

Submitted By :

Mahin Rashid Chowdhury

Roll : 1907021

Course : CSE 2200

Submitted To :

Prottoy Saha

Assistant Professor

Department of Computer Science and
Engineering
Khulna University of Engineering &
Technology

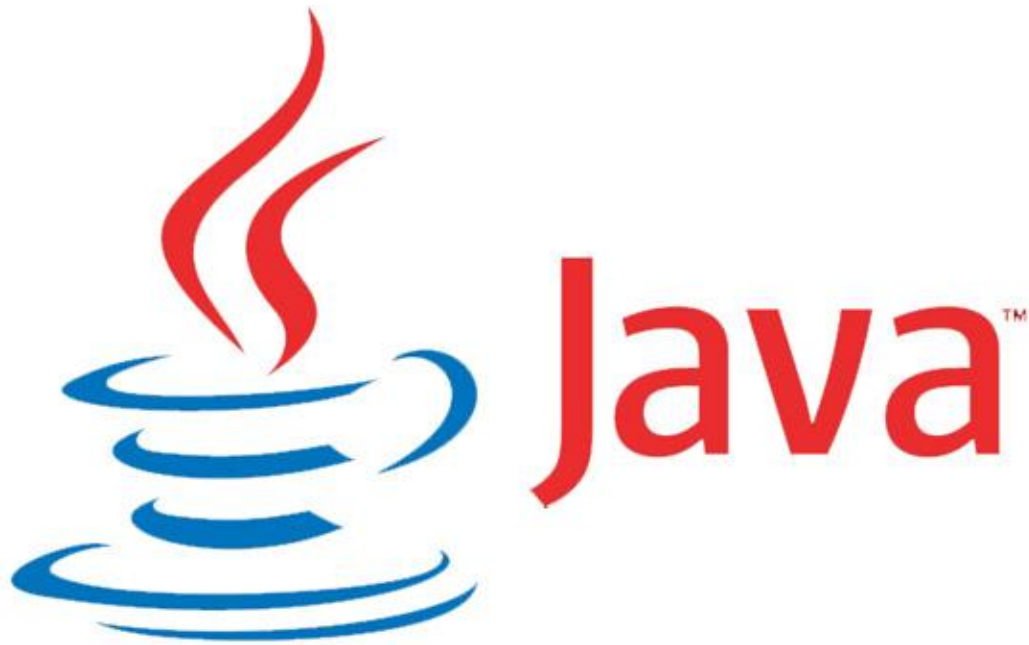
Objectives :

1. To Create a banking app using Java,Maven,JavaFX UI library.
2. To use MongoDB as our app database.
3. To use Java thread and multithreading for fast responsiveness and high scalability.
4. To use authentication.
5. To create an attractive and easy UI.
6. To check confidentiality every time a user logs in and if a new user wants to sign up and update the data in the database in real time.
7. To have the ability to transfer credit from one user to another user.
8. To have the ability to deposit funds to clients.

Introduction :

Macro is a multi user banking app. Users can transfer credits among them in real-time and also an admin section for administration processes. This project was divided into two sub-project. It is built in a

way that multiple clients can transfer funds between them that includes small texts in each transaction.



The software was built using Java 19, Maven, JavaFX 19 and scene builder for UI and MongoDB for database. Java is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. It is a general-purpose programming language intended to let application developers write once, run anywhere (WORA), meaning the compiled

Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of the underlying computer architecture. The syntax of Java is similar to C and C++, but has fewer low-level facilities than either of them. The java runtime provides dynamic capabilities that are typically not available in traditional compiled languages. Java has released many SDK in years. The The SDK used in this program is Java 19 which is the latest one. But in Macro program, Maven module is used for automation and for Dependencies.

Maven is a build automation tool used primarily for Java projects. Maven address two aspects of building software: how software is built and its dependencies. Most of Maven's functionality is in plug-ins. In Maven project plug-ins are handled by the POM file. POM stands for Project Object Model which provides all the configuration for a single project. Java by default runs on the console. To make the program user-A friendly GUI framework is needed. One of the popular GUI frameworks for Java is JavaFX.



JavaFX is a java library used to develop Desktop applications as well as Rich Internet Applications. The applications built in JavaFX, can run on multiple platforms including Web, Mobile and Desktops. JavaFX is intended to replace Swing in Java applications as a GUI framework. However, it provides more functionalities than Swing. Like Swing, JavaFX, also provides its own components and doesn't depend upon the operating system. It is lightweight and hardware accelerated. It supports various operating systems including Windows, Linux and Mac OS. Like Java, JavaFX has many versions. JavaFX 16 is used in this program. JavaFX using FXML file for UI design. For visual scripting Scene Builder is used.

MongoDB database is used in the program for storing confidential data like usernames, emails and passwords of the admins and clients. Also, for storing

previous transactions between users. MongoDB is a source-available cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas. In Macro MongoDB Atlas is used. It is a web-based NoSQL database and it is always running. It has many security features like IP access, User access etc. Also, database can be modified while database is online or offline.



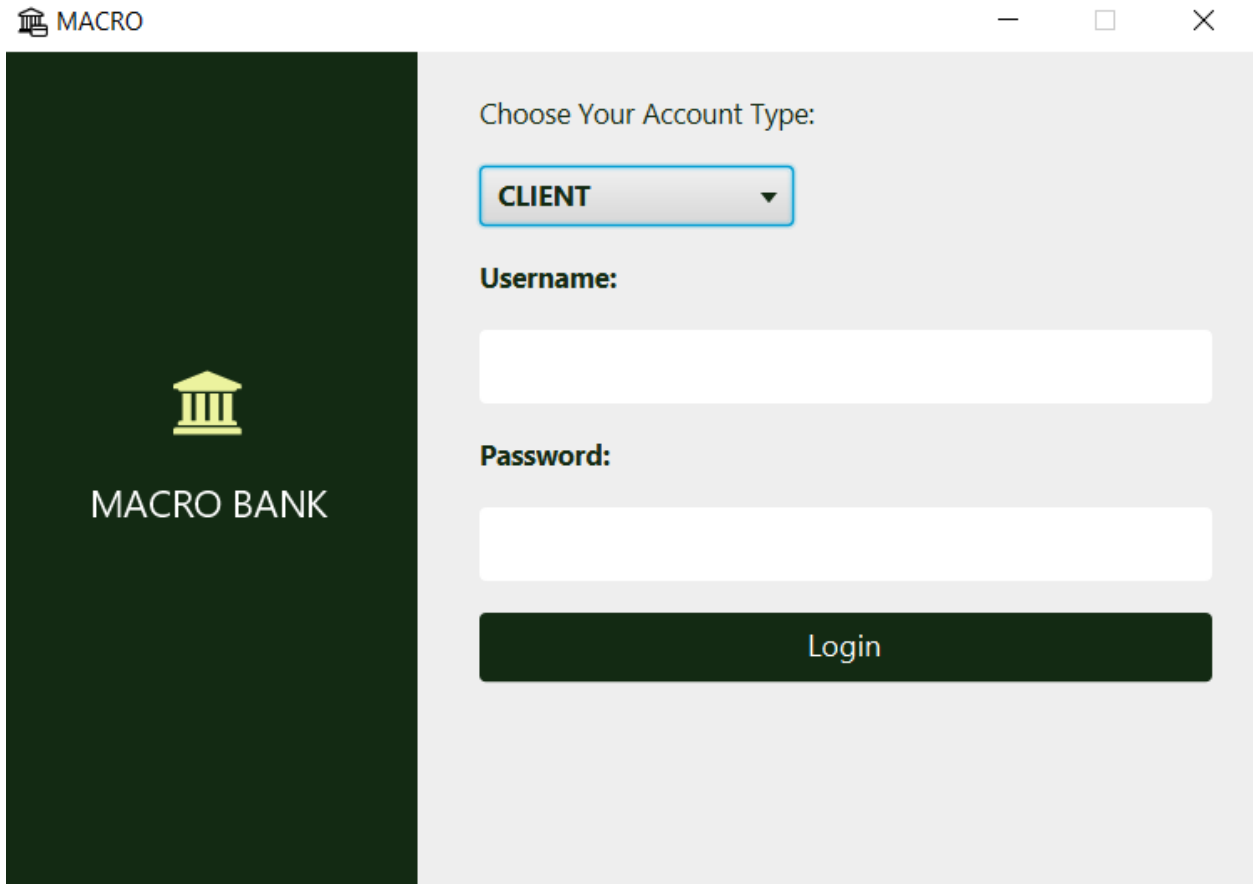
In MongoDB Atlas first a Cluster is created then databases can be added in them. Each database can contain multiple collection. And each collection can have as many as needed documents. For the Macro program, a MacroBank named Cluster is created. In this Cluster

There is a database named Macrobank.

In the users database there are collections called admin which holds admin login coordinates and client information such as name,userId,created date,currentBalance etc. information of the users. Information is held in BSON documents.

Implementation :

Login Page :



MACRO

Choose Your Account Type:

CLIENT

Username:

Password:

Login

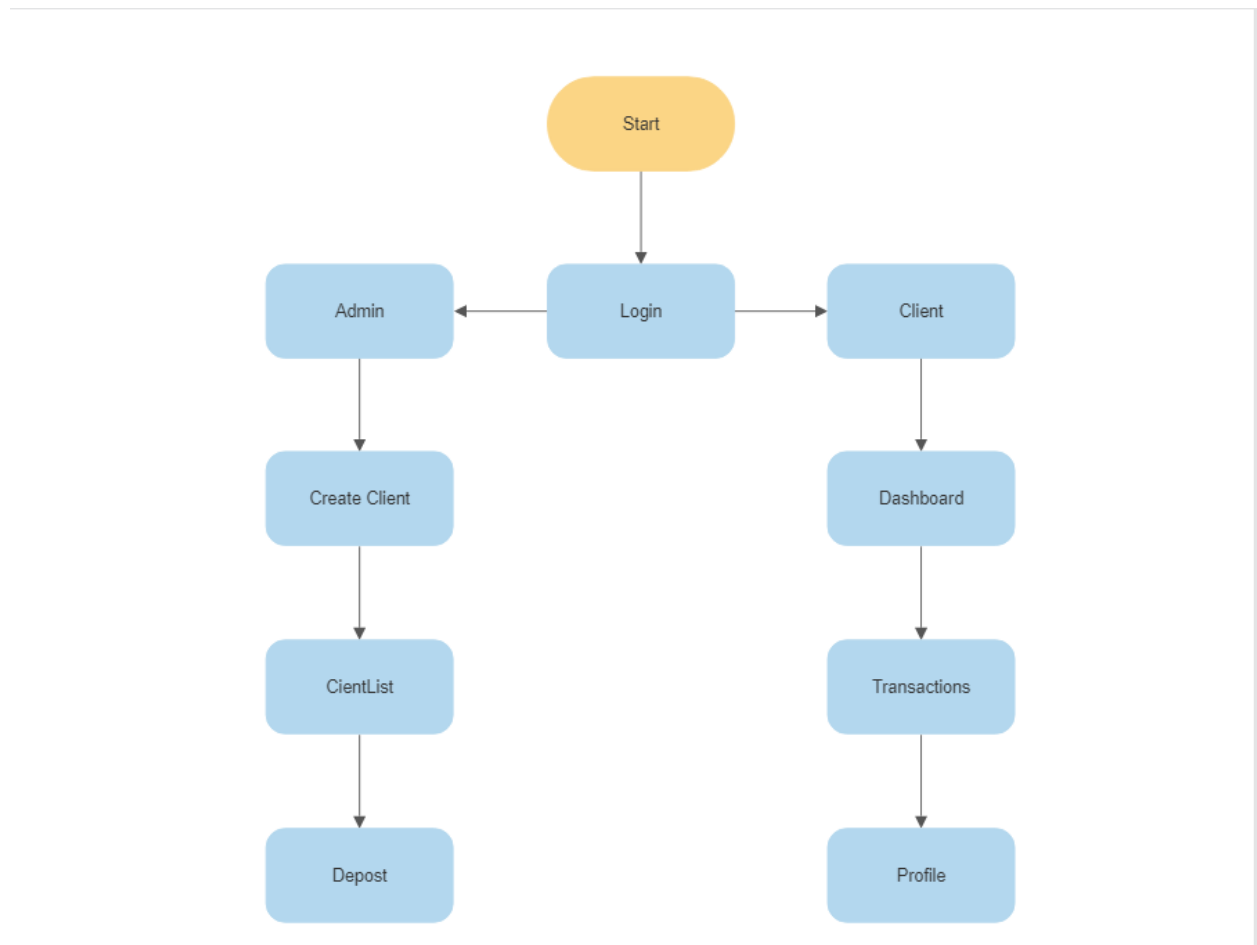
We can choose to go to both the admin section and client section by choosing the checkbox.

Primary window carries the options :

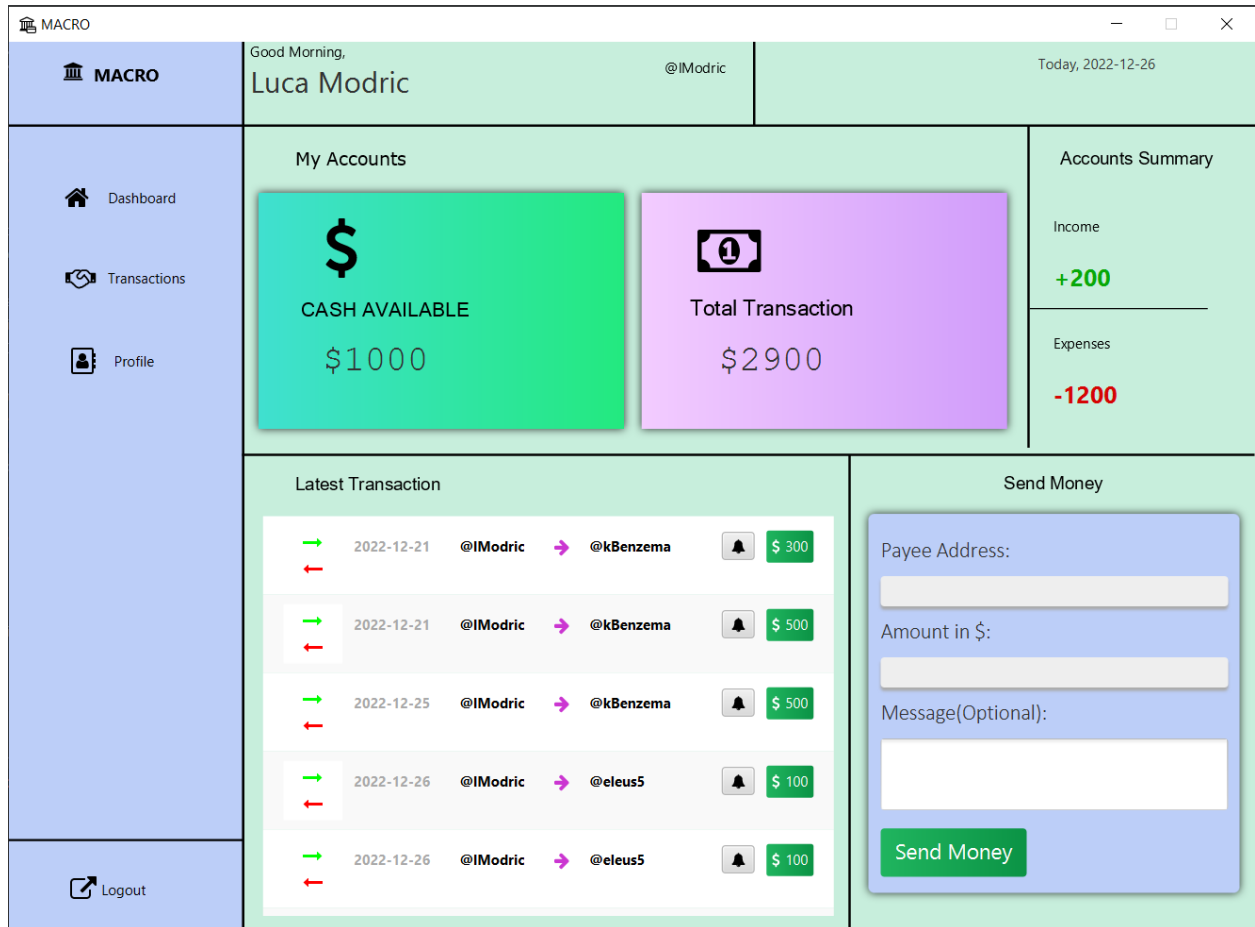
Username & Password.

Note that : This app is registration restricted. However new clients can be created by login as admin.

Flowchart :



Client Section :

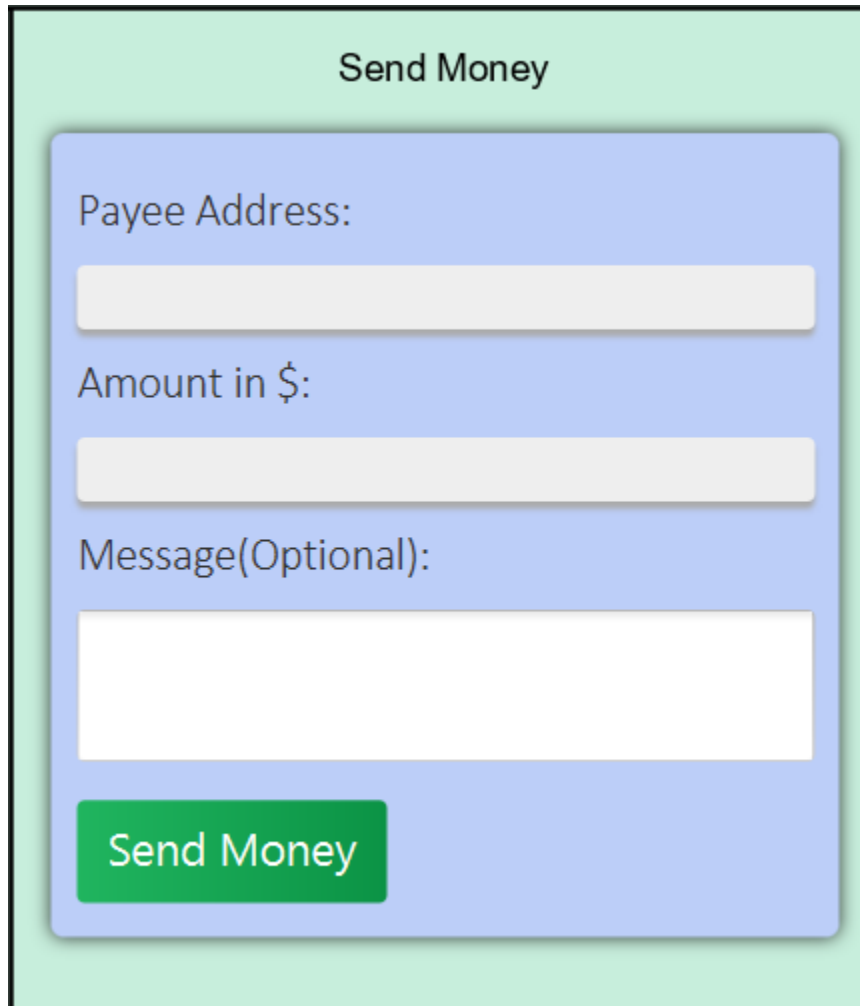


Client Dashboard contains:

- Available Cash
- Total Transactions made
- Username
- UserID
- Date
- Accounts Summary
- Income
- Expenses
- Latest Transactions

- Send money section

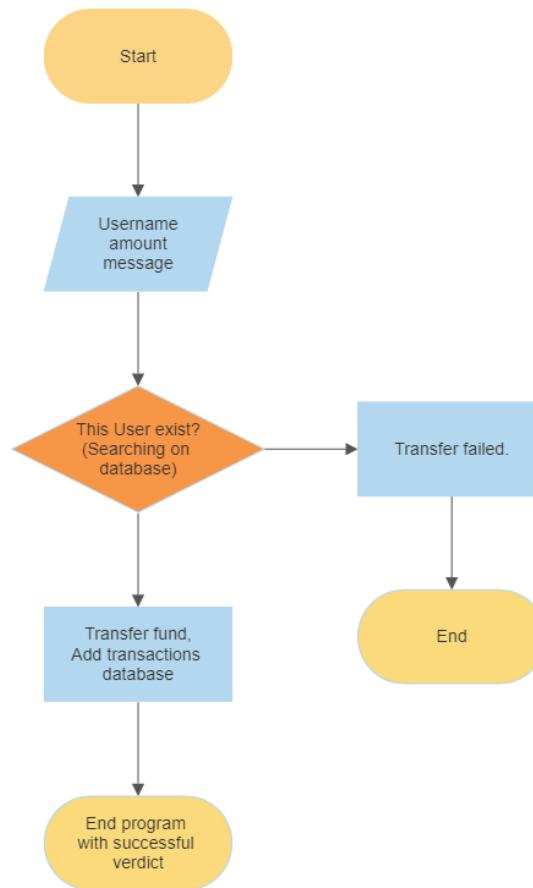
Send money Section :



The image shows a 'Send Money' form interface. It has a light green background with a title 'Send Money' at the top. Below the title is a light blue rounded rectangle containing the form fields. The fields are: 'Payee Address:' with a white text input field, 'Amount in \$:' with a white text input field, and 'Message(Optional):' with a larger white text area. At the bottom of the blue rectangle is a green button with the text 'Send Money' in white.

Money can be send by searching the user and entering amount with optional message.




























Flowchart :



Transaction section :

MACRO

Transactions

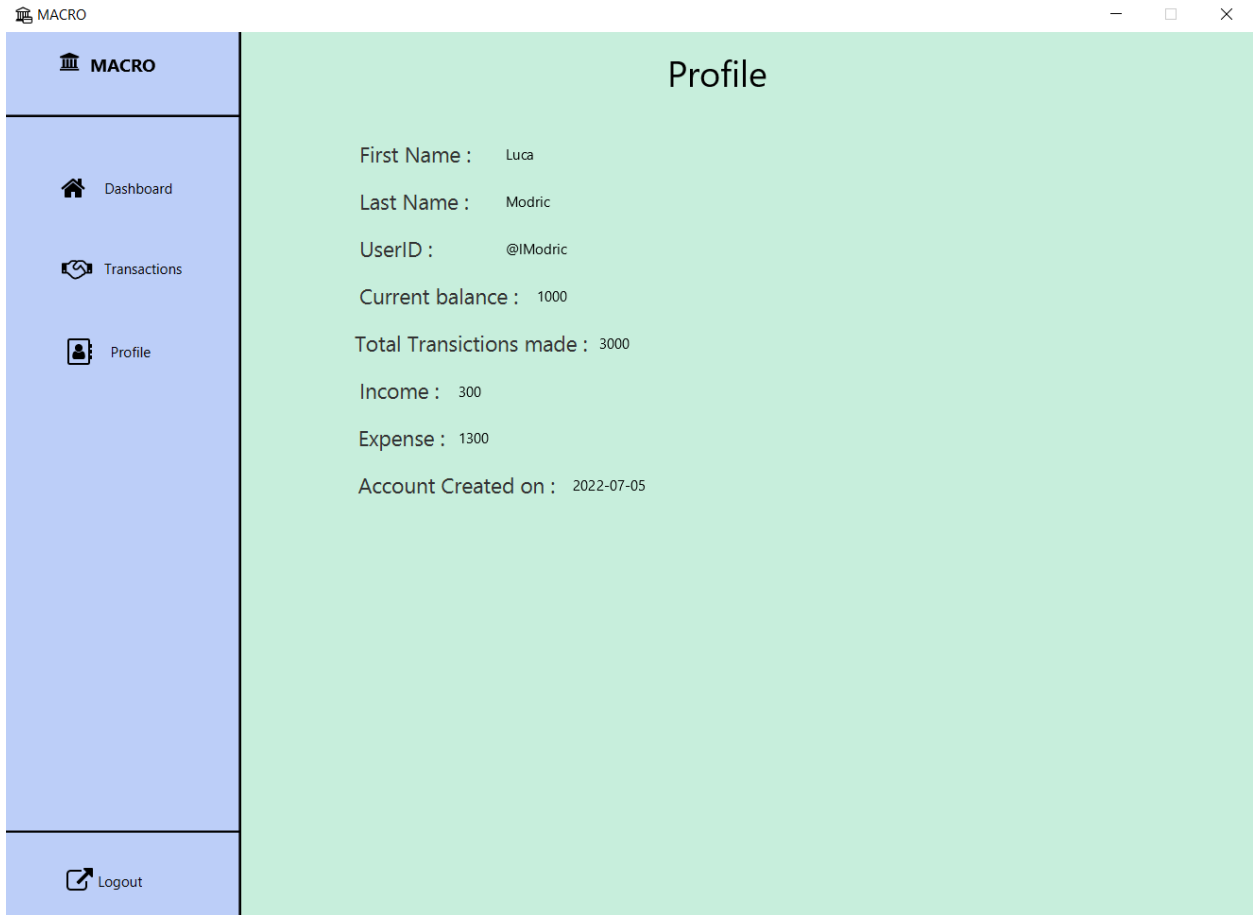
	2022-12-21	@IModric	→	@kBenzema		\$ 300
						
	2022-12-21	@IModric	→	@kBenzema		\$ 500
						
	2022-12-25	@IModric	→	@kBenzema		\$ 500
						
	2022-12-26	@IModric	→	@eleus5		\$ 100
						
	2022-12-26	@IModric	→	@eleus5		\$ 100
						
	2022-12-26	@eleus5	→	@IModric		\$ 200
						
	2022-12-26	@IModric	→	@eleus5		\$ 100
						
	2022-12-26	@IModric	→	@eleus5		\$ 900
						
	2022-12-26	@IModric	→	@eleus5		\$ 100
						

Logout

Client Transactions page contains:

- All transactions happened for the current user i.e both incoming and outgoing credits.
- Transactions also contain a message show option.

Client-Profile :




Client-Profile page contains :

- First Name
- Last Name
- UserID
- Current Balance
- Account creation date

And much more information about the current user.

Admin Login :



MACRO BANK

MACRO

—

□

×

Choose Your Account Type:

ADMIN

Username:

Admin

Password:

.....

Login

Login into the Admin section by choosing ADMIN from the checkbox.

Admin-Create Client:

MACRO

MACRO

+ Create Client

≡ Clients

🏧 Deposit

🔗 Logout

Create New Client Account

First Name:

Mason

Last Name:

Mount

Password:

12345

Account ID:

@mMount

New Balance:

4000

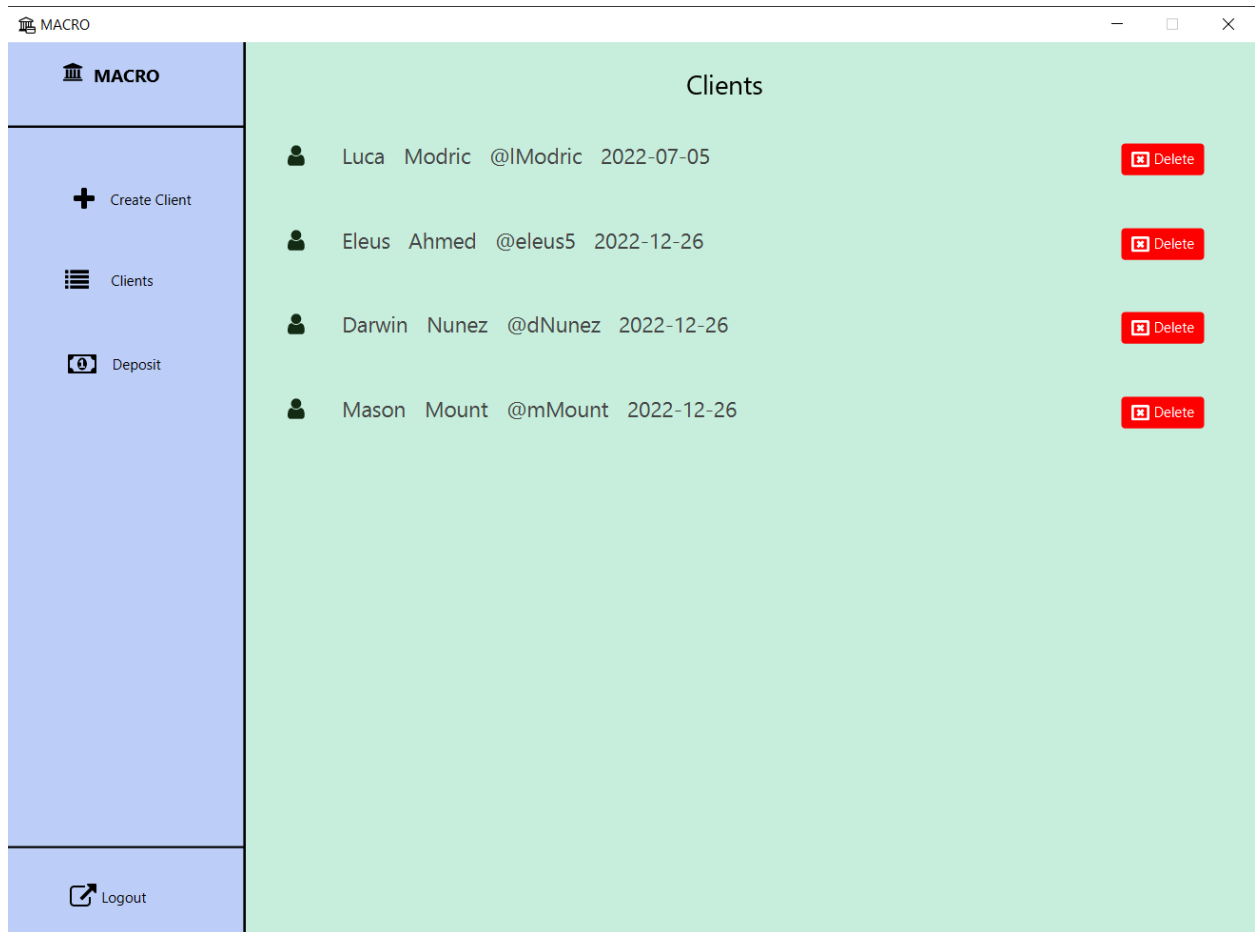
Create New Client

Successfully Created.

We can create new Clients from admin section by filling :

- First Name
- Last Name
- Password
- AccountID
- New Balance

Admin-Clientlist :



It is important for the admin to keep track of the clients.

In the Clients section a listview of clients with deleting functionality is implemented.

Admin-Deposit :

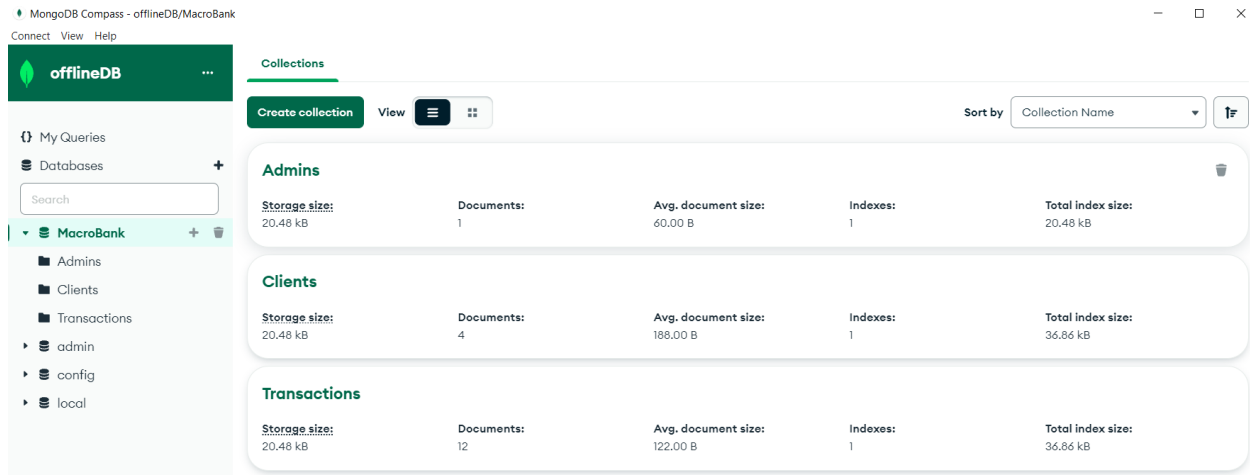
The screenshot displays a web application interface for a banking system. On the left is a blue sidebar with the 'MACRO' logo and navigation links: 'Create Client', 'Clients', 'Deposit', and 'Logout'. The main area has a light green background and is titled 'Deposit'. It features a search bar with the text 'Search by Account ID : @mMount' and a green 'Search' button. Below this, a client record is shown: 'Mason Mount @mMount 2022-12-26', with a red 'Delete' button. The 'Deposit Section' contains an 'Amount:' label, a text input field with '500', and a green 'Deposit' button.

An important feature of a banking App would be depositing credit In a client account.

In the Deposit section we can do just that.

1. Search the user with UserID
2. Enter amount to be deposited
3. Deposit with deposit button

MongoDB database :



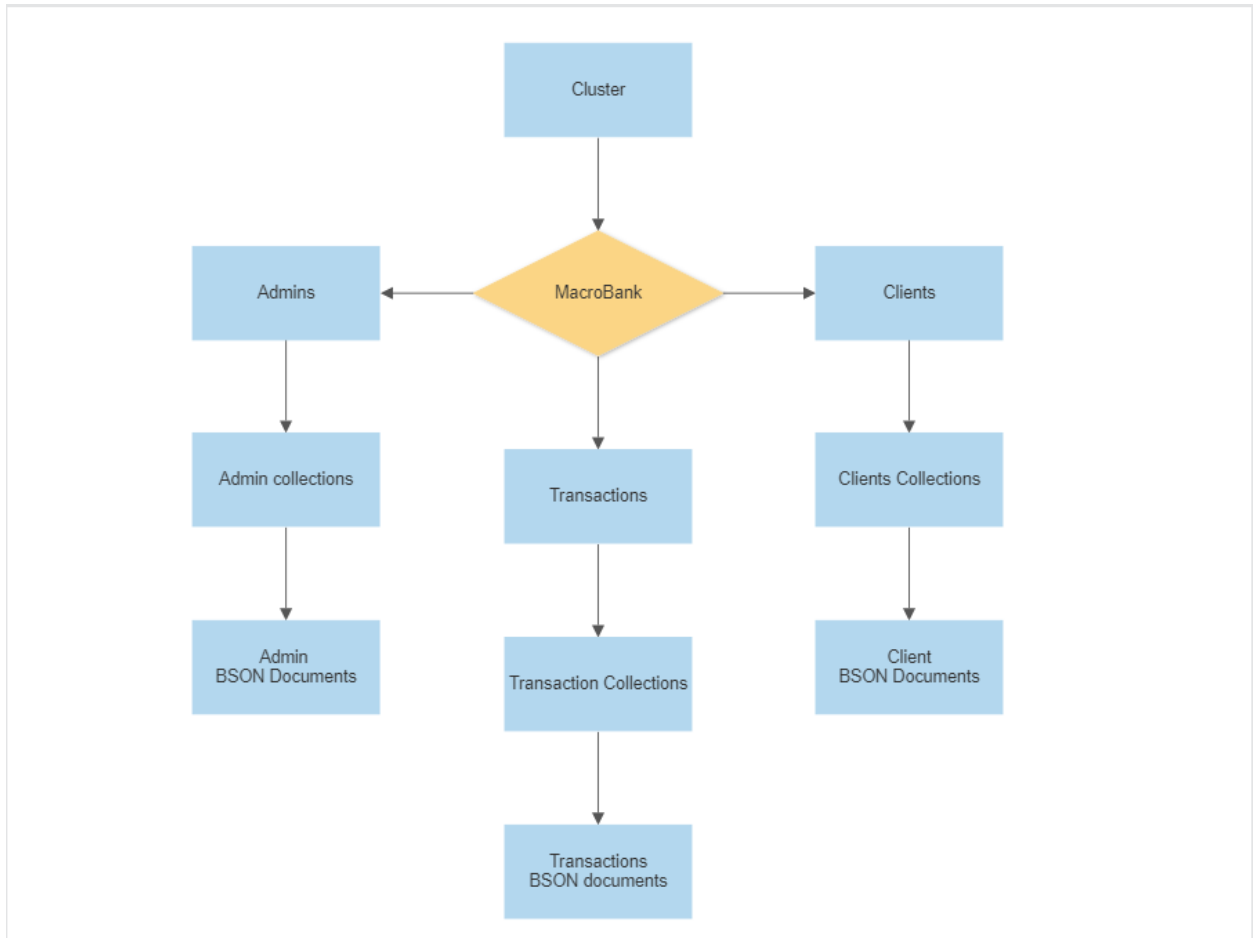
For the Macro program, a MacroBank named Cluster is created. In this Cluster

There is a database named Macrobank.

In the users database there are collections called admin which holds admin login coordinates and client information such as name,userId,created date,currentBalance etc.

information of the users. Information is held in BSON documents.

Database-flowchart :



MacroBank.Clients

Documents

Aggregations

Schema

Explain Plan

Filter 



Type a query: { field: 'value' }

 ADD DATA ▼

 EXPORT COLLECTION

```
_id: ObjectId('63a6ea6c7d1008a27237d719')
FirstName: "Luca"
LastName: "Modric"
Password: "12345"
Date: "2022-07-05"
UserID: "@lModric"
Balance: "1000"
total: "3000"
income: "300"
expense: "1300"
```

```
_id: ObjectId('63a890be5d3cd625af7ab925')
FirstName: "Eleus"
LastName: "Ahmed"
Password: "12345"
UserID: "@eleus5"
Date: "2022-12-26"
Balance: "4000"
total: "100"
income: "1100"
expense: "300"
```

```
_id: ObjectId('63a8baf34e35a47a248affa5')
FirstName: "Darwin"
```

MacroBank.Transactions

Documents

Aggregations

Schema

Explain Plan

Filter



Type a query: { field: 'value' }

ADD DATA

EXPORT COLLECTION

```
_id: ObjectId('63a7e4aac18c4c277d6806b2')
Sender: "@lModric"
Receiver: "@kBenzema"
Amount: "300"
Date: "2022-12-21"
Message: "Thanks"
```

```
_id: ObjectId('63a7e55bc18c4c277d6806b5')
Sender: "@lModric"
Receiver: "@kBenzema"
Amount: "500"
Date: "2022-12-21"
Message: "Thanks"
```

```
_id: ObjectId('63a80f4476e4547a6e6e7485')
Sender: "@lModric"
Receiver: "@kBenzema"
Amount: "500"
Date: "2022-12-25"
Message: "Hi there."
```

```
_id: ObjectId('63a85636803d1667da8fafcd')
```

Target vs Actual accomplishment :

Target :

- Making a banking software using Java programming.
- In the software users can transact with another client through server.
- User's confidential data will be saved in the database.
- User can use the software from anywhere in the world using internet.
- Previous transactions between users will be saved in the database.
So, when the login, it will be already available in dashboard window.
- Make it user-friendly like Bkash.
- Ability to request for deposit from admin.
- Ability to deposit funds to clients.
- Users can text other users.
- Make an individual profile and system for each user.
- Give admin the ability to create new clients and deposit funds.
- Add two step verifications for better security.
 - Updating data real-time and refreshing page after every operation.

- Making the Admin section more secured.

Accomplishment :

- Making a banking software using Java programming.
- In the software users can transact with another client through server.
- User's confidential data will be saved in the database.
- Previous transactions between users will be saved in the database. So, when the login, it will be already available in dashboard window.
- Ability to request for deposit from admin.
- Ability to deposit funds to clients.
- Users can text other users.
- Make an individual profile and system for each user.
- Give admin the ability to create new clients and deposit funds.

From above we can see that even though many things have been added to actual software, there is much left. From the target only have was able to be included in the software. The main function that could not be included was updating the page after every operation. Also the program could've been made more secure and can be introduced with a two step verification system.

RISK AND ISSUES :

Maco has some issues that can be solved by further development and updates over the period of time. Some of the issues or bugs can be seen in this version are:

- The UI needs to be more polished and responsive.
- The account balance may behave odd when the account balance becomes negative.
- Transfer can be done even when a client has no money.
- For searching user error can occur when we enter a invalid username or a client who has no account in our program.
- Page doesn't refresh every time an operation has been made
- Accounts can be created even leaving a text field blank.

Discussion and Conclusion :

The Macro banking app was inspired from the Bkash app and developed for learning purposes. It was made more user friendly with an attractive and easy dashboard section. Even though for the end product the program wasn't made secure and responsive enough. The program data may malfunction when out of bound operations are made and if

sequence is not maintained. These issues can be solved by further development and updates. But still the program will work fine with logical sequence.

In making MacroBank, Java's many core parts are learned. Like, Java basic, use of Maven module, use of JavaFX for making UI, access and use MongoDB database in Java projects. Some problems that we faced during development are user authentication, simultaneous transactions between users, creating a listview of the client list and listview of transaction history that were solved throughout developing the app.

References :

- (<https://docs.oracle.com/en>)
- (<https://docs.oracle.com/javase/8/javase-clienttechnologies.html>)
- <https://www.javatpoint.com/java-tutorial>
- (<https://www.geeksforgeeks.org/java>)
- (<https://www.youtube.com>)
- (<https://docs.mongodb.com/drivers/java/>)
- ([https://docs.oracle.com/javafx/2/api/javafx/scene/doc-](https://docs.oracle.com/javafx/2/api/javafx/scene/doc-files/cssref.html)
[files/cssref.html](https://docs.oracle.com/javafx/2/api/javafx/scene/doc-files/cssref.html))