

CS166 PROJECT

GROUP 79, Rongzhong Ye, 862326647

March 2024

1 Implementation Description

A system to simulate how a customer has a basic shopping experience, a manager manages his store, and also an administrator how to admin database of user and merchants in Amazon.

1. View Stores within 30 miles

```
MAIN MENU
-----
1. View Stores within 30 miles
2. View Product List
3. Place a Order
4. View 5 recent orders
5. Update Product
6. View 5 recent Product Updates Info
7. View 5 Popular Items
8. View 5 Popular Customers
9. Place Product Supply Request to Warehouse
10. Admin System.
.....
20. Log out
21. Exist the system
Please make your choice: 1
Enter your latitude: 22
Enter your longitude: 22
Stores within a 30.0-mile radius:
StoreID: 3, Latitude: 9.255290, Longitude: 14.251070, Date Established: 1938-12-30
StoreID: 10, Latitude: 9.746990, Longitude: 17.303860, Date Established: 1908-09-11
StoreID: 11, Latitude: 17.009880, Longitude: 10.882990, Date Established: 1998-05-19
StoreID: 13, Latitude: 3.444470, Longitude: 28.841020, Date Established: 1926-12-05
StoreID: 14, Latitude: 21.924100, Longitude: 22.354760, Date Established: 1966-10-17
```

Figure 1: Caption for the figure

- (a) QUERY:

```
String query = "SELECT s.storeID, s.latitude, s.longitude, s.dateEstablished " +
               "FROM Store s";
```

- (b) This Query looks for by provided longitude, latitude, pick up the satisfied Store's information of table Store.

2. View Product List

```
MAIN MENU
-----
1. View Stores within 30 miles
2. View Product List
3. Place a Order
4. View 5 recent orders
5. Update Product
6. View 5 recent Product Updates Info
7. View 5 Popular Items
8. View 5 Popular Customers
9. Place Product Supply Request to Warehouse
10. Admin System.
.....
20. Log out
21. Exist the system
Please make your choice: 2
Enter the storeID: 11
Products for Store 11:
Product Name      Number of Units  Price per Unit
7up                50              $3.00
Pepsi              21              $4.00
Lemonade           42              $8.00
Brisk              27              $3.00
Orange Juice       45              $6.00
Donuts             37              $7.00
Pudding            13              $3.00
Ice Cream          33              $6.00
Hot and Sour Soup  43              $5.00
Egg                15              $3.00
```

Figure 2: Caption for the figure

(a) QUERY:

```
String query = "SELECT p.productName, p.numberOfUnits, p.pricePerUnit " +
               "FROM Product p " +
               "WHERE p.storeID = " + storeID;
```

(b) This Query looks for productName, numberOfUnits, pricePerUnit in table Product if provided the storeID.

3. Place an Order

```

MAIN MENU
-----
1. View Stores within 30 miles
2. View Product List
3. Place a Order
4. View 5 recent orders
5. Update Product
6. View 5 recent Product Updates Info
7. View 5 Popular Items
8. View 5 Popular Customers
9. Place Product Supply Request to Warehouse
10. Admin System.
.....
20. Log out
21. Exist the system
Please make your choice: 3
Enter your latitude: 22
Enter your longitude: 22
Enter your name: rye014
Enter the storeID: 11
Enter the product name: 7up
Enter the number of units: 2
Order placed successfully!

```

Figure 3: Caption for the figure

(a) QUERY:

```

String query = "SELECT userID FROM Users
WHERE name = '" + userName + "'";
query = "SELECT latitude, longitude
FROM Store WHERE storeID = " + storeID;
String query = "SELECT p.productName, p.numberOfUnits,
p.pricePerUnit " +
"FROM Product p " +
"WHERE p.storeID = " + storeID;
query = "INSERT INTO Orders (customerID, storeID, productName, unitsOro

```

(b) User should provide his name, required delivery address and required storeID, then getting the required orders information, update to table Orders.

4. View 5 recent orders

```
MAIN MENU
-----
1. View Stores within 30 miles
2. View Product List
3. Place a Order
4. View 5 recent orders
5. Update Product
6. View 5 recent Product Updates Info
7. View 5 Popular Items
8. View 5 Popular Customers
9. Place Product Supply Request to Warehouse
10. Admin System.
.....
20. Log out
21. Exist the system
Please make your choice: 4
Enter your name: rye014
Your Recent Orders:
Order Number  Store ID  Product Name  Units Ordered  Order Time
502           11       7up           1             2024-03-13 15:51:44.329894
501           11       7up           2             2024-03-13 14:01:32.215671
```

Figure 4: Caption for the figure

(a) QUERY:

```
query = "SELECT o.orderNumber, s.storeID, o.productName, o.unitsOrdered,
        o.orderTime " +
        "FROM Orders o " +
        "JOIN Product p ON o.storeID = p.storeID AND o.productName =
        p.productName " +
        "JOIN Store s ON p.storeID = s.storeID " +
        "WHERE o.customerID = " + userID + " " +
        "ORDER BY o.orderTime DESC LIMIT 5";
```

(b) The query join three tables, orders, product, and store by looking for provided userID, then it is not null, it shows o.orderNumber, s.storeID, o.productName, o.unitsOrdered, o.orderTime limit 5.

5. Update Product

```
MAIN MENU
-----
1. View Stores within 30 miles
2. View Product List
3. Place a Order
4. View 5 recent orders
5. Update Product
6. View 5 recent Product Updates Info
7. View 5 Popular Items
8. View 5 Popular Customers
9. Place Product Supply Request to Warehouse
10. Admin System.
.....
20. Log out
21. Exist the system
Please make your choice: 5
Enter your ManagerID: 87
Enter the storeID: 11
Enter the product name: 7up
Enter the new number of units: 2000
Enter the new price per unit: 1
Product information updated successfully!
```

Figure 5: Caption for the figure

(a) QUERY:

```
String query = "SELECT storeID FROM Store WHERE managerID = " + managerID;
query = "SELECT * FROM Product WHERE storeID = " + storeID + " AND productName = " + productName;
query = "UPDATE Product SET numberOfUnits = " + numberOfUnits + ", pricePerUnit = " + pricePerUnit;
query = "INSERT INTO ProductUpdates (managerID, storeID, productName, numberOfUnits, pricePerUnit) VALUES (" + managerID + ", " + storeID + ", " + productName + ", " + numberOfUnits + ", " + pricePerUnit + ")";
```

(b) This query first selects the storeID from the Store table where the managerID matches the provided managerID. Then, it selects all columns from the Product table where the storeID matches the provided storeID and the productName matches the provided productName. Next, it updates the numberOfUnits and pricePerUnit columns in the Product table where the storeID matches the provided storeID

and the productName matches the provided productName. Finally, it inserts a new row into the ProductUpdates table with the provided managerID, storeID, productName, and the current timestamp.

6. View 5 recent Product Updates Info

```

MAIN MENU
-----
1. View Stores within 30 miles
2. View Product List
3. Place a Order
4. View 5 recent orders
5. Update Product
6. View 5 recent Product Updates Info
7. View 5 Popular Items
8. View 5 Popular Customers
9. Place Product Supply Request to Warehouse
10. Admin System.
.....
20. Log out
21. Exist the system
Please make your choice: 6
Enter your ManagerID: 87
+-----+-----+-----+-----+-----+
| Store ID | Product Name | New Units | New Price | Update Time |
+-----+-----+-----+-----+-----+
| 11       | 7up          |           | 2000      | 2024-03-13 16:07:26.417016 |
| 11       | Brisk        |           | 27        | 2016-09-10 13:43:00 |
| 11       | Lemonade     |           | 42        | 2016-09-10 13:33:00 |
| 11       | Pepsi        |           | 21        | 2016-09-10 13:17:00 |
| 11       | 7up          |           | 2000      | 2016-09-10 13:09:00 |

```

Figure 6: Caption for the figure

(a) QUERY:

```

query = "SELECT s.storeID, u.productName, p.numberofUnits AS newUnits, p.pr
        "FROM ProductUpdates u " +
        "JOIN Store s ON u.storeID = s.storeID " +
        "JOIN Product p ON u.storeID = p.storeID AND u.productName
        = p.productName " +
        "WHERE s.storeID IN (" + storeIDList + ") AND u.managerID
        = " + managerID + " " +
        "ORDER BY u.updatedOn DESC LIMIT 5";

```

(b) The query joins three tables store, product and productupdates and seeking the update ite by desc updatedon limit 5.

7. View 5 Popular Items

```
MAIN MENU
-----
1. View Stores within 30 miles
2. View Product List
3. Place a Order
4. View 5 recent orders
5. Update Product
6. View 5 recent Product Updates Info
7. View 5 Popular Items
8. View 5 Popular Customers
9. Place Product Supply Request to Warehouse
10. Admin System.
.....
20. Log out
21. Exist the system
Please make your choice: 7
Enter your ManagerID: 87
Top 5 Popular Products in Your Stores:
Product Name      Order Count
Pepsi              7
Pudding            6
7up                6
Ice Cream          6
Brisk              6
```

Figure 7: Caption for the figure

(a) QUERY:

```
String query = "SELECT storeID FROM Store WHERE managerID = " + managerID;
query = "SELECT p.productName, COUNT(o.orderNumber) AS orderCount " +
        "FROM Product p " +
        "JOIN Orders o ON p.storeID = o.storeID AND p.productName = o.productName " +
        "WHERE p.storeID IN (" + storeIDList + ") " +
        "GROUP BY p.productName " +
        "ORDER BY orderCount DESC LIMIT 5";
```

(b) This query looks for storeID, then joins table product and orders by knowned storeid to get the required information limit 5.

8. View 5 Popular Customers

```

MAIN MENU
-----
1. View Stores within 30 miles
2. View Product List
3. Place a Order
4. View 5 recent orders
5. Update Product
6. View 5 recent Product Updates Info
7. View 5 Popular Items
8. View 5 Popular Customers
9. Place Product Supply Request to Warehouse
10. Admin System.
.....
20. Log out
21. Exist the system
Please make your choice: 8
Enter your ManagerID: 87
Top 5 Customers Who Placed the Most Orders in Your Stores:
Name      Latitude      Longitude      Type      Order Count
Garrett    67.428760     91.785130     customer   2
rye014     22.000000     22.000000     Customer   2
Christa    48.637420     82.642730     customer   2
Delpha.Hand 72.145910     12.841660     customer   2
Ellis      55.358500     46.315520     customer   2

```

Figure 8: Caption for the figure

(a) QUERY:

```

String query = "SELECT storeID FROM Store WHERE managerID = " + managerID;
query = "SELECT u.name, u.latitude, u.longitude, u.type, COUNT(o.orderID) " +
        "FROM Users u " +
        "JOIN Orders o ON u.userID = o.customerID " +
        "JOIN Product p ON o.storeID = p.storeID AND o.productName = p.productName " +
        "WHERE p.storeID IN (" + storeIDList + ") " +
        "GROUP BY u.userID " +
        "ORDER BY orderCount DESC LIMIT 5";

```

(b) The query get storeID first, then join tables users, orders, product by known storeID group by userID, order by dsc ordercount limit 5.

9. Place Product Supply Request to Warehouse


```
MAIN MENU
-----
1. View Stores within 30 miles
2. View Product List
3. Place a Order
4. View 5 recent orders
5. Update Product
6. View 5 recent Product Updates Info
7. View 5 Popular Items
8. View 5 Popular Customers
9. Place Product Supply Request to Warehouse
10. Admin System.
.....
20. Log out
21. Exist the system
Please make your choice: 9
Enter your ManagerID: 87
Enter the storeID: 11
Enter the product name: 7up
Enter the number of units needed: 5
Enter the warehouseID: 2
Product supply request placed successfully!
```

Figure 9: Enter Caption

(a) QUERY:

```
String query = "SELECT storeID FROM Store WHERE managerID = " + managerID;
query = "SELECT * FROM Product WHERE storeID = " + storeID + " AND productName = " + productName;
query = "INSERT INTO ProductSupplyRequests (managerID, warehouseID, storeID, productName, units) "
        + "VALUES (" + managerID + ", " + warehouseID + ", " + storeID + ", '" + productName + "', " + units + ")";
query = "UPDATE Product SET numberOfUnits = numberOfUnits + " + units + " WHERE storeID = " + storeID + " AND productName = '" + productName + "'";
```

(b) The query needs to get storeID, find the product information, then get required update information, using update to insert them in table productSupplyRequests, and more important is update them in table product.

10. Admin System.

```

MAIN MENU
-----
1. View Stores within 30 miles
2. View Product List
3. Place a Order
4. View 5 recent orders
5. Update Product
6. View 5 recent Product Updates Info
7. View 5 Popular Items
8. View 5 Popular Customers
9. Place Product Supply Request to Warehouse
10. Admin System.
.....
20. Log out
21. Exist the system
Please make your choice: 10
Enter your AdminID: Admin
Admin Menu:
1. View all users
2. Update user information
3. View all products
4. Update product information
5. Quit
Enter your choice: █

```

Figure 10: Enter Caption

(a) QUERY:

```
String query = "SELECT * FROM Users WHERE name = '" + userIDInput + "'"
```

(b) The query needs to find AdminId to see if the user can be Admin, if it can be, then access to use Admin function.

11. EXTRA - TRIGGER

```

CREATE OR REPLACE FUNCTION store_logged_in_user() RETURNS TRIGGER AS $$
BEGIN
    DELETE FROM LoggedInUser;
    INSERT INTO LoggedInUser (userID, name)
    VALUES (NEW.userID, NEW.name);
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER store_logged_in_user_trigger
AFTER INSERT ON Users
FOR EACH ROW
EXECUTE PROCEDURE store_logged_in_user();

```

Figure 11: triggers.sql

```

#!/bin/bash
DIR="$( cd "$( dirname "${BASH_SOURCE[0]}" )" && pwd )"

# Create the database
cs166_createdb $USER"_project_phase_3_DB"

# Create tables
cs166_psql -p $PGPORT $USER"_project_phase_3_DB" < $DIR/../src/create_tables.sql

# Create indexes
cs166_psql -p $PGPORT $USER"_project_phase_3_DB" < $DIR/../src/create_indexes.sql

# Load data
cs166_psql -p $PGPORT $USER"_project_phase_3_DB" < $DIR/../src/load_data.sql

```

Figure 12: createdb.sh

```

CREATE TABLE LoggedInUser (
    userID INT PRIMARY KEY,
    name VARCHAR(50),
    FOREIGN KEY (userID) REFERENCES Users(userID)
);

```

Figure 13: createtable.sql

```
Connecting to database...Connection URL: jdbc:postgresql://localhost:24525/rye014
_project_phase_3_DB

Done
Welcome back, rye014!

  .-''''''-.
 / \       \
| \   \   / |
 \   '---'  /
  '-.....-'

MAIN MENU
-----
1. View Stores within 30 miles
2. View Product List
3. Place a Order
4. View 5 recent orders
5. Update Product
6. View 5 recent Product Updates Info
7. View 5 Popular Items
8. View 5 Popular Customers
9. Place Product Supply Request to Warehouse
10. Admin System.
.....
20. Log out
21. Exist the system
Please make your choice: 1
```

Figure 14: Enter Caption

- (a) By using the trigger, design an auto log-in function. To satisfy it, we need a new table for storing the userID information, when the system runs, checking the database exists the userid. If it exists, the user does not need to login in again.

2 Problems/Findings

1. has a lot of Java syntax issues and logic issues(major issue), such as string convert to int, how to identify to satisfy trigger on user auto log in
2. Not familiar with JAVA's list
3. Finally clarify we can split query into multiple parts to implement the required functions

3 Contributions

JOBS ALL DONE BY RONGZHONG YE