A
PROJECT REPORT ON

Grocery Management System

SUBMITTED IN
PARTIAL FULFILLMENT OF

DIPLOMA IN ADVANCED COMPUTING (PG-DAC)

BY

Geeta Patil
Yash Zope

UNDER THE GUIDENCE OF

Aditya Sabale

AT

SUNBEAM INSTITUTE OF INFORMATION TECHNOLOGY, PUNE

SUNBEAM INSTITUTE OF INFORMATION TECHNOLOGY, PUNE.

CERTIFICATE

This is to certify that the project

Grocery Management System

Has been submitted by

Geeta Patil
Yash Zope

In partial fulfillment of the requirement for the Course of **PG Diploma in Advanced Computing (PG-DAC AUG2024)** as prescribed by The **CDAC** SIIT, PUNE.

Place: Pune

Date: 11-Feb-2025

Aditya Sable
Project Guide

ACKNOWLEDGEMENT

A project usually falls short of its expectation unless aided and guided by the right persons at the right time. We avail this opportunity to express our deep sense of gratitude towards Mr. Nitin Kudale (Center Coordinator, SIIT, Pune) and Mr. Yogesh Kolhe (Course Coordinator, SIIT ,Pune) .We are deeply indebted and grateful to them for their guidance, encouragement and deep concern for our project. Without their critical evaluation and suggestions at every stage of the project, this project could never have reached its present form. Last but not the least we thank the entire faculty and the staff members of Sunbeam Institute of Information Technology, Pune for their support.

Geeta Patil

Yash Zope

SIIT, Pune

ABSTRACT

The **Grocery Management System (GMS)** is an advanced software solution designed to manage the core functions of grocery store operations, with a focus on improving efficiency in inventory management, product categorization, order processing, and user interaction. This system automates key aspects of grocery store operations to reduce human error, enhance decision-making, and ensure smooth customer experiences.

The system is built around several key features that allow for comprehensive management of a grocery store. The **Cart** feature enables users to easily add, view, and modify items they intend to purchase, providing a seamless shopping experience. The **Category** feature organizes products into specific groups, making it easier for both staff and customers to navigate and manage the store's inventory. Products are meticulously managed and tracked, ensuring that store operators always have up-to-date information on stock levels, product details, and availability.

The **Order** functionality allows customers to complete their purchases, which are then stored in the system for future reference. The **Order_Products** feature links the ordered products to individual customer orders, simplifying order tracking and inventory updates. The **Payments** module ensures that transactions are securely processed, providing a clear overview of customer payments, payment methods, and transaction status. This enables quick reconciliation of store sales and improves cash flow management.

A user-friendly **User** management feature is integrated to handle customer accounts, allowing users to sign up, log in, and track their order history. It also supports user authentication and roles, enabling different levels of access for store employees and administrators. With the help of this feature, managers can assign specific permissions to users, enhancing security and control within the system.

The GMS is designed to optimize the flow of operations, from managing product inventories to processing payments, all while offering a simple, intuitive interface that ensures efficient interaction between the system, employees, and customers. Real-time tracking of product availability, orders, and payments reduces the likelihood of errors, prevents overstocking or understocking, and helps store managers make informed decisions based on accurate data

INDEX

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

1.1 Introduction

The **Grocery Management System (GMS)** is an advanced software solution designed to streamline and automate the core operations of a grocery store. By integrating efficient inventory management, product categorization, order processing, and user interaction, the system enhances overall store efficiency while minimizing human error. This digital solution provides a structured and organized approach to managing grocery store operations, ensuring that both customers and store administrators benefit from a seamless and hassle-free experience.

In a rapidly evolving retail environment, grocery stores must maintain accurate inventory levels, efficiently process customer orders, and ensure smooth transactions. GMS addresses these challenges by providing a user-friendly platform that simplifies these processes. The system is equipped with essential features such as a **Cart**, allowing customers to add, view, and modify items effortlessly, ensuring a smooth and personalized shopping experience. The **Category** feature ensures products are systematically grouped, making it easier for both customers and store employees to browse and manage inventory effectively.

Additionally, GMS offers a comprehensive **Order Management** system, where customers can place orders, and store administrators can track, process, and fulfill them efficiently. The **Order Products** module links each purchased item to a specific order, facilitating order tracking, reducing mismanagement, and ensuring smooth inventory updates. The **Payments** module provides secure transaction processing, supporting various payment methods while offering real-time insights into sales and revenue tracking.

Beyond sales and inventory, the **User Management** feature plays a crucial role in handling customer and employee accounts. It allows users to register, log in, and track their order history while also providing store administrators with the ability to assign roles and permissions, ensuring controlled access to sensitive store operations. This enhances security, prevents unauthorized access, and streamlines workflow management.

The GMS is designed to optimize every aspect of grocery store operations**,** from inventory control and order processing to payment handling and customer management. The system ensures real-time tracking of stock levels, prevents overstocking or understocking, and helps store managers make informed decisions based on accurate data insights. By reducing manual workload and automating routine tasks, GMS not only enhances operational efficiency but also improves the overall customer experience, making grocery shopping more convenient and hassle-free.

By implementing GMS, grocery store owners and managers can significantly improve productivity, reduce losses due to stock mismanagement, enhance security, and provide a better shopping experience for their customers. Whether it's a small grocery shop or a large supermarket chain, GMS offers a scalable and customizable solution to meet the needs of modern retail businesses.

2. PRODUCT OVERVIEW AND SUMMARY

2.1 Purpose

The **Grocery Management System (GMS)** is designed to enhance the efficiency, accuracy, and convenience of managing grocery store operations. Its primary purpose is to **automate and streamline** key functions such as inventory management, order processing, product categorization, payments, and user management. By reducing human error and improving operational control, the system ensures a **smooth and seamless shopping experience** for both customers and store administrators.

The key objectives of the GMS include:

1. **Efficient Inventory Management** – The system provides real-time tracking of stock levels, preventing issues such as overstocking or running out of essential products.
2. **Seamless Order Processing** – Customers can easily browse products, add them to their cart, and complete transactions, while store administrators can efficiently process and track orders.
3. **Enhanced Customer Experience** – With an intuitive interface, customers can navigate the store easily, manage their orders, and make secure payments without complications.
4. **Product Categorization & Organization** – The system systematically organizes products into categories, improving accessibility and management for both customers and employees.
5. **Secure Payment Handling** – The GMS integrates various payment methods, ensuring smooth and secure transactions with proper record-keeping.
6. **User & Role Management** – It allows different levels of access and permissions for store administrators, staff, and customers, improving security and operational control.
7. **Accurate Sales & Revenue Tracking** – The system helps store managers track sales, generate reports, and make informed business decisions based on real-time data.
8. **Reduction of Human Error** – Automation of inventory updates, order tracking, and transaction processing reduces manual errors and enhances operational accuracy.

2.2 Scope

The **Grocery Management System (GMS)** is designed to serve as a **comprehensive solution** for managing grocery store operations, catering to both small and large-scale retail stores. The system **integrates various functionalities** that facilitate inventory control, order processing, payment handling, and user management.

The scope of GMS encompasses the following areas:

1. Inventory Management

- Real-time stock tracking to prevent overstocking or stockouts.
- Automatic updates when products are added, sold, or restocked.
- Categorization of products based on type, brand, or department.

2. Order and Cart Management

- Customers can browse products, add them to a cart, and modify their selections before checkout.
- Store administrators can process, track, and manage customer orders efficiently.
- Order history and status tracking for both customers and store staff.

3. Product Categorization

- Organizing products into relevant categories for better accessibility.
- Ensuring easy navigation for customers and quick identification for store employees.

4. Payment Processing

- Secure transaction handling with payment methods (card, digital wallets, etc.).
- Generating invoices and receipts for purchases.

5. Security and Data Management

- Secure login and authentication for different users.
- Data encryption to protect transaction and customer information.
- Backup and recovery options to prevent data loss.

Target Users

- **Grocery Store Owners & Managers** – For inventory tracking, sales analysis, and overall store management.
- **Store Employees** – For processing orders, managing products, and handling customer inquiries.
- **Customers** – For browsing products, placing orders, and making payments.

2.3 User Classes and Characteristics

The **Grocery Management System (GMS)** is designed to support multiple types of users, each with specific roles, responsibilities, and access levels. The system ensures smooth operations by providing role-based access control, allowing users to interact only with the functionalities relevant to their tasks.

1. Customers (End Users)

Characteristics:

- General shoppers who use the system to browse products, place orders, and make payments.
- Require a simple, intuitive interface for easy navigation and shopping.
- May have varying levels of technical proficiency.

Primary Functions:

- Browse and search for products using categorized listings.
- Add and remove items from the shopping cart.
- Place orders and complete payments securely.
- Track order history and delivery status.
- Manage personal profiles, including delivery addresses and payment preferences.

Frontend Experience:

- A **React-based UI** with an interactive shopping experience.
- Responsive design for mobile and desktop usage.
- Secure authentication for login and order tracking.

Backend Role (Spring Boot):

- Handles customer authentication and session management.
- Processes orders, payments, and order tracking.
- Fetches product data from the database and returns it to the frontend via REST APIs

2. Store Administrators

Characteristics:

- Have full control over store management, inventory, sales, and employee accounts.
- Require access to comprehensive reports and analytics.
- Must be able to make business-related decisions based on sales and inventory insights.

Primary Functions:

- Manage product inventory (add, update, delete stock).

- Monitor sales trends and generate business reports.
- Oversee order fulfillment and ensure smooth transactions.
- Manage store employees by assigning roles and permissions.
- Generate invoices and maintain financial records.
- Ensure system security and compliance with business policies.

Frontend Experience:

- Admin dashboard with **React-based UI** for data visualization and store management.
- Real-time notifications for low stock, pending orders, and sales insights.

Backend Role (Spring Boot):

- Manages authentication and authorization for admin users.
- Handles CRUD operations for products, orders, and users.
- Provides sales analytics and reporting functionalities.

3. Store Employees (Cashiers, Stock Managers, Order Handlers)

Characteristics:

- Require limited access to manage store operations efficiently.
- Need quick access to inventory and order details for their tasks.
- Work with order processing, stock updates, and assisting customers.

Primary Functions:

- Process customer orders and ensure timely fulfillment.
- Manage stock updates when new shipments arrive.
- Handle transactions at checkout if integrated with a POS system.
- Assist customers with inquiries about products and orders.
- Update order statuses (e.g., confirmed, packed, shipped, delivered).

Frontend Experience:

- Simple, **React-based UI** optimized for efficiency in order processing.
- Quick product search and stock update functionalities.

Backend Role (Spring Boot):

- Provides order tracking and updates inventory status.

2.4 Design and Implementation Constraints

1. Technology Constraints

- **Frontend:** The system's frontend must be developed using **React.js** to provide a dynamic and responsive user experience.
- **Backend:** The backend must be built using **Java with Spring Boot**, ensuring robust and scalable server-side processing.
- **Database:** The system must use **MySQL or PostgreSQL** for structured data storage, ensuring ACID compliance.
- **Authentication: JWT-based authentication** should be implemented for secure user sessions.
- **Payment Gateway:** Transactions must be handled using secure third-party payment gateways such as **Stripe, PayPal, or Razorpay**.
- **API Communication:** The frontend and backend must communicate using **RESTful APIs** for efficient data exchange.

2. Performance Constraints

- The system must **handle at least 1000 concurrent users** without significant performance degradation.
- **API response time** should not exceed **500ms** for standard queries.
- The system must be optimized to **handle bulk data transactions**, especially during inventory updates and order processing.

3. Compatibility Constraints

- The system must be **compatible with modern web browsers**, including **Chrome, Firefox, Edge, and Safari**.
- It must be **mobile-responsive**, ensuring usability across **smartphones, tablets, and desktops**.
- The backend must support **deployment on cloud platforms like AWS, Azure, or Google Cloud** for scalability.

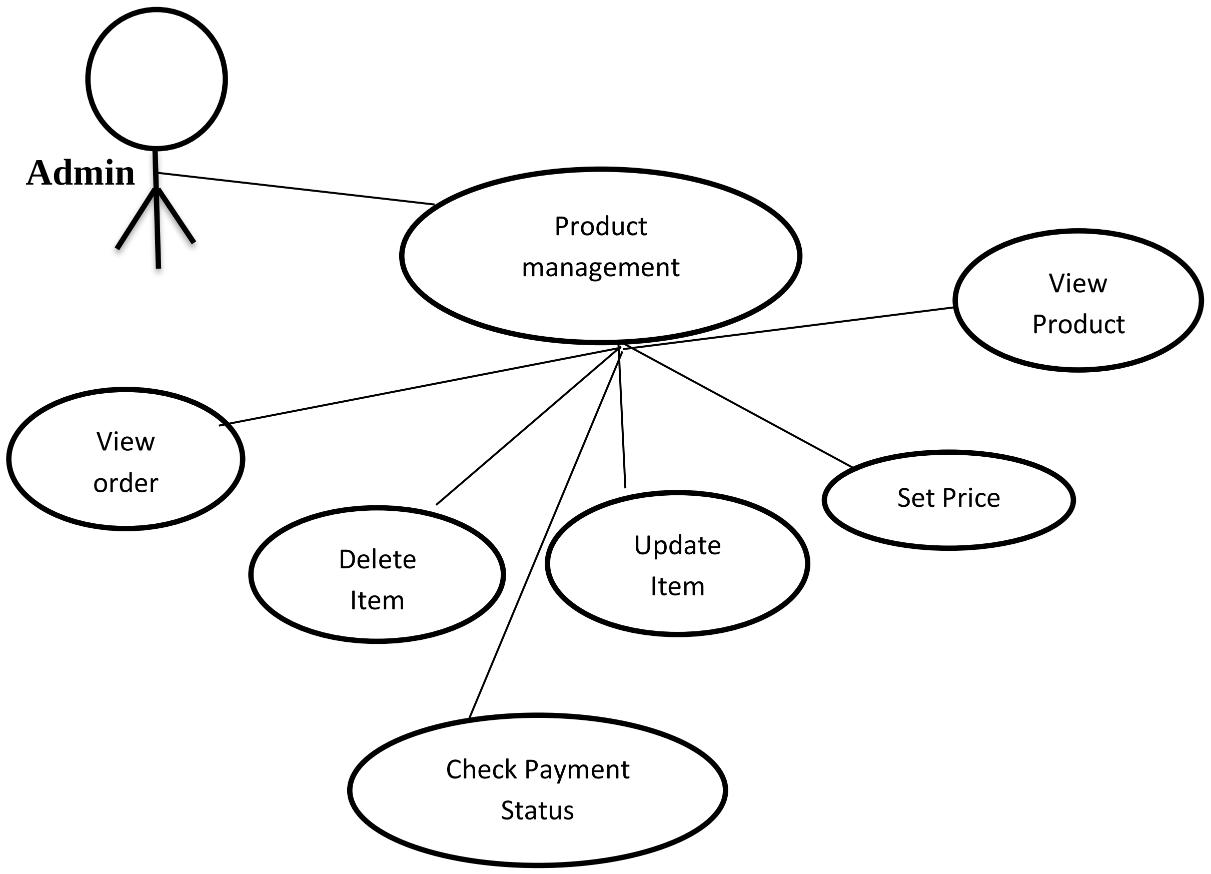
4. Hardware Constraints

- The server must have a minimum of **8GB RAM and a quad-core processor** to handle backend operations efficiently.
- The database storage should be **scalable**, allowing for future expansion as the grocery store grows.

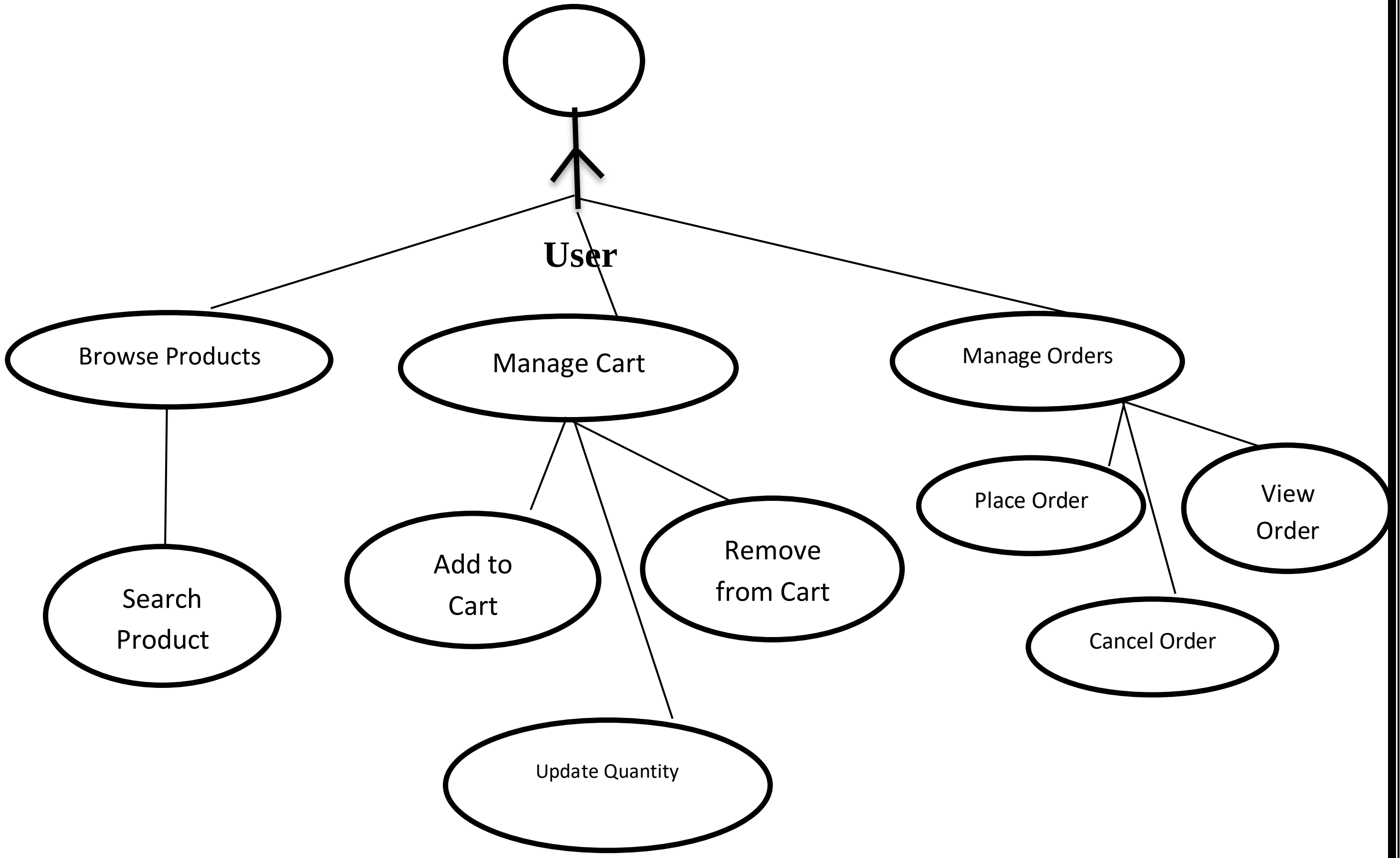
3. REQUIREMENTS

3.1 Functional Requirements

3.1.1 Use case for Administrator.



3.1.2 Use case for Customer.



3.2 Non - Functional Requirements

3.2.1 Usability Requirements

1. **Intuitive User Interface**: The system should feature a clear and straightforward interface that allows users to navigate and perform tasks with minimal training. This includes logical menu structures, easily identifiable icons, and consistent design elements.
2. **Responsive Design**: The system must be accessible across various devices, including desktops, tablets, and smartphones, ensuring a seamless experience regardless of the platform used.
3. **Efficient Task Flow**: Common tasks, such as adding items to the inventory, processing sales, and generating reports, should be streamlined to minimize the number of steps required. This reduces user effort and enhances productivity.
4. **Error Prevention and Recovery**: The system should include features that prevent common user errors and provide clear, actionable feedback when errors occur. This includes undo options and informative error messages.
5. **Consistent Terminology and Design**: The system should use consistent language and design patterns throughout, reducing cognitive load and making it easier for users to understand and operate the system.
6. **Accessibility Features**: The system should be designed to accommodate users with disabilities, including support for screen readers, keyboard navigation, and customizable display settings.
7. **Help and Support**: Integrated help resources, such as tooltips, user guides, and customer support options, should be readily available to assist users in resolving issues and learning system functionalities.
8. **Performance and Reliability**: The system should operate smoothly without significant delays or crashes, ensuring a reliable experience for users.

3.2.2 Performance Requirements

1. **System Response Time**: The system should process user inputs and display results within a specified time frame, typically within 2 seconds for most operations, to ensure a smooth user experience.
2. **Transaction Processing Speed**: The system must handle a high volume of transactions per minute, especially during peak hours, to maintain operational efficiency.

3. **Data Integrity and Accuracy**: The system should ensure that all data entered, processed, and stored is accurate and consistent, with mechanisms in place to detect and correct errors.
4. **Scalability**: The system must be capable of scaling to accommodate increased data volume, user load, and transaction frequency without degradation in performance.
5. **Availability and Uptime**: The system should be available 99.9% of the time, with minimal downtime for maintenance, to ensure continuous business operations.
6. **Backup and Recovery**: The system must have automated backup processes and a disaster recovery plan to restore data and functionality in case of system failures.
7. **Security Performance**: The system should implement security measures that do not significantly impact performance, ensuring both data protection and efficient operation.
8. **Load Handling**: The system must efficiently manage peak loads, such as high customer traffic during holidays, without performance degradation.

3.2.3 Reliability Requirements

1. **System Availability**: The system should be available 99.9% of the time, ensuring minimal downtime and continuous operation.
2. **Data Integrity**: The system must ensure that all data entered, processed, and stored is accurate and consistent, with mechanisms in place to detect and correct errors.
3. **Error Handling**: The system should gracefully handle unexpected errors, providing clear error messages and logging for troubleshooting without disrupting user operations.
4. **Backup and Recovery**: The system must have automated backup processes and a disaster recovery plan to restore data and functionality in case of system failures.
5. **Fault Tolerance**: The system should be designed to continue operating correctly even in the event of hardware or software failures, ensuring uninterrupted service.
6. **Maintenance Support**: The system should support regular maintenance activities, such as updates and patches, without significant disruption to operations.
7. **Audit Trails**: The system should maintain comprehensive logs of user activities and system events to support accountability and facilitate troubleshooting.

3.2.4 Portability Requirements

1. **Cross-Platform Compatibility**: The system should be designed to function seamlessly across different operating systems (e.g., Windows, macOS, Linux) and devices (e.g., desktops, tablets, smartphones) without requiring significant changes to the codebase.
2. **Web-Based Access**: Implementing a web-based interface allows users to access the system through standard web browsers, reducing the need for platform-specific installations and ensuring broader accessibility.
3. **Modular Architecture**: Developing the system with a modular architecture facilitates easier adaptation to different environments and simplifies maintenance and updates.
4. **Standardized Technologies**: Utilizing widely adopted programming languages, frameworks, and databases enhances the system's portability and ensures compatibility with various platforms.
5. **Cloud Integration**: Integrating with cloud services can provide scalability and flexibility, allowing the system to operate efficiently across different environments and platforms.
6. **Minimal External Dependencies**: Reducing reliance on platform-specific libraries or tools minimizes compatibility issues and enhances the system's portability.

3.2.5 Security Techniques

1. **Data Encryption**: Encrypt sensitive data both at rest and in transit to prevent unauthorized access. Utilize strong encryption algorithms and manage encryption keys securely.
2. **Access Control**: Implement role-based access control (RBAC) to ensure that users have access only to the data and functionalities necessary for their roles. Regularly review and update access permissions.
3. **Authentication and Authorization**: Use multi-factor authentication (MFA) to verify user identities and strengthen access security. Ensure that authorization mechanisms are in place to grant appropriate permissions based on user roles.
4. **Regular Security Audits**: Conduct periodic security audits to identify vulnerabilities and ensure compliance with security policies. Address any findings promptly to maintain a secure environment.
5. **Intrusion Detection and Prevention Systems (IDPS)**: Deploy IDPS to monitor network traffic for suspicious activities and potential threats, enabling timely responses to security incidents.

6. **Data Backup and Recovery**: Implement regular data backup procedures and establish a disaster recovery plan to ensure data integrity and availability in case of system failures or cyberattacks.
7. **Employee Training**: Educate employees on security best practices, phishing attacks, and data protection policies to reduce human errors that could lead to security breaches.
8. **Secure Software Development Lifecycle (SDLC)**: Integrate security into every phase of the software development process, from design to deployment, to identify and mitigate vulnerabilities early.

4. PROJECT DESIGN

4.1 Data Model

4.1.1 Database Design

Field	Type	Null	Key	Default	Extra
Cart_id	bigint	NO	PRI	NULL	auto_increment
Quantity	int	YES		NULL	
Product_id	bigint	YES	MUL	NULL	
User_id	bigint	NO	MUL	NULL	

4.1.1.1 Cart

Field	Type	Null	Key	Default	Extra
id	bigint	NO	PRI	NULL	auto_increment
image	longblob	YES		NULL	
name	varchar(255)	YES		NULL	

4.1.1.2 Category

Field	Type	Null	Key	Default	Extra
order_id	bigint	NO	PRI	NULL	auto_increment
order_date	date	YES		NULL	
user_id	bigint	YES	MUL	NULL	

4.1.1.3 Order

Field	Type	Null	Key	Default	Extra
id	bigint	NO	PRI	NULL	auto_increment
quantity	int	NO	PRI	NULL	
order_id	bigint	YES		NULL	
product_id	bigint	YES	MUL	NULL	

4.1.1.4 Order_Product

Field	**Type**	**Null**	**Key**	**Default**	**Extra**
id	bigint	NO	PRI	NULL	auto_increment
amount	double	YES		NULL	
payment_mode	varchar(255)	YES		NULL	
payment_status	varchar(255)	YES		NULL	
order_id	bigint	NO	MUL	NULL	

4.1.1.5 Payment

Field	**Type**	**Null**	**Key**	**Default**	**Extra**
product_id	bigint	NO	PRI	NULL	auto_increment
price	double	NO		NULL	
product_image	longblob	YES		NULL	
product_name	varchar(255)	YES		NULL	
quantity	double	NO		NULL	
category_id	bigint	YES	MUL	NULL	

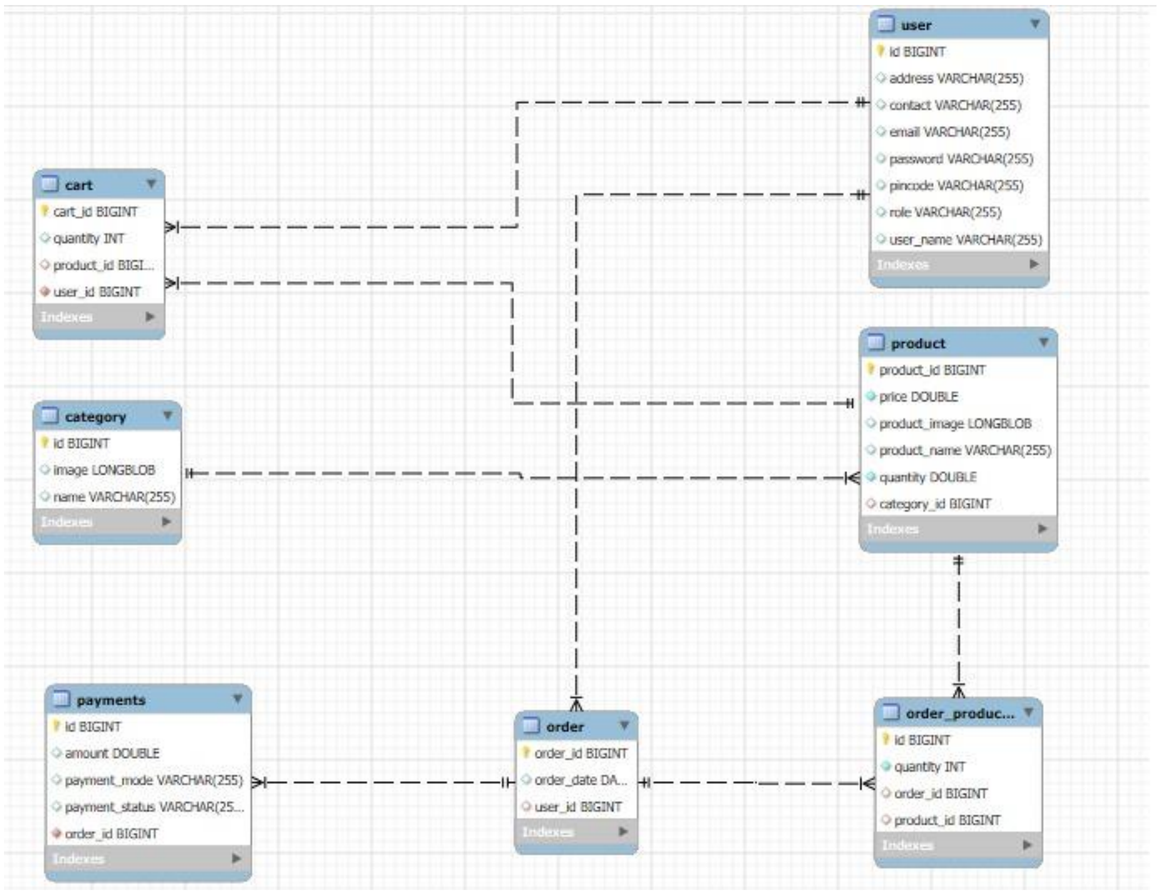
4.1.1.6 Product

Field	**Type**	**Null**	**Key**	**Default**	**Extra**
Id	bigint	NO	PRI	NULL	auto_increment
address	varchar(255)	YES		NULL	
contact	varchar(255)	YES		NULL	
email	varchar(255)	YES		NULL	
password	varchar(255)	YES		NULL	
pincode	varchar(255)	YES		NULL	
role	varchar(255)	YES		NULL	
user_Name	varchar(255)	YES		NULL	

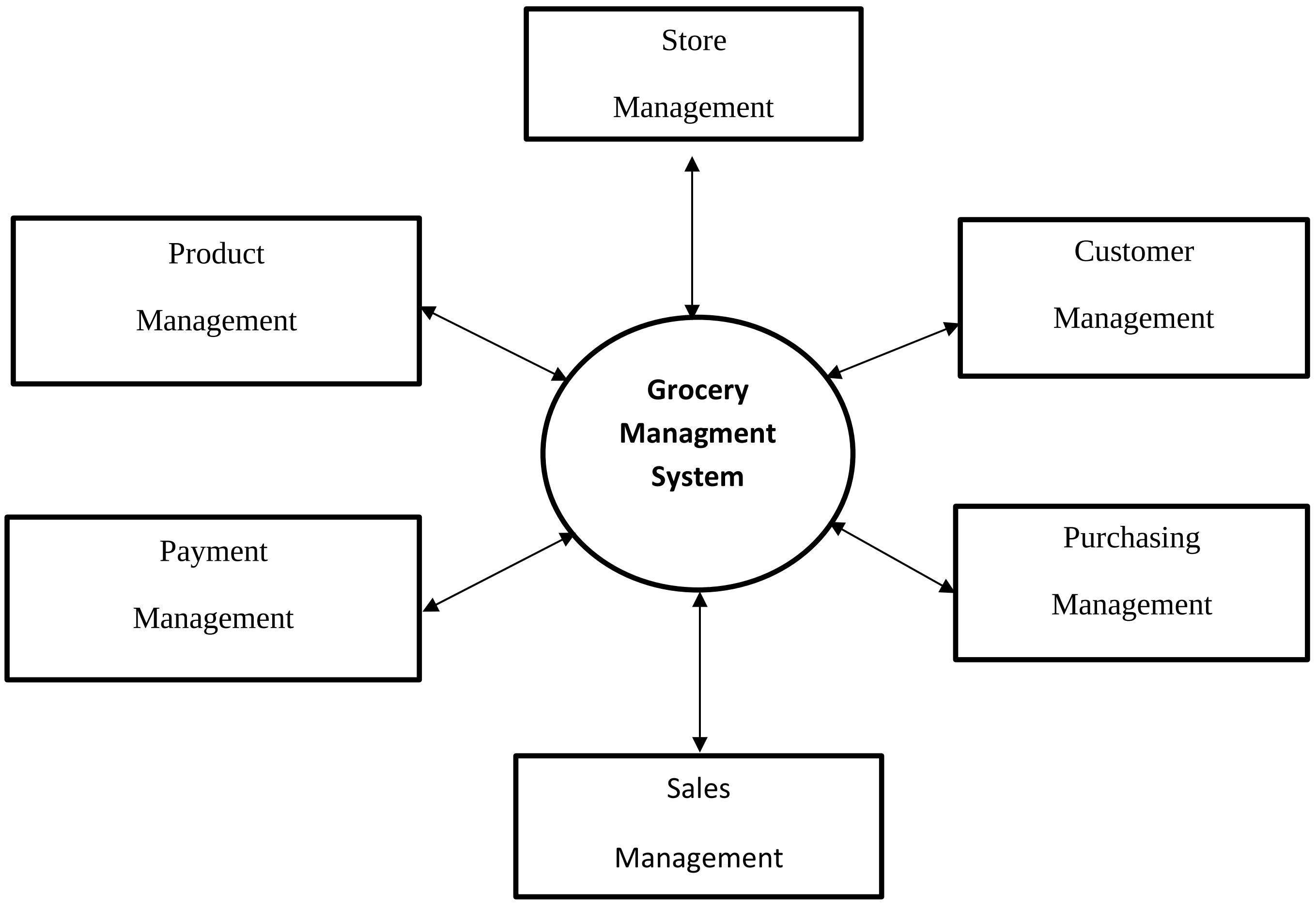
4.1.1.7 User

4.2 Process Model

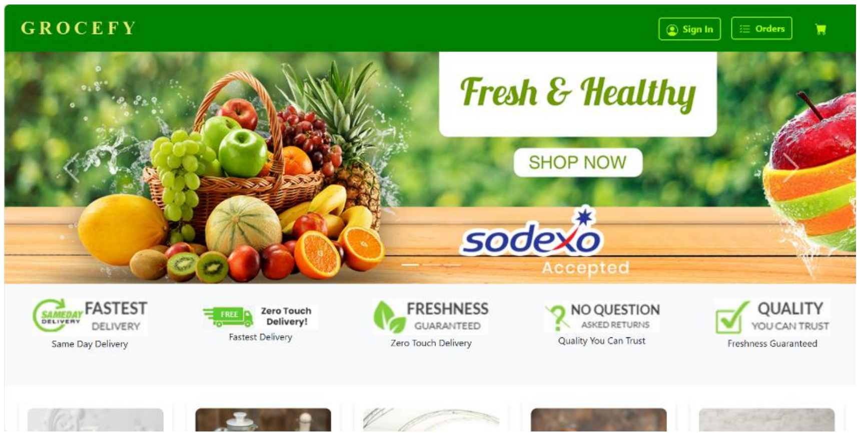
4.2.1 ER Diagram



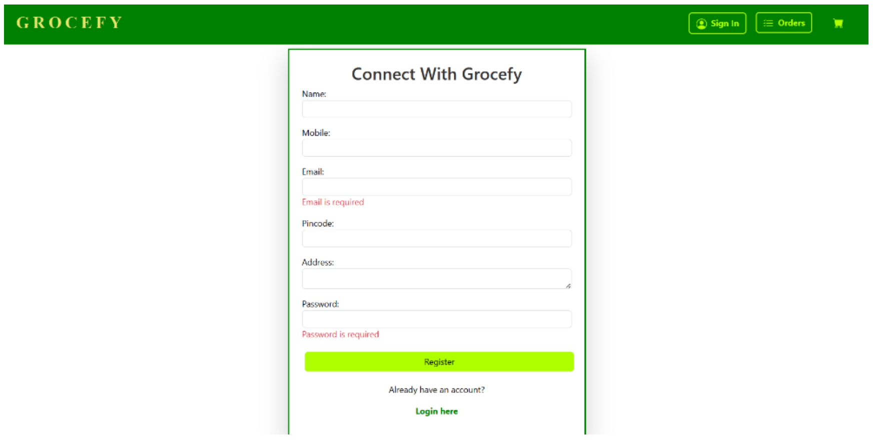
4.2.2 Data Flow Diagram



5. TEST REPORT



5.1 Home Page



Connect With Grocefy

Name:

Mobile:

Email:

Email is required

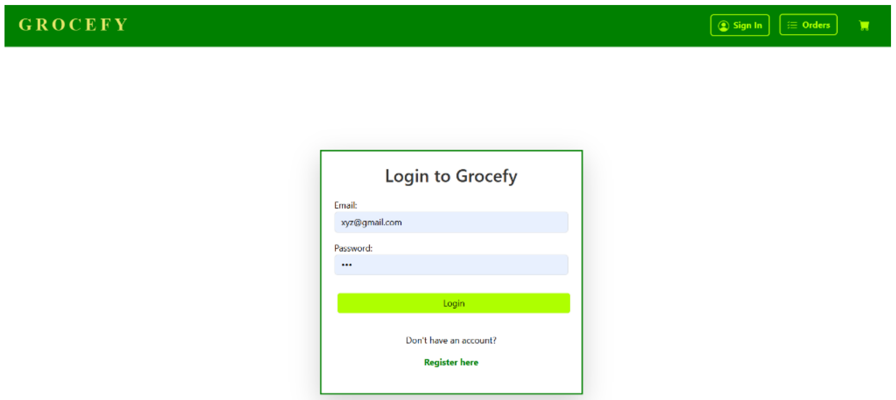
Pincode:

Address:

Password:

Password is required

5.2 Registration Page

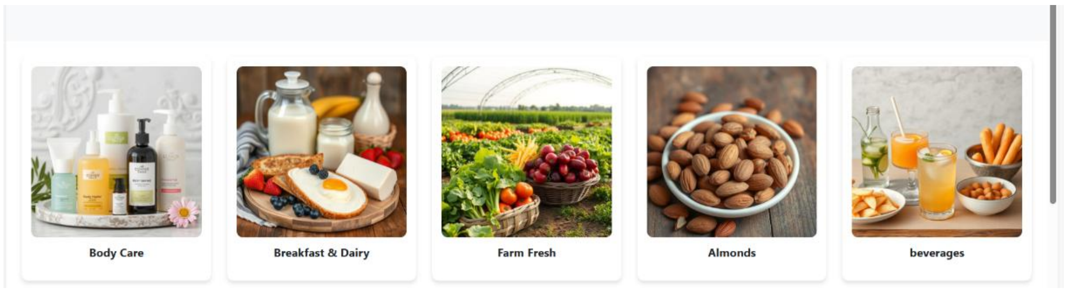


Login to Grocefy

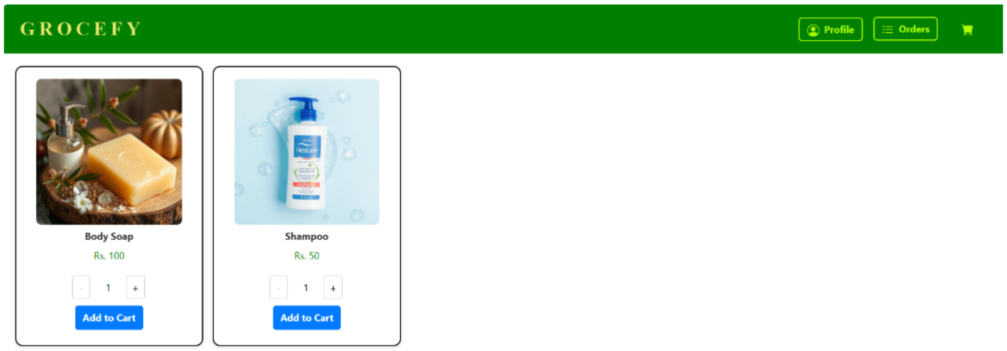
Email: xyz@gmail.com

Password: •••

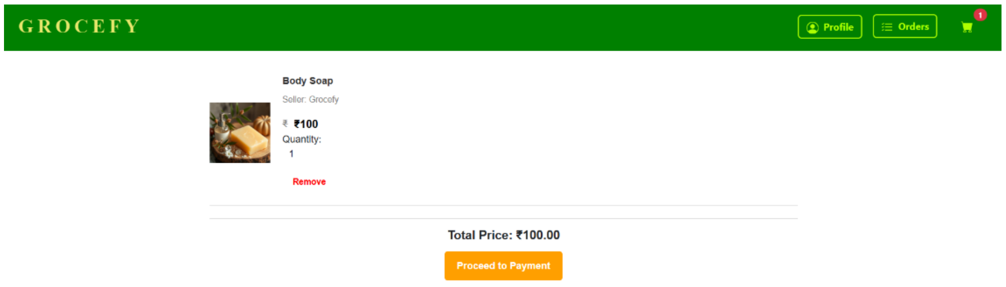
5.3 Login Page



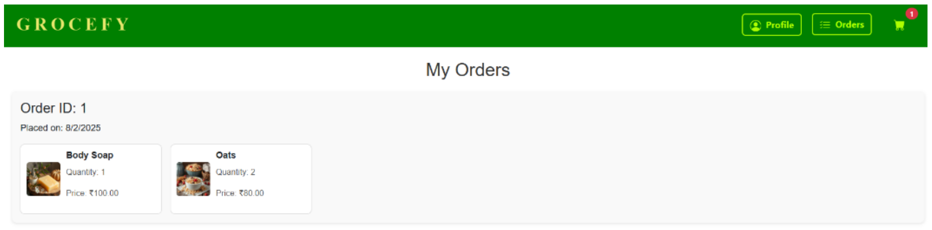
5.4 Category Page



5.5 Product Page



5.6 View Cart



My Orders

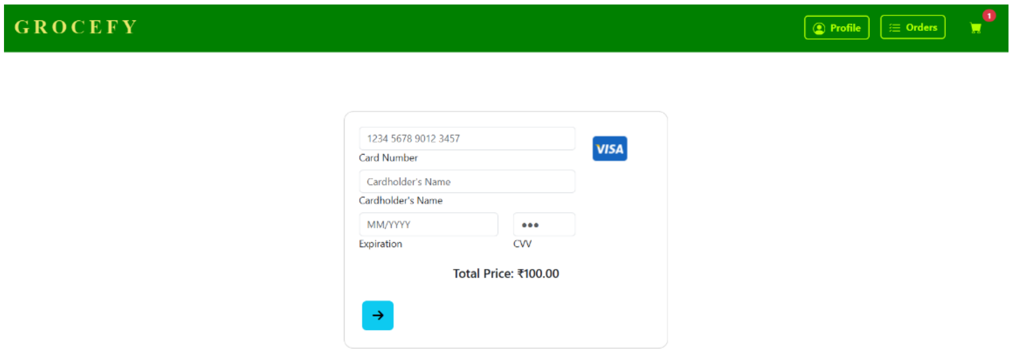
Order ID: 1

Placed on: 8/2/2025

Body Soap
Quantity: 1
Price: ₹100.00

Oats
Quantity: 2
Price: ₹80.00

5.7 Order Page



1234 5678 9012 3457
Card Number

Cardholder's Name
Cardholder's Name

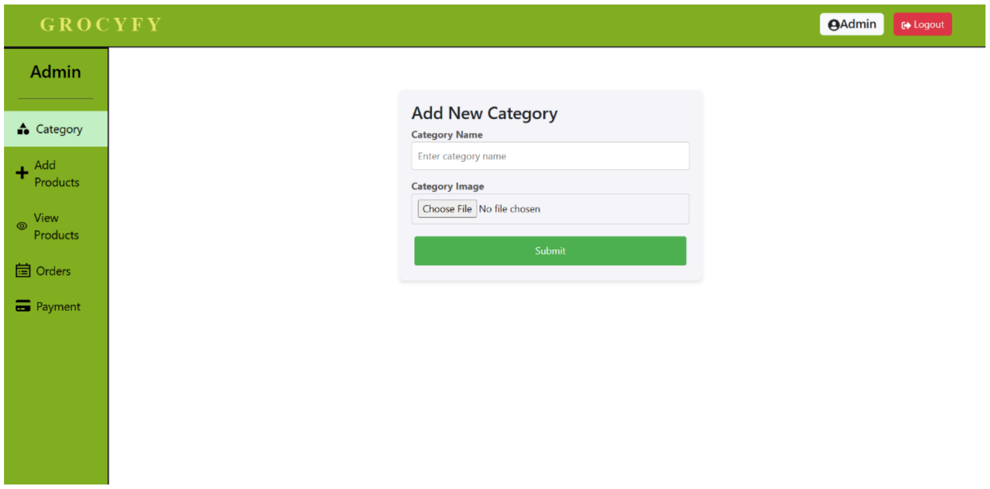
MM/YYYY
Expiration

•••
CVV

Total Price: ₹100.00

5.8 Payment Page

Admin Sections



Add New Category

Category Name

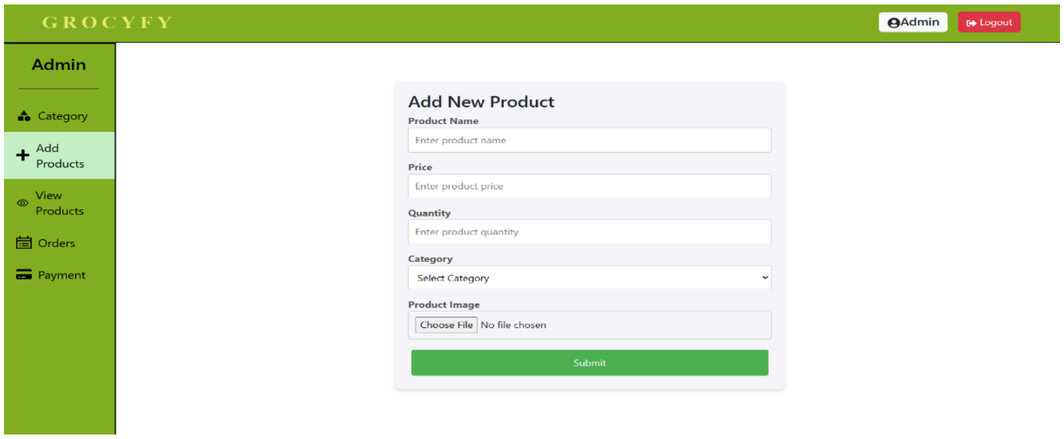
Enter category name

Category Image

Choose File No file chosen

Submit

5.9 Add Category Page



Add New Product

Product Name

Enter product name

Price

Enter product price

Quantity

Enter product quantity

Category

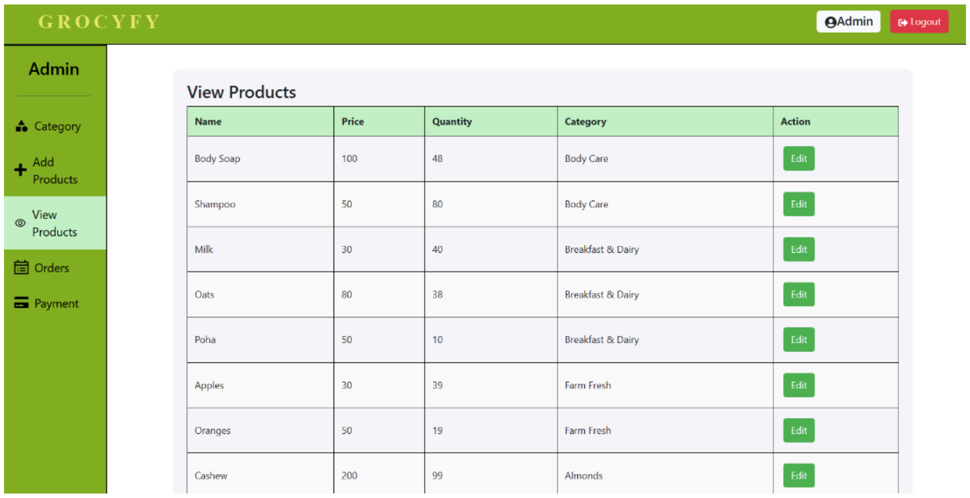
Select Category

Product Image

Choose File No file chosen

Submit

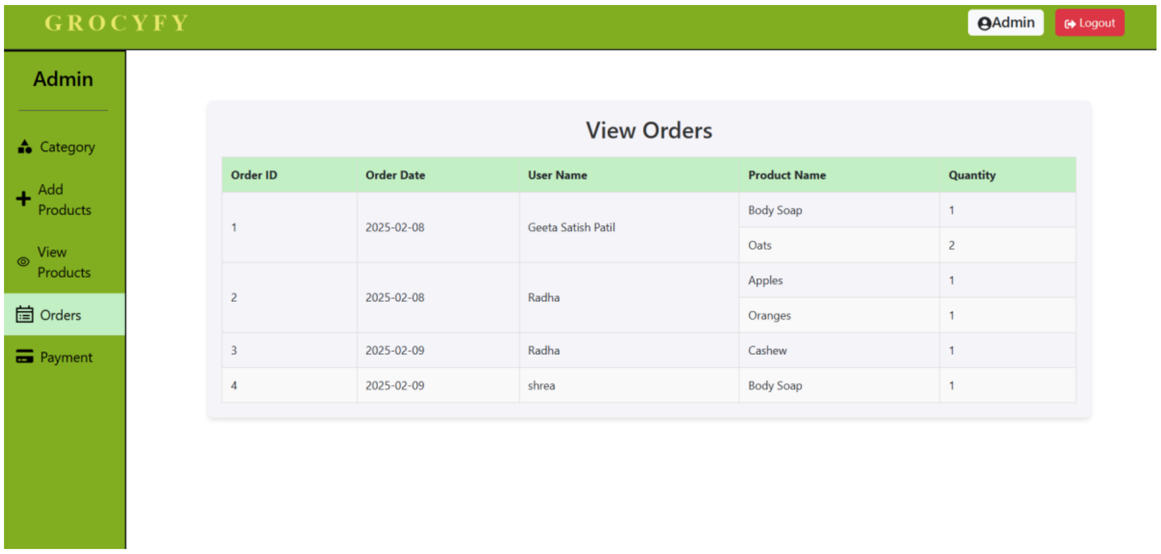
5.10 Add Product Page



View Products

Name	Price	Quantity	Category	Action
Body Soap	100	48	Body Care	Edit
Shampoo	50	80	Body Care	Edit
Milk	30	40	Breakfast & Dairy	Edit
Oats	80	38	Breakfast & Dairy	Edit
Poha	50	10	Breakfast & Dairy	Edit
Apples	30	39	Farm Fresh	Edit
Oranges	50	19	Farm Fresh	Edit
Cashew	200	99	Almonds	Edit

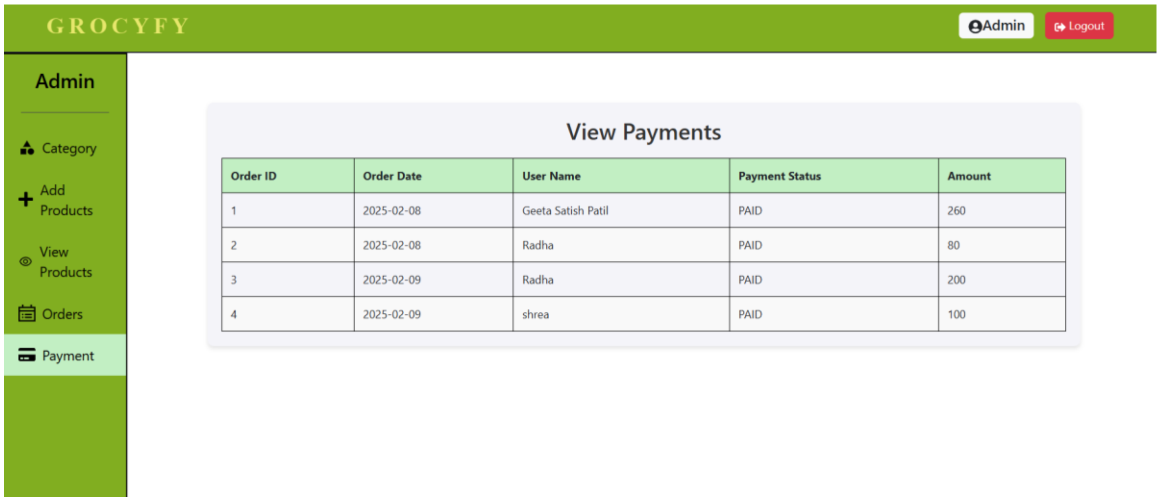
5.11 View Product Page



View Orders

Order ID	Order Date	User Name	Product Name	Quantity
1	2025-02-08	Geeta Satish Patil	Body Soap	1
			Oats	2
2	2025-02-08	Radha	Apples	1
			Oranges	1
3	2025-02-09	Radha	Cashew	1
4	2025-02-09	shrea	Body Soap	1

5.12 View Order Page



View Payments

Order ID	Order Date	User Name	Payment Status	Amount
1	2025-02-08	Geeta Satish Patil	PAID	260
2	2025-02-08	Radha	PAID	80
3	2025-02-09	Radha	PAID	200
4	2025-02-09	shrea	PAID	100

5.13 View Payment page

6. Conclusion

In conclusion, the development of the Grocery Management System addresses the critical need for efficient and streamlined operations within the grocery retail sector. By integrating functionalities such as inventory management, sales processing, customer relationship management, and supplier coordination, the system enhances operational efficiency and accuracy. The implementation of robust security measures ensures data integrity and user trust, while the emphasis on portability allows for flexible deployment across various platforms. Collectively, these features contribute to improved resource utilization, reduced operational costs, and elevated customer satisfaction, thereby providing a comprehensive solution to the challenges faced by modern grocery stores.

7. REFERENCES

1. Spring Boot Documentation

URL: https://spring.io/projects/spring-boot

2. React.js Documentation

URL: https://reactjs.org/docs/getting-started.html

3. Redux Documentation

URL: https://redux.js.org

4. Java Programming Language

URL: https://www.oracle.com/java/

5. MySQL Workbench Documentation

URL: https://dev.mysql.com/doc/workbench/en/

6. Spring Boot with React and Redux

URL: https://www.baeldung.com/spring-boot-react-and-redux

7. Java Persistence API (JPA) Documentation

URL: https://www.eclipse.org/eclipselink/documentation/2.7/

8. Swagger Documentation for Spring Boot

URL: https://springdoc.org/

9. MDN Web Docs

URL: https://developer.mozilla.org/

10. React Redux Integration Guide

URL: https://react-redux.js.org/