

# **A PROJECT REPORT**

On

**“Mini Bank Management System with  
Authentication”**

**BACHELOR OF TECHNOLOGY**

In

**Artificial Intelligence and Machine  
Learning**

**Submitted By**

**Gautam Kumar (AIMLLT2203)**

**Under the supervision of  
Mr. Rikendra**



**G.L. Bajaj Institute Of Technology & Management  
Greater Noida**

Affiliated to



**Dr. APJ Abdul Kalam Technical University**

**Lucknow**

**(2022-23)**

## **Declaration**

---

We hereby declare that the project work presented in this report entitled “**Mini Bank Management System**”, in partial fulfilment of the requirement for the award of the degree of Bachelor of Technology in Artificial Intelligence and Machine Learning, submitted to A.P.J Abdul Kalam Technical University, Lucknow, is Based on my own work carried out at Department of Computer Science & Engineering G.L. Bajaj Institute of Technology & Management, Greater Noida. The work contained in the report is original and project work reported in this report has not been submitted by me/us for award of any degree or diploma.

**Name:** GAUTAM KUMAR

**Admission No.** – AIMLLT2203

**Date:** 25.02.2023

**Place:-** Greater Noida, 201310

**GUIDED BY :  
( MR. RIKENDRA)**

## **Certificate**

---

This is to certify that the project report entitled “**MINI BANK MANAGEMENT SYSTEM WITH AUTHENTICATION**” done by **Gautam Kumar (AIMLLT2203)** is an original work carried out by them in Department of Computer Science & Engineering G.L. Bajaj Institute of Technology & Management, Greater Noida under my guidance. The matter embodied in this project work has not been submitted earlier for the award of any degree or diploma to the best of my knowledge and belief.

**Date:**

**Mr. Rikendra**

**Signature of the Supervisor**

**Dr. Sansar Singh Chauhan**

**Head of Department**

# Acknowledgement

---

The merciful guidance bestowed to us by the almighty made us stick to this project to a successful end. We humbly pray with sincere heart for his guidance to continue forever.

We pay thanks to our project guide **Mr. Rikendra** who has given guidance and light to us during this project. His versatile knowledge has helped us in the critical times during the span of this project.

We pay special thanks to our Head of Department **Dr. Sansar Singh Chauhan** who has been always present as a support and help us in all possible ways during this project.

We also take this opportunity to express our gratitude to all those who have been directly and indirectly with us during the completion of the project.

At the last but not least thanks to all the faculty of CSE department who provided valuable suggestions during the period of project.

# Abstract

---

Bank management system is a commonly used technique in our modern world . This bank management system creates an authentication before you go for the transaction of your money.

The main motive behind this project is to create a register and login with logout system which creates authentication and gives security to our payments transaction.

The performance of our approach was evaluated using some clients information and storing it in the file which is being created during the registration or signup and after that the name and phone number of the client is being stored in that particular file .

The separate file is the main approach for creating this bank management system so that every file store's every clients data separately.

In conclusion, our approach provides a reliable and efficient solution for the security and safe transaction of money from the bank.

## Table of Content

|   |       |
|---|-------|
| Declaration .....   | (ii)  |
| Certificate .....   | (iii) |
| Acknowledgement .....   | (iv)  |
| Abstract .....  | (v)   |
| Table of content .....  | (vi)  |
| List of Figures .....   | (vii) |
|   |       |
| <b>Chapter 1.</b> Objectives .....                                      | 8     |
| <b>Chapter 2.</b> Background of the Study... ..                         | 9     |
| <b>Chapter 3.</b> Software requirements and external import files ..... | 10    |
| <b>Chapter 4.</b> Aims and Objectives of the Study .....                | 12    |
| <b>Chapter 5.</b> Activity Diagram for Bank Management System .....     | 13    |
| <b>Chapter 6.</b> Transaction Processing Diagram .....                  | 15    |
| <b>Chapter 7.</b> Bank Management System Use Case Diagram.....          | 17    |
| <b>Chapter 8.</b> Bank Management System Class Diagram.....             | 20    |
| 8.1 Flowchart .....   | 21    |
| <b>Chapter 9.</b> Bank Management System Sequence Diagram... ..         | 21    |
| <b>Chapter 10.</b> Bank Management System ER Diagram... ..              | 23    |
| <b>Chapter 11.</b> Implementation Setup... ..                           | 25    |
| <b>Chapter 12.</b> Use Case... ..                                       | 29    |
| <b>Chapter 13.</b> Result (Output) and Discussion ... ..                | 32    |
| <b>CODE</b> .....   | 36    |
| <b>Final Review of the Project</b> .....                                | 41    |
| <b>Conclusion</b> .....   | 42    |
| <b>References</b> .....   | 43    |
| <b>User Manual</b> .....  | 44    |

## List of Figures

---

|                     |   |    |
|---------------------|---|----|
| <b>Figure 5.1.</b>  | Activity Diagram.....                   | 13 |
| <b>Figure 6.1.</b>  | Data Flow Diagram.....                  | 15 |
| <b>Figure 7.1.</b>  | Use Case Diagram.....                   | 17 |
| <b>Figure 8.1.</b>  | BMS Class Diagram.....                  | 18 |
| <b>Figure 9.1.</b>  | BMS Sequence Diagram .....              | 20 |
| <b>Figure 10.1.</b> | Entity Relationship Diagram .....       | 23 |
| <b>Figure 11.1.</b> | Implementation Setup 1 .....            | 24 |
| <b>Figure 11.2.</b> | Implementation Setup 2 .....            | 24 |
| <b>Figure 11.3.</b> | Implementation Setup 3 .....            | 25 |
| <b>Figure 11.4.</b> | Implementation Setup 4 .....            | 25 |
| <b>Figure 11.5.</b> | Implementation Setup 5 .....            | 26 |
| <b>Figure 11.6.</b> | Implementation Setup 6 .....            | 26 |
| <b>Figure 11.7.</b> | Implementation Setup 7.....             | 27 |
| <b>Figure 11.8.</b> | Implementation Setup 8 .....            | 27 |
| <b>Figure 12.1.</b> | Banking System .....                    | 29 |
| <b>Figure 12.2.</b> | Login and Authentication Use Case ..... | 30 |
| <b>Figure 13.1.</b> | Result Output 1.....                    | 31 |
| <b>Figure 13.2.</b> | Result Output 2 .....                   | 31 |
| <b>Figure 13.3.</b> | Result Output 3 .....                   | 32 |
| <b>Figure 13.4.</b> | Result Output 4 .....                   | 32 |
| <b>Figure 13.5.</b> | Result Output 5 .....                   | 33 |
| <b>Figure 13.6.</b> | Result Output 6.....                    | 33 |
| <b>Figure 13.7.</b> | Result output 7 .....                   | 34 |

## **Chapter-1**

### **Objectives**

---

The goal of the bank management system project is to create an organic and optimal software of interaction between the various banking components. This is to maximize the profit of the banking components. This is to maximize the profit of the banking mechanism. The implementation of competent bank management procedures is significantly responsible for the successful optimization of the bank's productivity and activities.

The project's main goal is to create an online banking system for banks. All banking work is done manually in the current system. To withdraw or deposit money, the user must go to the bank. Today, it is also hard to find account information for people who have accounts in the banking system.



## **Chapter-2**

### **Background and scope of the Study**

---

Depending on the bank's policies, bank personnel and /or customers can utilize the Banking Management System. It can be utilize the Banking Management System. It can be utilized by multiple employees at the same time if they have the necessary permissions. Any web browser with a graphical interface can be used to access it.

Bank management system can **keep the information of account type, account opening form, deposit , and searching the transaction, transaction report, individual account opening form, group account as a record.** it displays records of transaction reports, statistical summary of account type and interest information.

# Software Requirements and external import files

---

1. **Vs code** - Visual Studio Code in short it is said VS Code which is an Virtual Studio Code's IntelliSense code completion gives programmers hints on the best possible function parameter, syntax, or variable. In addition, it can sense coding errors and offer suggestions to correct them.
2. **<iostream>** - iostream stands for input output header file system in c++ language where we use this header file for taking as input and show output.
3. **<stdlib>** - this header file is stands for standard library header files. This library file is used to create exit(0) function . This command makes you exit from your program running in the console.
4. **<string.h>** - The functions inside string.h header file is mainly used to locate last occurrence of character in string, searching from the end. It returns a pointer to the last occurrence of character in the C string str and many more.....
5. **<istream.h>** - The istream class handles the input stream in c++ programming language. These input stream objects are used to read and interpret the input as a sequence of characters. The cin handles the input. ostream class – The ostream class handles the output stream in c++ programming language. -
6. **<fstream.h>** - The fstream term stands for **File Stream**. Stream refers to a sequence of characters moving from the disk to the C++ program or from the C+ program to the disk. Moving characters from a file in disk to the program is inputting. Moving characters from the program to a file in the disk is outputting. -

## **Chapter-4**

### **Aims and Objectives**

---

The bank management system project in C++ and file system database is a project that allows you to save your money in a detailed account. By entering Name, Amount, Phone No, and other information, the system can organize your transaction list. The system's goal is to efficiently arrange bank account management.

The Basic Bank Management System is a simple console program that allows you to access all of the system's features by entering the system password. The user has several options in the system, including creating new client accounts, depositing dollars, withdrawing cash, and changing account information. The system will offer you the tools you need to manage your bank accounts. Your data will be saved as a data file extension by the system. The Simple Bank Management System was written in the C++ programming language and is suitable for newcomers to coding.

## **Chapter-5**

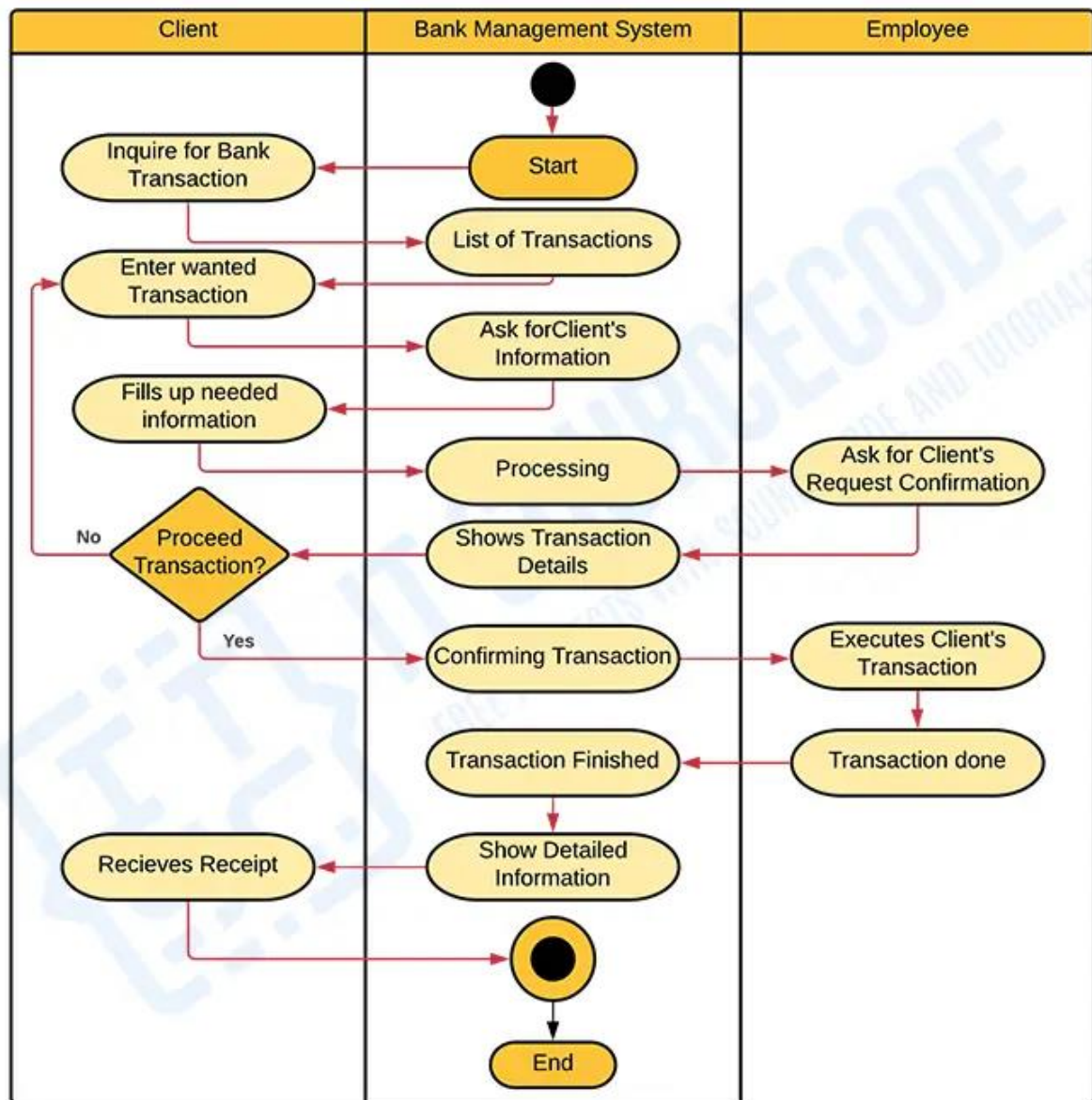
### **Activity Diagram for Bank Management System**

---

The bank management system activity diagram is a designed illustration that shows the system's behavioral aspect. It shows the bank management system's behavior in terms of responding to its users or clients.

It is designed for both customers and the admin of the Bank. It shows more detailed information on the interactions between the system and its users. This design is to explain and inform readers or users that the system provides security for the important transaction records and can only be accessed by authorized personnel.

# BANK MANAGEMENT SYSTEM



## ACTIVITY DIAGRAM

**Fig. 5.1**

## **Chapter-6**

### **Transaction Processing Diagram**

---

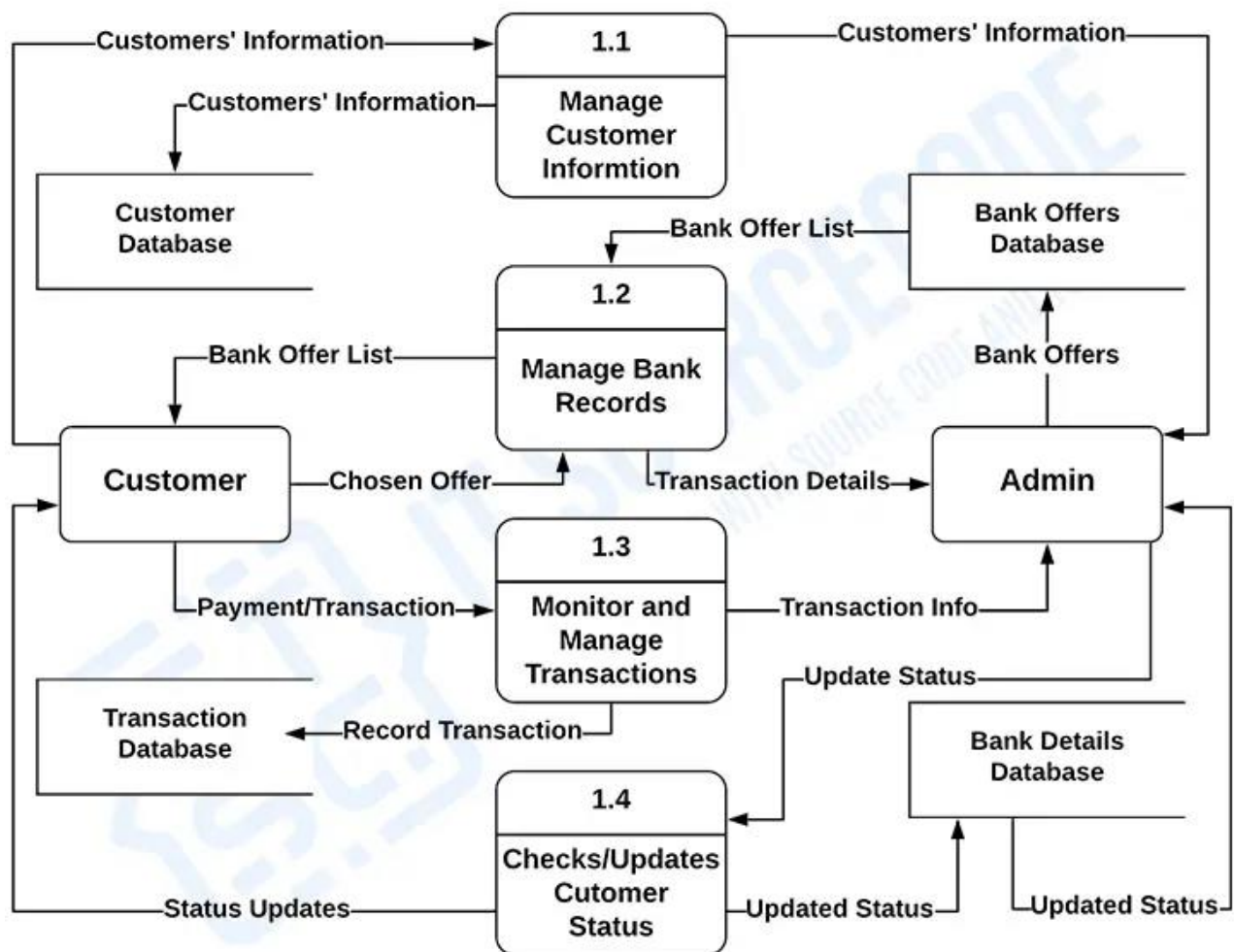
The Data Flow Diagram (DFD) represents the flow of data and the transformations in the Bank Management System. It discusses the overall definition of input, processing, and output.

There is a flow of how data passes and the input and output is being processed in all overall process is shown in below diagram with update status and the information of client and many more.....

The bank management system DFD has three levels explaining the content of the data flow diagram.

## 6.1 Flowchart

### BANK MANAGEMENT SYSTEM



### DATA FLOW DIAGRAM LEVEL 2

**Fig. 6.1**

### **Bank Management System Use Case Diagram**

This diagram discusses the meaning of the Bank Management System project UML as well as its use case diagram using include and extend.

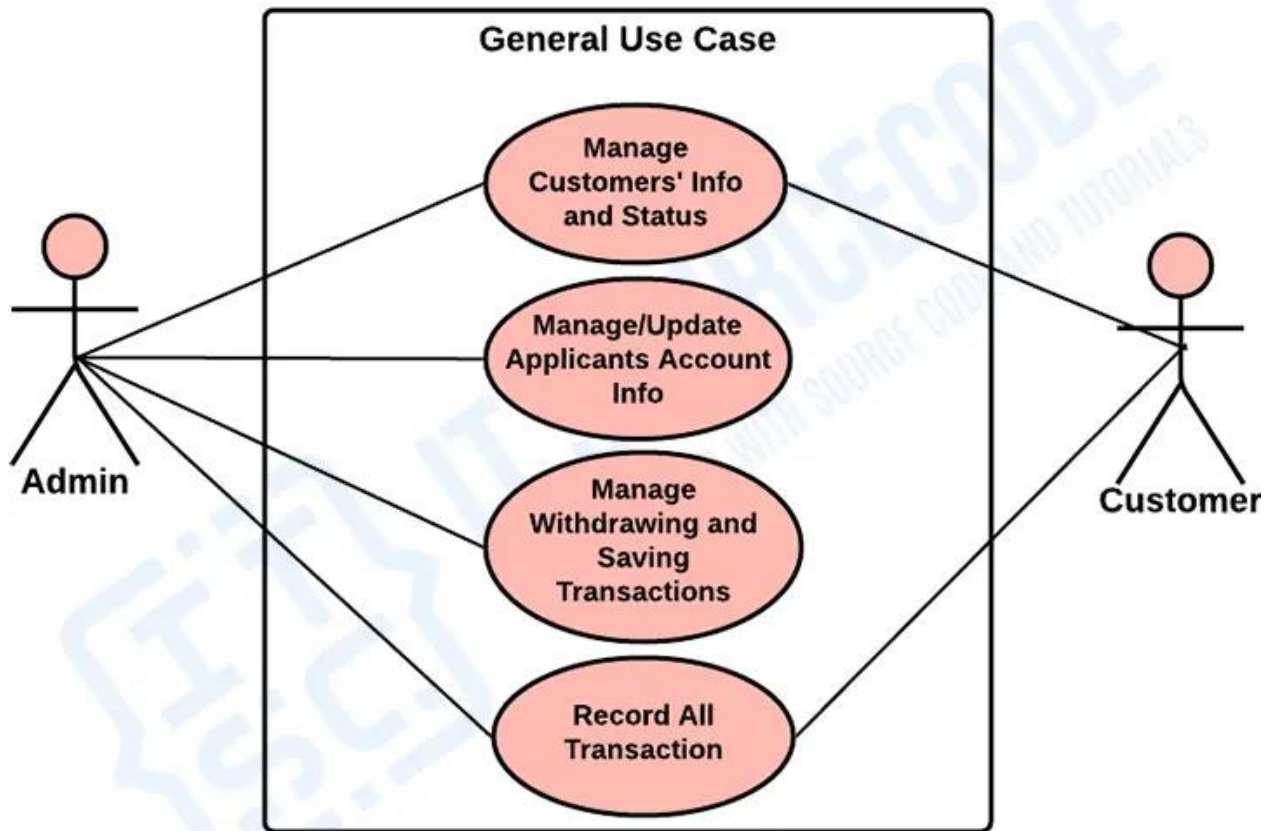
This use case diagram is a visual representation of how a user might interact with the bank management system. It depicts the system's numerous use cases and different sorts of users. The circles or ellipses are used to depict the use cases.

The best uses of this is the relation between customer and the admin of the bank account of that particular customer's account in the bank. The below diagram shows about how to manage customers info and maintain their status.

To manage/ update applicants account info like their phone number, ID proof number, any mistakes in name spelling and others. Next comes the main thing is to save transaction history into one's file so that they can track their overall money expenditure or transactions in a week/ month/ year. We have to keep those all records into a file management system or into a database that we will going to use it in next phase of this project.



# BANK MANAGEMENT SYSTEM

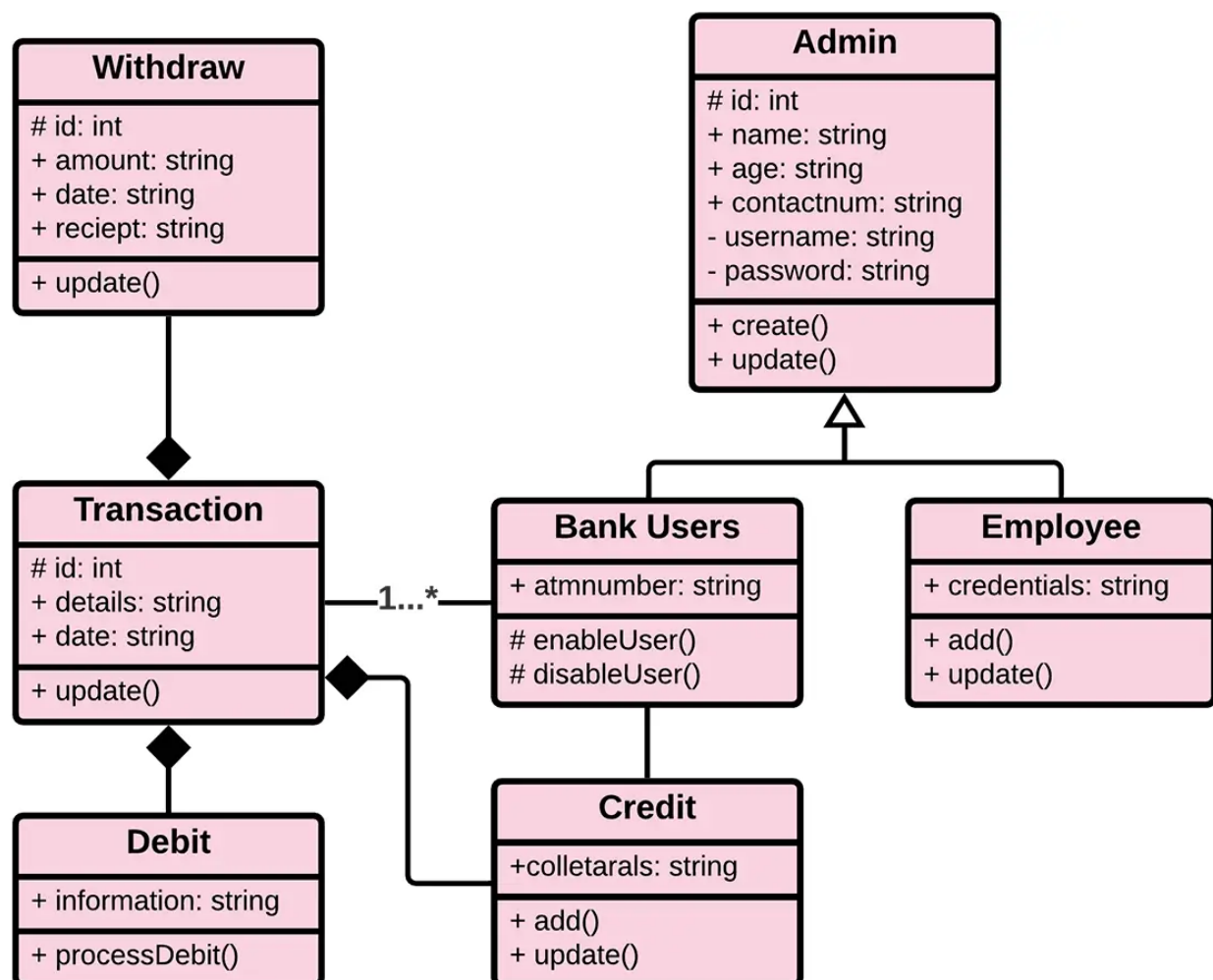


## USE CASE DIAGRAM

**Fig. 7.1**

### Bank Management System Class Diagram

The bank management system class diagram is a designed diagram that shows the system's relationships and classes. This UML Class Diagram is made to guide programmers along with the Bank management system development. It contains the class attributes, methods as well as the relationships between classes.



**Fig. 8.1**

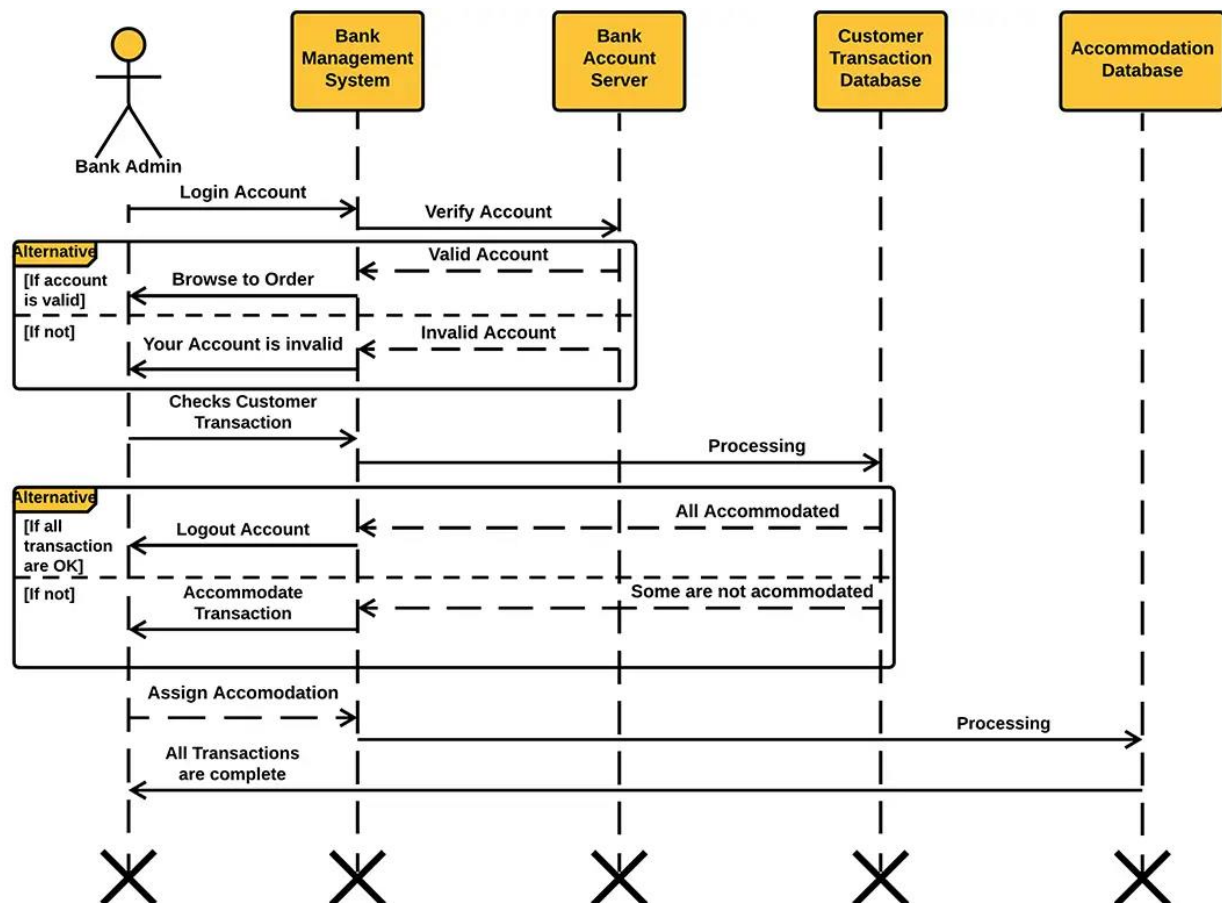
As you can see through the illustration, the classes were determined which is symbolized by

boxes. They were designated with their corresponding attributes and shows the class' methods. Their relationships are also plotted to show the connections between classes and their multiplicity.

There is a connectivity between all the classes used and the functions inside every class used. There is also some variables of local and global scoped variable which are to be used inside or outside of the class mentioning those as local or global variables. The arrow represents the connection between the classes and functions or variable that are used in the program.

### Bank Management System Sequence Diagram

This diagram gives enlightenment and guides the programmers and developers on how should they build the system. The idea presented in a sequence diagram will give efficiency on Bank Management System development.



**Fig. 9.1**

As you can see through the illustration, the conditions and interactions are emphasized, These interactions are essential for the Bank Management System development. The series of messages are shown and labeled to guide you in building a Bank Management System. You can modify the design if you have more ideas. You can also add more features to this design and use it as your project blueprint.

Here the map of the transaction history is being tried to show of how a transaction has been processed and the connection between customer and database with their username and password has been stored with authentication so that no one can steal their password or see their password.

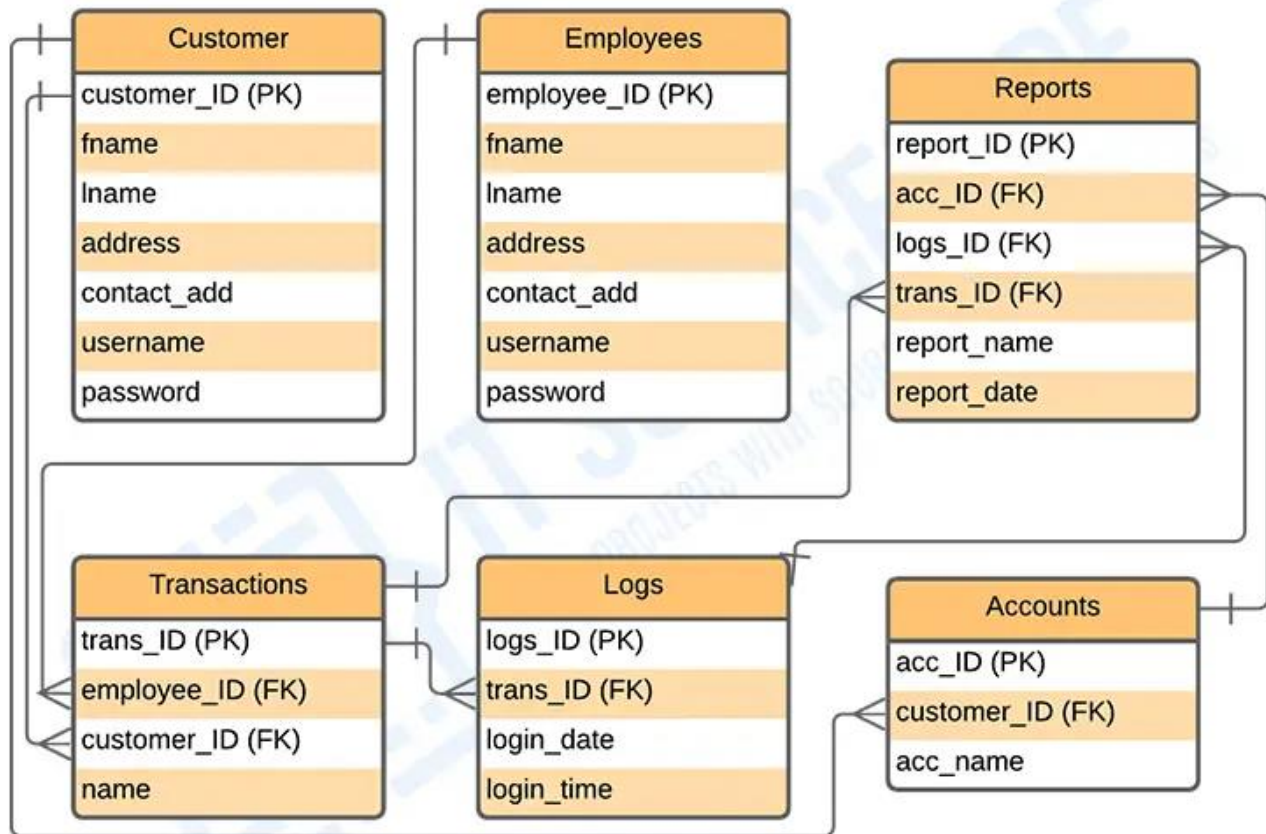
### **Bank Management System ER Diagram**

The ER diagram for the bank management system was made based on bank requirements. It can encode customer information and banking transactions.

The admin can have access to the customer status and information for the important transactions. They can handle the data needed in managing customer and employee files as well as the transactions made by the customer and staff.

The features included in the system ER diagram were the security and monitoring of the customer records, transactions, and status. These features were also listed and recorded in reports that served as the history of transactions done in the system.

# BANK MANAGEMENT SYSTEM

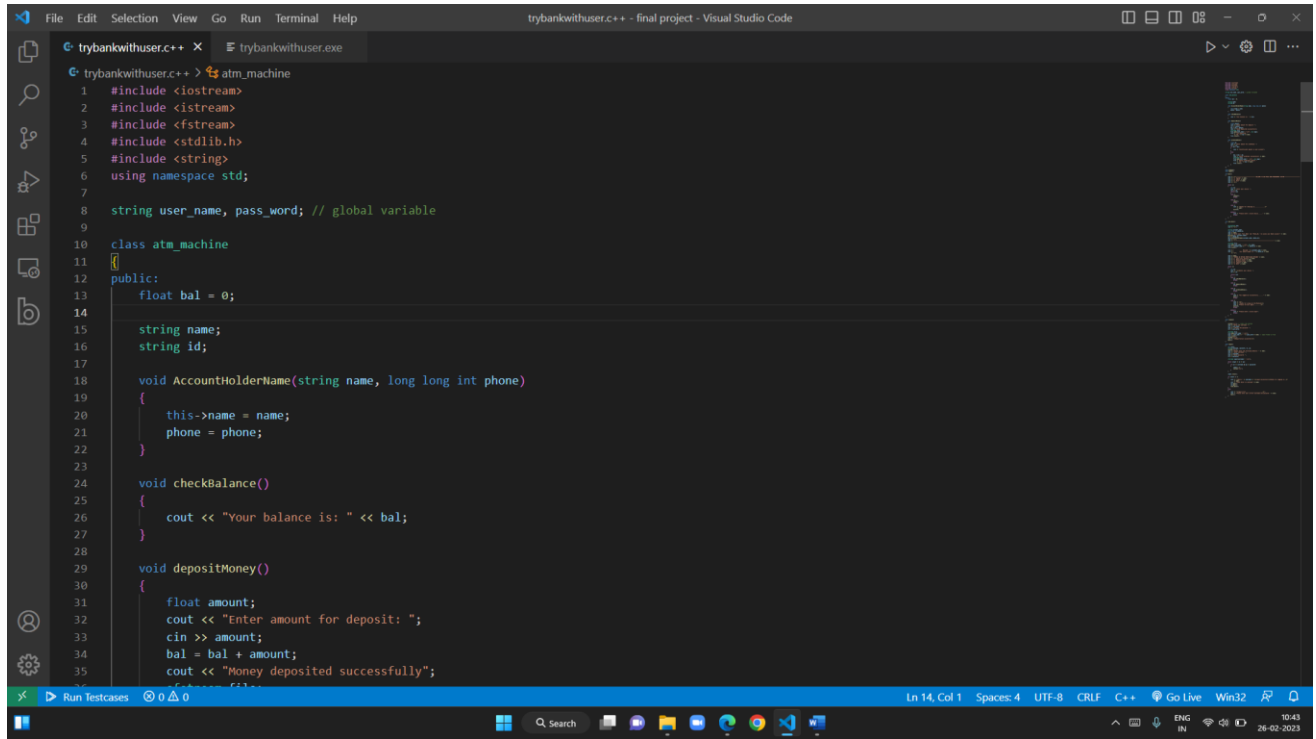


## ENTITY RELATIONSHIP DIAGRAM

**Fig. 10.1**

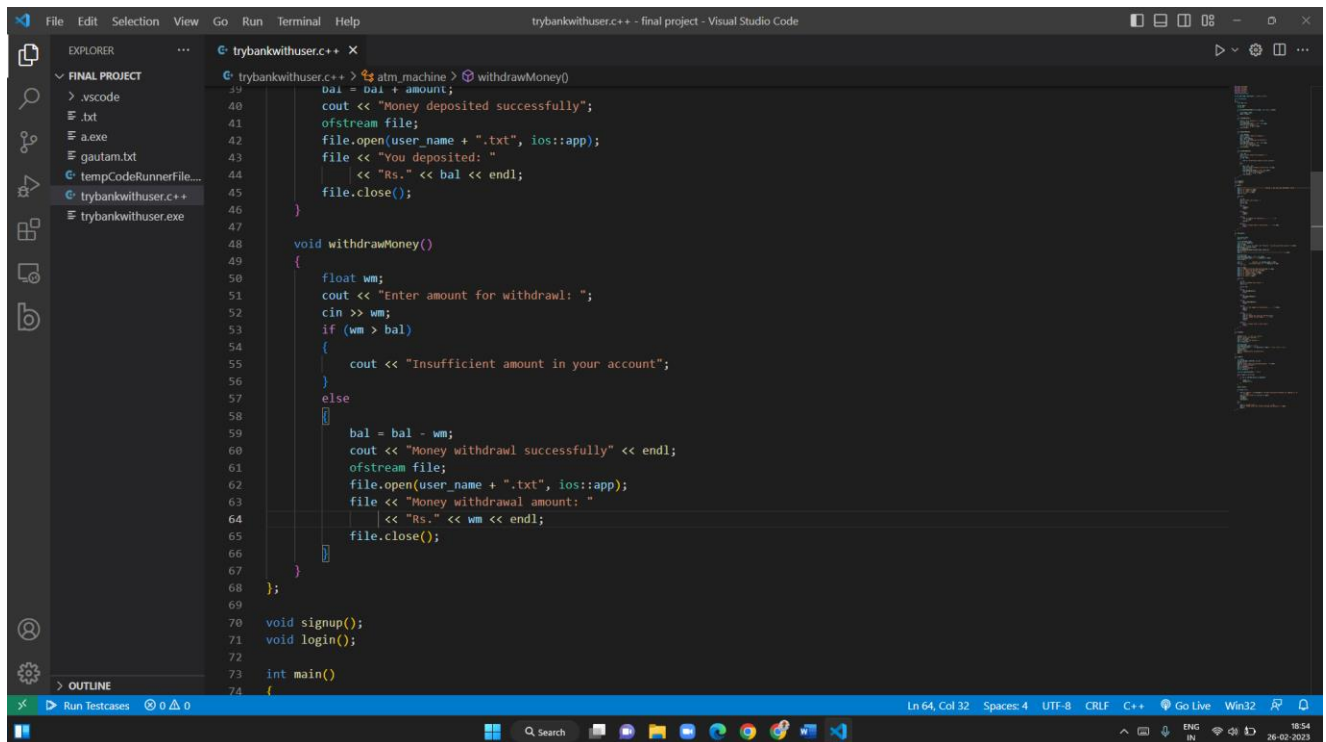
## Chapter-11

### Implementation Setup (CODE)



```
trybankwithuser.cpp - final project - Visual Studio Code
trybankwithuser.cpp X atm_machine
1 #include <iostream>
2 #include <istream>
3 #include <fstream>
4 #include <stdlib.h>
5 #include <string>
6 using namespace std;
7
8 string user_name, pass_word; // global variable
9
10 class atm_machine
11 {
12 public:
13     float bal = 0;
14
15     string name;
16     string id;
17
18     void AccountHolderName(string name, long long int phone)
19     {
20         this->name = name;
21         phone = phone;
22     }
23
24     void checkBalance()
25     {
26         cout << "Your balance is: " << bal;
27     }
28
29     void depositMoney()
30     {
31         float amount;
32         cout << "Enter amount for deposit: ";
33         cin >> amount;
34         bal = bal + amount;
35         cout << "Money deposited successfully";
36     }
```

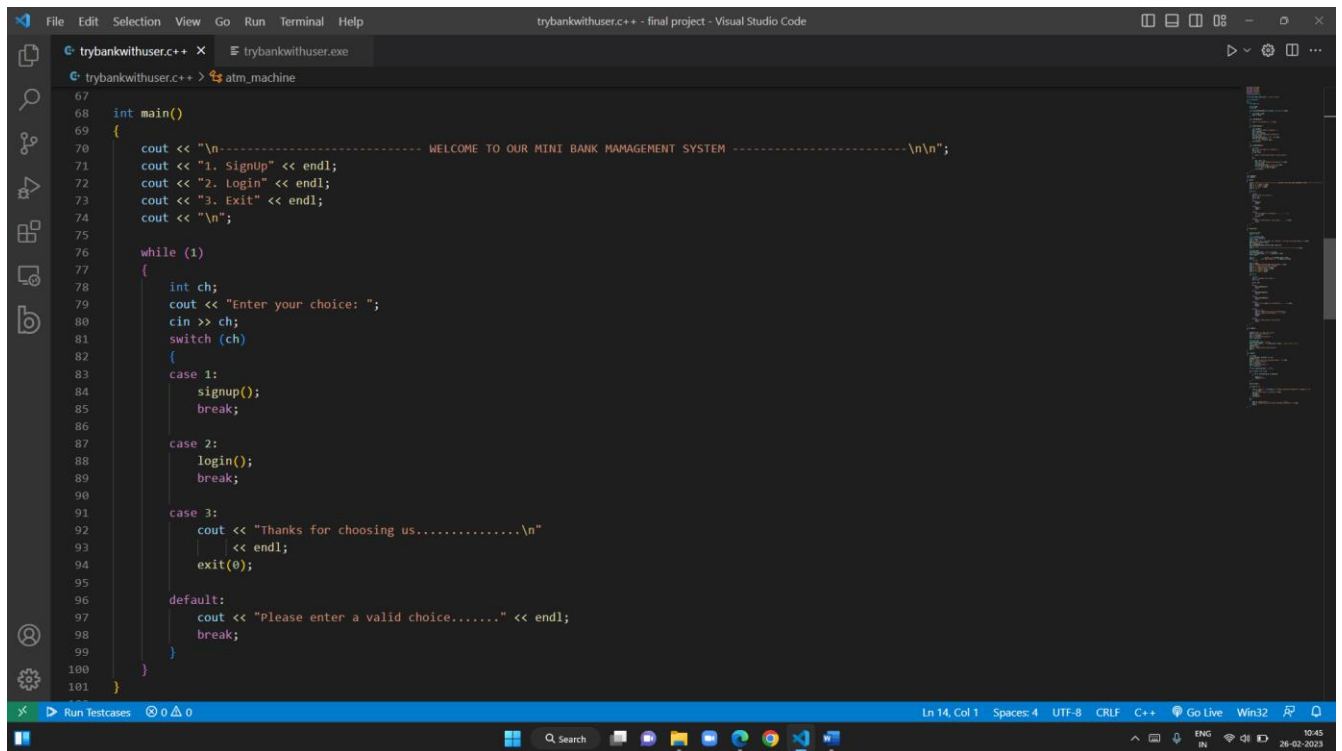
**Fig. 11.1**



```
trybankwithuser.cpp - final project - Visual Studio Code
trybankwithuser.cpp X atm_machine > withdrawMoney()
39 bal = bal + amount;
40 cout << "Money deposited successfully";
41 ofstream file;
42 file.open(user_name + ".txt", ios::app);
43 file << "You deposited: "
44     << "Rs." << bal << endl;
45 file.close();
46
47
48 void withdrawMoney()
49 {
50     float wm;
51     cout << "Enter amount for withdrawl: ";
52     cin >> wm;
53     if (wm > bal)
54     {
55         cout << "Insufficient amount in your account";
56     }
57     else
58     {
59         bal = bal - wm;
60         cout << "Money withdrawl successfully" << endl;
61         ofstream file;
62         file.open(user_name + ".txt", ios::app);
63         file << "Money withdrawal amount: "
64             << "Rs." << wm << endl;
65         file.close();
66     }
67 }
68
69 void signup();
70 void login();
71
72 int main()
73 {
```

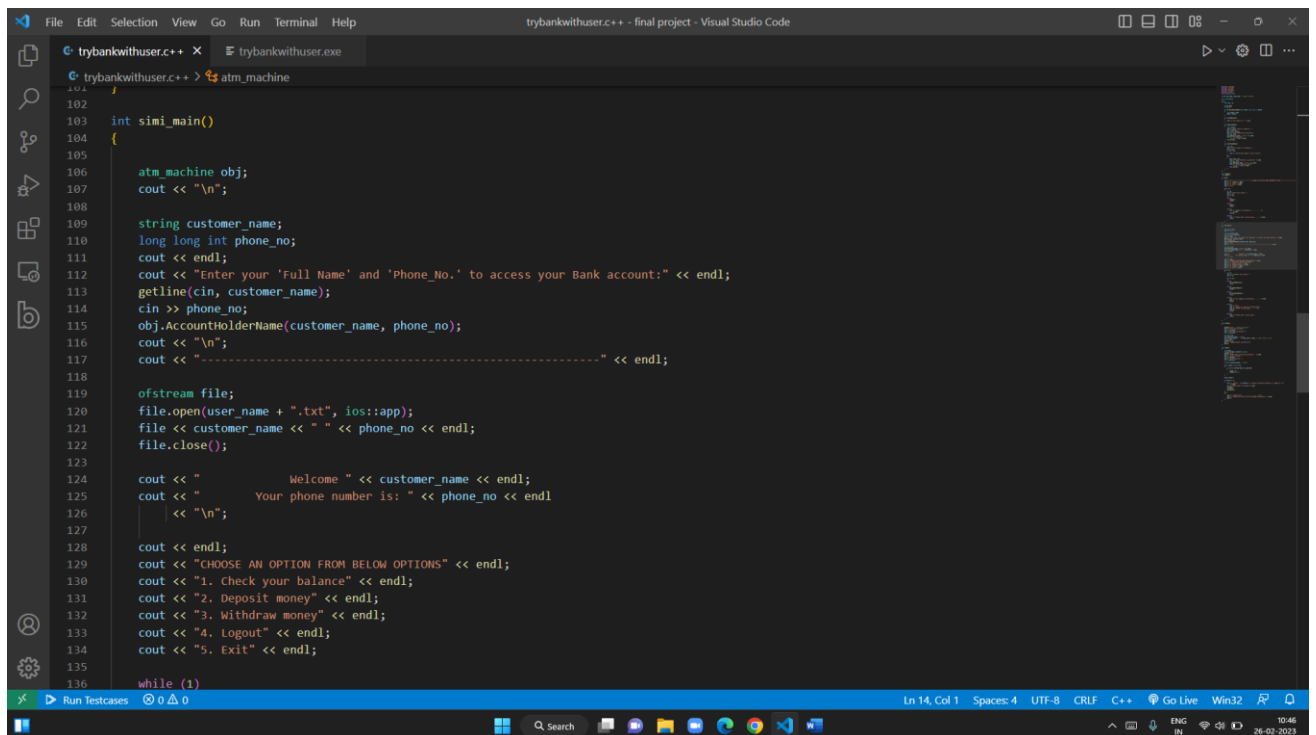
**Fig. 11.2**





```
67
68 int main()
69 {
70     cout << "\n----- WELCOME TO OUR MINI BANK MANAGEMENT SYSTEM ----- \n\n";
71     cout << "1. Signup" << endl;
72     cout << "2. Login" << endl;
73     cout << "3. Exit" << endl;
74     cout << "\n";
75
76     while (1)
77     {
78         int ch;
79         cout << "Enter your choice: ";
80         cin >> ch;
81         switch (ch)
82         {
83             case 1:
84                 signup();
85                 break;
86
87             case 2:
88                 login();
89                 break;
90
91             case 3:
92                 cout << "Thanks for choosing us.....\n"
93                 << endl;
94                 exit(0);
95
96             default:
97                 cout << "Please enter a valid choice....." << endl;
98                 break;
99         }
100     }
101 }
```

**Fig. 11.3**



```
102
103 int simi_main()
104 {
105
106     atm_machine obj;
107     cout << "\n";
108
109     string customer_name;
110     long long int phone_no;
111     cout << endl;
112     cout << "Enter your 'Full Name' and 'Phone_No.' to access your Bank account:" << endl;
113     getline(cin, customer_name);
114     cin >> phone_no;
115     obj.AccountHolderName(customer_name, phone_no);
116     cout << "\n";
117     cout << "-----" << endl;
118
119     ofstream file;
120     file.open(user_name + ".txt", ios::app);
121     file << customer_name << " " << phone_no << endl;
122     file.close();
123
124     cout << "        Welcome " << customer_name << endl;
125     cout << "        Your phone number is: " << phone_no << endl
126     << "\n";
127
128     cout << endl;
129     cout << "CHOOSE AN OPTION FROM BELOW OPTIONS" << endl;
130     cout << "1. Check your balance" << endl;
131     cout << "2. Deposit money" << endl;
132     cout << "3. Withdraw money" << endl;
133     cout << "4. Logout" << endl;
134     cout << "5. Exit" << endl;
135
136     while (1)
```

**Fig. 11.4**

```
135
136
137 while (1)
138 {
139     int ch;
140     cout << "\n\nEnter your choice: ";
141     cin >> ch;
142
143     switch (ch)
144     {
145     case 1:
146         obj.checkBalance();
147         break;
148     case 2:
149         obj.depositMoney();
150         break;
151     case 3:
152         obj.withdrawMoney();
153         break;
154     case 4:
155         cout << "You logged out successfully....." << endl;
156         main();
157         break;
158     case 5:
159         cout << "\n";
160         cout << "Thanks for using our Atm Machine\n";
161         cout << "Please re-visit again.....\n";
162         exit(0);
163         break;
164     default:
165         cout << "Please enter a valid input";
166     }
```

**Fig. 11.5**

```
167
168
169     default:
170         cout << "Please enter a valid input";
171         break;
172 }
173
174
175 void signup()
176 {
177
178     system("cls"); // clears the console
179     cout << "Enter the username: ";
180     cin >> user_name;
181     cout << "\nEnter the password: ";
182     cin >> pass_word;
183
184     ofstream file;
185     file.open(user_name + ".txt");
186     file << user_name << " " << pass_word << endl; // input format in file
187     file.close();
188     system("cls");
189     cout << "\nRegistration successfull\n";
190     main();
191 }
192
193 void login()
194 {
195     int count;
196     string username, password, un, pw;
197     system("cls");
198     cout << "Please enter the following details: " << endl;
199     cout << "Enter username: ";
200     cin >> username;
201     cout << "Enter password: ";
202     cin >> password;
```

**Fig. 11.6**

The screenshot shows the Visual Studio Code editor with the file `trybankwithuser.c++` open. The code defines a `login()` function. It starts by clearing the screen and prompting the user to enter details. It then reads the username and password. A while loop checks if the entered credentials match the stored ones in `username.txt`. If they match, it increments a counter, clears the screen, and prints a success message. If not, it prints an error message and prompts the user to re-enter their credentials. The function ends by calling `simi_main()`.

```
192 void login()
193 {
194     int count;
195     string username, password, un, pw;
196     system("cls");
197     cout << "Please enter the following details: " << endl;
198     cout << "Enter username: ";
199     cin >> username;
200     cout << "Enter password: ";
201     cin >> password;
202
203     ifstream input(username + ".txt");
204
205     while (input >> un >> pw)
206     {
207         if (un == username && pw == password)
208         {
209             count = 1;
210             system("cls");
211         }
212     }
213
214     input.close();
215
216     if (count == 1)
217     {
218         cout << "\nHello " << username << "\nLogin Successfull\nThanks for logging in.\n"
219             << endl;
220         cout << "Press Enter to continue" << endl;
221         cin.get();
222         cin.get();
223         simi_main();
224     }
225     else
226     {
227         cout << "\nlogin error.....\n";
228         cout << "Please enter your correct username and password." << endl;
229         main();
230     }
231 }
232
233
```

**Fig. 11.7**

The screenshot shows the Visual Studio Code editor with the file `trybankwithuser.c++` open. The code defines a `login()` function. It starts by clearing the screen and prompting the user to enter details. It then reads the username and password. A while loop checks if the entered credentials match the stored ones in `username.txt`. If they match, it increments a counter, clears the screen, and prints a success message. If not, it prints an error message and prompts the user to re-enter their credentials. The function ends by calling `simi_main()`.

```
201 cout << "Enter password: ";
202 cin >> password;
203
204 ifstream input(username + ".txt");
205
206 while (input >> un >> pw)
207 {
208     if (un == username && pw == password)
209     {
210         count = 1;
211         system("cls");
212     }
213 }
214
215 input.close();
216
217 if (count == 1)
218 {
219     cout << "\nHello " << username << "\nLogin Successfull\nThanks for logging in.\n"
220         << endl;
221     cout << "Press Enter to continue" << endl;
222     cin.get();
223     cin.get();
224     simi_main();
225 }
226 else
227 {
228     cout << "\nlogin error.....\n";
229     cout << "Please enter your correct username and password." << endl;
230     main();
231 }
232
233
```

**Fig. 11.8**

## **Chapter-12**

### **Use Case**

A Customer is required to create an account to avail services offered by Bank. Bank verifies detail and creates new account for each new customer. Each customer is an actor for the Use-Case Diagram and the functionality offered by Online Banking System to Add Account is Use-Case.

Each customer can check the balance in bank account and initiate request to transfer an account across distinct branches of Bank. Cashier is an employee at bank who supports service to the customer.

A customer can execute cash transactions where the customer must either add cash value to bank account or withdraw cash from account. Either of two or both that is credit as well as debit cash, might be executed to successfully execute one or multiple transactions.

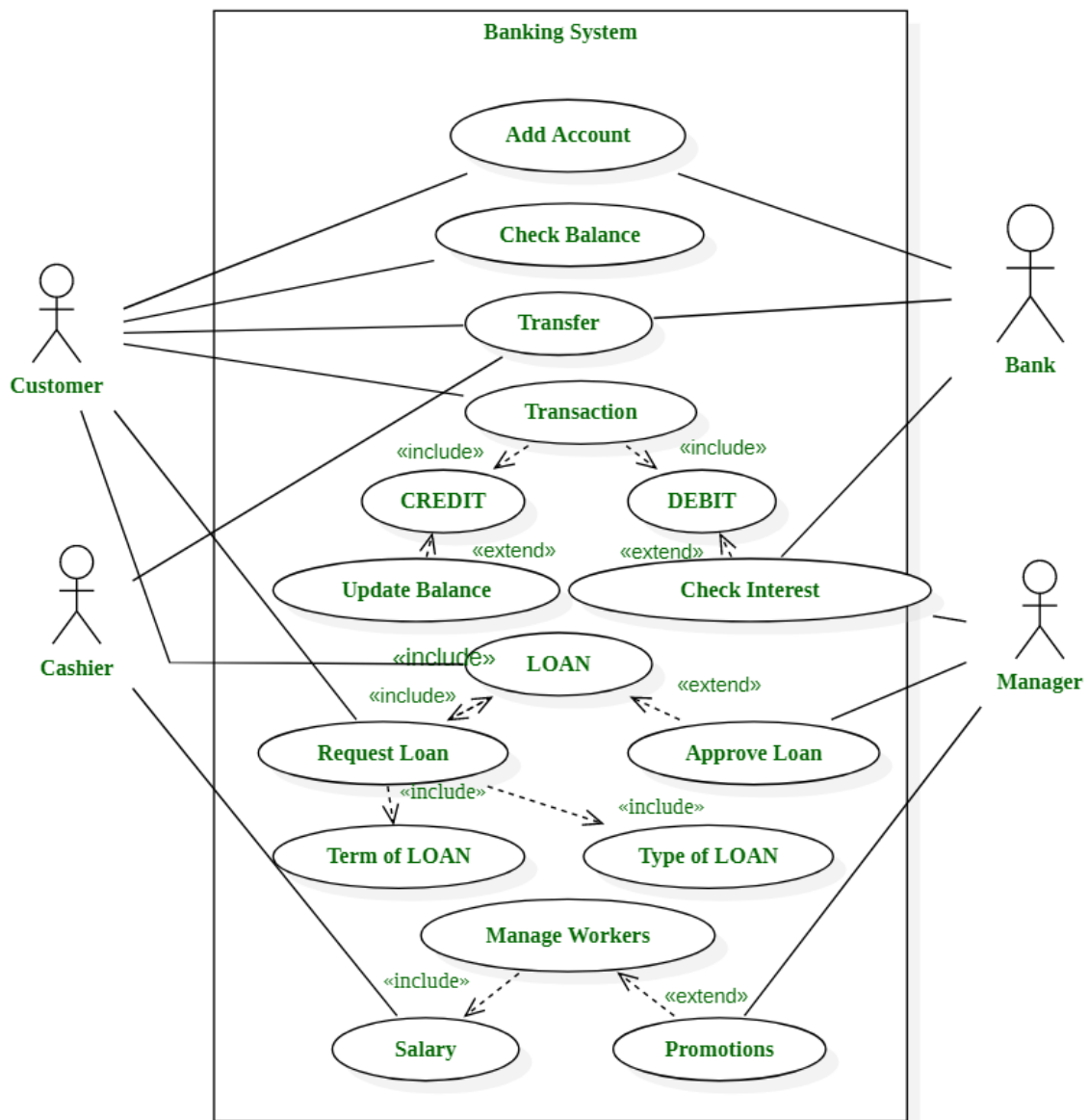
After each successful transaction customer might or might not want to get details for action. Manager can check interest value for each account corresponding to transaction to ensure and authenticate details.

A customer can also request loan from bank where customer must add request for loan with the appropriate details.

The type of loan in accordance with purpose or the need for loan and term or duration to pay back the loan must be provided by customer.

The manager of each branch of bank has choice to either accept or approve loan to initiate process further or just reject request for loan based on terms and conditions.

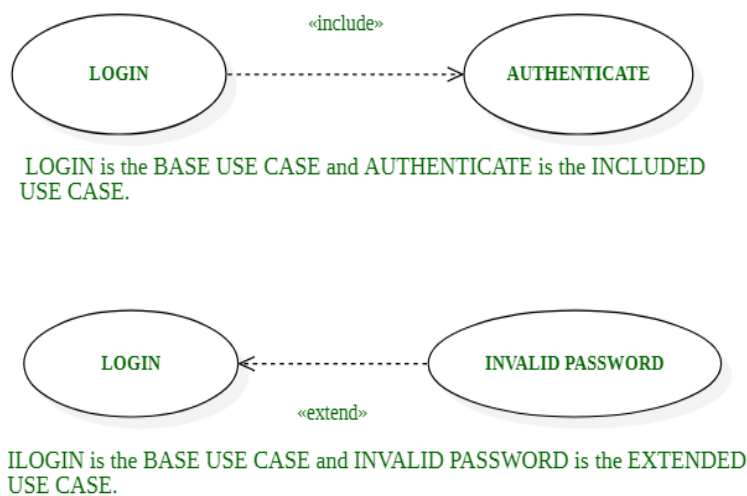
The record for each employee of bank is maintained by bank and bank manages all employees of each branch of bank. The manager of each branch has choice to offer bonus to employees. Note here that each employee is paid as part of management of staff but promotion or bonus might or might not be offered certainly to each employee.



**Fig. 12.1**

A user must provide appropriate details to securely Login. Software must check and verify the details at every attempt to Login. Here LOGIN is the Base Use Case and AUTHENTICATE is the Included Use Case.

If a user enters appropriate details , user is allowed to Login. However if the details entered by the user are incorrect, software must be able to catch and display problem to the user and allow the user to re-enter details. LOGIN is hence a complete use case. However under certain situations it might use action corresponding to INVALID PASSWORD. Here LOGIN is the Base Use Case and INVALID PASSWORD is Extended Use Case.

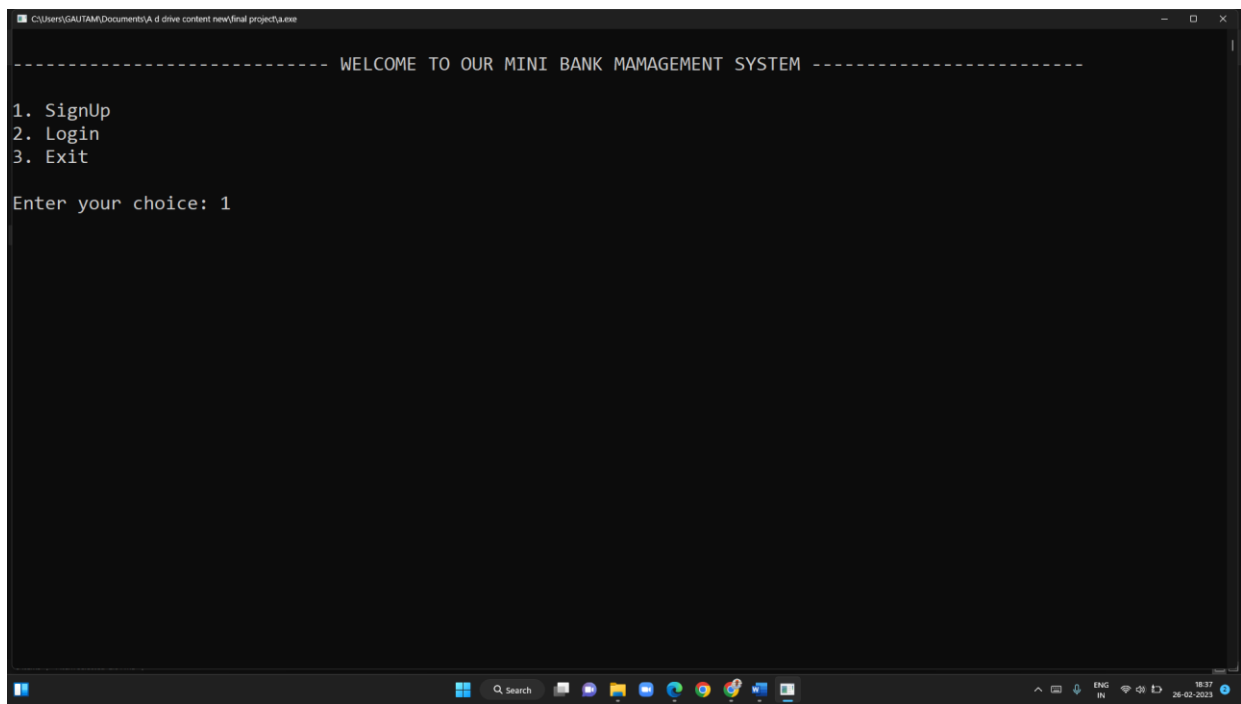


**Fig. 12.2**

## Chapter-13

### Result and Discussion

---



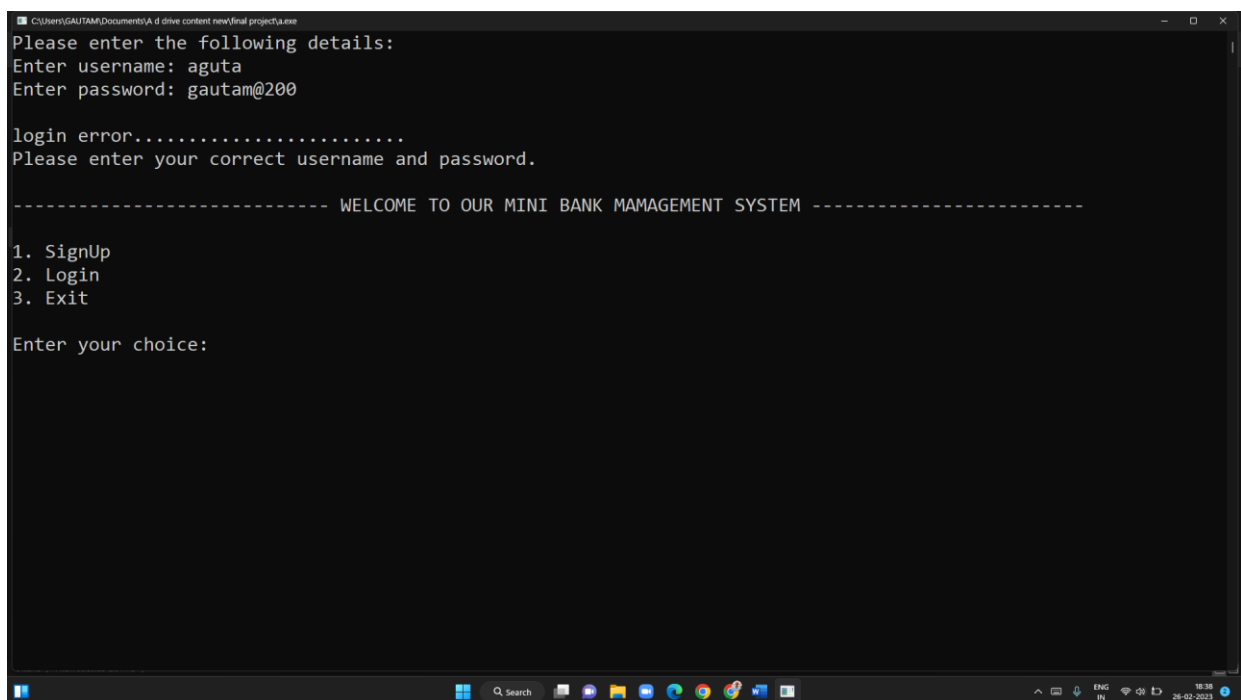
```
C:\Users\GAUTAM\Documents\A d drive content new\final project\>.exe

----- WELCOME TO OUR MINI BANK MANAGEMENT SYSTEM -----

1. SignUp
2. Login
3. Exit

Enter your choice: 1
```

Fig. 13.1



```
C:\Users\GAUTAM\Documents\A d drive content new\final project\>.exe

Please enter the following details:
Enter username: aguta
Enter password: gautam@200

login error.....
Please enter your correct username and password.

----- WELCOME TO OUR MINI BANK MANAGEMENT SYSTEM -----

1. SignUp
2. Login
3. Exit

Enter your choice:
```

Fig. 13.2

```
C:\Users\GAUTAM\Documents\A d drive content new\final project\main.exe

Hello gautam
<Login Successfull>
Thanks for logging in..

Press Enter to continue

Enter your 'Full Name' and 'Phone_No.' to access your Bank account:
Gautam kumar
9711329779

-----
Welcome Gautam kumar
Your phone number is: 9711329779

CHOOSE AN OPTION FROM BELOW OPTIONS
1. Check your balance
2. Deposit money
3. Withdraw money
4. Logout
5. Exit

Enter your choice: _
```

**Fig. 13.3**

```
C:\Users\GAUTAM\Documents\A d drive content new\final project\main.exe

Enter your choice: 1
Your balance is: 0

Enter your choice: 2
Enter amount for deposit: 5000
Money deposited successfully

Enter your choice: 3
Enter amount for withdrawl: 2000
Money withdrawl successfully

Enter your choice: 1
Your balance is: 3000

Enter your choice: 2
Enter amount for deposit: 7500
Money deposited successfully

Enter your choice: 2
Enter amount for deposit: 500
Money deposited successfully

Enter your choice: 1
Your balance is: 11000

Enter your choice: _
```

**Fig. 13.4**



```
C:\Users\GAUTAM\Documents\A d drive content new\final project\u.exe
Enter your choice: 2
Enter amount for deposit: 7500
Money deposited successfully

Enter your choice: 2
Enter amount for deposit: 500
Money deposited successfully

Enter your choice: 1
Your balance is: 11000

Enter your choice: 3
Enter amount for withdrawl: 15000
Insufficient amount in your account

Enter your choice: 1
Your balance is: 11000

Enter your choice: 3
Enter amount for withdrawl: 7500
Money withdrawl successfully

Enter your choice: 1
Your balance is: 3500

Enter your choice:
```

**Fig. 13.5**

```
C:\Users\GAUTAM\Documents\A d drive content new\final project\u.exe
Your balance is: 11000

Enter your choice: 3
Enter amount for withdrawl: 15000
Insufficient amount in your account

Enter your choice: 1
Your balance is: 11000

Enter your choice: 3
Enter amount for withdrawl: 7500
Money withdrawl successfully

Enter your choice: 1
Your balance is: 3500

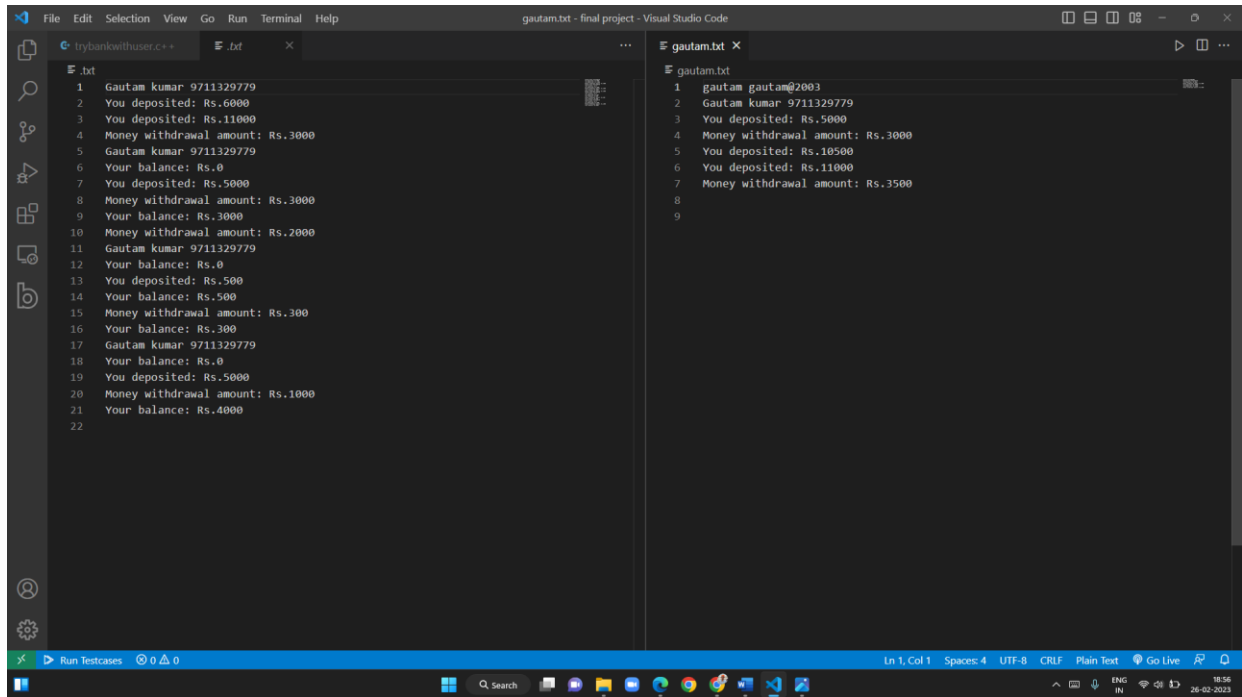
Enter your choice: 4
You logged out successfully.....

----- WELCOME TO OUR MINI BANK MANAGEMENT SYSTEM -----

1. SignUp
2. Login
3. Exit

Enter your choice: _
```

**Fig. 13.6**



**Fig. 13.7**

## CODE

```
#include <iostream>
#include <istream>
#include <fstream>
#include <stdlib.h>
#include <string>
using namespace std;

string user_name, pass_word; // global variable

class atm_machine
{
public:
    float bal = 0;

    string name;
    string id;

    void AccountHolderName(string name, long long int phone)
    {
        this->name = name;
        phone = phone;
    }

    void checkBalance()
    {
        cout << "Your balance is: " << bal;
        ofstream file;
        file.open(user_name + ".txt", ios::app);
        file << "Your balance: "
            << "Rs." << bal << endl;
        file.close();
    }

    void depositMoney()
    {
        float amount;
        cout << "Enter amount for deposit: ";
        cin >> amount;
        bal = bal + amount;
        cout << "Money deposited successfully";
        ofstream file;
        file.open(user_name + ".txt", ios::app);
        file << "You deposited: "
            << "Rs." << bal << endl;
        file.close();
    }
}
```

```

void withdrawMoney()
{
    float wm;
    cout << "Enter amount for withdrawl: ";
    cin >> wm;
    if (wm > bal)
    {
        cout << "Insufficient amount in your account";
    }
    else
    {
        bal = bal - wm;
        cout << "Money withdrawl successfully" << endl;
        ofstream file;
        file.open(user_name + ".txt", ios::app);
        file << "Money withdrawal amount: "
            << "Rs." << wm << endl;
        file.close();
    }
}

};

void signup();
void login();

int main()
{
    cout << "\n----- WELCOME TO OUR MINI BANK MAMAGEMENT
SYSTEM -----\\n\\n";
    cout << "1. SignUp" << endl;
    cout << "2. Login" << endl;
    cout << "3. Exit" << endl;
    cout << "\\n";

    while (1)
    {
        int ch;
        cout << "Enter your choice: ";
        cin >> ch;
        switch (ch)
        {
            case 1:
                signup();
                break;

            case 2:
                login();
                break;

```

```

        case 3:
            cout << "Thanks for choosing us.....\n"
                << endl;
            exit(0);

        default:
            cout << "Please enter a valid choice....." << endl;
            break;
    }
}

int simi_main()
{
    atm_machine obj;
    cout << "\n";

    string customer_name;
    long long int phone_no;
    cout << endl;
    cout << "Enter your 'Full Name' and 'Phone_No.' to access your Bank
account:" << endl;
    getline(cin, customer_name);
    cin >> phone_no;
    obj.AccountHolderName(customer_name, phone_no);
    cout << "\n";
    cout << "-----" <<
endl;

    ofstream file;
    file.open(user_name + ".txt", ios::app);
    file << customer_name << " " << phone_no << endl;
    file.close();

    cout << "            Welcome " << customer_name << endl;
    cout << "            Your phone number is: " << phone_no << endl
        << "\n";

    cout << endl;
    cout << "CHOOSE AN OPTION FROM BELOW OPTIONS" << endl;
    cout << "1. Check your balance" << endl;
    cout << "2. Deposit money" << endl;
    cout << "3. Withdraw money" << endl;
    cout << "4. Logout" << endl;
    cout << "5. Exit" << endl;

    while (1)
    {

```

```

    int ch;
    cout << "\n\nEnter your choice: ";
    cin >> ch;

    switch (ch)
    {
    case 1:
        obj.checkBalance();
        break;

    case 2:
        obj.depositMoney();
        break;

    case 3:
        obj.withdrawMoney();
        break;

    case 4:
        cout << "You logged out successfully....." << endl;
        main();
        break;

    case 5:
        cout << "\n";
        cout << "Thanks for using our Atm Machine\n";
        cout << "Please re-visit again.....\n";
        exit(0);
        break;

    default:
        cout << "Please enter a valid input";
        break;
    }
}

void signup()
{
    system("cls"); // clears the console
    cout << "Enter the username: ";
    cin >> user_name;
    cout << "\nEnter the password: ";
    cin >> pass_word;

    ofstream file;
    file.open(user_name + ".txt");
    file << user_name << " " << pass_word << endl; // input format in file
}

```

```

    file.close();
    system("cls");
    cout << "\nRegistration successfull\n";
    main();
}

void login()
{
    int count;
    string username, password, un, pw;
    system("cls");
    cout << "Please enter the following details: " << endl;
    cout << "Enter username: ";
    cin >> username;
    cout << "Enter password: ";
    cin >> password;

    ifstream input(username + ".txt");

    while (input >> un >> pw)
    {
        if (un == username && pw == password)
        {
            count = 1;
            system("cls");
        }
    }

    input.close();

    if (count == 1)
    {
        cout << "\nHello " << username << "\n<Login Successfull>\nThanks for logging in..\n" << endl;
        cout << "Press Enter to continue" << endl;
        cin.get();
        cin.get();
        simi_main();
    }
    else
    {
        cout << "\nlogin error.....\n";
        cout << "Please enter your correct username and password." << endl;
        main();
    }
}

```

## **FINAL REVIEW OF THE PROJECT**

The Mini Project for Bank Management System in C++ is a console-based application created using the C++ programming language. This system is a simple mini project and compiled in Visual Studio Code IDE using GCC compiler. This system is centered around client account administrations in banks, so it is named “Client Account Bank Management System”. This simple mini project for Bank Management System is complete and totally error-free and includes a downloadable Source Code for free, just find the downloadable source code below and click to start downloading. Before you start to click the download now first you must click the Run Quick Scan for secure Download.

Here, you can make another record, update data of a current record, see and oversee exchanges, check the information of a current record, eliminate existing records and view clients’ lists. The source code for Bank Management System is generally short and straightforward. I have partitioned this C++ smaller than expected task into numerous capacities, a large portion of which are identified with various financial exercises.



## **Conclusion**

---

The Bank Management System Project in C++ can be very helpful for banks as this will automate all the tasks of a bank. In this project we have added most of the major operations of a bank like **depositing, Withdrawing, Transferring, Creating a secure pin, and viewing all transaction details**. I hope this project was helpful and you learned something valuable.

This project uses classes and file handling features of C++. In order to store all the user's data, an external file (DAT file) is created by the system, so every time we get into the system we can operate with the existing accounts. Bank Management System is developed using C++ Programming Language and difference variables, strings have been used for the development of it.

### **Features:**

- 1. Create an account**
- 2. Login to your account**
- 3. Deposit amount**
- 4. Withdraw amount**
- 5. Balance enquiry**
- 6. List account holder's detail**
- 7. Logout of your account**

## References

---

<https://www.w3schools.com/>

<https://www.geeksforgeeks.org/>

<https://cplusplus.com/>

<https://itsourcecode.com/uml/bank-management-system-uml-diagrams/>

## **USER MANUAL**

### **About Project**

Bank Management System is based on a concept of recording of customer's account details. Here the user can perform all the tasks like creating an account, then login for the account and then deposit amount, withdraw amount, check balance, view all account holders detail, close an account and logout it.

### **Talking about the features of the Bank Management System**

Talking about the features of the Bank Management System,

- ➔ A user can create an account by providing a username and password by signing up.
- ➔ This creates the user's account.
- ➔ Now login to your account by logging in using your username and password.
- ➔ Now after login to your account, set your name, and provide your permanent phone number.
- ➔ Now you can deposit and withdraw your money just by providing his/her account, then the system displays his/her profile and entering an amount.
- ➔ For certain purpose, he/she can also check for the balance inquiry which displays the account holder's name amount.
- ➔ All the transaction history including debit and credit from bank account is being stored in .txt file.
- ➔ After all the procedure of the bank account he/she can be logged out of their account for the safe transaction of their money.
- ➔ After logging out of the bank account he/she will be out of the transaction process and for again going for transaction he/ she will again need for login to their account.
- ➔ After their transactions he/she can now exit from the bank system.