YATHAARTHI:SHOP MANAGEMENT SYSTEM

Project report submitted for

Database Management Systems - Course Project

By

Aman Sahu (211020410, DSAI)
Rigved Desai (211020438, DSAI)

Dr. Shyama Prasad Mukherjee

International Institute of Information Technology, Naya Raipur
(A Joint Initiative of Govt. of Chhattisgarh and NTPC)
Email: iiitnr@iiitnr.ac.in, Tel: (0771) 2474040,

Web: www.iiitnr.ac.in

Contents:

ACKNOWLEDGMENTS

The completion of this project involves the sincere efforts of Rigved Desai and Aman Sahu. We have been lucky enough to have received a lot of help and support from the Internet during the making of this project, so with gratitude; we take this opportunity to acknowledge all those whose guidance and encouragement helped us emerge successful.

We are thankful for the resourceful guidance, timely assistance, and graceful gesture of our guide **Mr. Srinivas Naik N,** Assistant Professor, Department of Computer Science and Engineering, who has helped us in every aspect of our project work.

We express our sincere thanks to our most respected and beloved Vice-Chancellor and Director, Sir **Dr. Pradeep K Sinha, for supporting** us in our academic endeavors.

Moreover, last but not least, we would be delighted to express our heart full thanks to all the teaching and non-teaching staff of the DSAI department and our friends who have rendered their help, motivation, and support.

AMAN SAHU

RIGVED DESAI

ABSTRACT/ PROBLEM STATEMENT

Our Aim:

To create a database management system which will help an offline/online store manager manage all their transactions, inventory and seamlessly run a customer loyalty scheme alongside it.

Abstract:

This system will have a portal for shopkeepers to view and manage their transaction and customer data while also allowing them to run a points-based loyalty scheme. The customer will also have a portal which they can access in order to view their buying history and their points balance.

This system will have 2 levels:

1. Administrator
2. Customer

Administrator Rights:

- Can VIEW, ADD, UPDATE and DELETE inventory and customer data.
- Can VIEW, ADD and UPDATE transaction data.
- Can manage the rewards scheme for each customer.

Customer Rights:

- Can VIEW personal transaction data.
- Can VIEW personal customer data.

INTRODUCTION

A shop is a place where a shopkeeper can sell goods and commodities to people. Shopkeepers have the need to keep a record of the items they are have in stock, their prices and the items which have been sold. These records can help them analyze their sales, make key decisions and grow their business. This problem calls for the need of a Shop Database Management System.

A Shop Management System maintains and modifies the records of suppliers, hardware, software, sales, purchases, and customer information.

These tasks are usually completed and controlled manually, increasing the possibility of mistakes caused by humans. As a result, it needs to be managed by a safe and trustworthy automatic computerized system.

The above-described Shop Management System can result in an error-free, secure, dependable, and quick management system. It can help the employees focus on their other tasks rather than record keeping so they can better serve their clients. Consequently, it will aid small shopkeepers and organizations in making better use of their resources.

By definition, a Customer Loyalty Program is a marketing approach that recognizes and rewards customers who purchase or engage with a brand on a recurring basis. A company may dole out points or perks, and graduate customers to higher levels of loyalty the more they buy. These incentives and specific benefits often result in the customer becoming a more regular consumer or the ideal — a brand promoter. Benefits may involve free merchandise, rewards, coupons, or insider perks like early access to new products.

There are several different types of customer loyalty programs such as tiered loyalty programs, points-based loyalty programs, subscription-based loyalty programs, etc.

Points-based programs reward customers with points for each purchase, making them popular in retail environments, like restaurants. When customers reach a certain number of points, they can cash those points in to get a product or receive a discount.

This project deals with shop inventory and transaction management and structuring the current customer loyalty schemes of small shops in India. A store is a place where the work is to sell products, prepare their bills, keep track of their inventory and purchases, keep records, and create reports. Often shopkeepers give out discounts to their regular customers all of which many shopkeepers still store in written format in their notebooks. Many don’t even store it anywhere. The customers also do not have a proper way to actually track their purchase history. The goal is to automate its current manual system with the aid of computerized tools and comprehensive computer software that satisfies the needs of both the shopkeepers and their customers.

BACKGROUND/LITERATURE REVIEW

There have been countless research studies studying the relationship between proper management and increasing sales, all pointing towards a positive correspondence between the two. By implementing effective inventory management practices, companies can improve their overall performance and profitability. This can be achieved by reducing inventory costs, maximizing the utilization of available resources, and effectively tracking and managing inventory levels. By adopting these strategies, companies can improve their inventory management practices and ultimately achieve better financial performance. [1]

The proper implementation of an inventory and transaction management system can provide a wealth of data that can be used for advanced analysis and machine learning. By collecting and organizing data on inventory levels, sales transactions, and customer behavior, companies can gain valuable insights and make more informed business decisions. Through data mining and the development of machine learning models, companies can uncover hidden patterns and trends, and use this information to optimize their operations and improve their overall performance. [2]

The findings of this study [3] indicate that loyalty programs are an effective way for companies to build and maintain customer retention. This means that by implementing a loyalty program that rewards customers with points that can be redeemed for discounts, free products, or other incentives, businesses can benefit from increased customer retention and long-term success. It suggests that customer retention is a key benefit of loyalty programs, and that by offering rewards and incentives to customers, businesses can improve customer satisfaction and encourage repeat business. In addition to the overall benefits of loyalty programs, point-based schemes can provide additional advantages, such as the ability to track customer behavior and preferences, and tailor rewards and incentives accordingly. Overall, it highlights the importance of loyalty programs in building and maintaining customer retention, and the potential benefits that shopkeepers can gain from implementing such programs, particularly point-based schemes.

A lot of shop transaction and inventory management models already exist such as this [4] on GitHub and a few customer loyalty trackers [5] exist as well. However, no such system which combines both the inventory management system and the customer loyalty management system focused mainly on small shopkeepers. Hence, the need was identified and this management system was created which can serve as useful portal for both shopkeepers and customers.

METHODOLOGY

This project has been built with several new technologies such as MySQL and Node.js. The code for this project was written in Javascript. Following are some brief details about them:

MySQL:

MySQL is a popular open-source relational database management system that is based on SQL. It is widely used in web applications and other applications where data is stored and accessed.

MySQL is known for its speed, reliability, and ease of use. It is easy to set up and use, and it is often used in combination with other technologies, such as PHP, to build dynamic web applications.

In MySQL, data is organized into tables, which are similar to excel sheets. Each table has rows, which are called records, and columns, which are called fields. MySQL allows users to define the structure of the tables in a database, as well as the relationships between different tables.

MySQL also provides various commands, or statements, that can be used to perform operations on the data. Some of the most commonly used MySQL commands include SELECT, INSERT, UPDATE, DELETE, and CREATE.

MySQL is a widely used database management system that is suitable for many different applications and use cases. It is an important tool for working with data in a database, and it is an essential skill for many data professionals.

One of the key advantages of MySQL is its open-source nature, which allows users to access and modify the source code as needed. This allows users to customize the software to meet their specific needs and requirements.

MySQL is also known for its strong security features, which protect data against unauthorized access and tampering. It also provides various tools and features for backing up and restoring data, ensuring that data is always available and recoverable in case of an emergency.

Overall, MySQL is a powerful and versatile database management system that is widely used in a variety of industries and applications. Its ease of use, strong security features, and open-source nature make it an excellent choice for managing data in a database.

In this project, MySQL has been used to implement the backend database relations along with performing basic CRUD operations on the database.

Node.js:

Node.js is a popular open-source, cross-platform JavaScript runtime environment that is used for building server-side applications. It is built on Chrome's V8 JavaScript engine and allows developers to create scalable, high-performance network applications using JavaScript on the server-side.

One of the key advantages of Node.js is its asynchronous, non-blocking I/O model, which makes it highly efficient and lightweight. This allows Node.js applications to handle many concurrent connections with minimal overhead, making it well-suited for real-time, high-performance applications.

Node.js also has a large and active community, with thousands of open-source libraries and frameworks available that can be easily integrated into Node.js applications. This allows developers to quickly and easily build and deploy complex applications using Node.js.

Node.js is widely used in a variety of industries, including e-commerce, healthcare, finance, and education. It is often used in combination with other technologies, such as React and Angular, to build modern web applications.

Overall, Node.js is a powerful and versatile runtime environment that is widely used for building server-side applications. Its asynchronous, non-blocking I/O model and large ecosystem make it an excellent choice for building high-performance, scalable applications.

In this project, Node.js has been used to deploy our application on the server, connect to our database on the backend, fetch data and provide it to the frontend to be displayed to the user.

A Node.js framework called Express.js has been used as middleware which serves the webpage files stored on the server to the client. It is designed to male building web applications with Node.js easy and fast by providing a simple set of tools and features. It has also been used for routing which is essentially process of determining how an application responds to a client request for a specific endpoint, or URI (Uniform Resource Identifier), on the web server.

Other technologies used include .EJS (Embedded JavaScript) files to design the basic framework for our webpages and integrate server-side variables in it and CSS (Cascading Style Sheets) to properly style it.

Database Design:

The database is designed to contain three relations namely:

1. Customer Data
2. Inventory Data
3. Transaction Data

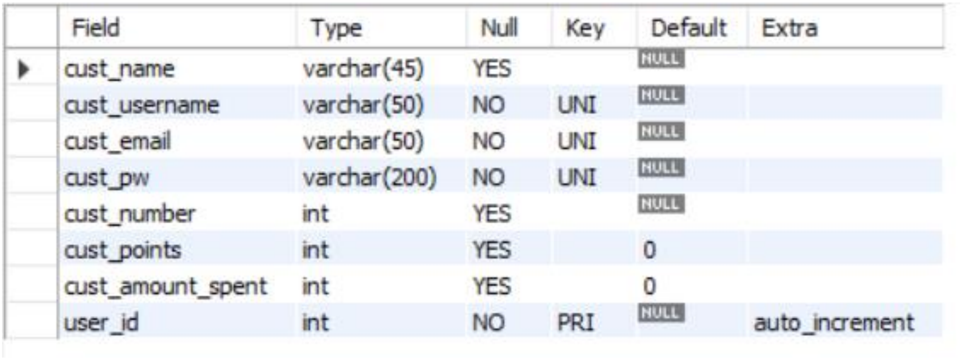
The details of each the relations are as follows:

Customer Data:

The relation Customer Data considers the Customer as an entity set with the following entities attached to it:

1. User ID: The User ID is the primary key of the customer data relation. It is an integer helps us to identify a customer individually. It is set to auto increment, i.e., its value is automatically set in the database when a new record is added in the database according to the number of records which were added prior to adding it.
2. Customer Name: The Customer Name attribute stores the full real name of the customer. It can be duplicate but cannot be NULL.
3. Customer Username: The Customer Username attribute is a string with which the user can log into the portal. It is set upon account creation and also with which the system identifies the user. It qualifies as a candidate key and cannot be NULL.
4. Customer Email: The Customer Email attribute stores the email ID of the user in the database. It cannot be duplicate or NULL and qualifies as a candidate key.
5. Customer Password: The Customer Password is a safe word string which the user can set upon account creation which restricts the access of their account to anyone but themselves. It can be duplicate but cannot be NULL. The requirements to login are correct username and its corresponding password.
6. Customer Number: The Customer Number is the telephone/mobile number of the customer. It can be NULL.
7. Customer Amount Spent: The Customer Amount spent the total amount spent of money spent by the customer in the shop. It is an integer and set to 0 by default. It is used to calculate the total number of points gained by the customer after a transaction.

8. Customer Points: The Customer Points attribute is the total number of points the user currently has. It is an integer and is set to 0 by default.



Field	Type	Null	Key	Default	Extra
cust_name	varchar(45)	YES		NULL	
cust_username	varchar(50)	NO	UNI	NULL	
cust_email	varchar(50)	NO	UNI	NULL	
cust_pw	varchar(200)	NO	UNI	NULL	
cust_number	int	YES		NULL	
cust_points	int	YES		0	
cust_amount_spent	int	YES		0	
user_id	int	NO	PRI	NULL	auto_increment

Customer Data Attributes

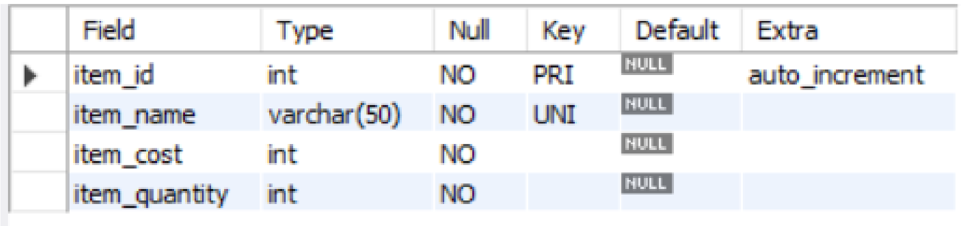
Implementation Of Customer Data in MySQL:

```
CREATE TABLE IF NOT EXISTS Customer_Data (

cust_name VARCHAR(45),

    cust_username VARCHAR(50) NOT NULL UNIQUE,

    cust_email VARCHAR(50) NOT NULL UNIQUE,

    cust_pw VARCHAR(200) NOT NULL UNIQUE,

    cust_number INT UNIQUE,

    cust_points INT DEFAULT 0,

    cust_amount_spent INT DEFAULT 0,

    user_id INT NOT NULL AUTO_INCREMENT PRIMARY KEY

);
```

Inventory Data:

The relation Inventory Data considers the Item as an entity set with the following entities attached to it:

1. Item ID: The Item ID is the primary key of the inventory data relation which helps us to identify an item in the database individually. It is set to auto increment, i.e., its value is automatically set in the database when a new record is added in the database according to the number of records which were added prior to adding it.
2. Item Name: The Item Name attribute is a string stores the name of the item in the inventory, it cannot be duplicate or NULL and qualifies as a candidate key
3. Item Cost: The Item Cost attribute is an integer which stores the cost of the item. It can be duplicate but cannot be NULL.
4. Item Quantity: The Item Quantity is an integer which stores the quantity of the item currently in stock, it can be duplicate but cannot be NULL.



	Field	Type	Null	Key	Default	Extra
▶	item_id	int	NO	PRI	NULL	auto_increment
	item_name	varchar(50)	NO	UNI	NULL	
	item_cost	int	NO		NULL	
	item_quantity	int	NO		NULL	

Inventory Data

Implementation Of Inventory Data in MySQL:

```
CREATE TABLE IF NOT EXISTS Inventory_Data (
      item_id INT AUTO_INCREMENT PRIMARY KEY NOT NULL,
    item_name VARCHAR(50) NOT NULL UNIQUE,
    item_cost INT NOT NULL,
    item_product INT NOT NULL
);
```

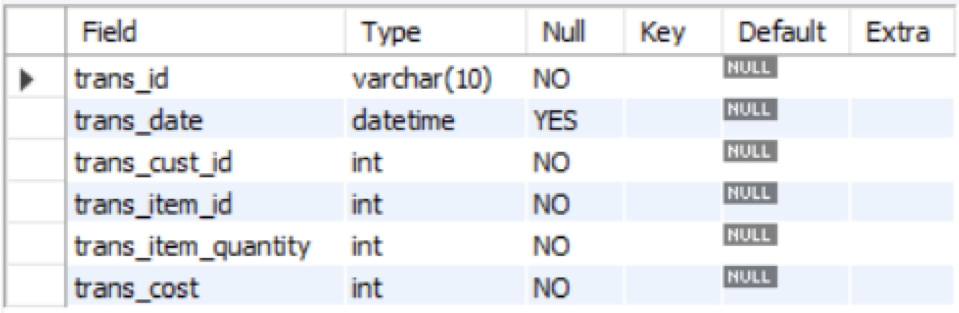
Transaction Data:

In our model, a transaction record has been defined as when the unique case when a certain unique item is bought at a certain nth time, I.e., the primary key is composite consisting of the transaction item ID and the transaction ID. Therefore, a transaction can be defined as a collection of **distinct** items being bought by a user at an instance. The reason why we can determine a transaction record using the Transaction ID and the Transaction Item ID is because in a given transaction is because all the items in a transaction are assumed distinct since their quantities are measured by a separate variable altogether.

The relation Transaction Data considers the above given description of transaction as an entity set with the following entities attached to it:

1. Transaction ID: The Transaction ID is the attribute of the transaction data relation which is a part of the composite primary key. Its value is calculated on the server side in such a way that it is ensured that it when it is combined with the Transaction Customer ID. Since it is a subset of the primary key, it cannot be NULL.
2. Transaction Date: The Transaction Date is the attribute of the Transaction which stores the time and the date of when the transaction took place. It can be duplicate but cannot be NULL.
3. Transaction Customer ID: The Transaction Customer ID is the attribute of the Transaction which is the customer ID the customer making the transaction. It is a foreign key referencing the Customer ID in the Customer Data relation. It cannot be NULL but can be duplicate.
4. Transaction Item ID: The Transaction Item ID is the attribute of the Transaction Data relation which stores the item ID of the item which the customer is purchasing in the transaction and is a part of the composite primary key. Its value is calculated on the server side in such a way that it is ensured that it when it is combined with the Transaction User ID. Since it is a subset of the primary key, it cannot be NULL but can be duplicate. It is also a foreign key referencing the Item ID attribute in the Inventory Data relation.
5. Transaction Item Quantity: The Transaction Item Quantity is the attribute of the Transaction Data relation which stores the quantity of the item which the customer is purchasing in the transaction. It cannot be NULL but can be duplicate.
6. Transaction Cost: The Transaction Cost Quantity is the attribute of the Transaction Cost relation which stores the cost of the quantity specified of the

item which the customer is purchasing in the transaction. It cannot be NULL but can be duplicate.



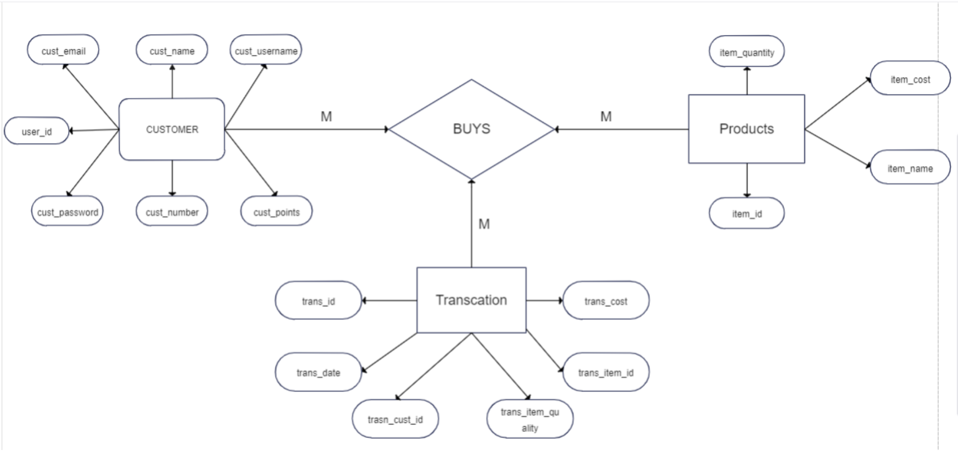
Field	Type	Null	Key	Default	Extra
trans_id	varchar(10)	NO		NULL	
trans_date	datetime	YES		NULL	
trans_cust_id	int	NO		NULL	
trans_item_id	int	NO		NULL	
trans_item_quantity	int	NO		NULL	
trans_cost	int	NO		NULL	

Transaction Data Attributes

Implementation Of Transaction Data in MySQL:

```
CREATE TABLE IF NOT EXISTS Transaction_Data (
trans_id VARCHAR(10) NOT NULL,
    trans_date DATETIME NOT NULL,
    trans_cust_id INT NOT NULL,
    trans_item_id INT NOT NULL,
    trans_item_quantity INT NOT NULL,
    trans_cost INT NOT NULL,
      FOREIGN KEY (trans_cust_id) REFERENCES Customer_Data(user_id),
      FOREIGN KEY (trans_item_id) REFERENCES Inventory_Data(item_id)
);
```

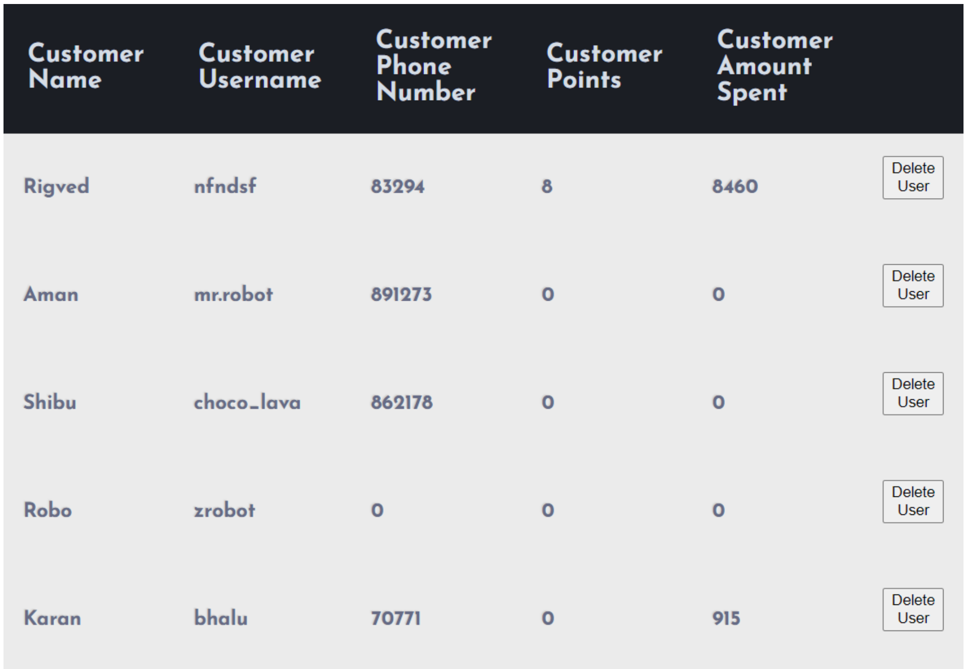
ER MODEL:



Examples of MySQL Queries to Perform Tasks:

1. Getting customer data from the database:

```
SELECT cust_name, cust_username, cust_number, cust_points, cust_amount_spent
FROM Customer_Data WHERE user_id != 1;
```

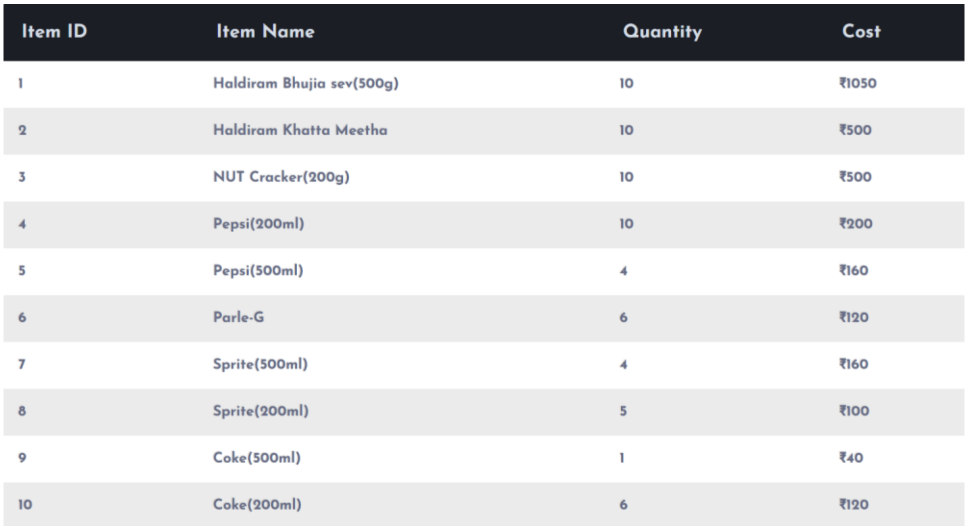


Customer Name	Customer Username	Customer Phone Number	Customer Points	Customer Amount Spent	
Rigved	nfndsf	83294	8	8460	Delete User
Aman	mr.robot	891273	0	0	Delete User
Shibu	choco_lava	862178	0	0	Delete User
Robo	zrobot	0	0	0	Delete User
Karan	bhalu	70771	0	915	Delete User

Result as Shown on the Web Portal

2. Getting Details of an individual Transaction ID:

```
SELECT transaction_data.trans_item_id, transaction_data.trans_item_quantity,
transaction_cost , Inventory_data.item_name FROM transaction_data,
inventory_data WHERE trans_id = `T14` AND inventory_data.item_id =
transaction_data.trans_item_id;
```

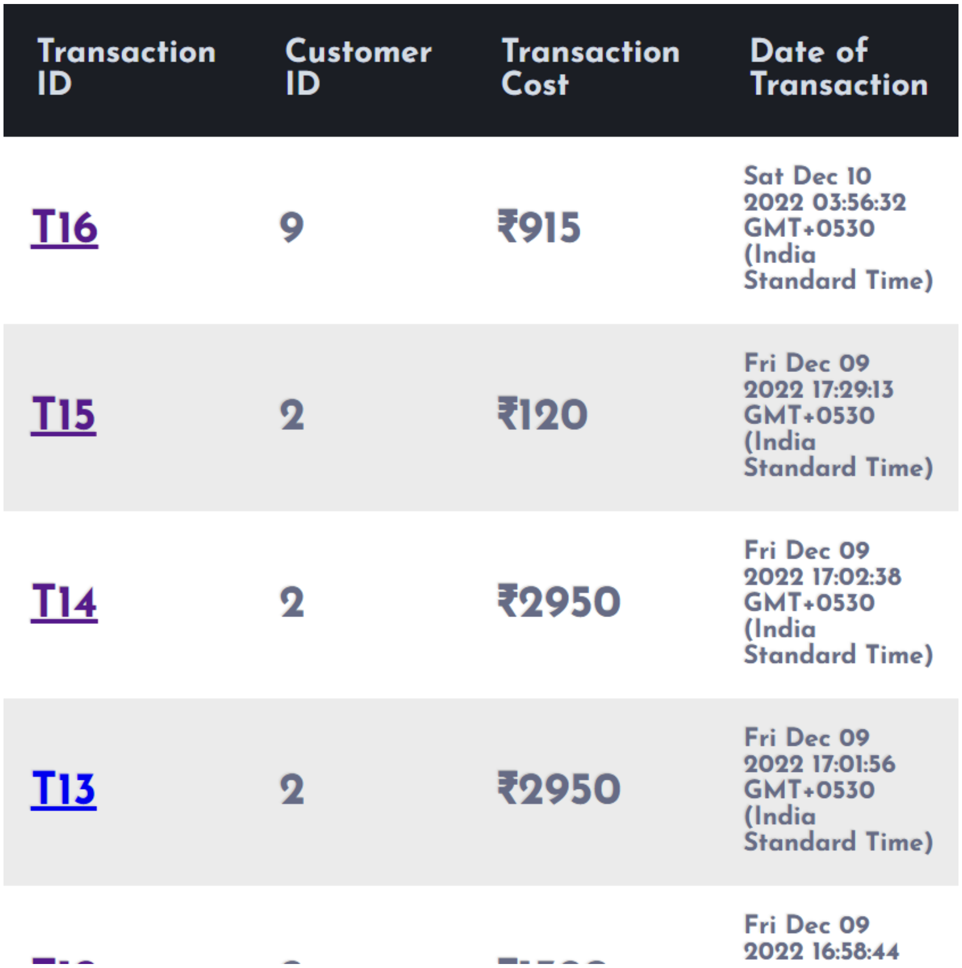


Item ID	Item Name	Quantity	Cost
1	Haldiram Bhujia sev(500g)	10	₹1050
2	Haldiram Khatta Meetha	10	₹500
3	NUT Cracker(200g)	10	₹500
4	Pepsi(200ml)	10	₹200
5	Pepsi(500ml)	4	₹160
6	Parle-G	6	₹120
7	Sprite(500ml)	4	₹160
8	Sprite(200ml)	5	₹100
9	Coke(500ml)	1	₹40
10	Coke(200ml)	6	₹120

Result as Shown on the Web Portal

3. Getting Transaction History of the Shop:

```
SELECT trans_id, trans_cust_id, SUM(trans_cost) as total_cost, trans_date
FROM Transaction_Data GROUP BY trans_id ORDER BY trans_date DESC;
```

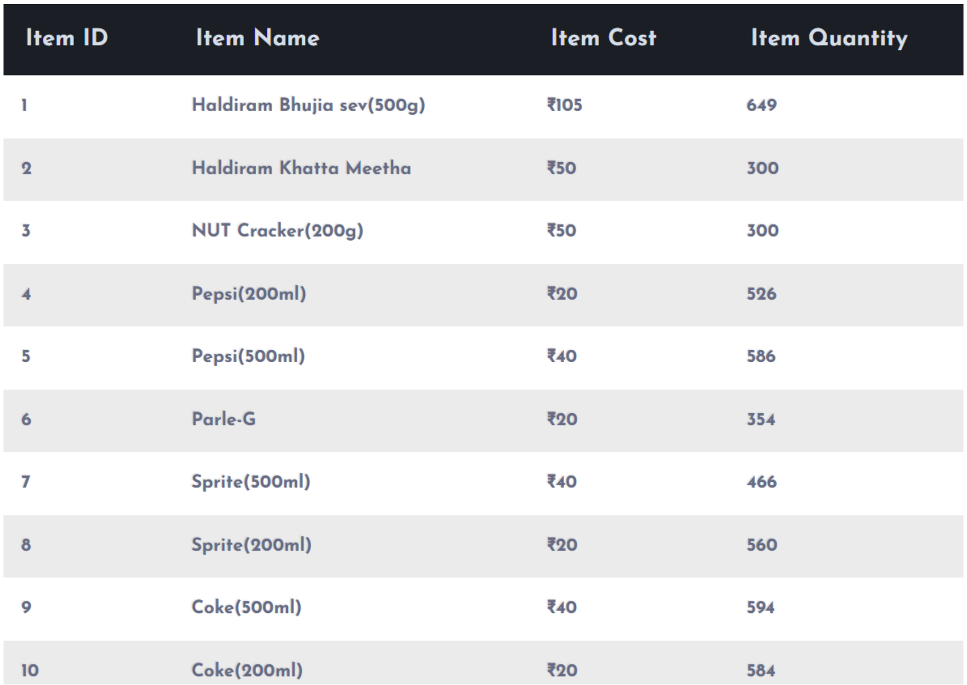


Transaction ID	Customer ID	Transaction Cost	Date of Transaction
T16	9	₹915	Sat Dec 10 2022 03:56:32 GMT+0530 (India Standard Time)
T15	2	₹120	Fri Dec 09 2022 17:29:13 GMT+0530 (India Standard Time)
T14	2	₹2950	Fri Dec 09 2022 17:02:38 GMT+0530 (India Standard Time)
T13	2	₹2950	Fri Dec 09 2022 17:01:56 GMT+0530 (India Standard Time)
			Fri Dec 09 2022 16:58:44

Result as Shown on the Web Portal

4. Getting Inventory Details of Shop:

```
SELECT * FROM Inventory_Data;
```



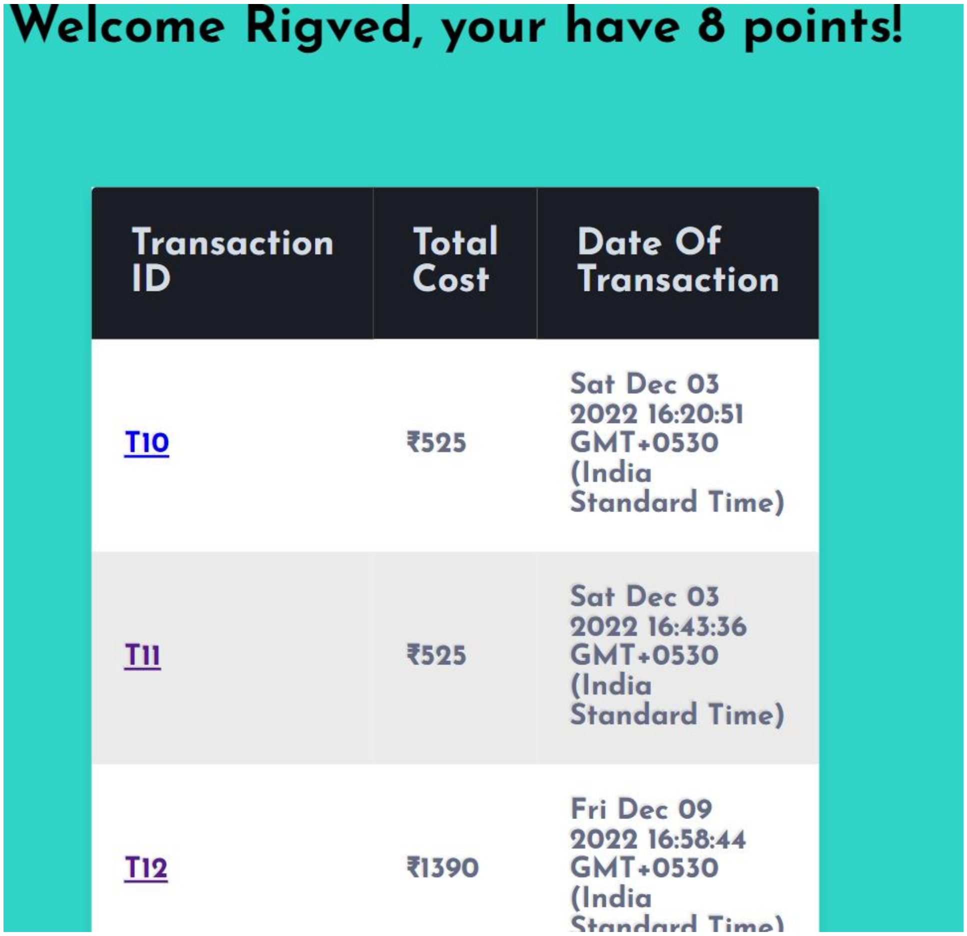
Item ID	Item Name	Item Cost	Item Quantity
1	Haldiram Bhujia sev(500g)	₹105	649
2	Haldiram Khatta Meetha	₹50	300
3	NUT Cracker(200g)	₹50	300
4	Pepsi(200ml)	₹20	526
5	Pepsi(500ml)	₹40	586
6	Parle-G	₹20	354
7	Sprite(500ml)	₹40	466
8	Sprite(200ml)	₹20	560
9	Coke(500ml)	₹40	594
10	Coke(200ml)	₹20	584

Result as Shown on the Web Portal

5. Getting point count and individual transaction details for a customer:

```
SELECT cust_points, cust_name FROM Customer_Data WHERE user_id =2;
```

```
SELECT trans_id, SUM(trans_cost) as total_cost, trans_date FROM
Transaction_Data WHERE trans_cust_id = `+ req.session.user_id + ` GROUP BY
trans_id;
```



Welcome Rigved, your have 8 points!

Transaction ID	Total Cost	Date Of Transaction
T10	₹525	Sat Dec 03 2022 16:20:51 GMT+0530 (India Standard Time)
T11	₹525	Sat Dec 03 2022 16:43:36 GMT+0530 (India Standard Time)
T12	₹1390	Fri Dec 09 2022 16:58:44 GMT+0530 (India Standard Time)

Result as Shown on the Web Portal

There are many more SQL queries in our project which are built specifically according to what users enter in the forms on the client side. We have only shown examples of those queries which are used to display something on the client side. Other examples include SQL query to add a new user's registration data into the Customer table, verifying user's login details and granting permission accordingly, adding/deleting quantity of an item in the inventory, adding new items in the inventory altogether, etc.

ANALYSIS OF DATABASE

As we previously demonstrated, the database is composed of three distinct relations. In the following analysis, we will evaluate each of these relations to determine their level of normalization and identify any potential issues or areas for improvement.

Before that, we will briefly normalization is as follows:

"*Normalization is a process in database design that aims to organize data in a manner that reduces redundancy and dependency. This helps to ensure the integrity and efficiency of the data. Normalization typically involves dividing a large, complex table into smaller, more specialized tables and defining relationships between them.*"

There are four main types of normal forms:

- First Normal Form (1NF) requires that each cell in a table contain a single, atomic value.
- Second Normal Form (2NF) requires that all non-primary attributes be dependent on the entire primary key, i.e., no partial dependencies should be present.
- Third Normal Form (3NF) requires that all non-primary attributes be dependent on the primary key, and that there are no transitive dependencies.
- Boyce-Codd Normal Form (BCNF) is a higher level of normalization than 3NF, and it requires that every determinant in a table must be a candidate key.

Each successive normal form builds upon the previous ones, adding additional constraints and requirements to ensure the data is organized in the most efficient and non-redundant way possible.

ANALYSIS OF CUSTOMER DATA:

Candidate Keys: Customer ID, Customer Username, Customer Email ID

1. This table satisfies the requirements for first normal form (1NF) because each cell in the table contains a single, atomic value. In other words, there are no repeating groups or arrays of values in any of the cells in the table, which

satisfies the requirement for 1NF that states that a table must be made up of unique, atomic values. Additionally, the table has a primary key ('Customer ID') that uniquely identifies each row in the table, further satisfying the requirements for 1NF.

2. This table satisfies the requirement for 2NF, which states that all non-primary attributes must be dependent on the entire primary key. In other words, the attributes in this table are all related to the primary key (Customer ID) and they do not contain any partial dependencies on any subset of the primary key.

3. This table satisfies the requirement for 3NF as well, since we have 3 prime attributes in the table and all the non-prime attributes can be determined from all of them.

4. Finally, this table also satisfies the requirement for BCNF, since no non-prime attribute is determining any attribute, let alone a non a prime attribute.

Hence, we can conclude that Customer Relation is in BCNF.

ANALYSIS OF INVENTORY DATA:

Candidate Key: Item ID

Dependencies: Item ID --> Item Name, Item ID --> Item Cost, Item ID --> Item Quantity

Since it is clear that all attributes in this table are dependent on a single candidate key (Item ID) and no non-prime attribute is determining a non-prime attribute, we can conclude that Inventory Data Relation is in BCNF.

ANALYSIS OF TRANSACTION DATA:

Candidate Key: (Transaction ID, Item ID)

Since there is a composite primary key in this table, which alone can determine any other attribute, and no non-prime attribute determines any other non-prime attributes, we can conclude that the Transaction Data relation is in BCNF.

RESULT

The main result of the project is the creation of a computerized system and a web portal using modern software technologies such as MySQL and Node.js for managing shop inventory, transactions, and customer loyalty schemes. This system was designed to automate the manual processes currently used by small shopkeepers in India, and provide a convenient way for both shopkeepers and customers to track purchase history and access relevant information. In the course of the project, we also studied database management systems and concluded that all of our tables are in Boyce-Codd Normal Form (BCNF).

CONCLUSION

The conclusion of the report is that the new computerized system for managing shop inventory and transactions, as well as structuring customer loyalty schemes, has been successful in meeting the goals of the project.

One of the unique aspects of our project is that we were unable to find any similar projects on the internet. This indicates that our approach to automating the current manual system and addressing the needs of both shopkeepers and customers is innovative and potentially valuable for other small shops in India and beyond.

There is also significant potential for future development and expansion of the system.

In terms of future scope, our system has the potential to be expanded and refined to include additional features and functionality, some of which include the ability for different shopkeepers to host multiple shops on the same portal. For example, it could be integrated with other systems, such as accounting software or online payment systems, to provide even greater convenience and efficiency for shopkeepers and customers. Overall, the successful implementation of this project indicates that there is significant potential for further development and growth in this area.

REFERENCES:

[1] Magatef, S. (2015) *The impact of customer loyalty programs on customer retention*, *Research Gate*. Available at: https://www.researchgate.net/profile/Sima-Magatef/publication/331833326_The_Impact_of_Customer_Loyalty_Programs_on_Customer_Retention/links/5c8f7199a6fdcc38175acc35/The-Impact-of-Customer-Loyalty-Programs-on-Customer-Retention.pdf (Accessed: December 1, 2022).

[2] Mohamad, S.N.B.S. (2016) *A study on relationship between inventory management and company ...*, *Research Gate*. Available at: https://www.researchgate.net/publication/283202157_A_Study_on_Relationship_between_Inventory_Management_and_Company_Performance_A_Case_Study_of_Textile_Chain_Store (Accessed: December 3, 2022).

[3] ABDUL ALEEM, R.A.H.M.A.T.B.E.E. (2013) *SALES AND INVENTORY MANAGEMENT SYSTEM*, *utpedia.utp.edu.m*. Available at: https://utpedia.utp.edu.my/id/eprint/13591/1/Rahmet%20Bee_13743.pdf (Accessed: December 2022).

[4] github.com : GitHub - akashjain04/Shop-Management-System: The project is developed to get some basic understanding of developing management systems using c# which can manage the small shops efficiently.

[5] GitHub - raoufswe/Customer-loyalty-Management: Customer Loyalty Management System in C

[6] https://stackoverflow.com/