PROJECT REPORT THE BANK MANAGEMENT SYSTEM

GROUP MEMBERS

ABBAS RAZA CT-75

OWAIS ANSARI CT-69

ZARYAB HAIDER CT-68

SALMAN SARWAR CT-71

Table of Contents

INTRODUCTION

The main objective of bank management is to build an organic and optimal interaction system between the elements of banking mechanisms with a view to profit its customers.
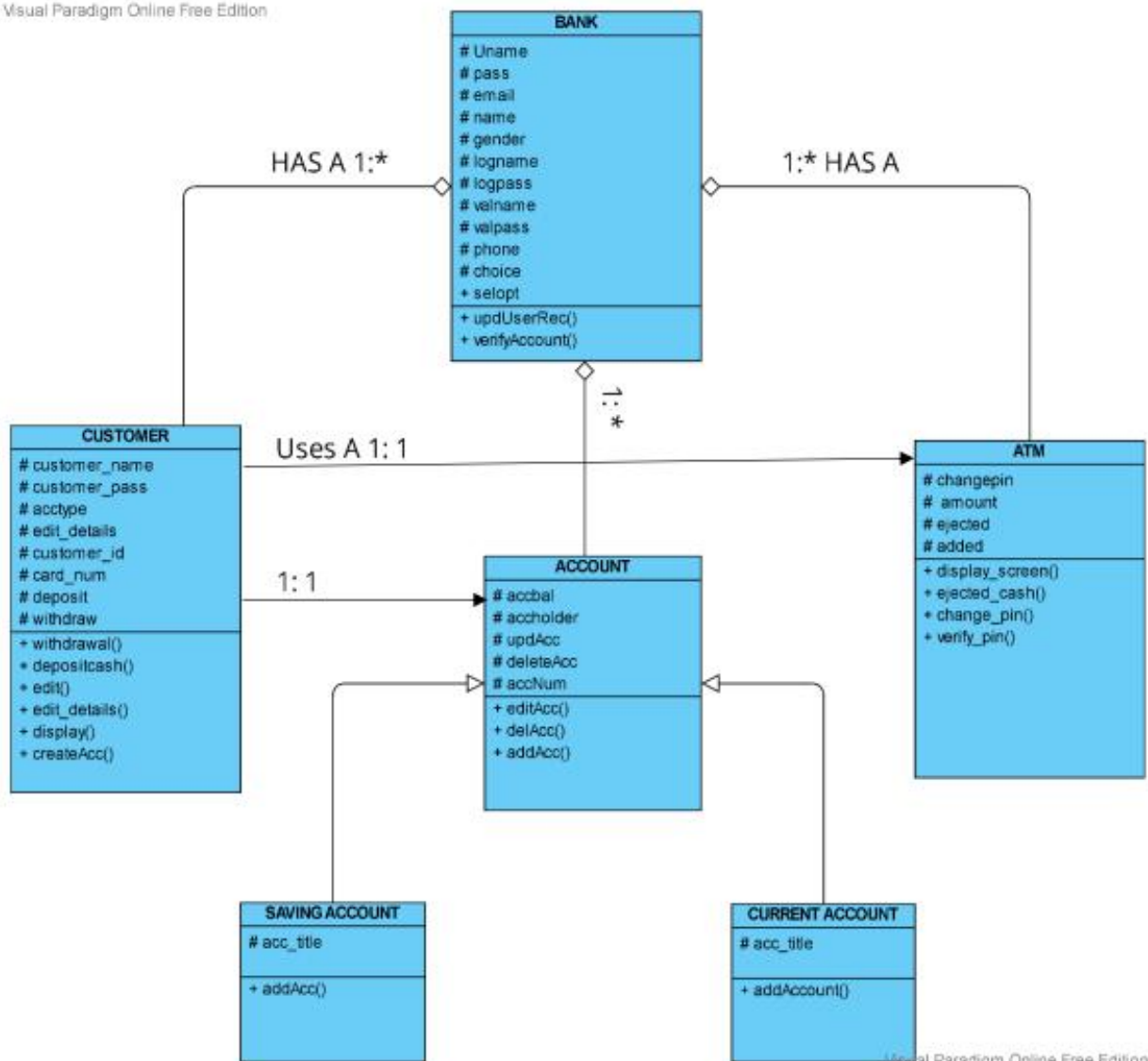
Bank Management System is based on a concept of recording customer's account details. Here the user can perform all the tasks like creating an account, deposit amount, withdraw amount, check balance, view all account holders detail, close an account and modify an account, ATM ETC. All the main features for the banking system are set in this project.

.

SCOPE

Talking about the scope of the Bank Management System, a user can create an account by providing the name of the account holder, account number, select amount type whether its Saving account or Current account and providing an initial amount. Then the user can also deposit and withdraw money just by providing his/her account, then the system displays his/her profile and enters an amount. For certain purposes, he/she can also check for the balance inquiry which displays the account holder's name with account number type and amount. He/she can also check for all the account holder's list. Another feature is that the user can also close their account by providing their account number and he/she can modify their account details and type if they want to.

UML DIAGRAM

BANK
Uname
pass
email
name
gender
logname
logpass
valname
valpass
phone
choice
+ selopt
+ updUserRec()
+ verifyAccount()

HAS A 1:*

1:* HAS A

1:*

CUSTOMER
customer_name
customer_pass
acctype
edit_details
customer_id
card_num
deposit
withdraw
+ withdrawal()
+ depositcash()
+ edit()
+ edit_details()
+ display()
+ createAcc()

Uses A 1: 1

1: 1

ACCOUNT
accbal
accholder
updAcc
deleteAcc
accNum
+ editAcc()
+ delAcc()
+ addAcc()

ATM
changepin
amount
ejected
added
+ display_screen()
+ ejected_cash()
+ change_pin()
+ verify_pin()

SAVING ACCOUNT
acc_title
+ addAcc()

CURRENT ACCOUNT
acc_title
+ addAccount()

CODE

```cpp
1    #include <iostream>
2    #include <string>
3    #include <fstream>
4    using namespace std;
5
6    //**BANK CLASS**//
7    class BANK
8    {
9    protected:
10       string uName, pass, name, email, gender, logname, logpass, valname, valpass;
11       int phone, choice;
12   public:
13       string selOpt;
14
15   public:
16       void updUserRec()
17       {
18           ofstream output;
19           output.open("/home/faiz/Desktop/C++/Project/record.txt", ios::app);
20           cout << "Enter username: " << endl;
21           cin >> uName;
22           cout << "Enter password: " << endl;
23           cin >> pass;
24           cout << "Enter name: " << endl;
25           cin >> name;
26           cout << "Enter email: " << endl;
27           cin >> email;
28           cout << "Enter gender: " << endl;
29           cin >> gender;
30           cout << "Enter phone number: " << endl;
31           cin >> phone;
32           output << endl
33                   << "Username: " << uName << endl
34                   << "Password: " << pass << endl
35                   << "Name: " << name << endl
36                   << "Password: " << pass << endl
37                   << "Email: " << email << endl
38                   << "Gender: " << gender << endl
39                   << "Phone: " << phone << endl;
40           output.close();
```

```cpp
40              output.close();
41              cout << "Registration complete!" << endl;
42          }
43      void verifyAccount()
44          {
45              updUserRec();
46              //  login
47
48              ifstream input("/home/faiz/Desktop/C++/Project/record.txt");
49
50              if (!input)
51              {
52                  cout << "Error creating file!";
53              }
54
55              cout << "Enter username:" << endl;
56              cin >> logname;
57              cout << "Enter password:" << endl;
58              cin >> logpass;
59
60              if (uName == logname && pass == logpass)
61              {
62                  // input??uName trying to get to the file and read the already entered uName and match with the current entered name by user, same with pass
63                  cout << "Login successful" << endl;
64                  cout << "\n*Welcome To your Account*\nEnter The Option Number You want to Perform \n1) Customer \n2) Account \n3) ATM\n";
65                  cin >> selOpt;
66              }
67              else
68              {
69                  cout << "Login failed!" << endl;
70              }
71
72              input.close();
73          }
74      };
75
76      //**ATM CLASS**//
77      class ATM : public BANK
78      {
79      protected:
```

```cpp
75
76  //**ATM CLASS**//
77  class ATM : public BANK
78  {
79  protected:
80      string changepin, Y;
81      int amount = 50000, ejected, added;
82
83  public:
84      void display_screen()
85      {
86          cout << "\n*Welcome to ATM*" << endl;
87      }
88      void eject_cash()
89      {
90          cout << "Enter the amount you want to eject: ";
91          cin >> ejected;
92          amount = amount - ejected;
93          cout << "You have ejected the cash amount: " << ejected << endl;
94          cout << "Amount left in account: " << amount << endl;
95      }
96      void change_pin()
97      {
98          cout << "If you want to change password choose yes (Y/N): ";
99          cin >> changepin;
100         if (changepin == "Y" || changepin == "y")
101         {
102             cout << "Enter new password: ";
103             cin >> pass;
104             cout << "Your password has been changed\n";
105         }
106     }
107     void verify_pin()
108     {
109         cout << "--------Welcome to your account--------\n";
110         change_pin();
111         cout << "You have " << amount << " in your account\n";
112         eject_cash();
113     }
114 };
```

```cpp
//**ACCOUNT CLASS**//
class ACCOUNT
{
protected:
    float accBal;
    string accHolder, updAcc, deleteAcc;
    int accNum;

public:
    void editAcc()
    {
        cout << "Enter New Account Holder name: ";
        cin >> accHolder;
        accHolder = accHolder;
        cout << "Enter Your New Account Number: ";
        cin >> accNum;
        accNum = accNum;
        cout << "Enter the New Money you want to deposit: ";
        cin >> accBal;
        accBal = accBal;
        cout << "The account has been updated with name " << accHolder << "\nThe account number is " << accNum << "\nTotal Amount is " << accBal << endl;
    }
    void delAcc()
    {
        accBal = 0;
        accHolder = "";
        accNum = 0;
        cout << "**ACCOUNT HAS BEEN DELETED**" << endl;
    }
    void addAcc()
    {
        cout << "Enter Account Holder name: ";
        cin >> accHolder;
        cout << "Enter Your Account Number: ";
        cin >> accNum;
        cout << "Enter the Money you want to deposit: ";
        cin >> accBal;
        cout << "The account has been created with name " << accHolder << "\nThe account number is " << accNum << "\nTotal Amount is " << accBal << endl;
        cout << "Do you want to update your account details? (Y/N): ";
        cin >> updAcc;
        if (updAcc == "Y" || updAcc == "y")
        {
            editAcc();
        }
        cout << "Do you want to Delete your account details? (Y/N): ";
        cin >> deleteAcc;
        if (deleteAcc == "Y" || deleteAcc == "y")
        {
            delAcc();
        }
    }
};

//**SAVING ACCOUNT CLASS**//
class SAVING_ACCOUNT : public ACCOUNT
{
protected:
    string accTitle = "Saving account";

public:
    void addAcc()
    {
        cout << "*You are creating a saving account...*\n";
        ACCOUNT::addAcc();
    }
};

//**CURRENT ACCOUNT CLASS**//
class CURRENT_ACCOUNT : public ACCOUNT
{
protected:
    string accTitle = "Current account";

public:
    void addAcc()
    {
```

```cpp
190         void addAcc()
191         {
192             cout << "*You are creating a current account...*\n";
193             ACCOUNT::addAcc();
194         }
195     };
196
197     //**CUSTOMER CLASS**//
198     class CUSTOMER
199     {
200     protected:
201         string customer_name, customer_pass, acctype, edit_details;
202         int customer_id, card_num, deposit, withdraw;
203
204     public:
205         void withdrawal()
206         {
207             cout << "* WELCOME TO CASH WITHDRAWAL SECTION *" << endl;
208             cout << "Enter the amount you want to withdraw: ";
209             cin >> withdraw;
210             deposit = deposit - withdraw;
211             cout << "You have ejected the cash amount: " << withdraw << endl;
212             cout << "Amount left in account: " << deposit << endl;
213
214             if (withdraw > deposit)
215             {
216                 cout << "Insufficient balance " << endl;
217             }
218             else
219             {
220                 cout << "Withdrawal successful" << endl;
221             }
222         }
223         void depositcash()
224         {
225             cout << "*WELCOME TO CASH DEPOSIT SECTION*" << endl;
226             cout << "Enter the cash you want to deposit: ";
227             cin >> deposit;
228             cout << "\n\n Cash deposit of amount " << deposit << " successful " << endl;
229         }
229         }
230         void edit()
231         {
232             cout << "Enter customer name: ";
233             cin >> customer_name;
234             customer_name = customer_name;
235             cout << "Enter customer id: ";
236             cin >> customer_id;
237             customer_id = customer_id;
238             cout << "Enter account type(S/C): ";
239             cin >> acctype;
240             acctype = acctype;
241         }
242         void edit_detail()
243         {
244             cout << "*WELCOME TO ACCOUNT UPDATION SECTION*" << endl;
245             cout << "Do you want to update your account details?(Y/N): ";
246             cin >> edit_details;
247             if (edit_details == "Y" || edit_details == "y")
248             {
249                 edit();
250             }
251         }
252         void display()
253         {
254             cout << "*THIS IS THE BANK STATEMENT FOR YOUR " << acctype << "ACCOUNT*" << endl;
255             cout << "THE NAME OF THE BANK ACCOUNT HOLDER IS: " << customer_name << endl;
256             cout << "THE CUSTOMER ID FOR " << customer_name << " is :" << customer_id << endl;
257             cout << "YOUR ACCOUNT TYPE IS " << acctype << endl;
258             cout << "Current deposit in your account is: " << deposit << endl;
259             cout << "You have withdrawal an amount of:  " << withdraw << endl;
260         }
261         void createAcc()
262         {
263             cout << "*Enter your details to create an account*: " << endl;
264             cout << "Enter the name as per your national identity card: " << endl;
265             cin >> customer_name;
266             cout << "Please enter your bank account number: " << endl;
267             cin >> customer_id;
268             cout << "Please enter the type of bank account(Savings/Current): " << endl;
```

```
268            cout << "Please enter the type of bank account(Savings/Current): " << endl;
269            cin >> acctype;
270            depositcash();
271            withdrawal();
272            edit_detail();
273            display();
274        }
275    };
276
277    //**MAIN**//
278    int main()
279    {
280        string cont, accType;
281        BANK B1;
282        CUSTOMER C1;
283        ATM A1;
284        SAVING_ACCOUNT SA1;
285        CURRENT_ACCOUNT CA1;
286        cout << "******* WELCOME TO THE BANKING MANAGEMENT SYSTEM ********" << endl;
287        cout << "*******    HOPE YOU WILL ENJOY OUR SERVICES     ********" << endl;
288        B1.verifyAccount();
289        if (B1.selOpt == "CUSTOMER" || B1.selOpt == "customer" || B1.selOpt == "Customer" || B1.selOpt == "1")
290        {
291            C1.createAcc();
292        }
293        else if (B1.selOpt == "ACCOUNT" || B1.selOpt == "account" || B1.selOpt == "Account" || B1.selOpt == "2")
294        {
295            cout << "Which type of account do you want to create? \n1) Saving Account (Write: 1) \n2) Current Account (Write: 2)\n ";
296            cin >> accType;
297            if (accType == "1")
298            {
299                SA1.addAcc();
300            }
301            else if (accType == "2")
302            {
303                CA1.addAcc();
304            }
305            else
306            {
```
```
300            }
301            else if (accType == "2")
302            {
303                CA1.addAcc();
304            }
305            else
306            {
307                cout << "Invalid input!!";
308            }
309        }
310        else if (B1.selOpt == "ATM" || B1.selOpt == "atm" || B1.selOpt == "Atm" || B1.selOpt == "3")
311        {
312            A1.display_screen();
313            A1.verify_pin();
314        }
315        else
316        {
317            cout << "Invalid input!!";
318        }
319        cout << "\nContinue using bank? (Y/N): ";
320        cin >> cont;
321        if (cont == "Y" || cont == "y")
322        {
323            B1.verifyAccount();
324        }
325        return 0;
326    }
327
```

OUTPUT

```
faiz@faiz-HP-ProBook-450-G2:~/Desktop/C++/Project$ cd "/home/faiz/Desktop/C++/Project/" && g++ project.cpp -o proj
ect && "/home/faiz/Desktop/C++/Project/"project
******* WELCOME TO THE BANKING MANAGEMENT SYSTEM ********
*******      HOPE YOU WILL ENJOY OUR SERVICES      ********
Enter username:
salmansarwar
Enter password:
123
Enter name:
Salman
Enter email:
muhammadsalmansarwar32gmail.com
Enter gender:
male
Enter phone number:
033333333
Registration complete!
Enter username:
salmansarwar
Enter password:
123
Login successful

*Welcome To your Account*
Enter The Option Number You want to Perform
1) Customer
2) Account
3) ATM

                                                                              Tweet Fee
```

LIMITATIONS

Although we've tried our best to maintain the quality and accuracy of the system but being a beginner to the coding and implementation of the Programs we know that we might lack some minor issues that can be learned from the experiences that we'll be having in the future.

Some limitations of the existing system:

Less security of customer and bank information.

All the manual entry and editing will take more time.

No backup of the information.

Some improvements by executing the proposed system:

More secure information will give a layer of security of authentication and authorization.

Required very little manpower.