Departmental Store Management System

A Project Report Submitted

to

MANIPAL ACADEMY OF HIGHER EDUCATION

For Partial Fulfillment of the Requirement for the

Award of the Degree

Of

Bachelor of Technology

in

Computer and Communication Engineering

by

Priyanshu Vij, Tavish Gupta, Aryaman Singhi

210953330, 210953332, 210953396

Under the guidance of

Mr. Akshay KC
Associate Professor – Senior Scale
Department of I&CT
Manipal Institute of Technology
Manipal, Karnataka, India

Dr. Diana Olivia
Associate Professor
Department of I&CT
Manipal Institute of Technology
Manipal, Karnataka, India

MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
A Constituent Unit of MAHE, Manipal

March 2023

ABSTRACT

A departmental source is an essential community service center. It acts as a source of fulfilling human wants and needs of members of a geographical community. They act as a middleman between the wholesale suppliers of essential goods and the end consumer – for example, households.

This project aims to create an efficient management system for owners and employees of a departmental store, allowing them to carry out various tasks such as refer to the suppliers, existing customers, inventory details, loyalty program depending on the time's need. It stores all the relevant information in an easy to view manner.

TABLE OF CONTENTS

INTRODUCTION

This project report outlines the development of a departmental store management system that utilizes database design and user interface (UI) creation. The system aims to streamline the store's inventory management, sales tracking, and customer relations management.

The database design includes tables for product information, employee data, and sales transactions, among others. The UI was designed with a user-friendly interface, providing quick access to relevant information, and allowing for efficient data entry and retrieval. Overall, this project demonstrates how the integration of database design and UI creation can enhance the management of a departmental store.

PROBLEM STATEMENT

Departmental stores are facing various challenges in managing their inventory, tracking sales, and maintaining customer relations. Many stores rely on manual systems that are time-consuming and error-prone. This can result in poor customer service, stockouts, and loss of sales.

To address these issues, there is a need for a digital management system that can automate store processes, provide real-time inventory updates, and track sales data. This project aims to create a departmental store management system that uses database design and UI creation to improve the efficiency and effectiveness of store operations, resulting in improved customer satisfaction, increased sales, and reduced costs.

OBJECTIVES

1. Develop a departmental store management system that automates inventory management, sales tracking, and customer relations management processes.

2. Implement a user-friendly UI that allows store employees to easily access and input data into the system.

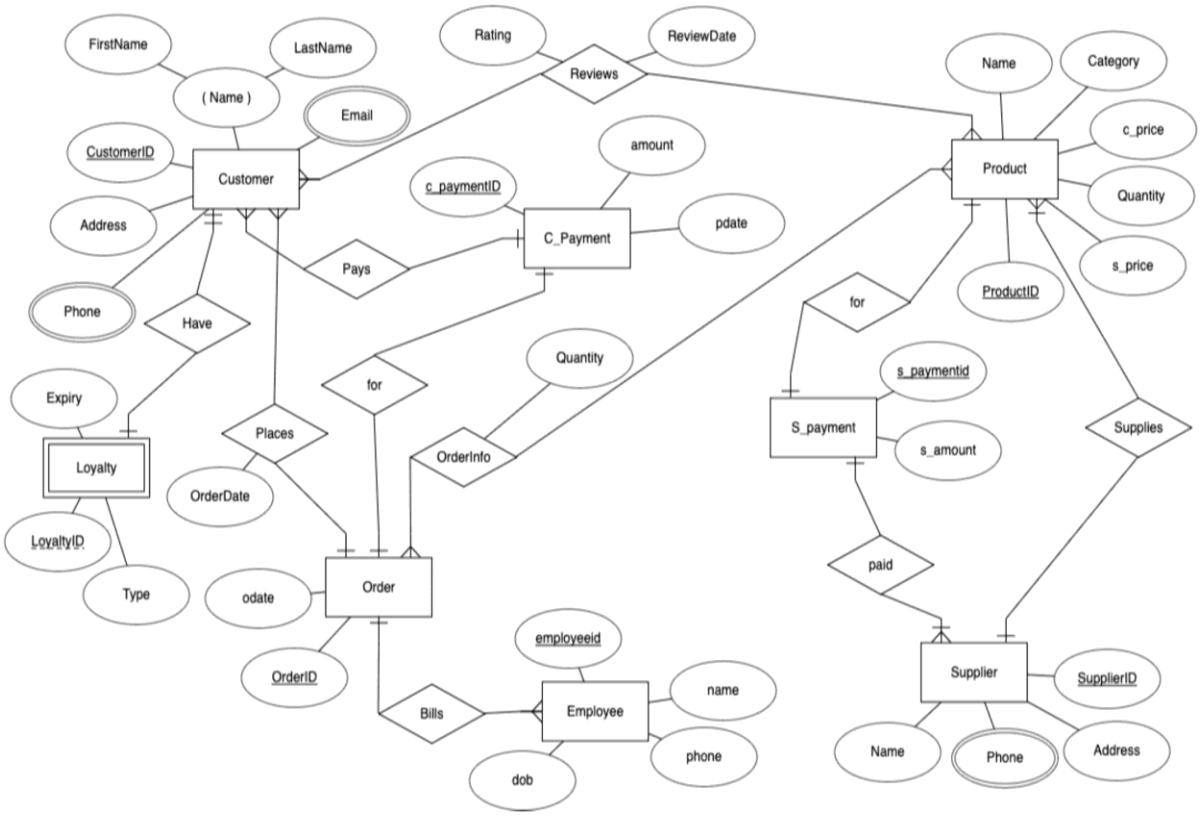
3. Create a database design that accurately stores and organizes product information, employee data, and sales transactions, among others.

4. Provide real-time updates on inventory levels and sales data to improve decision-making and restocking processes.

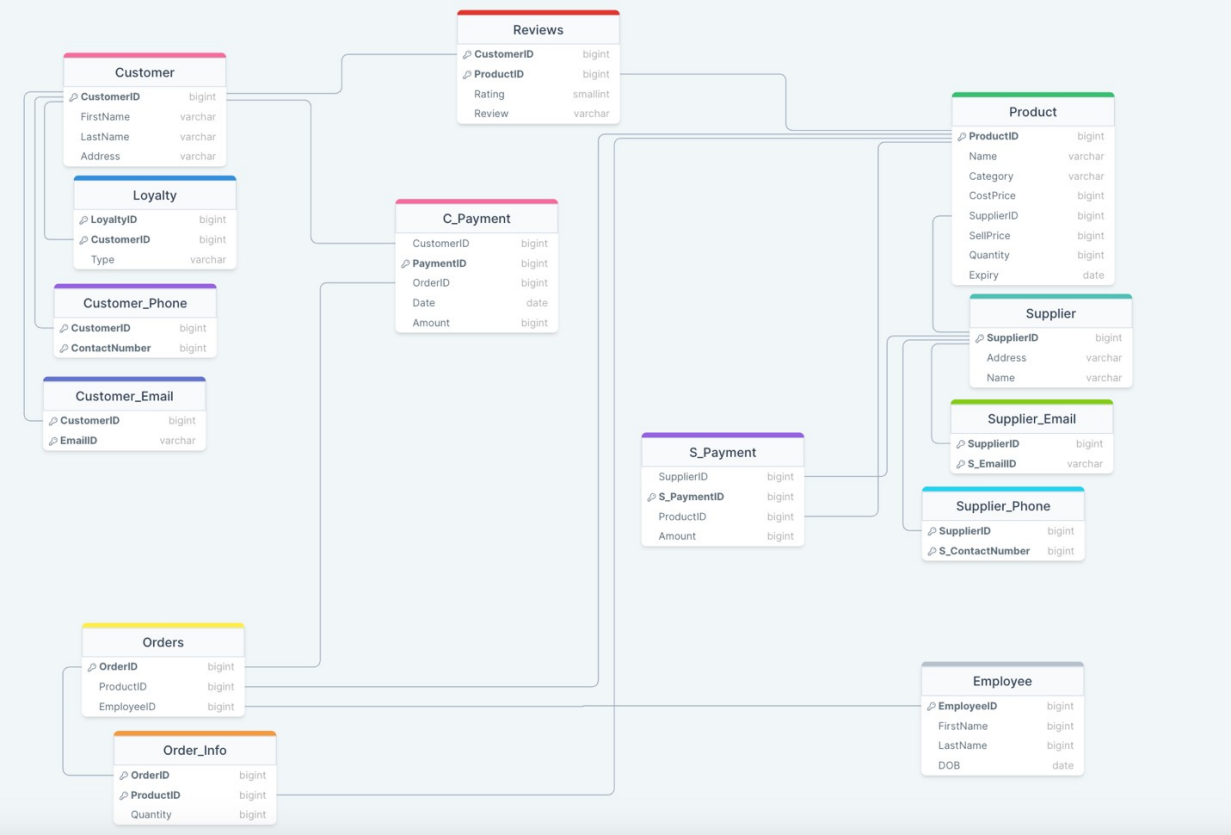
5. Improve overall store efficiency and profitability by reducing errors and streamlining operations through the implementation of the digital management system.

DATABASE DESIGN

ER DIAGRAM



SCHEMA DIAGRAM



REDUCTION

Customer(**CustomerID**, FirstName, LastName, Address)

Loyalty(**LoyaltyID, CustomerID,** Type)

Customer_Phone(**CustomerID, ContactNumber**)

Customer_Email(**CustomerID, EmailID**)

Orders(**OrderID**, ProductID, EmployeeID)

Order_Info(**OrderID, ProductID,** Quantity)

Reviews(**CustomerID**, **ProductID**, Rating, Review)

C_Payment(CustomerID, **PaymentID**, OrderID, Date, Amount)

S_Payment(SupplierID, S_**PaymentID**, ProductID, Amount)

Product(**ProductID**, Name, Category, CostPrice, SupplierID, SellPrice, Quantity, Expiry)

Supplier(**SupplierID**, Address, Name)

Supplier_Phone(**SupplierID, S_ContactNumber**)

Supplier_Email(**SupplierID, S_EmailID**)

Employee(**EmployeeID**, FirstName, LastName, DOB)

NORMALIZATION

The functional dependencies are as follows:

- In the Customer table, CustomerID → FirstName, LastName, Address.
- In the Loyalty table, LoyaltyID → CustomerID, and CustomerID → LoyaltyID, Type (since each customer can have at most one loyalty record).
- In the Customer_Phone table, (CustomerID, ContactNumber) → none (since a customer can have multiple phone numbers).
- In the Customer_Email table, (CustomerID, EmailID) → none (since a customer can have multiple email addresses).
- In the Orders table, OrderID → ProductID, EmployeeID, and ProductID, EmployeeID → OrderID (since an order can have multiple products and employees, but each product/employee can only be associated with one order).
- In the Order_Info table, (OrderID, ProductID) → Quantity (since the quantity of a product in an order depends on both the order and the product).
- In the Reviews table, (CustomerID, ProductID) → Rating, Review (since each customer can submit at most one review for a given product).
- In the C_Payment table, PaymentID → CustomerID, OrderID, Date, Amount, and (CustomerID, OrderID) → PaymentID (since each payment is uniquely identified by a payment ID, and a customer can make multiple payments for the same order).

- In the S_Payment table, S_PaymentID → SupplierID, ProductID, Amount, and (SupplierID, ProductID) → S_PaymentID (since each payment is uniquely identified by a payment ID, and a supplier can receive multiple payments for the same product).
- In the Product table, ProductID → Name, Category, CostPrice, SupplierID, SellPrice, Quantity, Expiry, and SupplierID → Address, Name (since a product is uniquely identified by its ID, and a supplier can supply multiple products).
- In the Supplier table, SupplierID → Address, Name.
- In the Supplier_Phone table, (SupplierID, S_ContactNumber) → none (since a supplier can have multiple phone numbers).
- In the Supplier_Email table, (SupplierID, S_EmailID) → none (since a supplier can have multiple email addresses).
- In the Employee table, EmployeeID → FirstName, LastName, DOB.

1NF

The given schema is already in 1NF because it satisfies the criteria of atomicity, i.e., each column in a table contains only a single value.

2NF

To convert the given schema to 2NF, we need to ensure that all non-key attributes in each table depend only on the primary key of that table.

We can start by creating a new table for Order_Info, which has a composite primary key of OrderID and ProductID, and contains only the Quantity attribute. This separates out the non-key attribute Quantity from the Orders table and eliminates the partial dependency.

Orders(OrderID, EmployeeID)
Order_Product(OrderID, ProductID)
Product(ProductID, Name, Category, CostPrice, SupplierID, SellPrice, Quantity, Expiry)
C_Payment(PaymentID, CustomerID, OrderID, Date, Amount)
S_Payment(S_PaymentID, SupplierID, ProductID, Amount)

Next, we can create a new table for Loyalty, which has a composite primary key of LoyaltyID and CustomerID, and contains only the Type attribute. This separates out the non-key attribute Type from the Customer table and eliminates the partial dependency.

Customer(CustomerID, FirstName, LastName, Address)
Loyalty(LoyaltyID, CustomerID, Type)
Customer_Phone(CustomerID, ContactNumber)
Customer_Email(CustomerID, EmailID)
Reviews(CustomerID, ProductID, Rating, Review)
Supplier(SupplierID, Address, Name)
Supplier_Phone(SupplierID, S_ContactNumber)
Supplier_Email(SupplierID, S_EmailID)
Employee(EmployeeID, FirstName, LastName, DOB)

3NF

To convert the schema to 3NF, we need to eliminate transitive dependencies. A transitive dependency occurs when a non-key attribute depends on another non-key attribute rather than the primary key.

We can start by creating a new table for the Supplier_Products, which has a composite primary key of SupplierID and ProductID, and contains only the SupplierCost attribute. This separates out the non-key attribute SupplierCost from the Product table and eliminates the transitive dependency between the SupplierID and the CostPrice.

Customer(CustomerID, FirstName, LastName, Address)
Loyalty(LoyaltyID, CustomerID, Type)
Customer_Phone(CustomerID, ContactNumber)
Customer_Email(CustomerID, EmailID)
Orders(OrderID, EmployeeID)
Order_Product(OrderID, ProductID)
Order_Info(OrderID, ProductID, Quantity)
Reviews(CustomerID, ProductID, Rating, Review)
C_Payment(PaymentID, CustomerID, OrderID, Date, Amount)
S_Payment(S_PaymentID, SupplierID, ProductID, Amount)
Product(ProductID, Name, Category, SellPrice, Quantity, Expiry)
Supplier(SupplierID, Address, Name)
Supplier_Products(SupplierID, ProductID, SupplierCost)
Supplier_Phone(SupplierID, S_ContactNumber)
Supplier_Email(SupplierID, S_EmailID)
Employee(EmployeeID, FirstName, LastName, DOB)

BCNF

To convert the schema to BCNF, we need to ensure that all functional dependencies are dependencies on the primary key, and there are no non-trivial functional dependencies between non-key attributes.

There are no non-trivial functional dependencies between non-key attributes in the current schema, so it is already in BCNF.

METHODOLOGY

To develop the departmental store management system, a comprehensive methodology was followed, encompassing the stages of data gathering, schema design, and user interface (UI) creation. This section provides an overview of the step-by-step process undertaken to achieve the desired outcomes.

Based on the gathered data, a detailed database schema was created to represent the various entities and their relationships. The team analyzed the requirements and identified the key entities, including products, employees, customers, suppliers, orders, and payments. These entities were organized into separate tables, ensuring the appropriate attributes and relationships were defined.

The schema consisted of primary keys, foreign keys, and the necessary constraints to maintain data integrity. For example, the Customer table contained fields such as CustomerID, FirstName, LastName, and Address, while the Orders table included attributes like OrderID, ProductID, and EmployeeID. The schema design phase aimed to capture all the essential information needed for effective store management.

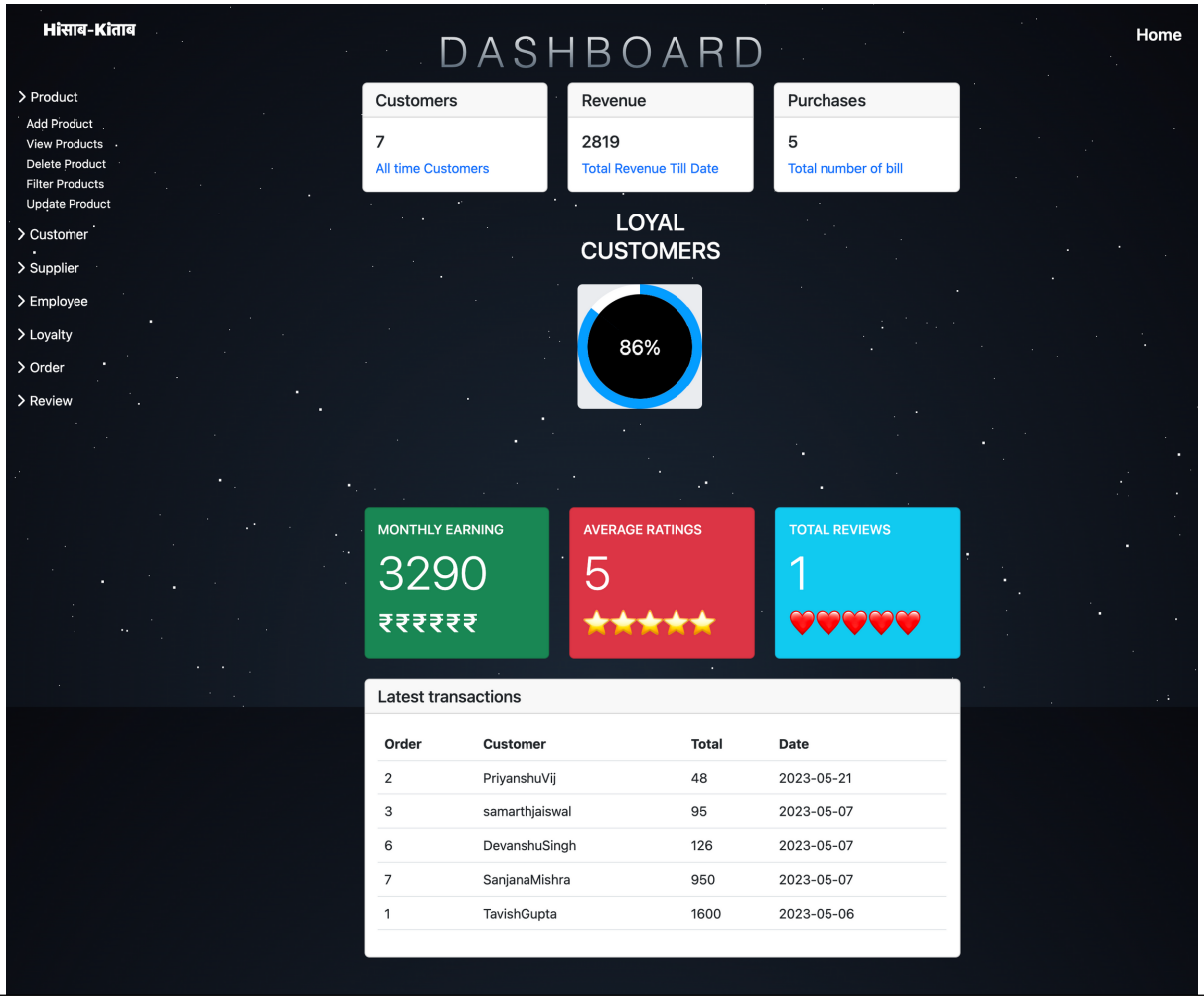
Simultaneously with the schema design, we focused on developing the user interface (UI) for the management system. The UI was created using a combination of HTML, CSS, JavaScript, and Bootstrap framework. The goal was to design an interface that would facilitate easy navigation and efficient data entry.

The UI design incorporated a visually appealing layout with well-organized sections for different functionalities. The team worked on creating elements to facilitate data entry, retrieval, and manipulation. CSS styling was applied to enhance the visual presentation, while JavaScript and Bootstrap were utilized to add functionality and responsiveness to the UI.

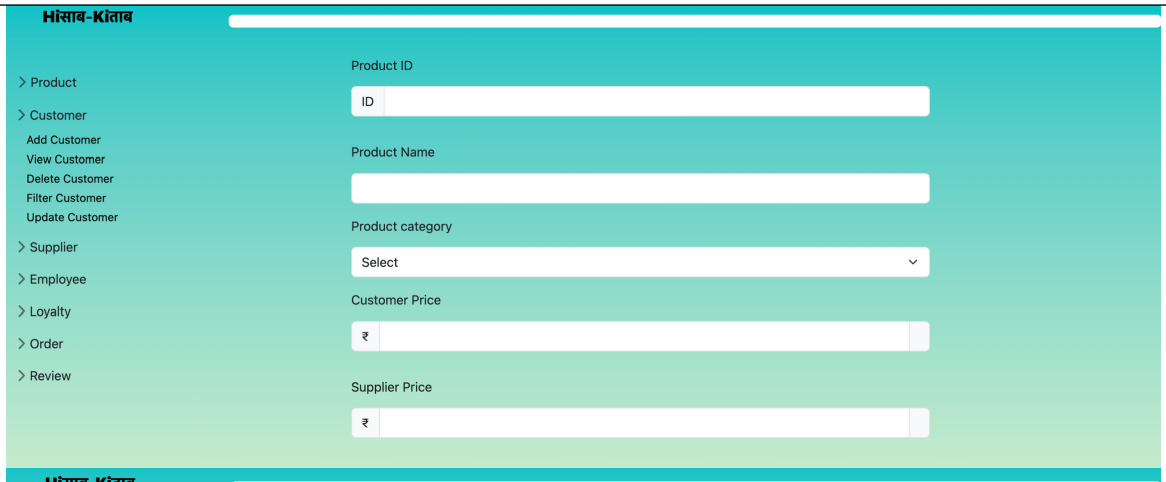
In order to ensure data integrity, eliminate redundancy, and improve database performance, the tables in the departmental store management system were normalized. Normalization is organizing data into efficient structures by minimizing data duplication and establishing relationships between entities. The normalization process involved analysing the data requirements and applying the principles of normalization, specifically up to the third normal form (3NF). This involved breaking down tables into smaller, more focused entities and arranging them in a way that reduced redundancy.

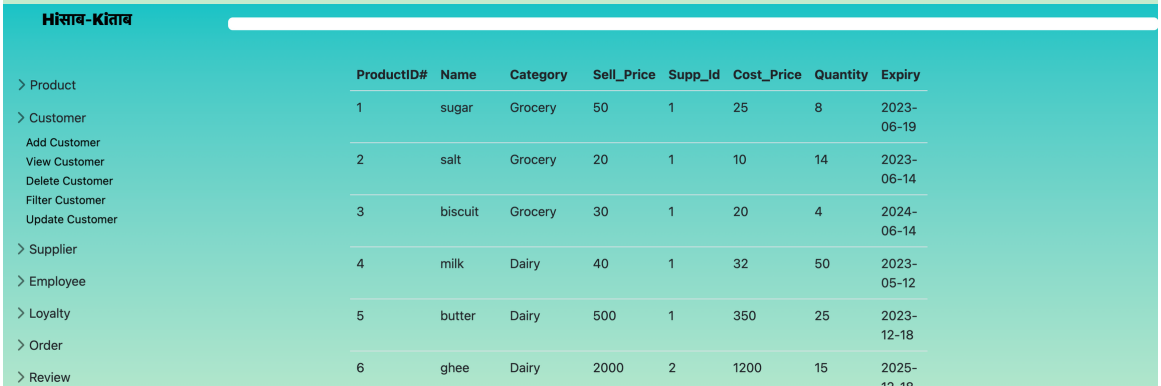
Overall, the methodology encompassed data gathering, schema design, and UI creation, all aimed at developing a robust departmental store management system. By combining the insights gathered from stakeholders, a well-structured database schema was created, enabling efficient data management. Simultaneously, the UI design process focused on delivering a user-friendly interface that met the needs of the store staff. The seamless integration of database design and UI creation laid the foundation for an effective system that streamlines inventory management, sales tracking, and customer relations in the departmental store

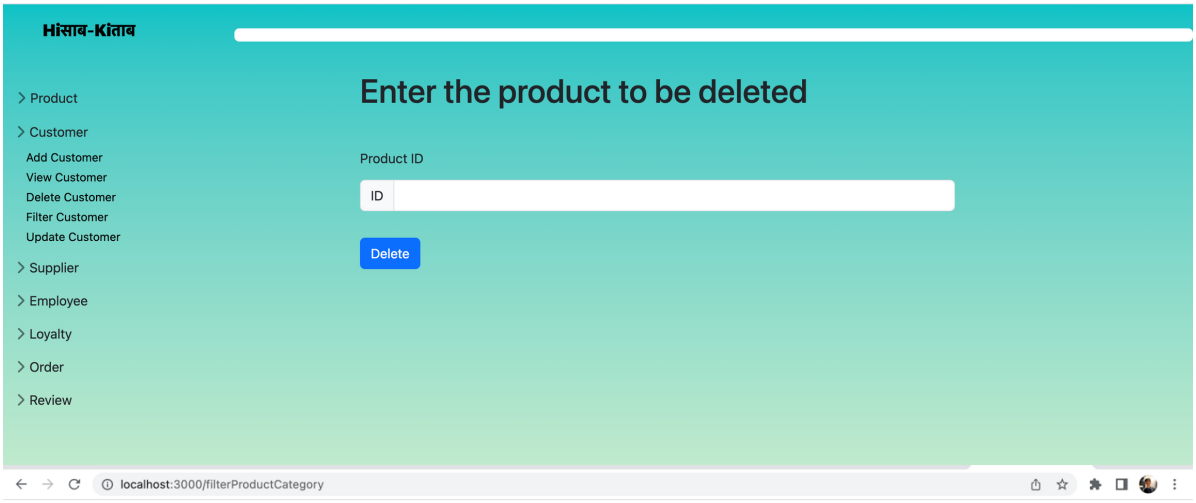
RESULTS



Dashboard | Homepage

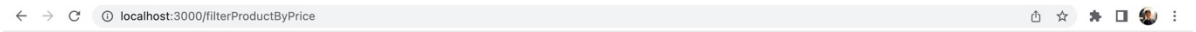




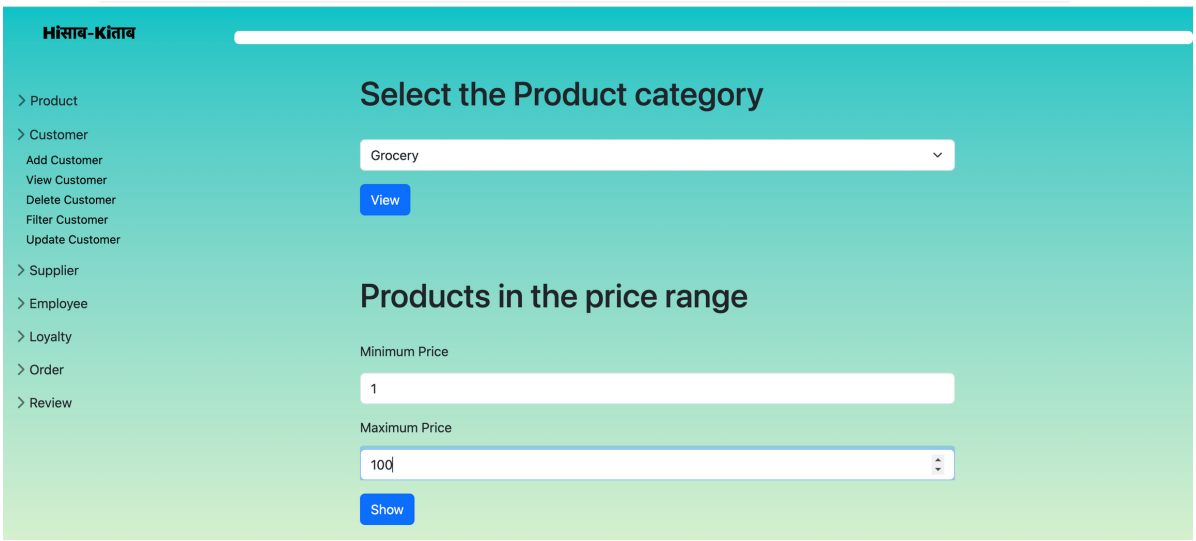


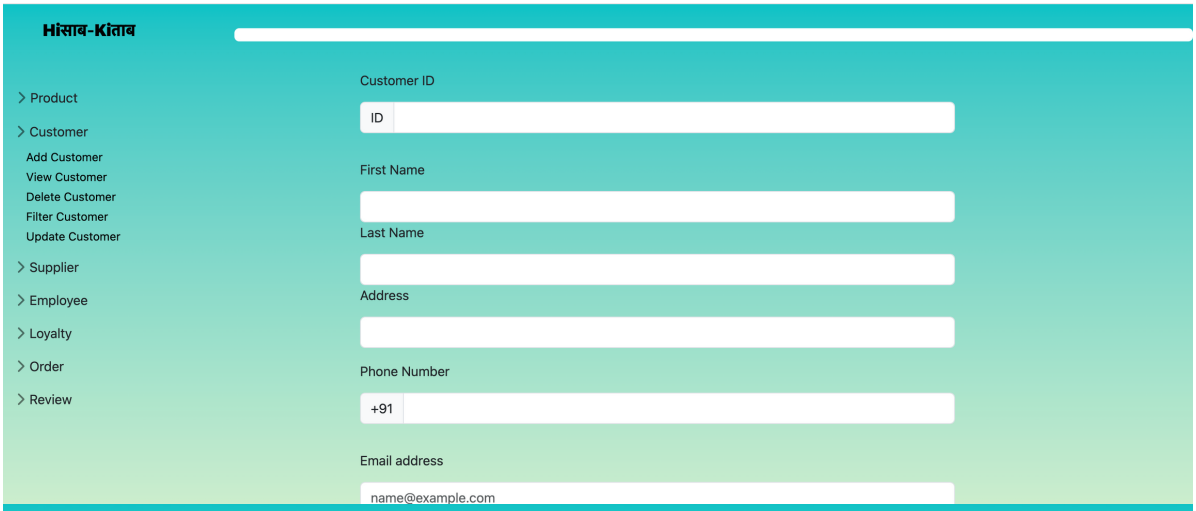
Supplier Name	ID#	Category	Number of products ordered
Gatik	1	Grocery	4

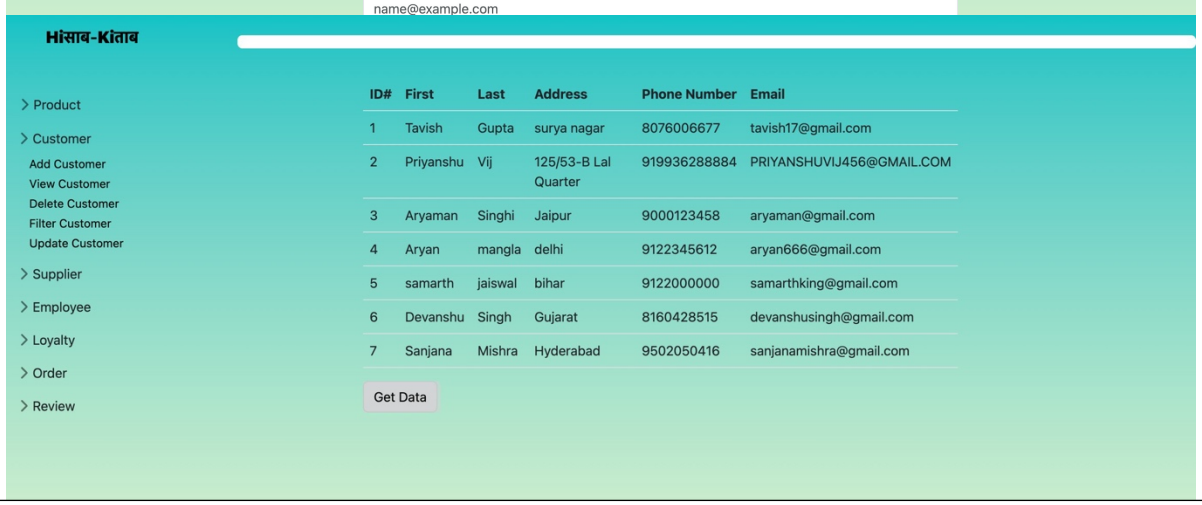
Delete Product



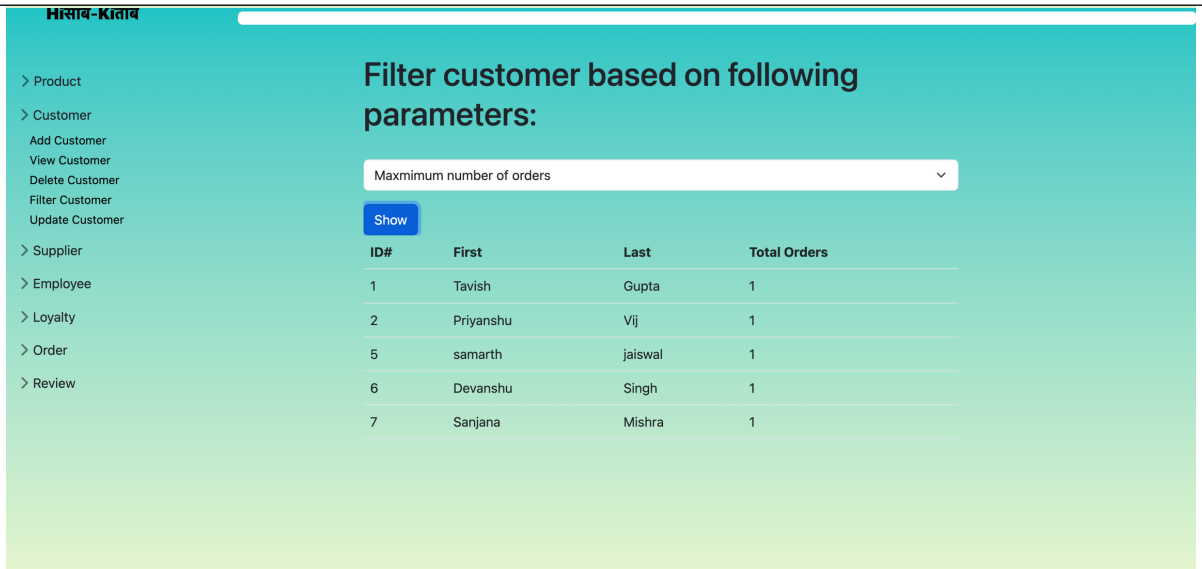
ID#	Name	Category	SP	CP	Supplier ID#	Quantity
1	sugar	Grocery	50	25	1	8
2	salt	Grocery	20	10	1	14
3	biscuit	Grocery	30	20	1	4
4	milk	Dairy	40	32	1	50
7	pen	Stationary	20	17	3	98
8	sheet	Stationary	5	3	3	65
9	stapler	Stationary	50	33	3	73
10	scrunchie	Accessories	100	60	4	40

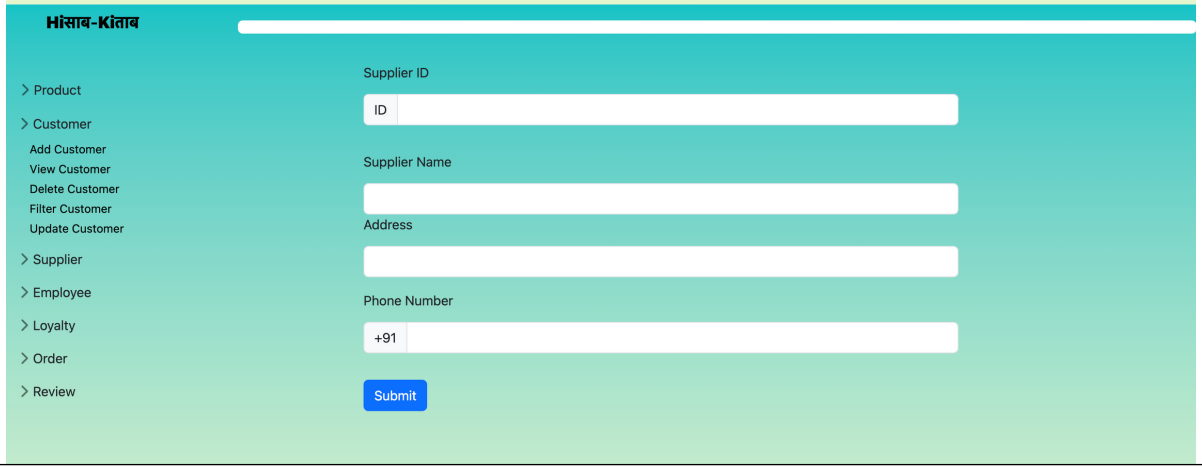


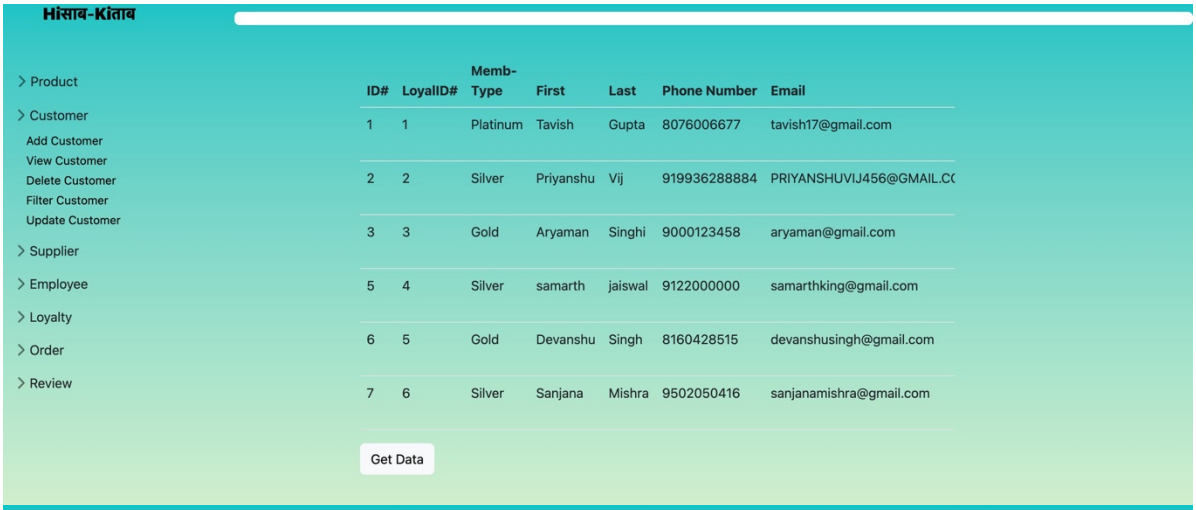


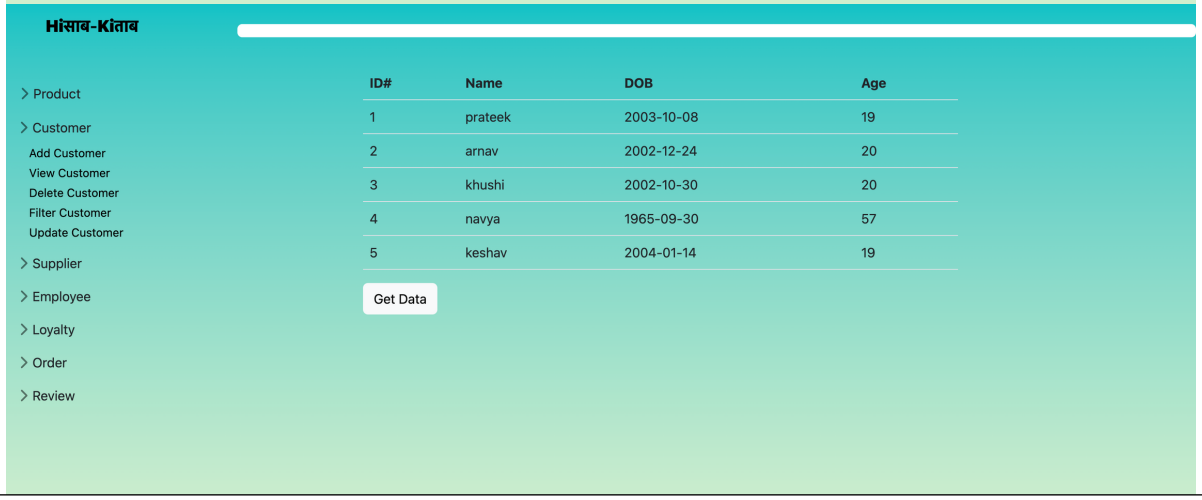


Add & View Customer

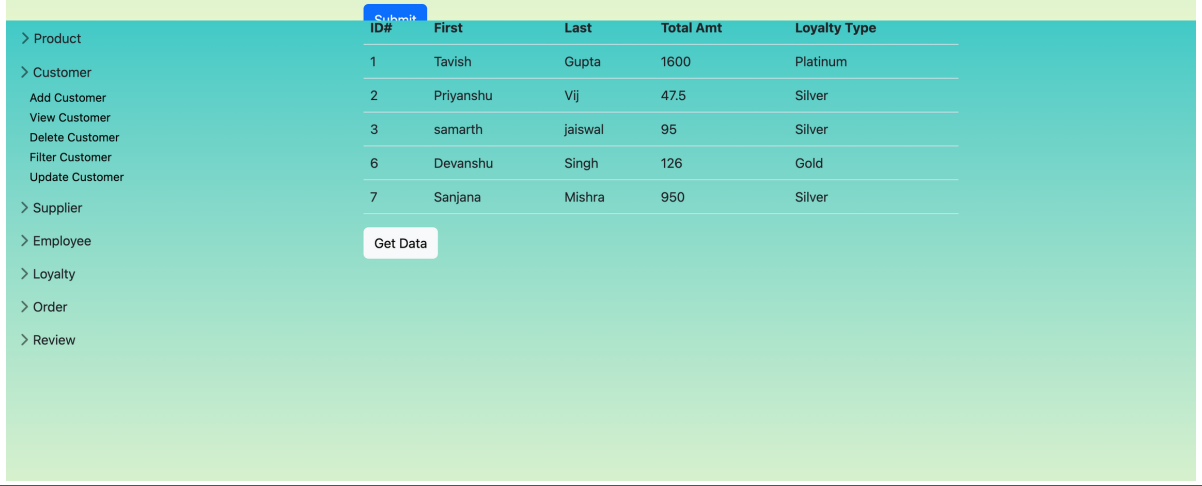




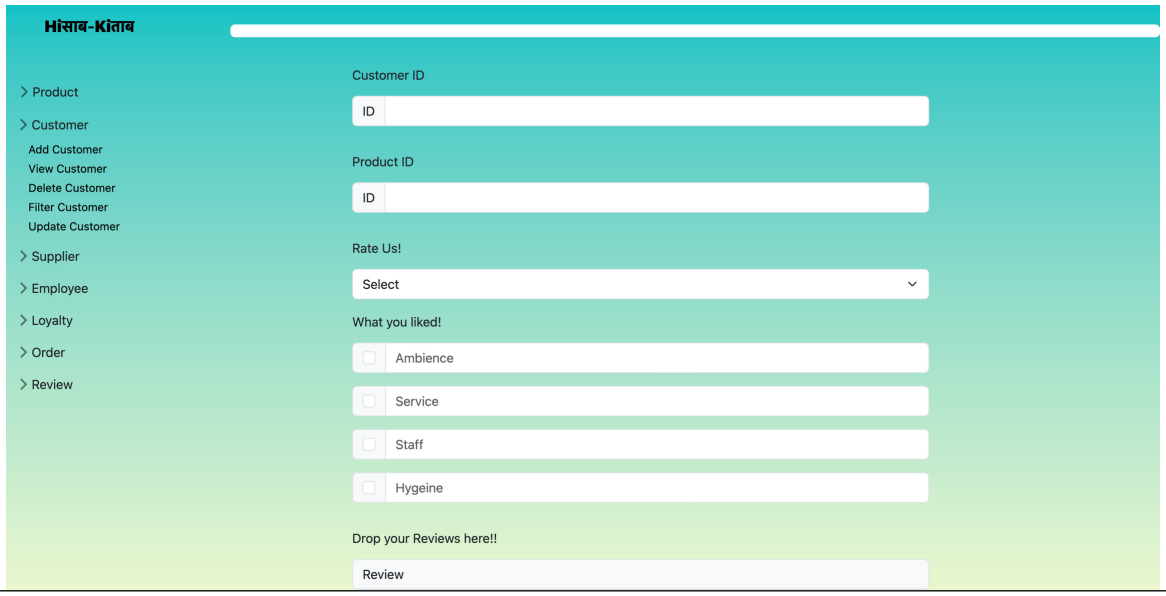




View Loyalty & View Employee



View Review



Add Review

SUSTAINABLE DEVELOPMENT GOAL

A well-designed database management system for a departmental store can contribute to achieving the SDG of Decent Work and Economic Growth by increasing efficiency, productivity, and job satisfaction for employees. It can also provide valuable data insights for informed decision-making on products, pricing, and marketing strategies, leading to growth and expansion of the store, creating more job opportunities and contributing to the overall economic growth of the community.

FUTURE WORK

Future work for this departmental store management system project involves several potential areas for improvement and expansion. One area for further development could be the incorporation of machine learning algorithms to predict customer demand and optimize inventory levels. Another possibility is to integrate the system with e-commerce platforms to provide an omnichannel shopping experience for customers.

Additionally, the system could be extended to include supply chain management features, allowing for better coordination with suppliers and more efficient order fulfillment. Finally, further improvements to the UI could be made to enhance user experience and improve the system's usability. These potential future developments have the potential to further increase the system's efficiency and effectiveness in managing departmental store operations.

REFERENCES

- Stack Overflow - https://stackoverflow.com/
- BootStrap Documentation - https://getbootstrap.com/docs/5.3/getting-started/introduction/
- Geeks For Geeks - https://www.geeksforgeeks.org/plsql-introduction/