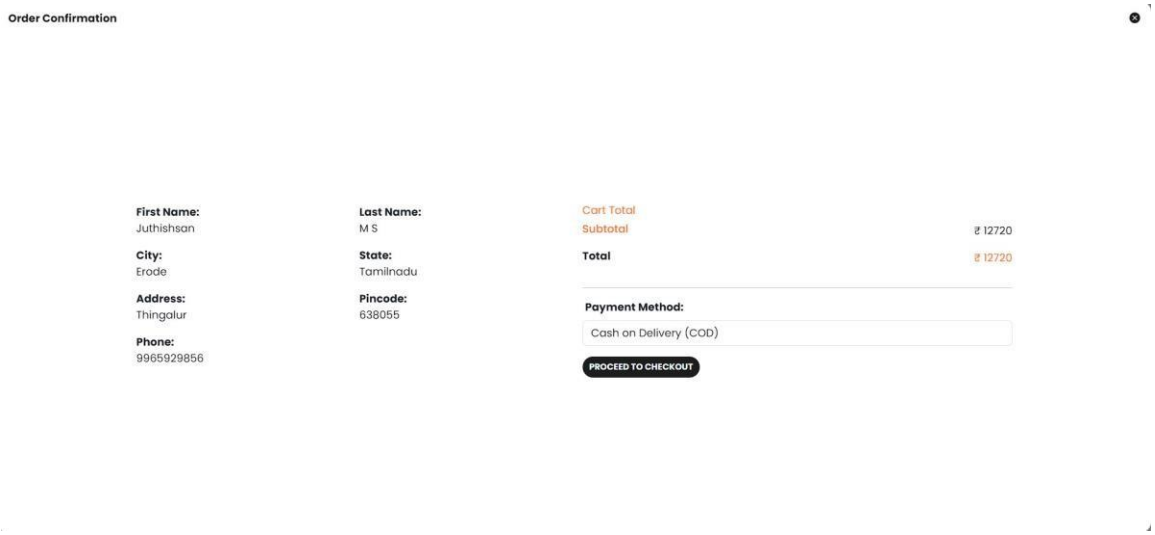

Figure A 2.7 Order Confirmation

Figure A 2.8 Admin Login Page

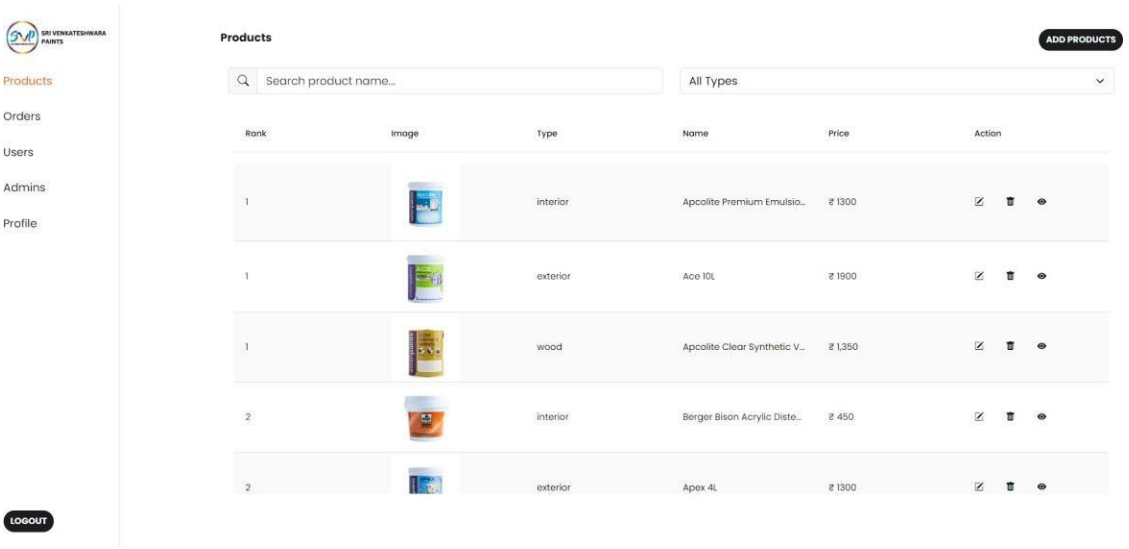

Figure A 2.9 Admin Page

REFERENCES

BOOK REFEERENCES

[1] Alex Banks & Eve Porcello (2020), “Learning React” 2nd Edition, O'Reilly Media.

[2] David Herron (2020), “Node.js Web Development” 5th Edition, O'Reilly Media.

[3] Jon Duckett (2011), “HTML and CSS: Design and Build Websites” 1st Edition, Wiley.

[4] Mark Otto and Jacob Thornton (2016), “Bootstrap in Practice”, Manning Publications.

[5] M. Murphy (2019), “Firebase: Up and Running” 2nd Edition, O'Reilly Media.

WEBSITE REFERENCES

[1] www.javapoint.com , “Learn HTML, CSS and React.js”.

[2] www.w3schools.com and www.getbootstrap.com , “Learn Java Script and Bootstrap.