Criterion B: Solution Overview (Design)

Hierarchal Structure diagram of the system:

This displays how different interfaces would be linked together in the proposed program.

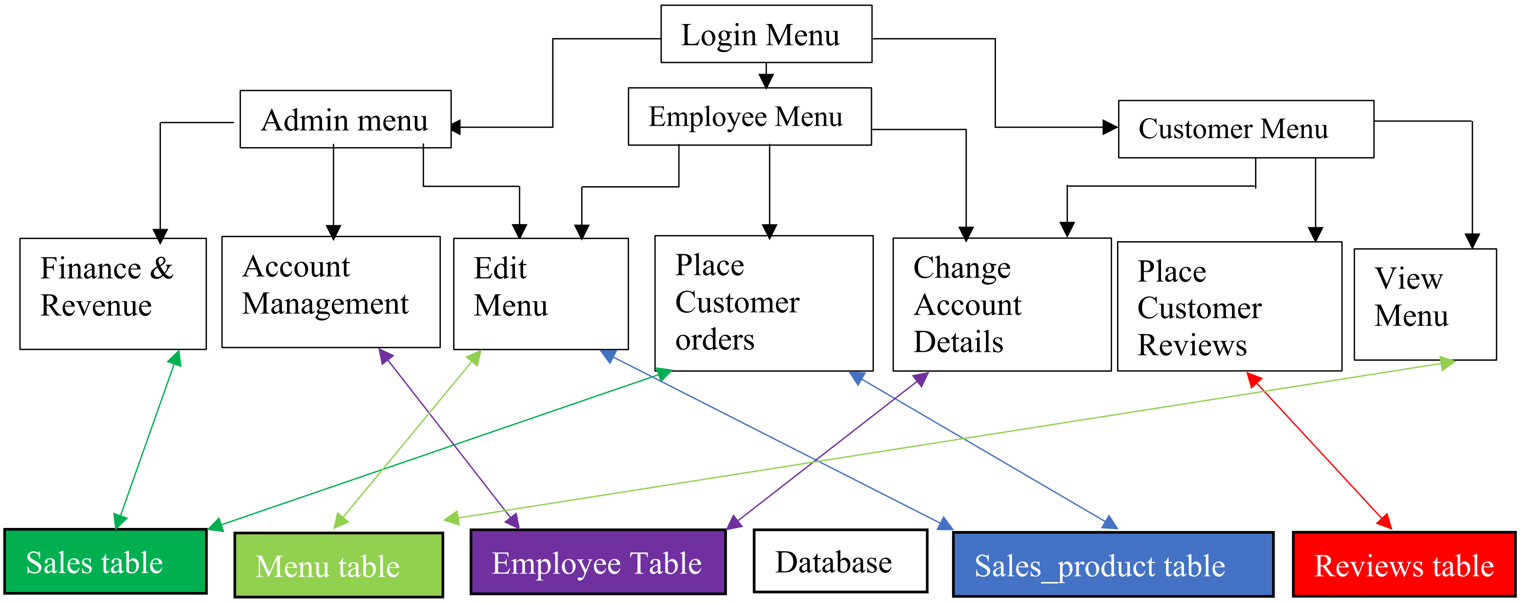


Table structures (MySQL):

Table 1 (Employee1):

Field Names	***Data Type***	***Description***	***Data Validation***	***Extra Information***
ID	int	ID is used to identify each user.	Not Editable	Primary key
First Name	text	The Name of the user (General information).	N/A	Not Null
Last Name	text	The surname of the user (General information).	N/A	Not Null
Role	text	Type of user admin, employee and customer (General information).	N/A	Can only be an admin or employee or customer.
Age	int	The age of the user (General information).	Between the numbers 1-100	Only numbers can be inputted.
Email	text	In order to contact the user.	Must follow Email format "@" ".com"	Not Null
Password	text	Password in order to login.	N/A	Not Null

Table 2 (Menu):

Field Names	***Data Type***	***Description***	***Data Validation***	***Extra information***
ID	int	ID is used to identify each item in the menu.	Not Editable	Primary key

ProductName	text	ProductName is the name of the item on the menu.	N/A	Not Null
Price	int	The price of the particular item on the menu.	Only numbers can be inputted	Null
Photo	BLOB	The Photo shows the image of the particular item on the menu.	Only PNG or JPEG	Null

Table 3 (Sales_Product):

Field Names	***Data Type***	***Description***	***Data Validation***	***Extra information***
ID	int	The ID is a customer ID used to identify the customer who is purchasing coffee.	Existence Validation	Null
Sales_ID	int	Sales_ID is used to identify each transaction/ purchase.	Only Numbers	Null
ProductName	text	ProductName is the name of the item being transacted/ purchased.	N/A	Not Null
Price	int	The price of the each item/ product.	Only Numbers	Null
Quantity	int	The amount of each item/ product.	Only Numbers	Null
Total	int	The multiplication of the price and quantity of the item/ product.	Only Numbers	Null
OrderDate	DATE	The data of when item was purchased.	Only Date Format	YYYY-MM-DD

Table 4 (Sales):

Field Names	***Data Type***	***Description***	***Data Validation***	***Extra information***
ID	int	ID is used to identify each sale.	Not Editable	Primary key
Subtotal	int	The subtotal is the total of the purchased items.	Only Numbers	Null
Pay	int	The pay is the amount of money the consumer is paying for the product.	Only Numbers	Null
Balance	int	The balance is the subtotal subtracted by the pay.	Only Numbers	Null

Table 5 (Reviews):

Field Names	***Data Type***	***Description***	***Data Validation***	***Extra information***
ID	int	ID is used to identify each review.	Not Editable	Primary key
Email	text	Email to Identify the customer.	Email Format with “@” and “.com”	Not Null
Rating	int	The rating/feedback of the customer out of 5. (e.g. 5/5).	Range check from 1-5	Null

Example Databases (MySQL):

Table 1 (Accounts):

ID	First Name	Last Name	Age	Role	Age	Email	Password
1	Ignacio	Sar	20	Admin	31	IS@gmail.com	423
2	Adam	Goorkani	27	Employee	26	AG@gmail.com	AG222
3	Mazen	Nassour	23	Costumer	22	MN@gmail.com	MN3298

Table 2 (Menu):

ID	ProductName	Price	Photo
1	Americano	3	Americano.PNG BLOB 3KB
2	Latte	3	Latte.PNG BLOB 3KB
3	Sandwich	5	Sandwich.PNG BLOB 3KB

Table 3 (Sales_Product):

ID	Sales_ID	ProductName	Price	Quantity	Total	OrderDate
2	1	Americano	3	1	3	2022-10-11
2	2	Latte	3	1	3	2022-10-12
2	2	Sandwich	5	2	10	2022-10-12

Table 4 (Sales):

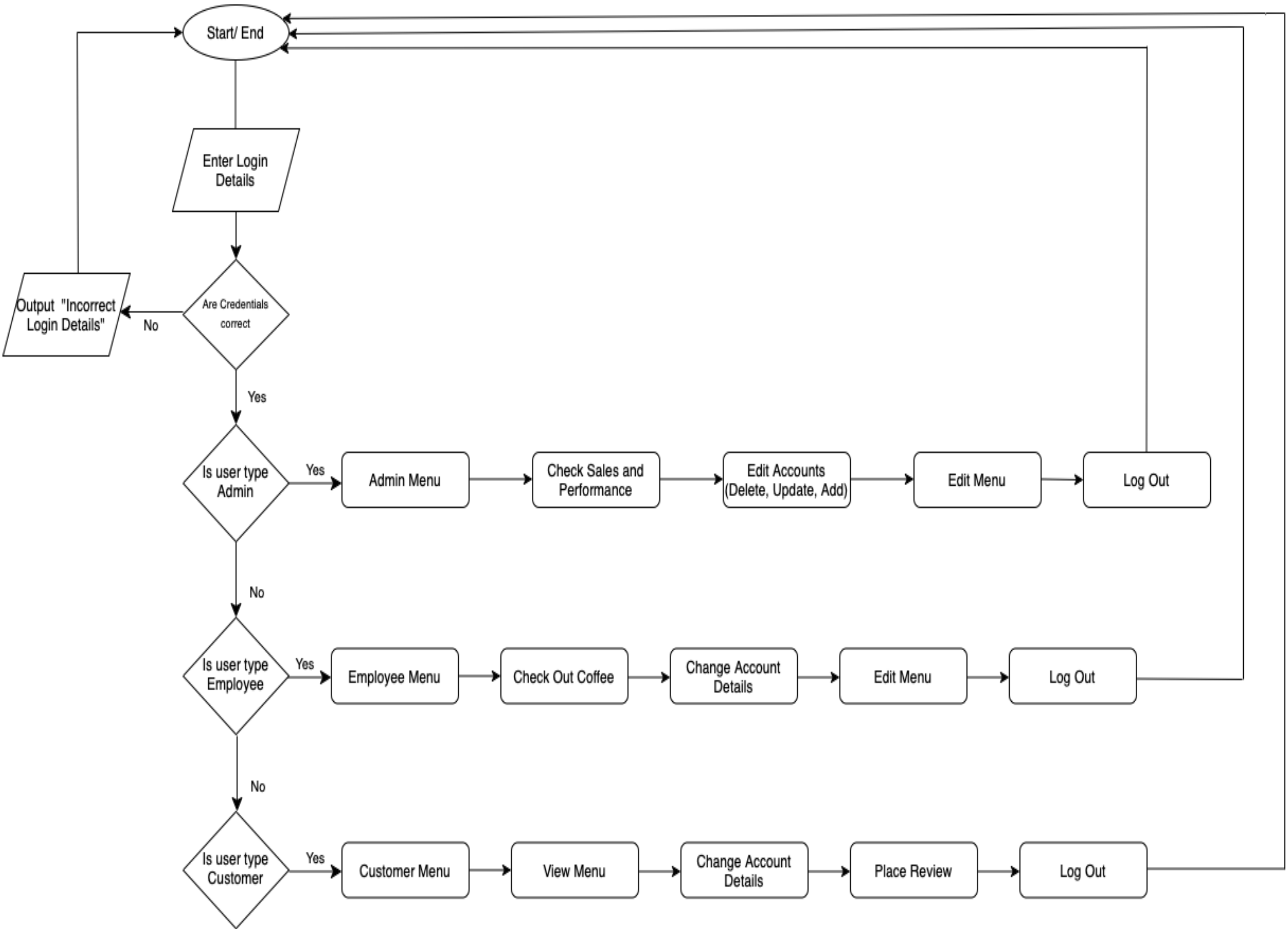
ID	Subtotal	Pay	Balance
1	3	4	1
2	13	13	0

Table 5 (Reviews):

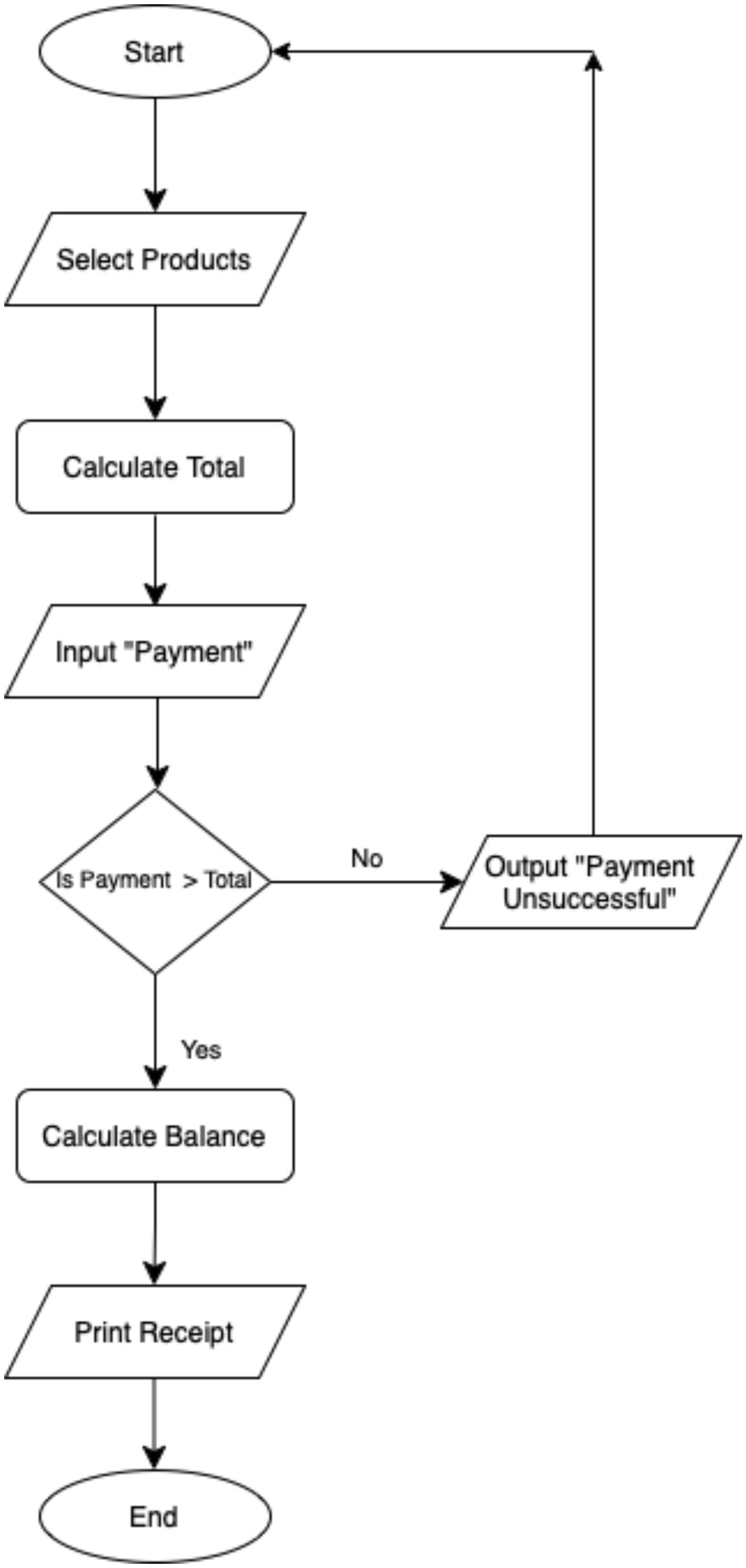
ID	Email	Review
1	IS@gmail.com	4
2	AG@gmail.com	5
3	MN@gmail.com	2

Flowchart Diagrams:

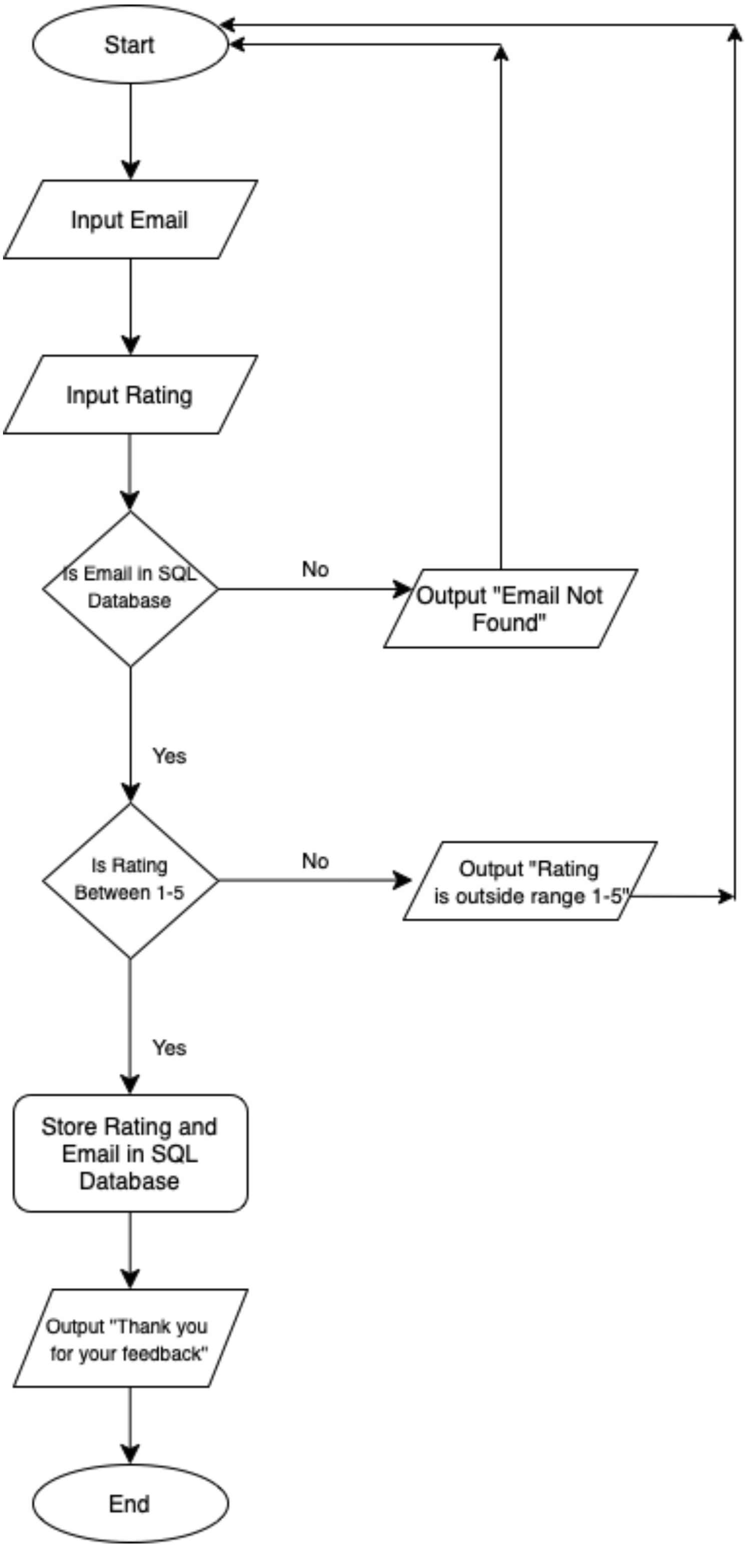
Flowchart 1:
This flowchart outlines how the software operates and how the user interacts with the programme. It demonstrates how different user types access different features.



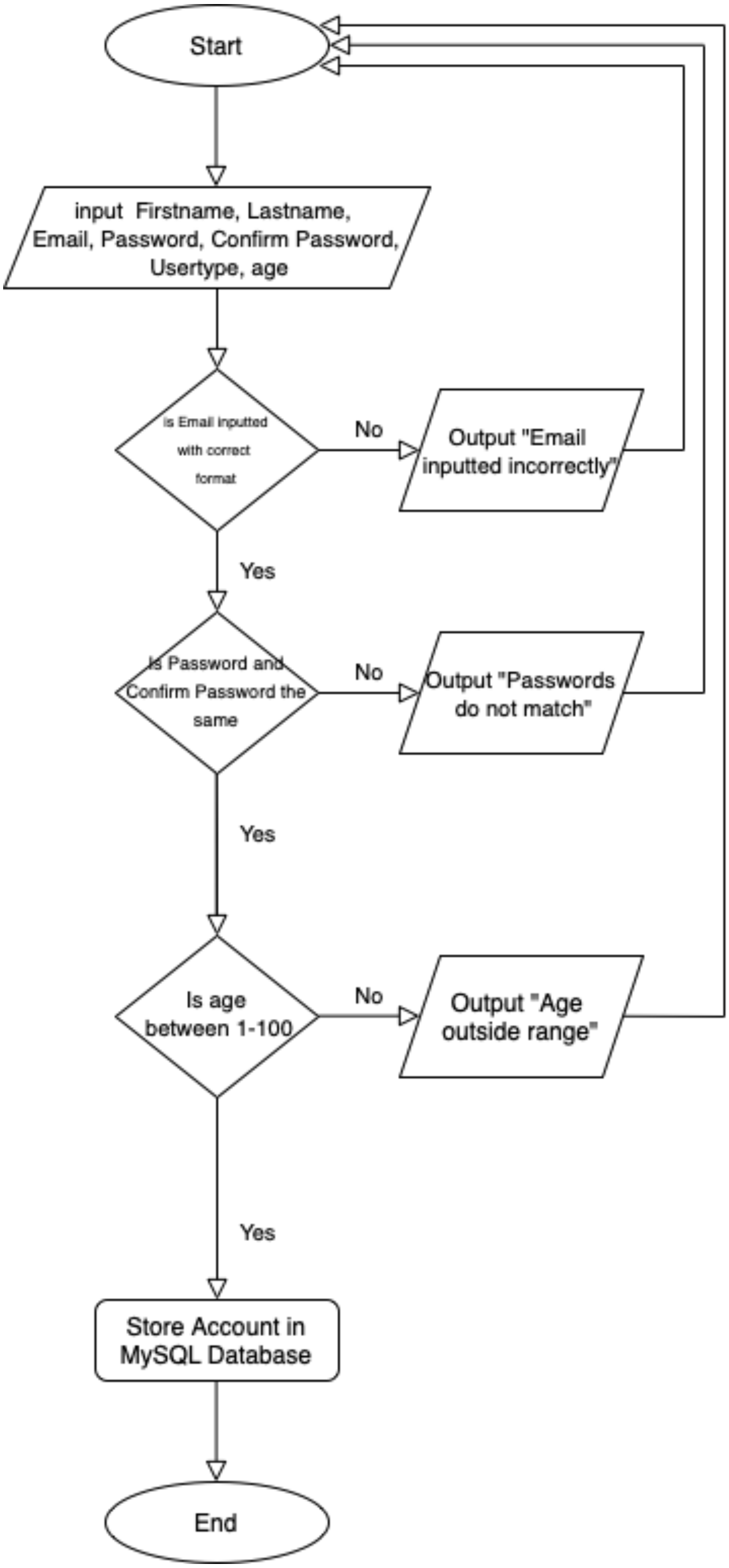
Flowchart 2 (Checkout System):



Flowchart 3 (Rating System):


Start
Input Email
Input Rating
Is Email in SQL Database
No
Output "Email Not Found"
Yes
Is Rating Between 1-5
No
Output "Rating is outside range 1-5"
Yes
Store Rating and Email in SQL Database
Output "Thank you for your feedback"
End

Flowchart 4 (Register System):


Start
input Firstname, Lastname, Email, Password, Confirm Password, Usertype, age
Is Email inputted with correct format
No
Output "Email inputted incorrectly"
Yes
Is Password and Confirm Password the same
No
Output "Passwords do not match"
Yes
Is age between 1-100
No
Output "Age outside range"
Yes
Store Account in MySQL Database
End

Pseudocode for login page:

```
// Prompt user for email, password, and role input
output "Please enter your email:"
input  EMAIL
output "Please enter your password:"
input PASSWORD
output "Please enter your role:"
input ROLE

// Check if input fields are empty, output error message if true
if email == "" OR password == "" OR role == "" THEN
    PRINT "Error: Please fill in all required fields."
else
    // Check if email, password, and role are correct
    if checkCredentials(email, password, role) then
        output "Login successful."
    else
        output "Error: Incorrect email, password, or role."
    end if
end if

// Check if email, password, and role match a user in the database
FOR EACH user IN usersDatabase
    IF user.email == email AND user.password == password AND user.role == role THEN
        RETURN TRUE
    END IF
END FOR

// If no user matches the input credentials, return false
RETURN FALSE
```

Pseudocode adding items to the menu:

```
// Define variables for the menu
CONST MAX_MENU_ITEMS = 50
DIM menu_names[MAX_MENU_ITEMS] // Array to hold the names of the menu items
DIM menu_prices[MAX_MENU_ITEMS] // Array to hold the prices of the menu items
DIM num_menu_items = 0 // Number of items currently in the menu

// Example usage: add a latte to the menu
menu_names.add("Latte")
menu_prices.add(3.99)
num_menu_items = num_menu_items + 1

// Example usage: add a cappuccino to the menu
menu_names.add("Cappuccino")
menu_prices.add(4.49)
num_menu_items = num_menu_items + 1
```

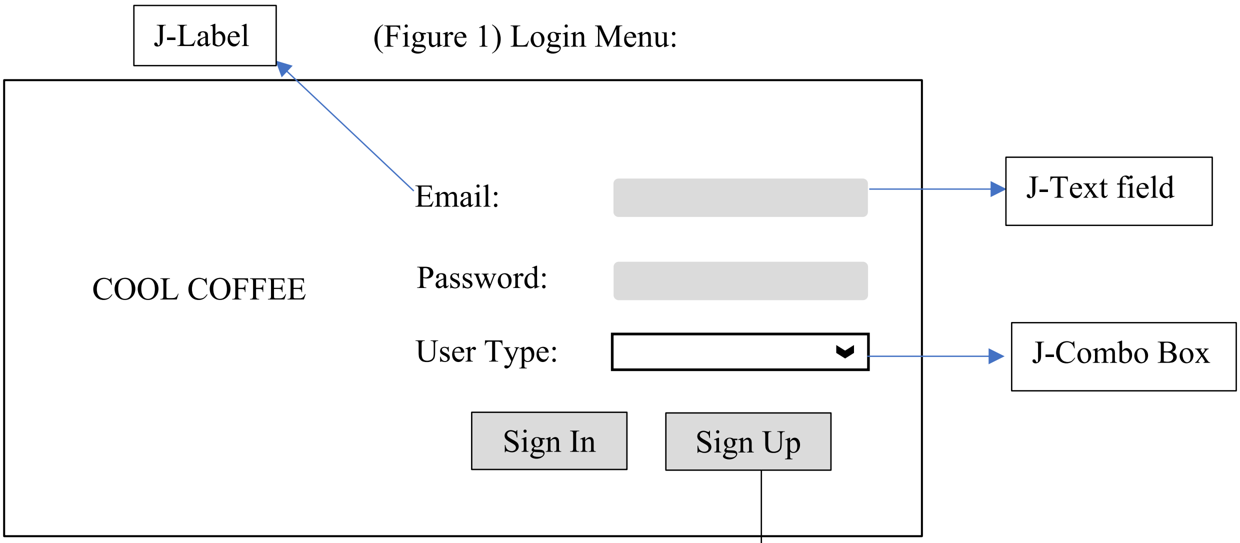
```
// Print out the entire menu
FOR i = 0 TO num_menu_items - 1
   PRINT menu_names[i] + " - JOD" + menu_prices[i]
END FOR
```

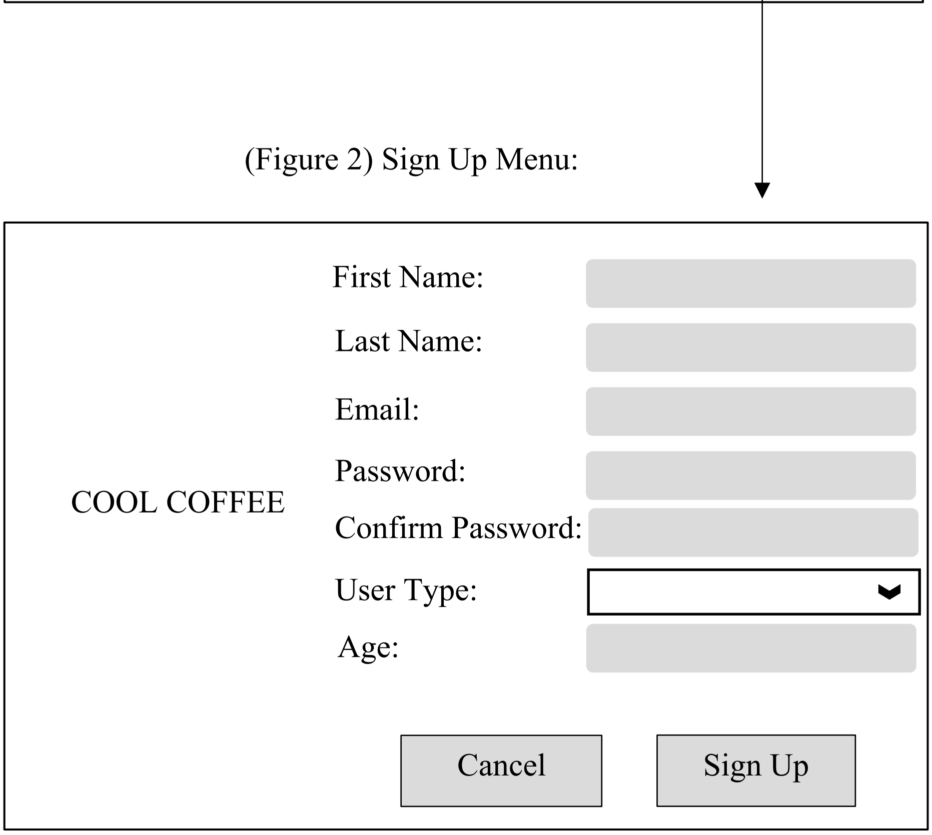
Pseudocode to prevent data entry:

```
accountsList = new List()

loop while true // prompt the user to input account details
input ID, FIRSTNAME, LASTNAME, ROLE, EMAIL, PASSWORD, AGE
// create a new account object with the input details
newAccount = new Account(ID, FIRSTNAME, LASTNAME, ROLE, EMAIL,
PASSWORD, AGE)
// add the new account to the list of accounts
accountsList.addItem(newAccount)
// prompt the user to continue adding accounts or quit the program
input CONTINUE
if CONTINUE == "quit" then
exit loop
         end if
 end loop
accountsList.resetNext()

loop while accountsList.hasNext()
output accountsList.getNext()
end loop
```

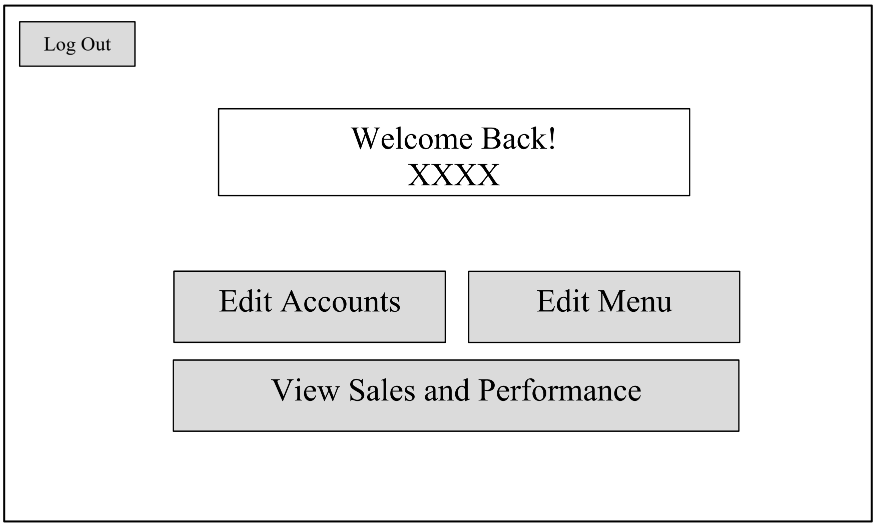
GUI design:
The diagrams below showcases the graphical user interface for the software.

Menus:

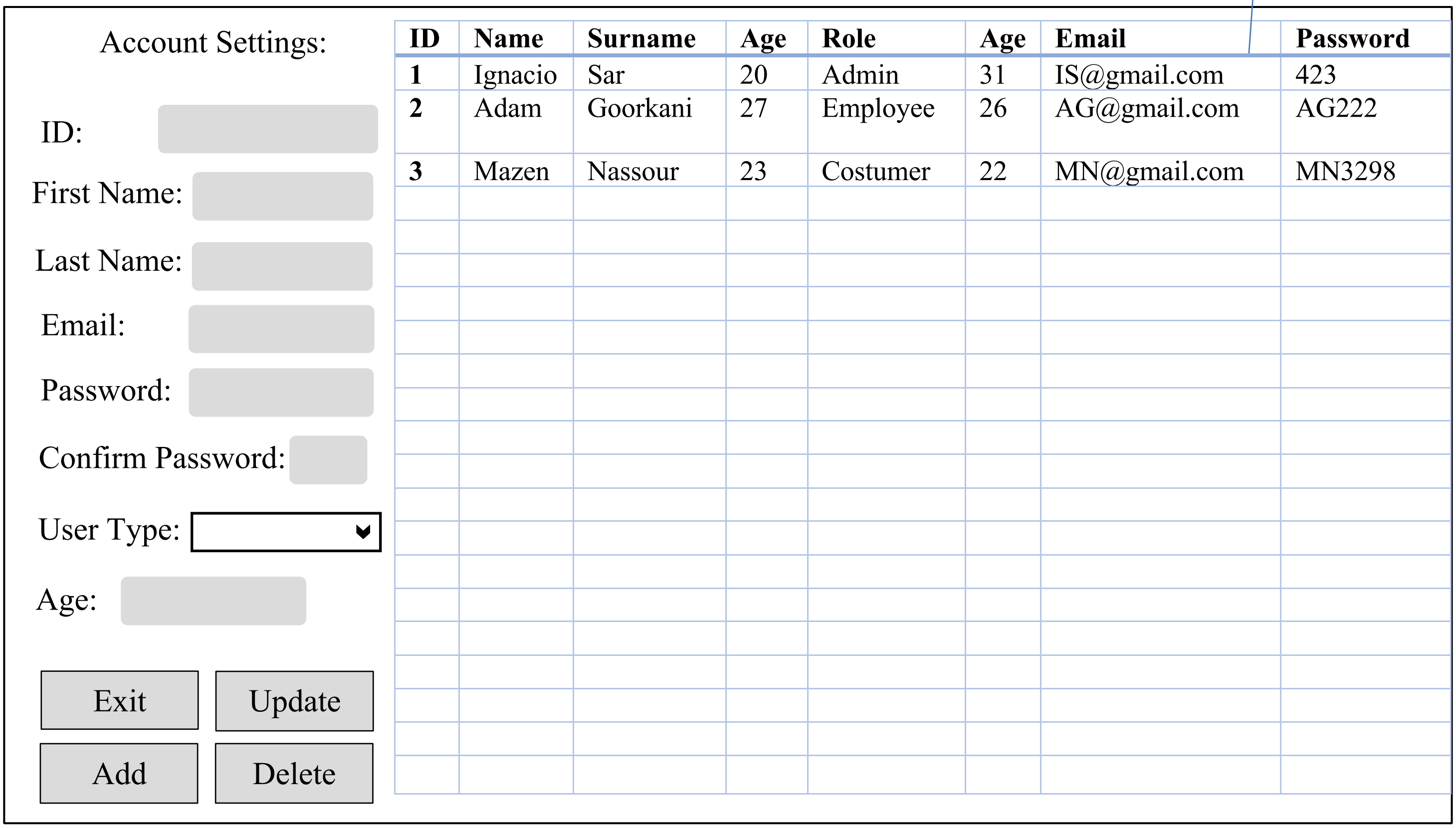

(Figure 1) Login Menu:


(Figure 2) Sign Up Menu:

(Figure 3) Admin Menu:

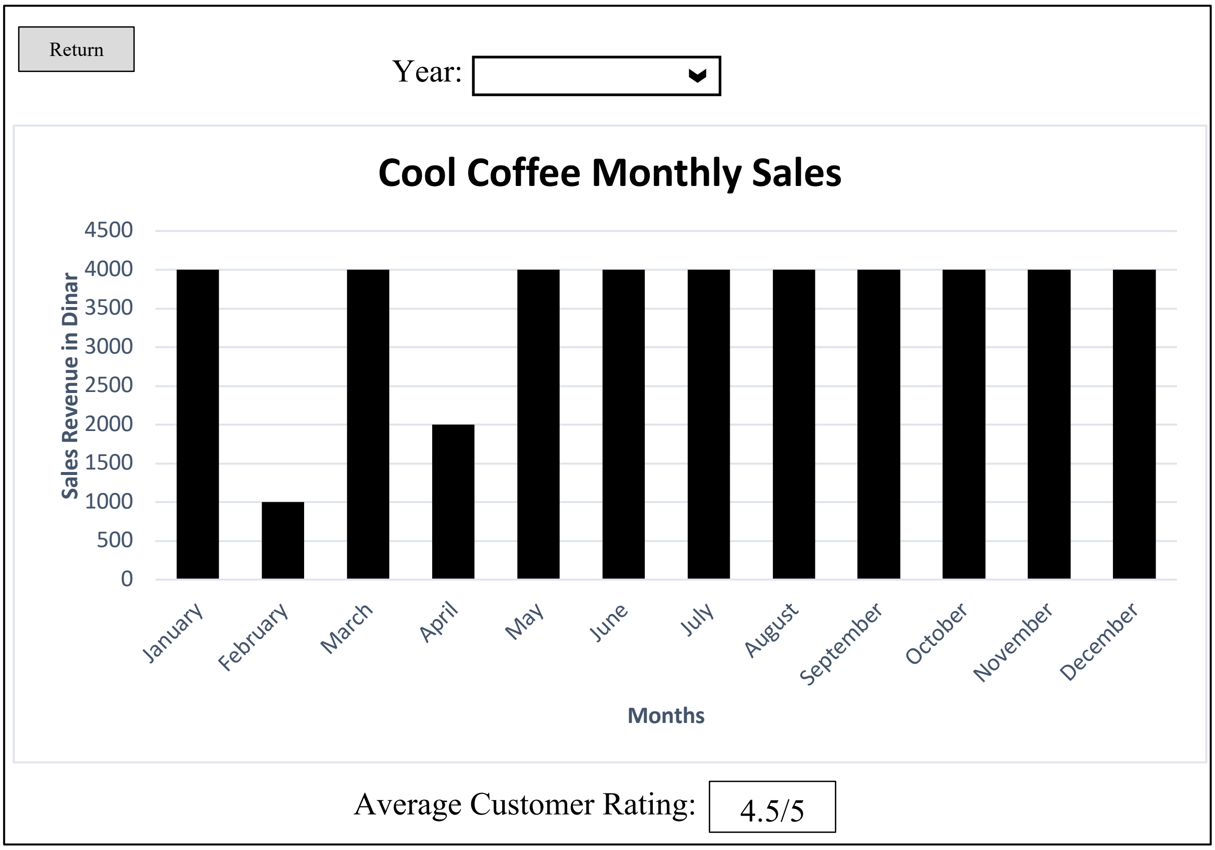


(Figure 4) Admin - Edit Accounts Menu:



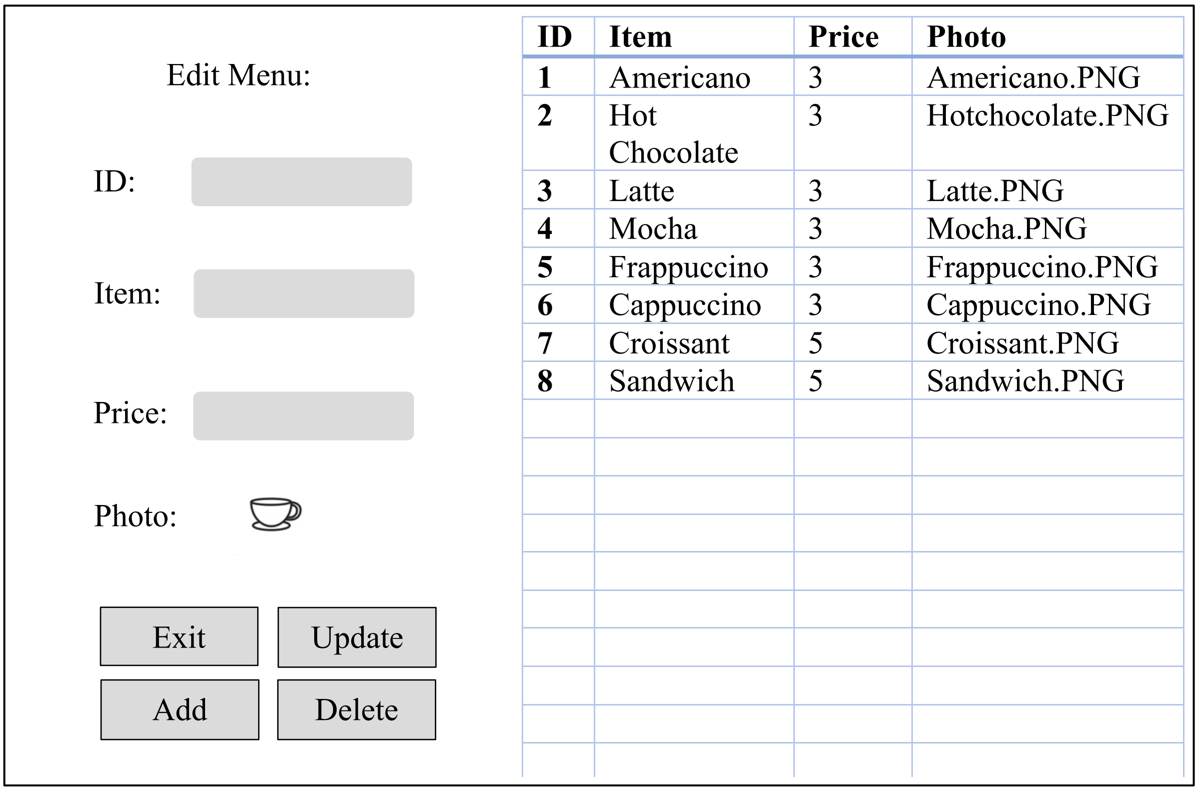
ID	Name	Surname	Age	Role	Age	Email	Password
1	Ignacio	Sar	20	Admin	31	IS@gmail.com	423
2	Adam	Goorkani	27	Employee	26	AG@gmail.com	AG222
3	Mazen	Nassour	23	Costumer	22	MN@gmail.com	MN3298

(Figure 5) Admin - View Sales and Performance:



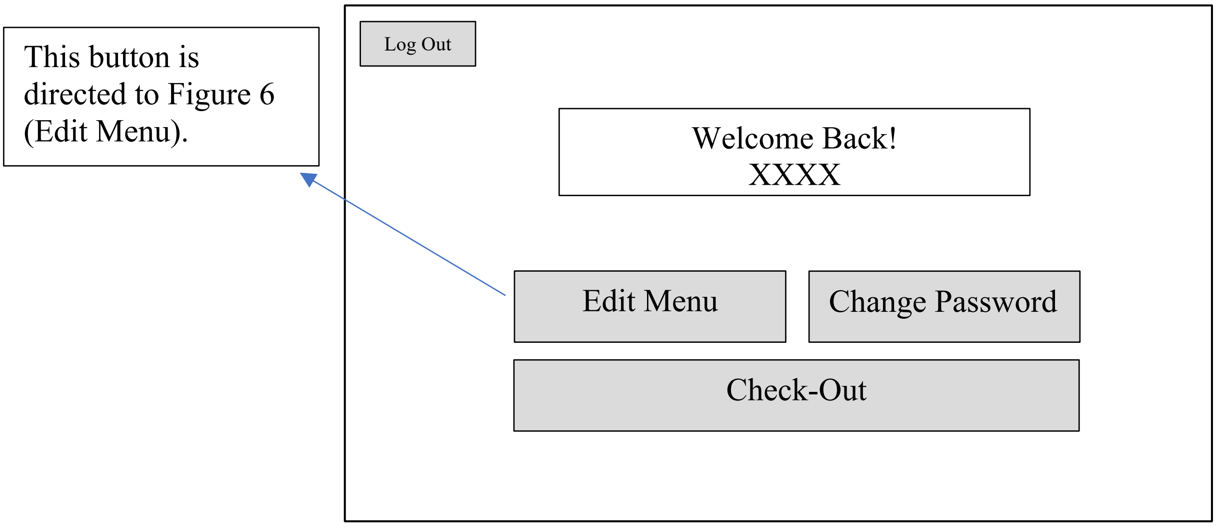
Average Customer Rating: 4.5/5

(Figure 6) Admin / Employee - Edit Café Menu:



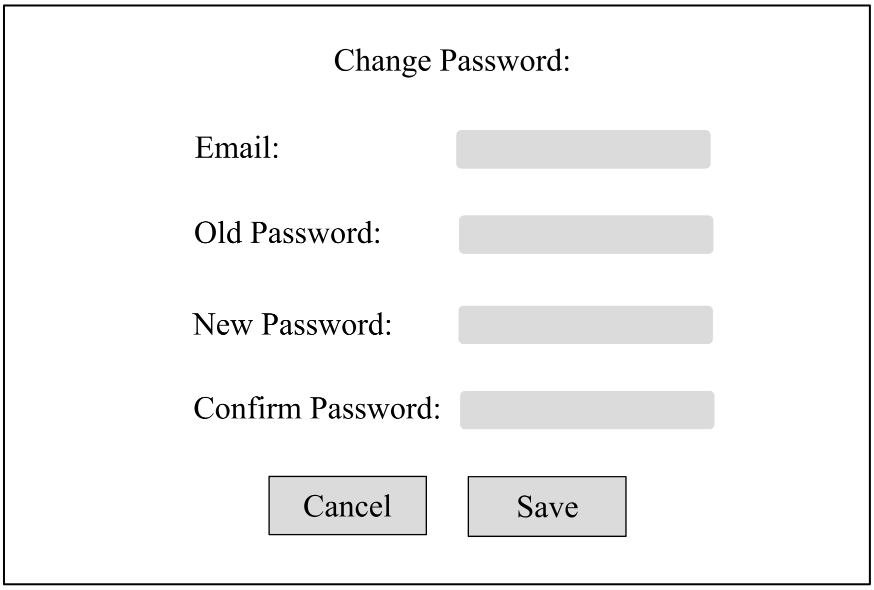
ID	Item	Price	Photo
1	Americano	3	Americano.PNG
2	Hot Chocolate	3	Hotchocolate.PNG
3	Latte	3	Latte.PNG
4	Mocha	3	Mocha.PNG
5	Frappuccino	3	Frappuccino.PNG
6	Cappuccino	3	Cappuccino.PNG
7	Croissant	5	Croissant.PNG
8	Sandwich	5	Sandwich.PNG

(Figure 7) Employee Menu:

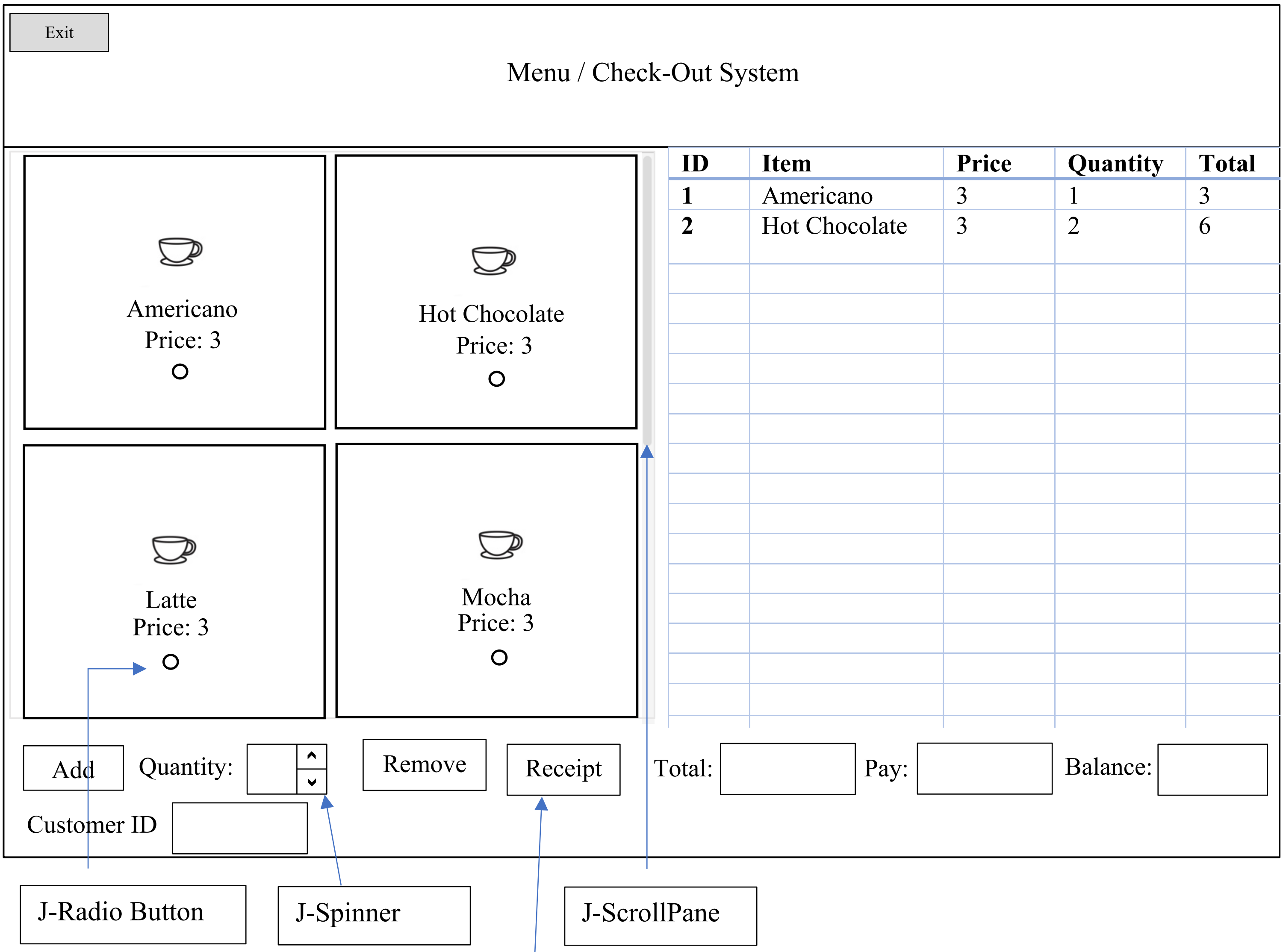


This button is directed to Figure 6 (Edit Menu).

(Figure 8) Employee/ Customer - Change Password Menu:

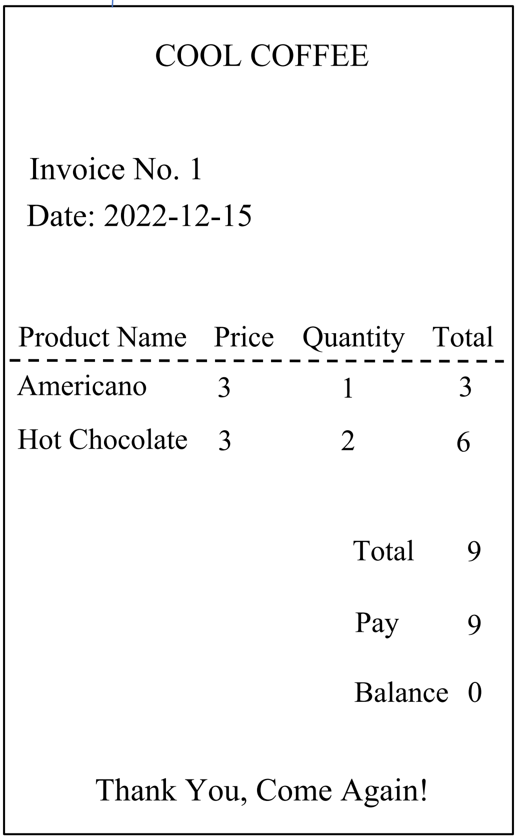


(Figure 9) Employee - Menu/ Check-Out:



ID	Item	Price	Quantity	Total
1	Americano	3	1	3
2	Hot Chocolate	3	2	6

(Figure 10) Receipt:



COOL COFFEE

Invoice No. 1
Date: 2022-12-15

Product Name	Price	Quantity	Total
Americano	3	1	3
Hot Chocolate	3	2	6

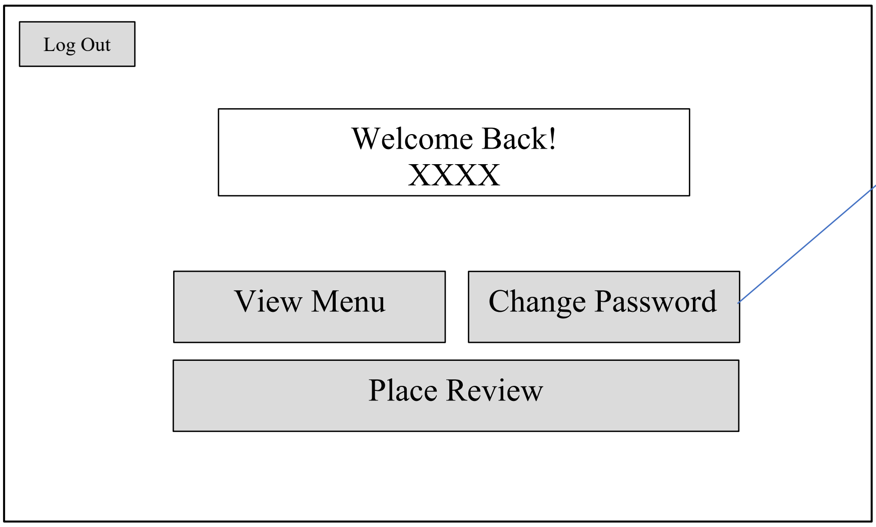
Total 9

Pay 9

Balance 0

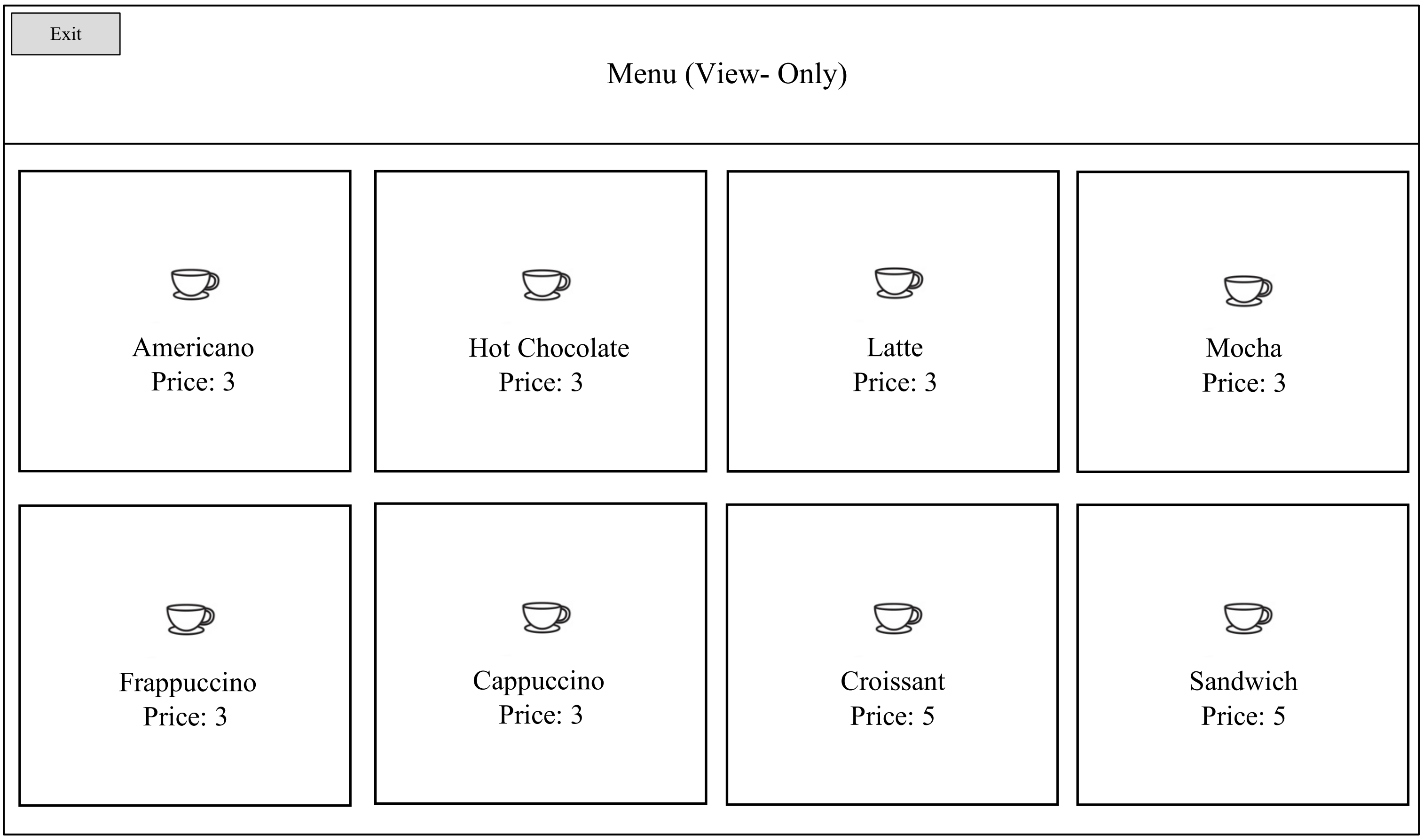
Thank You, Come Again!

(Figure 10) Customer Menu:



This button is directed to Figure 8 (Change account details menu).

(Figure 11) Customer - View Menu:



Americano
Price: 3

Hot Chocolate
Price: 3

Latte
Price: 3

Mocha
Price: 3

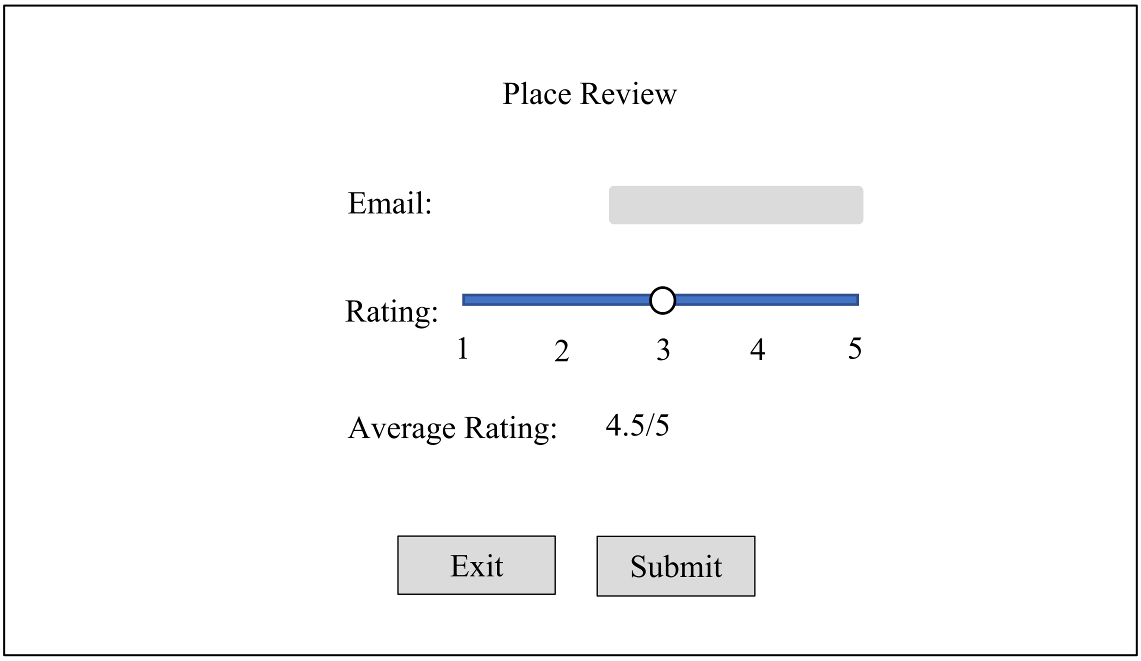
Frappuccino
Price: 3

Cappuccino
Price: 3

Croissant
Price: 5

Sandwich
Price: 5

(Figure 12) Customer - Place Review Menu:



Place Review

Email:

Rating:

Average Rating: 4.5/5

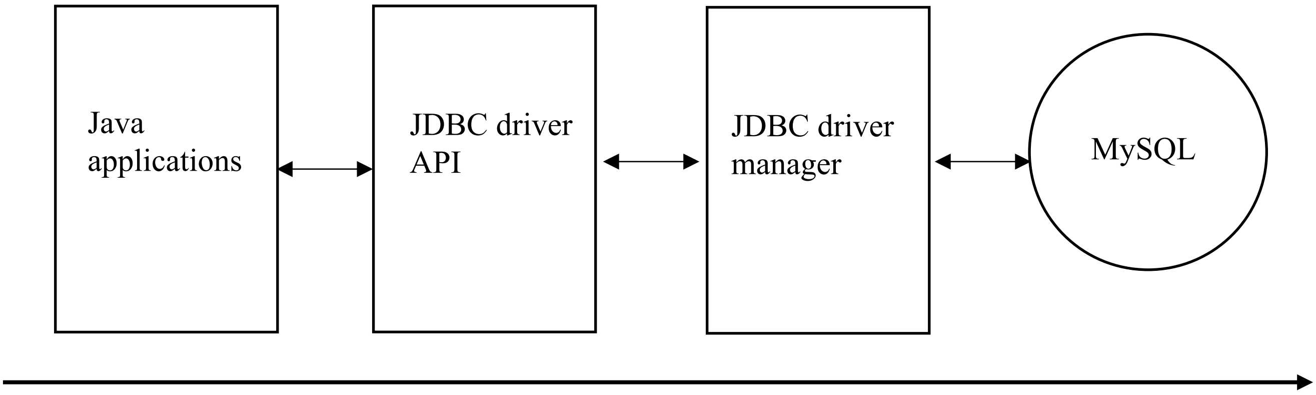
JDBC Connectivity:
JDBC stands for Java Database Connectivity, and it is a Java API (Application Programming Interface) that allows Java programs to access and change relational databases, such as MySQL, through a standard interface.

JDBC provides a set of classes and interfaces for connecting to a database, executing SQL statements, and handling results. With JDBC, a Java program can send SQL statements to a database, and receive and process the results. JDBC also supports transactions, prepared statements, and stored procedures.

JDBC driver used:
jdbc:mysql://test, username, password
com.mysql.cj.jdbc.Driver

Imports used:
import java.sql.Connection;
import java.sql.DriverManager;

Diagram showing connection:



Testing Plan:

What is tested	Method of testing	Expected outcome
Login functionality for all user types	Black box testing: Attempt to login with valid and invalid credentials for each user type.	User is able to login with valid credentials and denied access with invalid credentials.
Registration functionality for all user types	Black box testing: Attempt to register a new account with valid and invalid input, and ensure that the data is correctly stored in the MySQL database.	User is able to register with valid information and denied registration with invalid information.
Admin adding and deleting user accounts	White box testing: Attempt to add and delete user accounts as an admin, and ensure that the changes are correctly reflected in the database.	User accounts are successfully added and deleted from the database.
Admin viewing sales and performance	Black box testing: Access the sales and performance page as an admin, and ensure that the data displayed is accurate based on the data stored in the database.	Sales and performance data is accurately displayed.
Admin editing cafe menu	White box testing: Attempt to edit the cafe menu as an admin and employee, and ensure that the changes are correctly reflected in the database and the updated menu is displayed to all users.	Cafe menu is successfully updated in the database.
Employee editing cafe menu	White box testing: Employee edits cafe menu through the program.	Cafe menu is successfully updated in the database.
Employee check-out system	Black box testing: Attempt to check-out coffee and ensure that a receipt is printed with accurate information.	Order is successfully placed and receipt is generated.
Employee changing own account details	White box testing: Attempt to change account details for Employee and ensure that the changes are correctly reflected in the database and the updated account details are displayed to the user.	Employee account details are successfully updated in the database.
Customer viewing café menu	Black box testing: Access the cafe menu as a customer, and ensure that the data displayed is accurate based on the data stored in the database.	The cafe menu displayed is accurate and up-to-date based on the data stored in the database.
Customer changing own account details	White box testing: Attempt to change account details for Customer and ensure that the changes are correctly reflected in the database and the updated account details are displayed to the user.	Customer account details are successfully updated in the database.
Customer Placing review	Black box testing: Attempt to submit review through the program.	Review is successfully submitted and displayed on the database.
Full Application operating	Give the full application to a peer to test out and ensure all features are working as expected.	Peer successfully uses application however, any minor errors can be pointed out.