

CS 5200: Database Management Systems

SUPERMARKET MANAGEMENT SYSTEM

Group Name: ChandakRTilwaniV

Group Members

Vanshita Tilwani

Rahul Chandak

Professor

Kathleen Durant

Contents

1. Introduction
2. Features
3. Technical design
 - a. Backend design
 - b. Frontend design
 - c. Database design
 - d. Libraries used
4. Database design
 - a. UML diagram
 - b. Logical design
 - c. Database components
 - i. Tables
 - ii. Procedures
 - iii. Functions
 - iv. Triggers
5. Setup/ Working
6. Flowchart
7. Conclusion
8. Limitations
9. Lessons learnt
10. Future work

1.Introduction

One of the places almost everyone can relate to daily is the ‘Supermarket’. This project aims to create a desktop-based prototype application which will simulate a Supermarket management system. The application will involve various functionalities which a consumer as well as an administrator/ employee can use and benefit from.

For the consumer, they will be able to create their own account in which they can create a new order by buying from a list of predefined items, update their existing orders, view their order history, and view a specific order they had placed in the past.

On the employee side, the application will support functionalities according to the type of employee. For regular employees, they will be able to log in their working hours (start and end time) on any specific date. For administrators, they will supervise the regular staff and will be able to register newer employees in the database not to forget can also log their own working hours. There will be one more specific type of employee called the ‘Inventory Manager’ who will be responsible for keeping track of the items sold by the supermarket. This inventory manager will decide if the supermarket should sell newer items, restock their existing items, remove/ stop the selling of a particular item.

The types of products sold by the supermarket are broadly classified into ‘Perishable’ (for example dairy products, and vegetables) and ‘Non-Perishable’ (for example Books, Toys) items.

2.Features

1. Support for CRUD operations on a local database.
2. Login option for customer and employee.
3. Customer
 - a. Create a new account
 - b. Create an order (add items and quantity)
 - c. View a specific order
 - d. View their entire order history
 - e. Update an existing order
 - f. Delete an item.
4. Employee
 - a. Support for three types of employees: Regular staff, Administrator, Inventory Manager.
 - b. Timesheet: Each employee can register their login and logout time on a specific date.
 - c. Administrators can add more employees.
 - d. Inventory manager handles the products in the supermarket. This includes:
 - i. Incorporating new set of products
 - ii. Tracking existing set of products
 - iii. Updating current stock of products
 - iv. Removing (deleting) a product from the product list
5. Products
 - a. Products are classified into categories (perishable and non-perishable). This restriction is just for demonstration purposes.

3. Technical design

A. Back-end design

- a. Programming language used: Java
- b. MVC based design technique is used to develop the full application.
- c. This ensures a sturdy design to integrate new features without affecting the existing & working functions which is extremely necessary during new product development.

B. Front-end design

- a. GUI implementation is carried out using Java swing.
- b. Uses single frame throughout the application except while throwing errors.
- c. IDE used: NetBeans, IntelliJ

C. Database Design

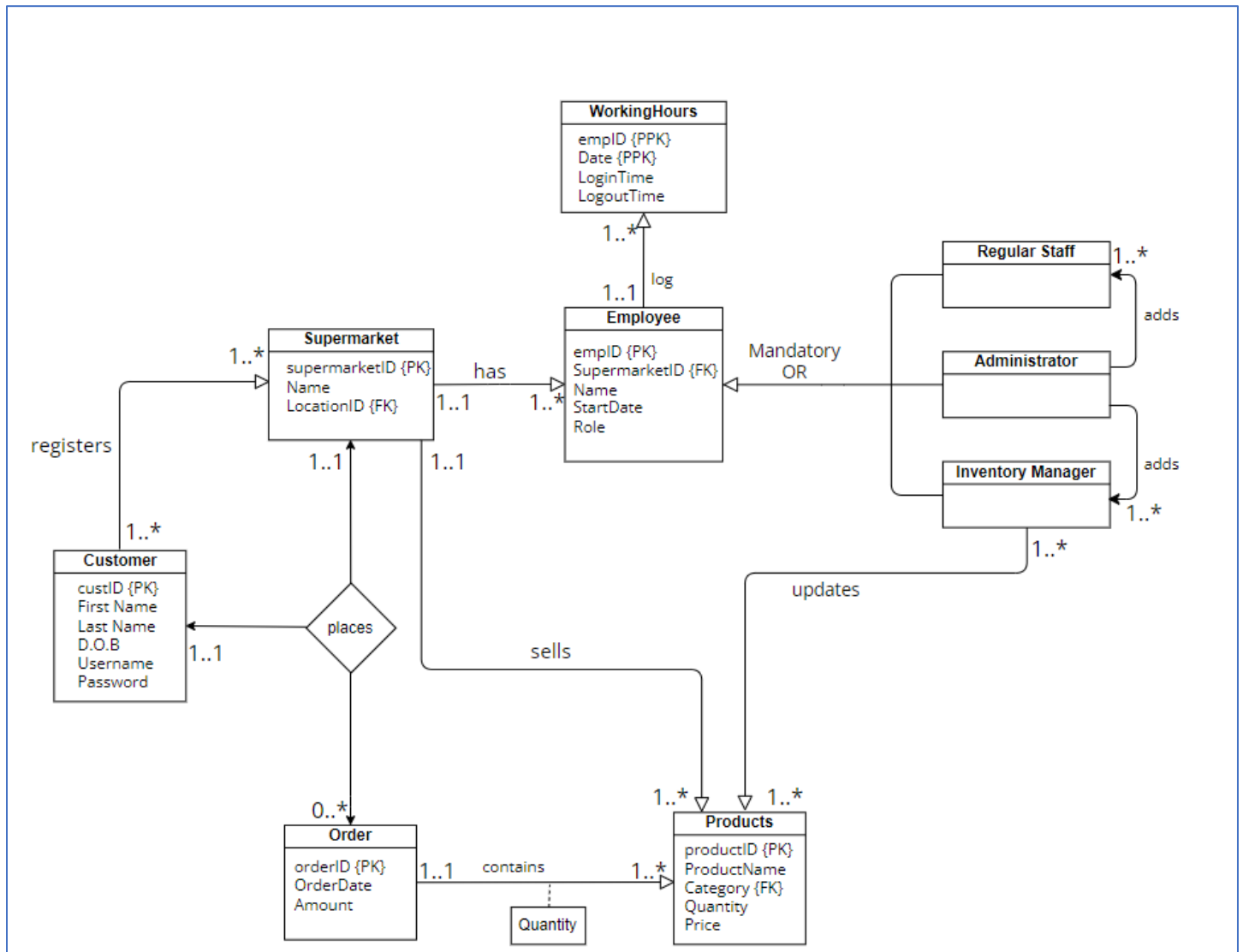
- a. DB language: MySQL
- b. DB design tool: MySQL Workbench 8.0.31

D. Libraries/ Drivers used

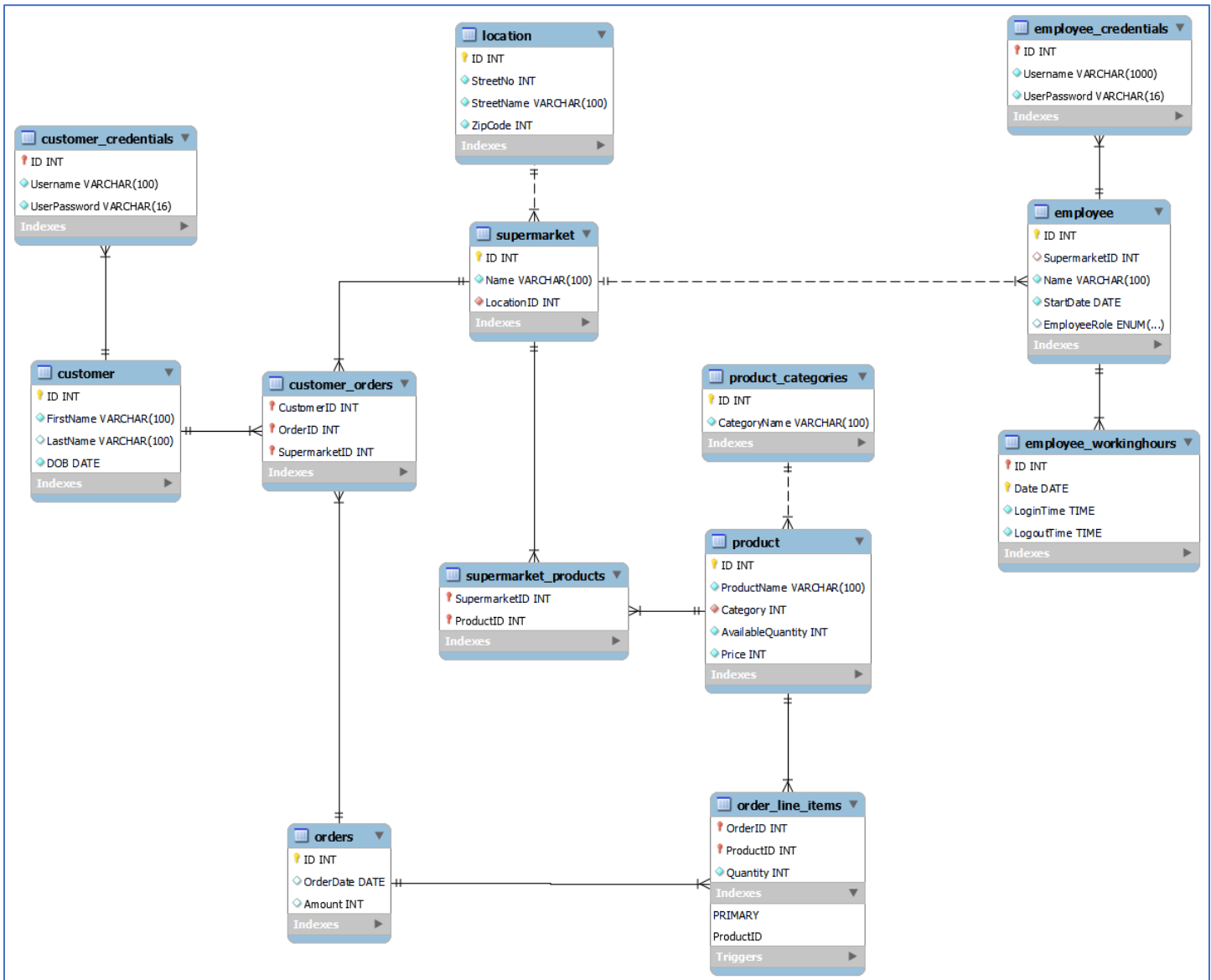
- a. Jdbc driver
- b. jDatePicker

4. Database Design

4.1 UML diagram



4.2 Logical Design (after reverse engineer)



4.3 Database components

- a. The database is broadly classified into two domains, each for the customer and the employee.
- b. Tables in 3rd normal form have been created to minimize the anomalies during update or delete of any records.
- c. Various database programming objects like procedures, functions and triggers are used to hide direct SQL queries from the code and improve the performance of the application to achieve successful CRUD database operations.

Detailed description of each component is mentioned below:

Database name: supermarketDB

A. Tables:

1. Customer

Keeps a record of the customer details who holds an account. Records include personal details like first name, last name, date of birth. A unique customer ID is allocated after each successful insertion of a customer.

2. Customer_credentials

Keeps a record of the username and password unique to each customer.

3. Location

Contains location details of a supermarket.

4. Product

List of all the products sold by the supermarket. Can be updated by the inventory manager anytime.

5. Supermarket

List of functional supermarkets. For this project we are supporting a single supermarket.

6. Supermarket_products

List of products sold at the supermarket.

7. Order_line_items

Keeps a record of all the orders placed by all the customers. Each combination of the customer id and order id will be unique in this table.

8. Customer_orders

Contains data of each order placed by each customer at a specific supermarket.

9. Orders

Keeps records of each order placed at the supermarket. Consists of the date the order was placed and the amount of each order. Can be used by the administrators to track all the orders at once if needed.

10. Product_Categories

Master list for the type of products sold by the supermarket. Have been limited to 'Perishable' and 'Non-Perishable' items.

11. Employee

Keeps a record of the employee's details like name, date of birth, role. Auto generates a unique employee id for every new employee inserted in this table.

12. Employee_workinghours

Data in this table reflects the login-logout times on specific dates. Has an 'ID' field which references the employee id.

13. Employee_Credentials

Keeps a record of the username and password unique to each employee.

B. Procedures:

1. addNewProduct

Whenever the inventory manager adds new products to be sold at the supermarket, this procedure updates the 'product' table.

2. addOrderItem

Add order details in order_line_items table. Contains the products and their quantities in an order.

3. addWorkHours

Called to add the date and login-logout times of all employees.

4. createEmployee

If the administrator adds a new employee, this procedure is called which takes the name, date of birth, role, username, and password as parameters and after validations, adds it to the employee and employee_credentials table respectively

5. deleteEmployee

Called to delete an employee's records from the database by the administrator.

6. getAllAvailableProducts

Called to get all the products with their quantities > 0 for inventory manager's reference.

7. getAllProducts

Called to get all the products sold at the supermarket.

8. `getEmployeeWorkingHours`

Returns a combined detail of employee details and their working hours which can be tracked by the administrator.

9. `orderDetails`

If the user requests for the details of a specific order, this procedure is called

10. `orderHistory`

Takes username as an input and joins multiple tables to return all the orders the consumer has placed from this specific account.

11. `Register`

This procedure is called if a customer tries to create a new account on the application. A check for unique username is made before inserting data in the 'Customers' table

12. `updateOrderItem`

Updates the order items

13. `updateProductQuantity`

Called to update the product quantity in product table in cases where new order is created, number of items in old order is updated or inventory manager buys more stock.

C. Functions:

1. `isValidEmployee:`

To check if the employee the administrator is trying to add already exists in the database or not. Validation happens on the employee ID.

2. `getEmployeeRole`

Returns the type (Admin, Inventory Manager or Regular) of employee.

3. IsValidUser

To check if the customer's new account already exists in db or not. Validation happens on the username, which should be unique for each account.

4. createOrder

Used to assign the current date to any new order being created.

D. Triggers:

1. on_order_create

This trigger will work after inserting it on order_line_items. Basically, whenever a user buys some products, the quantity bought should be subtracted from the inventory (product table) automatically.

2. on_order_delete

This trigger will work after deleting order_line_items. Whenever a user cancels any order, the quantity bought should be added back to the inventory (product table) automatically.

3. on_order_update

This trigger will work after update on order_line_items. Basically, whenever a user tries to edit the quantity of the products they have already bought this should also be reflected in the inventory (product table) automatically.

5. Setup/ Working:

The working of the entire project is explained in the below steps:

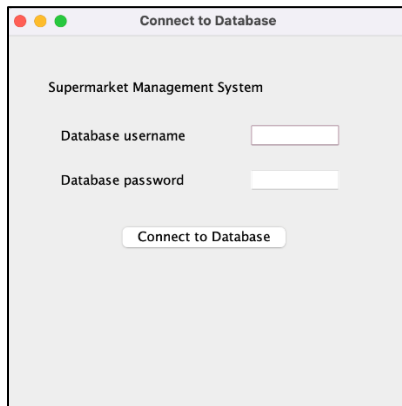
The setup is broadly classified into 3 parts: Setup, consumer, and employee.

Part 1: Setup

1. Execute the dump provided in the database design tool on your local server. (It will also have some dummy data for reference).
2. There is a .jar file provided in the 'Runnable' folder. Open terminal and navigate to this path. Enter the following command:

```
java -jar CS5200.jar
```

3. Since we are not hosting our database on some web server, records are stored locally. Hence, as soon as the application starts, the user must connect to the database by giving the appropriate database credentials.



The screenshot shows a window titled "Connect to Database". Inside the window, the text "Supermarket Management System" is displayed. Below this, there are two input fields: "Database username" and "Database password". At the bottom of the window, there is a button labeled "Connect to Database".

Part 2: The Consumer

1. Select consumer from the below options. A panel with sign up and log in tab will pop up.

Supermarket System - Login/Sign up Options

Back

Select the supermarket to order from

Supermarket1

Select the login option

☐ Customer

☐ Employee

Submit

2. If a new user, go to the 'Sign Up' tab, and the user will have to enter all the details and click submit. If a user already has an account (username provided is not unique), the new account creation will be denied. This is one of the 'Create and Read' operations the application handles.
3. If an existing user, go to the 'Log In' tab and enter the username and password.

Customer Login/Sign Up Page

Back

Customer Login Page

Sign up Login Form

Username

First Name

Last Name

Date of Birth Dec 9, 2022 ...

Password

Submit Clear

Customer Login/Sign Up Page

Back

Customer Login Page

Sign up Login Form

Username

Password

Log in Clear

4. Once login is successful, A new window with various CRUD operations will pop up. Navigate to the appropriate tabs to perform various operations.
5. To create a new order, the user will be displayed with a set of available items at the supermarket. The user will have to enter the product name and the quantity they want to buy. They can keep track of their cart in the window at the bottom right.

Main Menu - Customer

Logout

Welcome

Add Order Order History Order Details Order Update

ID	Name	Quantity	Category	Price
1	Apple	11	Perishable	2
2	Mango	10	Perishable	2
3	Grapes	18	Perishable	4
4	Milk	2	Perishable	5
5	Books	4	Non Peri...	5
6	Pen	8	Non Peri...	5
7	Cap	24	Non Peri...	20
8	Laptop	22	Non Peri...	100
9	Oranges	20	Perishable	10

Product ID: 1

Enter quantity: 0

- Once all items are added, click the 'create' button and your order is successfully placed.
- The 'order history' tab will display all the orders placed till date using the logged in account.

Main Menu - Customer

Logout

Welcome

Add Order Order History Order Details Order Update

Order ID	Order Date	Order Amount
1	2022-08-01	112.0
2	2022-01-01	10.0

- The 'view specific order' and 'update order' tabs both take order id and perform the said operations on the contents tagged to this specific order id.

Main Menu - Customer

Logout

Welcome

Add Order Order History Order Details Order Update

Order ID: 1

Submit Clear

ID	Name	Quantity	Category	Price
1	Apple	4	Perishable	8
2	Mango	7	Perishable	10
3	Grapes	7	Perishable	28
6	Pen	2	Non Peri...	10

Save

Main Menu - Customer

Logout

Welcome

Add Order Order History Order Details Order Update

Order ID: 1

Submit Clear

ID	Name	Quantity	Category	Price
1	Apple	4	Perishable	8
2	Mango	5	Perishable	10
3	Grapes	7	Perishable	28
6	Pen	2	Non Peri...	10

Part 3: The employee

1. After step 2 of 'Part 1', select employee. Anyone cannot create an employee account, hence only login option would be available.
2. After valid credentials are entered, the application is designed to display the appropriate window according to the type of the employee (normal, administrator or inventory manager).
3. For a regular employee, they can enter their login-logout times and date only.

Employee - Main Page

Logout

Welcome jerryroy

Date Dec 9, 2022 ...

Log in Time 16:31:00

Log out Time 16:31:00

Submit

4. The administrator account will allow adding more employees to the system. Thus, the administrator can choose to log his 'in' and 'out' times or can choose to add a new staff member to the employee list.

Admin Employee Main Page

Logout

Welcome johndavid

Add Staff Employee Working Hours Log Hours

Username

Name

Hire Date Dec 9, 2022 ...

Role Regular

Password

Add Staff

Admin Employee Main Page

Logout

Welcome johndavid

Add Staff Employee Working Hours Log Hours

ID	Name	Start Date	Start Time	End Time
1	Mark	2021-01-01	09:00:00	17:00:00
1	Mark	2021-02-03	09:00:00	17:00:00
1	Mark	2021-06-05	09:00:00	17:00:00
2	John	2021-02-03	09:00:00	17:00:00
2	John	2021-02-04	09:00:00	17:00:00
3	Steve	2022-01-01	09:00:00	17:00:00
3	Steve	2022-01-02	09:00:00	17:00:00
3	Steve	2022-02-02	09:00:00	17:00:00
4	Chris	2021-06-05	09:00:00	17:00:00
5	Christ...	2021-02-04	09:00:00	17:00:00

5. The inventory manager has more number features to handle. These again include CRUD operations like adding new products to the master list, viewing existing item list, update existing stock of items and deleting some products indicating that the supermarket doesn't sell this product anymore. Finally, the inventory manager can also choose to just login his work hours for the day.

The screenshot shows the 'Inventory Manager - Main Page' window. At the top left is a 'Logout' button. Below it is a welcome message 'Welcome chrisjoseph'. A tab bar contains 'Add new Items' (selected), 'Item list', and 'Update Product Stock'. The form includes fields for 'Enter item name', 'Enter quantity' (with a spinner set to 0), 'Select Category' (a dropdown menu showing 'Perishable'), and 'Price' (a spinner set to 0). An 'Add Product' button is at the bottom.

Add new items

The screenshot shows the 'Update Product Stock' tab selected. The form contains 'Product ID' (a text field with '5' and a dropdown arrow) and 'Quantity' (a spinner set to '10'). 'Submit' and 'Clear' buttons are at the bottom.

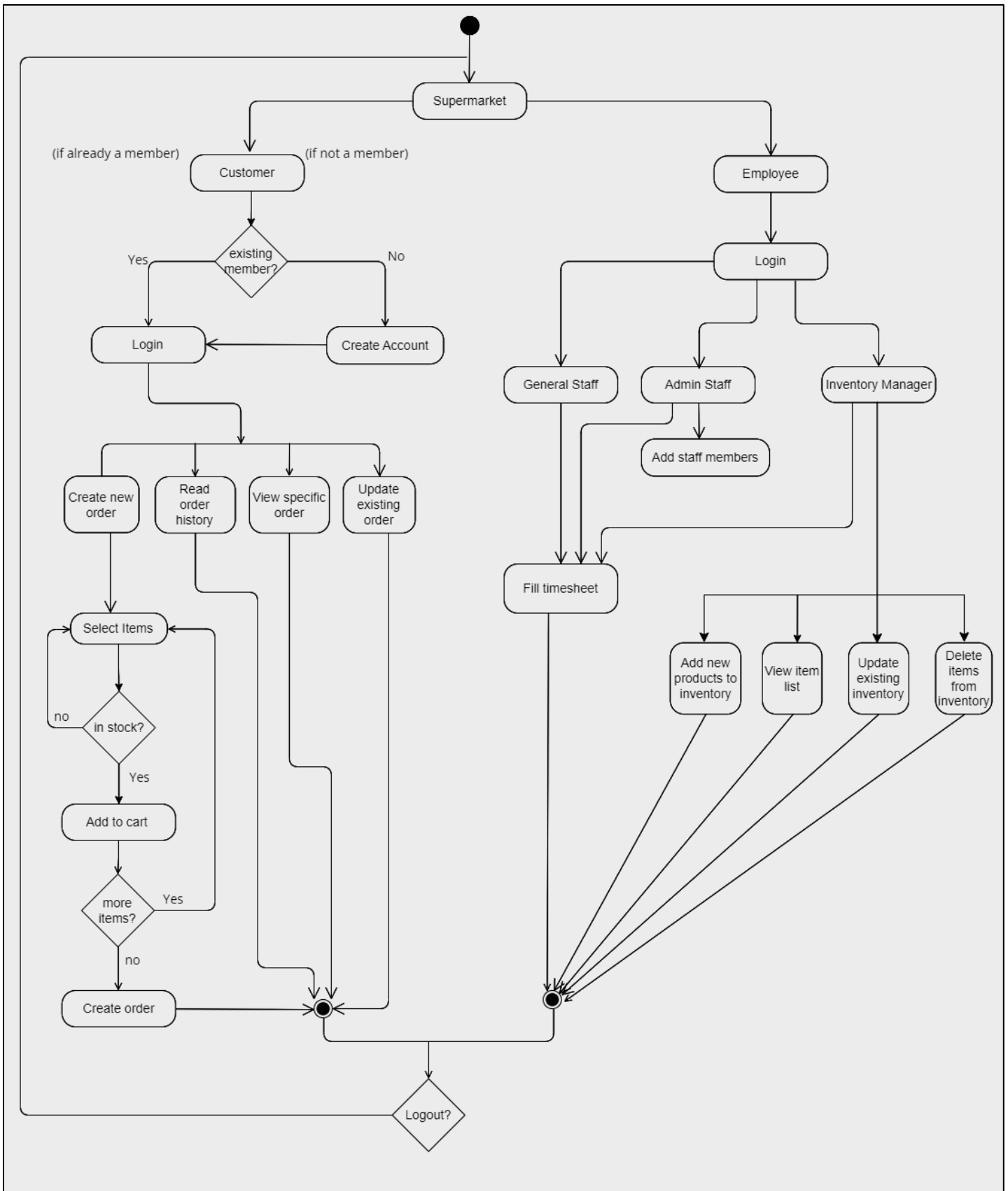
Update existing stock items

The screenshot shows the 'Delete Items' tab selected. The form contains 'Product ID' (a text field with '5' and a dropdown arrow). 'Submit' and 'Clear' buttons are at the bottom.

Delete stock items

The user at any point of time can logout with a 'logout' button present at the top left of the frame.

6. Flowchart



7. Conclusion

- a. Hence an attempt was made to simulate a supermarket management system, which can help demonstrate the use of CRUD commands using database programming objects and thus ensuring minimalistic exposure of the sql queries in the code.
- b. The focus of the project was on database design, although instead of command line arguments, GUI was also implemented as an attempt to make the application a bit more interactive.

8. Lessons Learnt

Technical expertise/ Insights gained

- a. Designing a relational database management system with its components in 3rd NF.
- b. SQL commands to create complex queries to fetch data from different tuples from multiple components.
- c. Working with database programming objects procedures, functions and triggers instead of using select and update queries everywhere.
- d. Handling different types of keys and implementing constraints on foreign keys.
- e. GUI implementation using Java Swing.
- f. MVC code design architecture.

Realized or contemplated alternative design / approaches to the project

- a. While creating an order, the customer gives the number of items of a product they want. The 'order' table will reflect the quantity bought but at the same time the number of items should be reduced on the 'product' table as well, which tells the overall available quantity. Our approach missed this corner case initially, but later we fixed it by using a trigger.
- b. We realized that we could've handled multiple supermarkets as well.
- c. To get the order history, we could've also used a view as an alternative approach.

9. Limitations

1. This application doesn't include support for multiple supermarkets.
2. Payment functionality is not developed while creating an order.
3. Since, major focus was on database design, the front-end design lacks efficiency and could be improved in various domains.

10. Future Scope

The project is open to a lot of improvements. Some of them are listed below:

1. Payment functionality while creating or updating an order.
2. Inventory managers can observe trends and declare a list of items which are in high demand. This will ensure that these in-demand items never run out of stock. (Data visualization)