COMP315

Project Documentation

FingerTip

Members	
1. Sipho Gwala [219041311]	3.Kaylene Chetty [220003579]
2. Siduduzo Mthethwa [219007384]	4.Xolisani Jam Jam [218051894] 5.Luyanda Gumede [218012664]

Contents

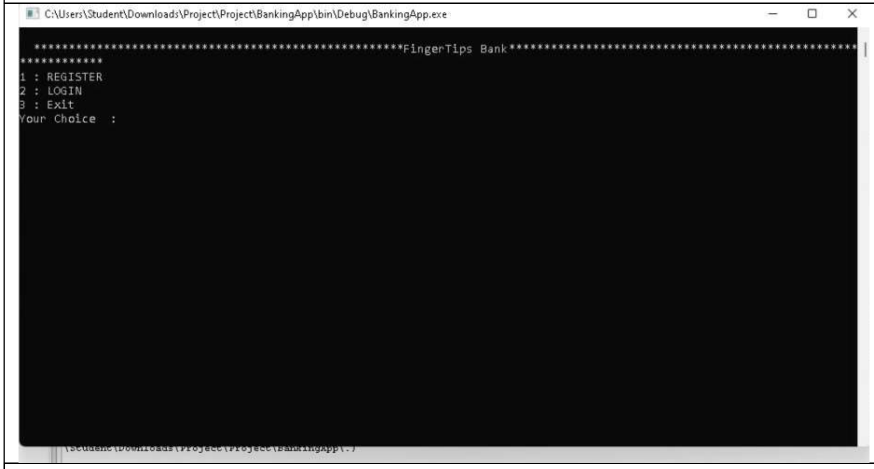
Introduction

Many customers found it difficult (and even impossible) to open bank account and do online transactions. The net result was accumulation of overdue payments, inability to make online purchases and transactions leading many customers being frustrated and left financially inert.

And so, we have developed a banking app that runs solely on the console that aim at solving these problems, called “FingerTip”. FingerTip is a mobile banking app that allows customers without bank account to register for a bank account and generate a virtual bank card at the tip of their hands within minutes. The virtual bank card will allow the customers to make online purchases and transactions at the comfort of their living spaces. To create a bank account and acquire virtual card, the customers will only need to provide their email address, cell phone number, physical address, identity number and their names as they appear of their identity document without the need to visit a bank branch that might be closed or only allows a limited number of customers. Thus, making the customers financially active again amidst the pandemic.

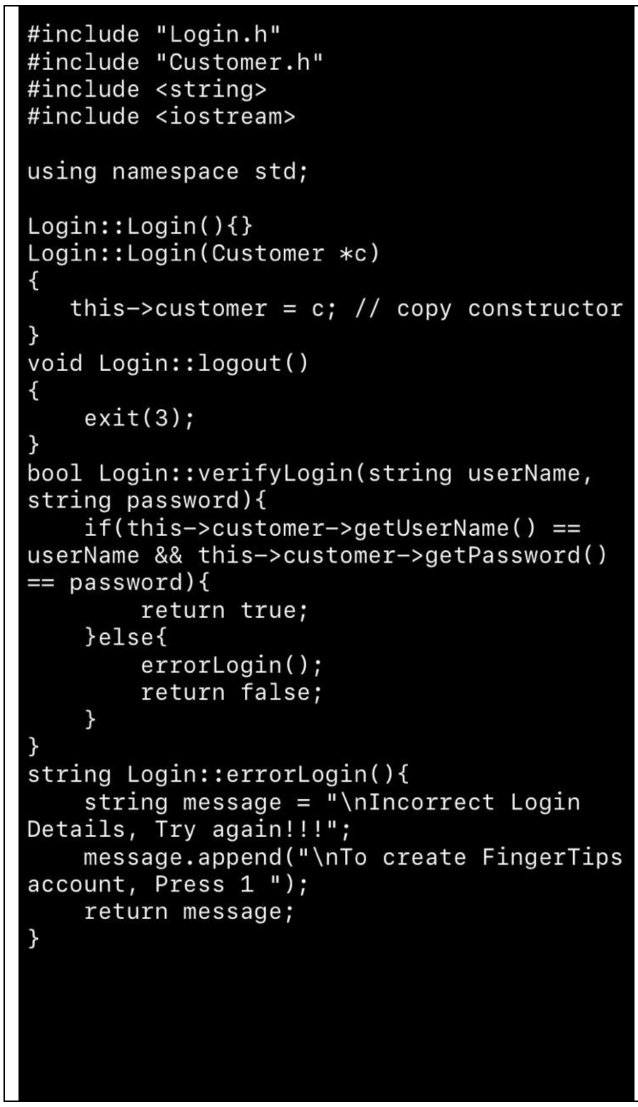
User Interaction

1. User input

Screenshot	Explanation
	When you run the code, on the console is a user prompt console that allows you to choose different options using numbers, to select options
	The console displays the console that allows user to enter their details in order to proceed with the registration process.

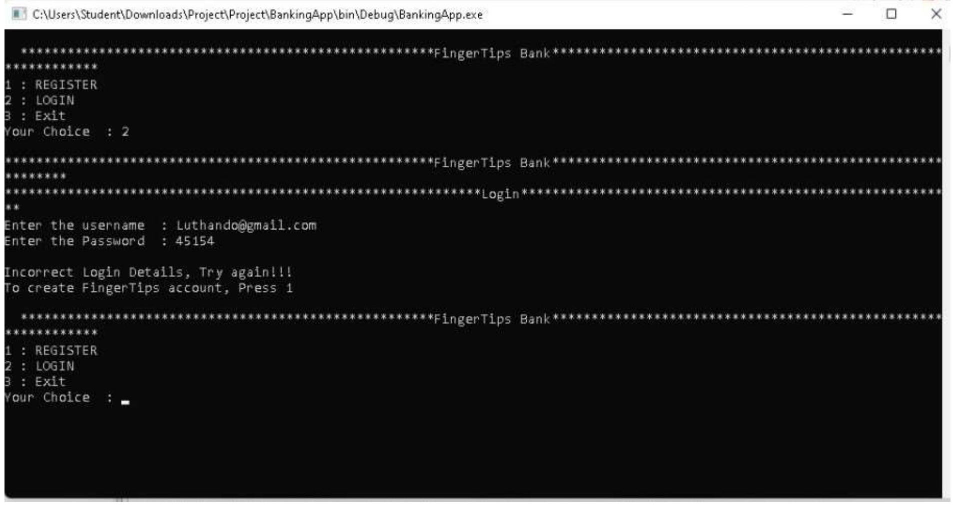
```
C:\Users\Student\Downloads\Project\Project\BankingApp\bin\Debug\BankingApp.exe

 *****************************************************FingerTips Bank*****************************************************
************
1 : REGISTER
2 : LOGIN
3 : Exit
Your Choice  :
```

Code Screenshot



```cpp
#include "Login.h"
#include "Customer.h"
#include <string>
#include <iostream>

using namespace std;

Login::Login(){}
Login::Login(Customer *c)
{
   this->customer = c; // copy constructor
}
void Login::logout()
{
    exit(3);
}
bool Login::verifyLogin(string userName,
string password){
    if(this->customer->getUserName() ==
userName && this->customer->getPassword()
== password){
        return true;
    }else{
        errorLogin();
        return false;
    }
}
string Login::errorLogin(){
    string message = "\nIncorrect Login
Details, Try again!!!";
    message.append("\nTo create FingerTips
account, Press 1 ");
    return message;
}
```

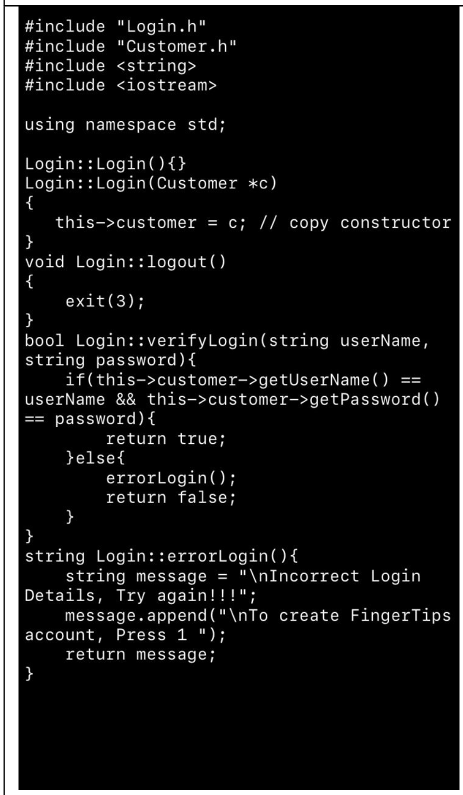
2. User feedback/Output

Screenshot	Explanation
	If the customer with an existing account the app will transfer to the next terminal, however, if the customer does have a registered account the app will transfer back to login terminal (the first console) to

```
C:\Users\Student\Downloads\Project\Project\BankingApp\bin\Debug\BankingApp.exe

  ******************************************************FingerTips Bank*******************************************************
************
1 : REGISTER
2 : LOGIN
3 : Exit
Your Choice  : 2

********************************************************FingerTips Bank*******************************************************
********
*************************************************************Login***********************************************************
**
Enter the username  : Luthando@gmail.com
Enter the Password  : 45154

Incorrect Login Details, Try again!!!
To create FingerTips account, Press 1

  ******************************************************FingerTips Bank*******************************************************
************
1 : REGISTER
2 : LOGIN
3 : Exit
Your Choice  : _
```

	register an account. If the customer entered an incorrect password, they will be redirected into the first console. To restart the whole process from the beginning.

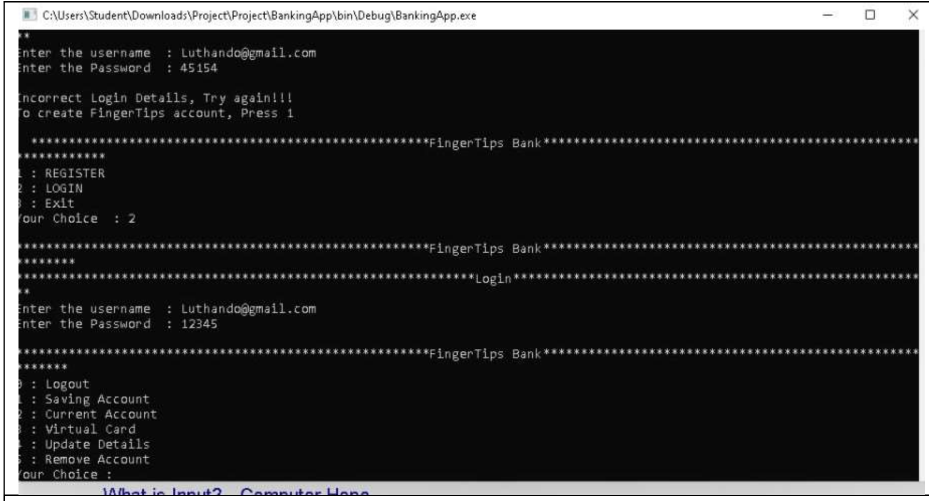
Code Screenshot



```
#include "Login.h"
#include "Customer.h"
#include <string>
#include <iostream>

using namespace std;

Login::Login(){}
Login::Login(Customer *c)
{
   this->customer = c; // copy constructor
}
void Login::logout()
{
    exit(3);
}
bool Login::verifyLogin(string userName,
string password){
    if(this->customer->getUserName() ==
userName && this->customer->getPassword()
== password){
        return true;
    }else{
        errorLogin();
        return false;
    }
}
string Login::errorLogin(){
    string message = "\nIncorrect Login
Details, Try again!!!";
    message.append("\nTo create FingerTips
account, Press 1 ");
    return message;
}
```

3. Login

Screenshot	Explanation

	If the entered login details are correct you will be prompt into the main window, else will have to register an account with the bank.

```
C:\Users\Student\Downloads\Project\Project\BankingApp\bin\Debug\BankingApp.exe
**
nter the username  : Luthando@gmail.com
nter the Password  : 45154

ncorrect Login Details, Try again!!!
o create FingerTips account, Press 1

 ******************************************************FingerTips Bank*****************************************************
************
 : REGISTER
 : LOGIN
 : Exit
our Choice  : 2

*******************************************************FingerTips Bank*****************************************************
********
****************************************************************Login******************************************************
**
nter the username  : Luthando@gmail.com
nter the Password  : 12345

*******************************************************FingerTips Bank*****************************************************
*******
 : Logout
 : Saving Account
 : Current Account
 : Virtual Card
 : Update Details
 : Remove Account
our Choice  :
```

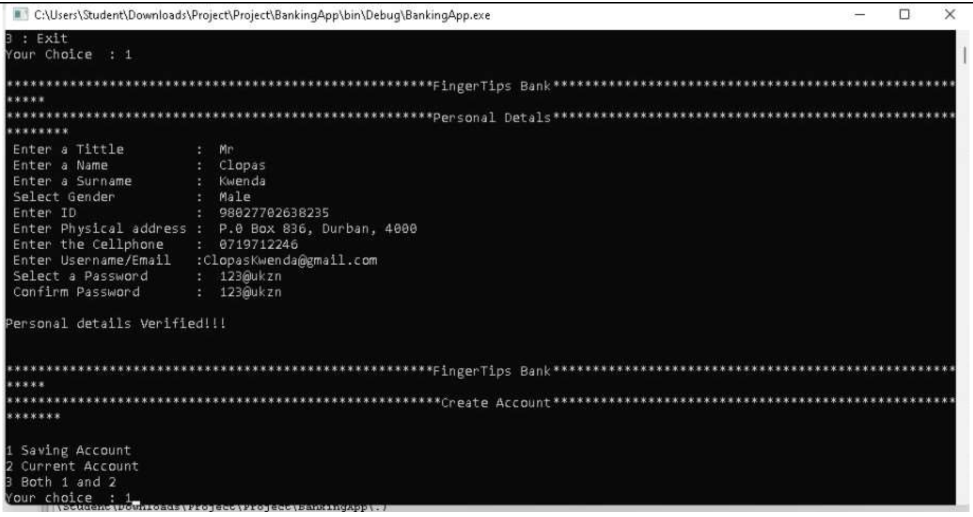
3.31Code Screenshot

Levels and Progression

[*Provide screenshots of your program in action. These screenshots may be at different points in time of the system. Include whatever you think is necessary.*]

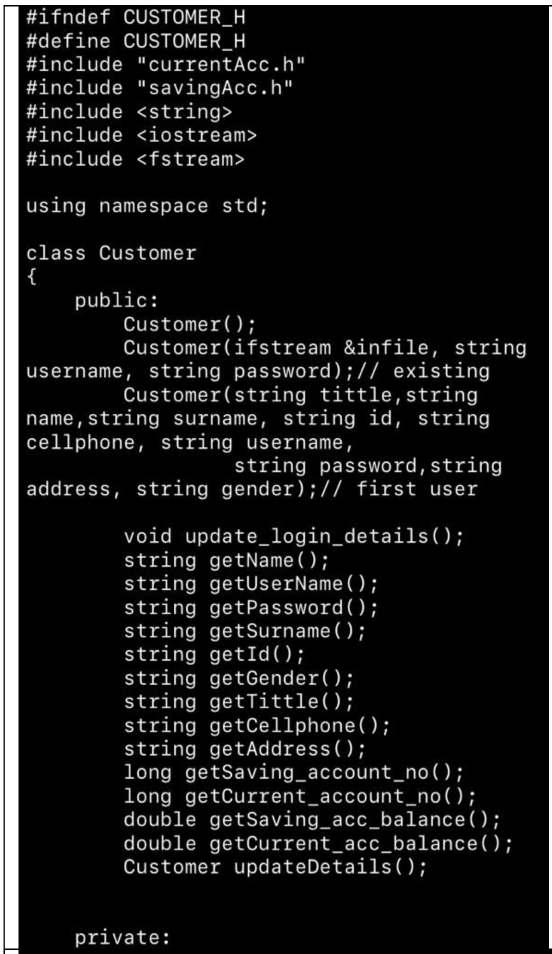
1. Customer

Screenshot	Explanatio n

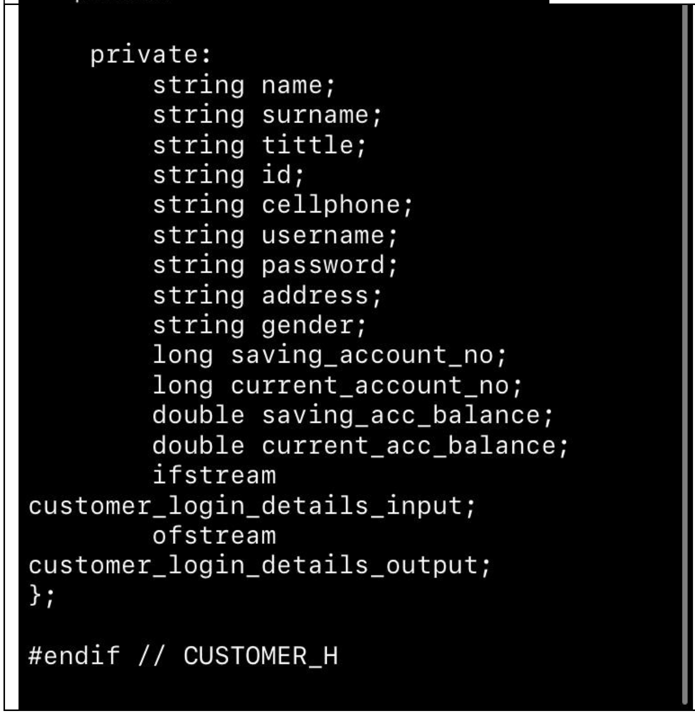
	The customer class gives the allows the user to register by allowing the user to enter their personal details from the title to the cell phone number that will be used to register the customer, if conditions are met the app will prompt a ‘personal details verified’ message then the details are saved as a account.txt file

```
C:\Users\Student\Downloads\Project\Project\BankingApp\bin\Debug\BankingApp.exe
3 : Exit
Your Choice  : 1

******************************************************FingerTips Bank***************************************************
*****
******************************************************Personal Detals***************************************************
********
 Enter a Tittle          :  Mr
 Enter a Name            :  Clopas
 Enter a Surname         :  Kwenda
 Select Gender           :  Male
 Enter ID                :  98027702638235
 Enter Physical address  :  P.0 Box 836, Durban, 4000
 Enter the Cellphone     :  0719712246
 Enter Username/Email    :ClopasKwenda@gmail.com
 Select a Password       :  123@ukzn
 Confirm Password        :  123@ukzn

Personal details Verified!!!


******************************************************FingerTips Bank***************************************************
*****
*******************************************************Create Account***************************************************
*******

1 Saving Account
2 Current Account
3 Both 1 and 2
Your choice  : 1
```

Code Screenshot



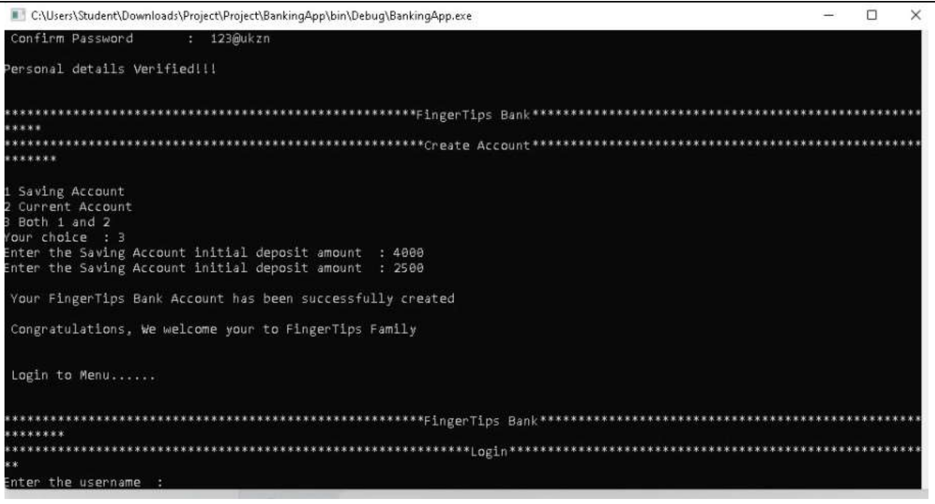
```
#ifndef CUSTOMER_H
#define CUSTOMER_H
#include "currentAcc.h"
#include "savingAcc.h"
#include <string>
#include <iostream>
#include <fstream>

using namespace std;

class Customer
{
    public:
        Customer();
        Customer(ifstream &infile, string
username, string password);// existing
        Customer(string tittle,string
name,string surname, string id, string
cellphone, string username,
                 string password,string
address, string gender);// first user

        void update_login_details();
        string getName();
        string getUserName();
        string getPassword();
        string getSurname();
        string getId();
        string getGender();
        string getTittle();
        string getCellphone();
        string getAddress();
        long getSaving_account_no();
        long getCurrent_account_no();
        double getSaving_acc_balance();
        double getCurrent_acc_balance();
        Customer updateDetails();


    private:
```



```
    private:
        string name;
        string surname;
        string tittle;
        string id;
        string cellphone;
        string username;
        string password;
        string address;
        string gender;
        long saving_account_no;
        long current_account_no;
        double saving_acc_balance;
        double current_acc_balance;
        ifstream
customer_login_details_input;
        ofstream
customer_login_details_output;
};

#endif // CUSTOMER_H
```

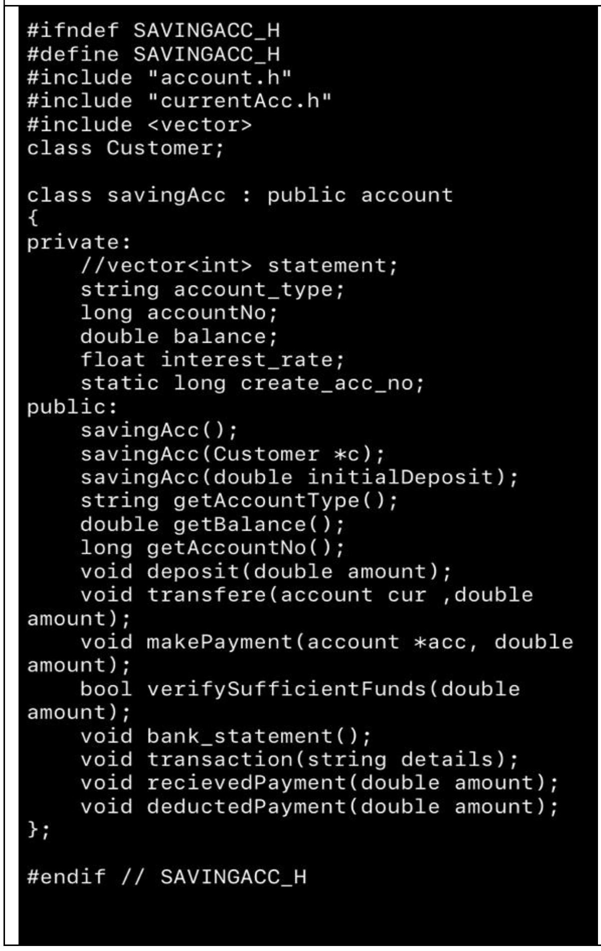
2. Type Of Account

Screenshot	Explanation

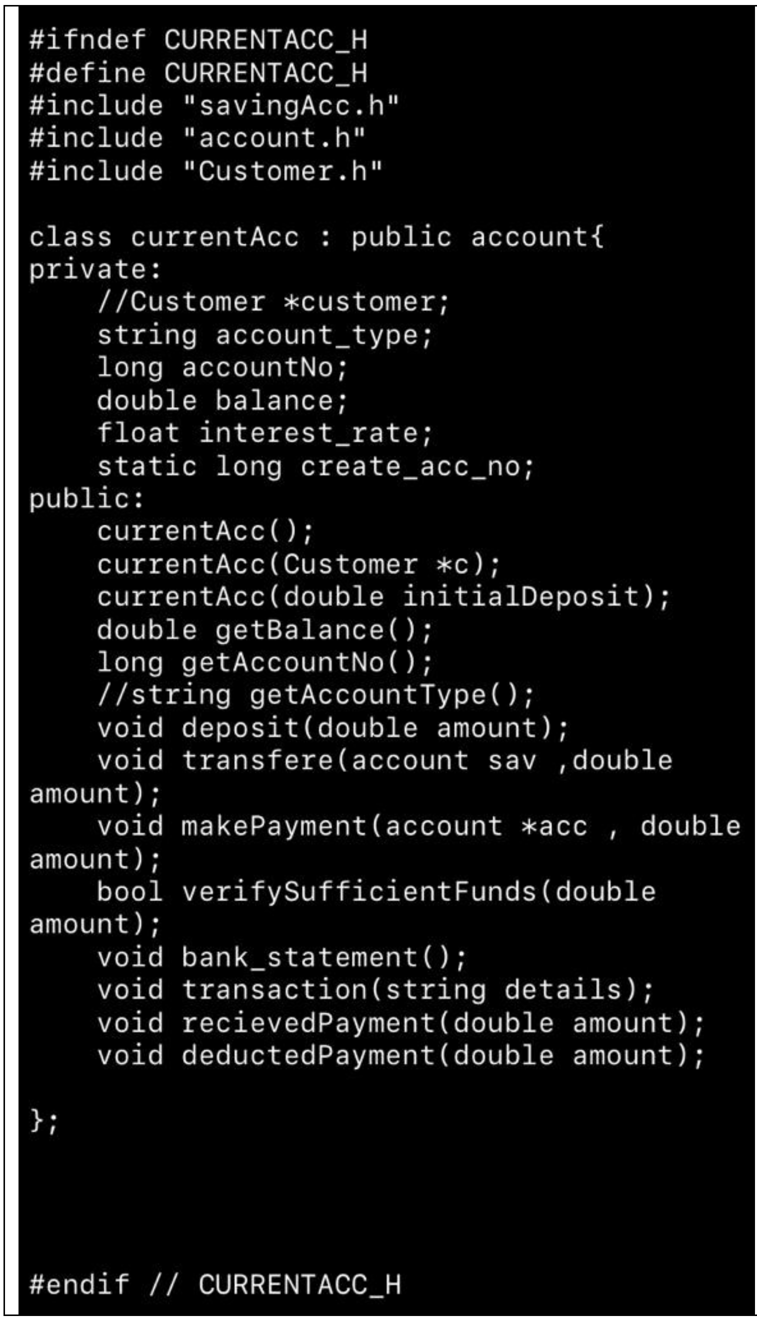
	After registering your account, you are prompt to select the type of account you want to open, between savings account or current account or both. The app will then allow the customer to make an initial deposit to the respective account.

```
C:\Users\Student\Downloads\Project\Project\BankingApp\bin\Debug\BankingApp.exe
 Confirm Password       :  123@ukzn

Personal details Verified!!!


*****************************************************FingerTips Bank*******************************************************
*****
******************************************************Create Account******************************************************
*******

1 Saving Account
2 Current Account
3 Both 1 and 2
Your choice  : 3
Enter the Saving Account initial deposit amount  : 4000
Enter the Saving Account initial deposit amount  : 2500

 Your FingerTips Bank Account has been successfully created

 Congratulations, We welcome your to FingerTips Family


 Login to Menu......


******************************************************FingerTips Bank*****************************************************
********
************************************************************Login*********************************************************
**
Enter the username  :
```

Code Screenshot



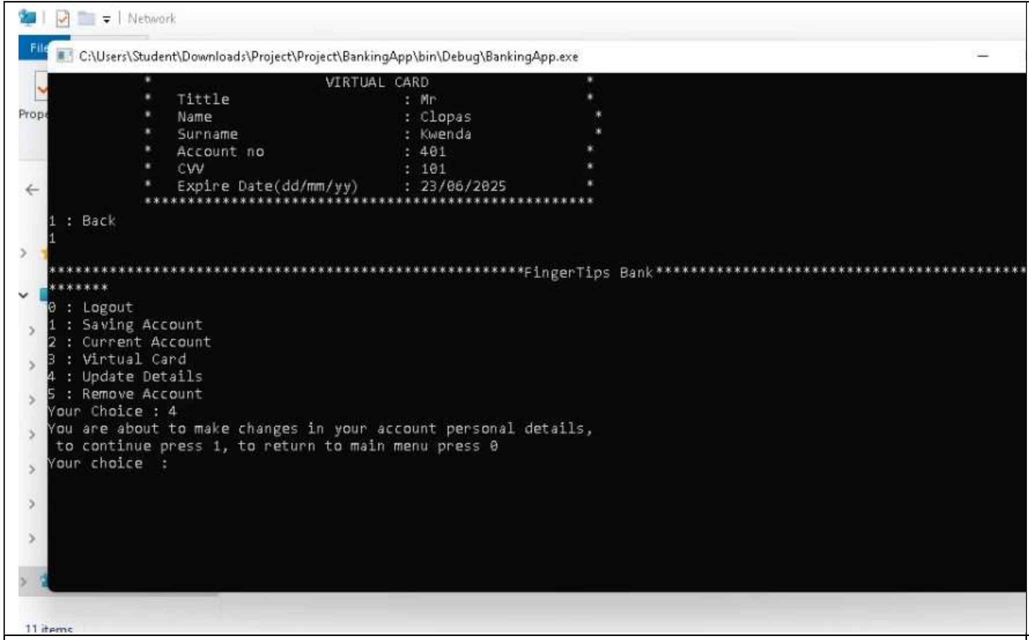
```
#ifndef SAVINGACC_H
#define SAVINGACC_H
#include "account.h"
#include "currentAcc.h"
#include <vector>
class Customer;

class savingAcc : public account
{
private:
    //vector<int> statement;
    string account_type;
    long accountNo;
    double balance;
    float interest_rate;
    static long create_acc_no;
public:
    savingAcc();
    savingAcc(Customer *c);
    savingAcc(double initialDeposit);
    string getAccountType();
    double getBalance();
    long getAccountNo();
    void deposit(double amount);
    void transfere(account cur ,double
amount);
    void makePayment(account *acc, double
amount);
    bool verifySufficientFunds(double
amount);
    void bank_statement();
    void transaction(string details);
    void recievedPayment(double amount);
    void deductedPayment(double amount);
};

#endif // SAVINGACC_H
```

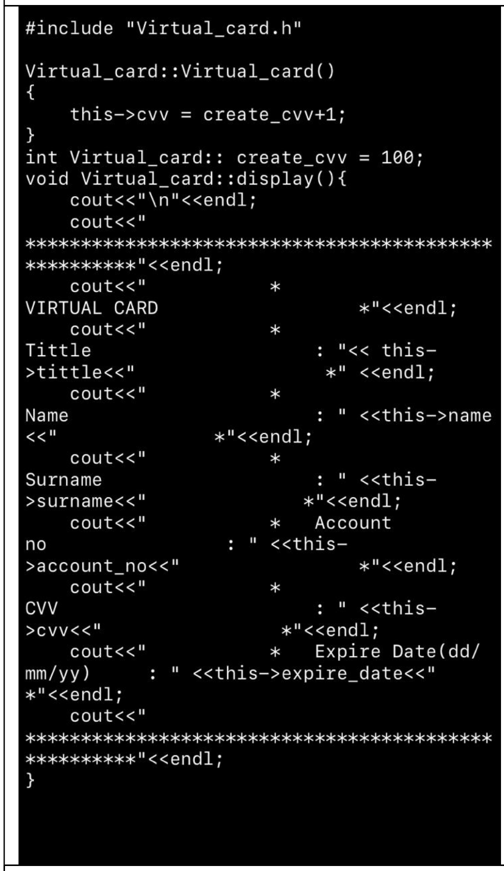
```
#ifndef CURRENTACC_H
#define CURRENTACC_H
#include "savingAcc.h"
#include "account.h"
#include "Customer.h"

class currentAcc : public account{
private:
    //Customer *customer;
    string account_type;
    long accountNo;
    double balance;
    float interest_rate;
    static long create_acc_no;
public:
    currentAcc();
    currentAcc(Customer *c);
    currentAcc(double initialDeposit);
    double getBalance();
    long getAccountNo();
    //string getAccountType();
    void deposit(double amount);
    void transfere(account sav ,double
amount);
    void makePayment(account *acc , double
amount);
    bool verifySufficientFunds(double
amount);
    void bank_statement();
    void transaction(string details);
    void recievedPayment(double amount);
    void deductedPayment(double amount);

};



#endif // CURRENTACC_H
```

3. Virtual Card

Screenshot	**Explanation**

	Here the customer will be able to gain access their virtual card which they can use to make online purchases, that have an expiry date, CVC(Security Number) and your initials.

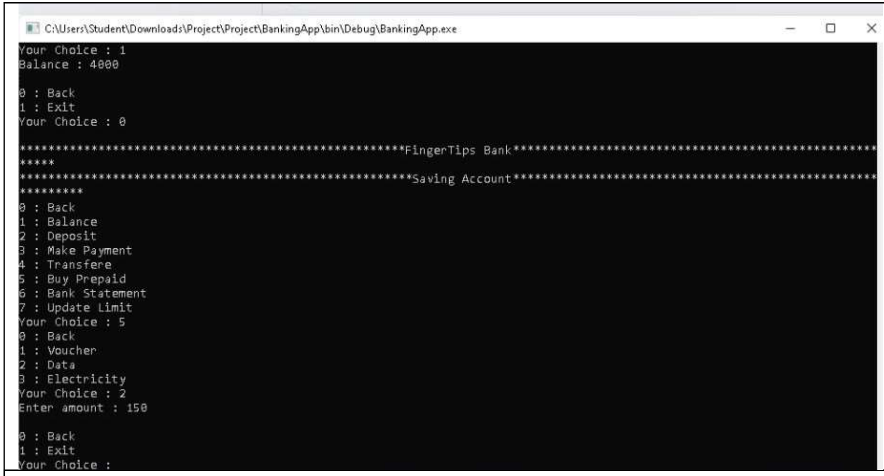
```
C:\Users\Student\Downloads\Project\Project\BankingApp\bin\Debug\BankingApp.exe
         *                   VIRTUAL CARD                  *
         *   Tittle                  : Mr                  *
         *   Name                    : Clopas               *
         *   Surname                 : Kwenda               *
         *   Account no              : 401                 *
         *   CVV                     : 101                 *
         *   Expire Date(dd/mm/yy)   : 23/06/2025          *
         ****************************************************
1 : Back
1

*******************************************************FingerTips Bank*****************************************
*******
0 : Logout
1 : Saving Account
2 : Current Account
3 : Virtual Card
4 : Update Details
5 : Remove Account
Your Choice : 4
You are about to make changes in your account personal details,
 to continue press 1, to return to main menu press 0
Your choice  :
```

Code Screenshot


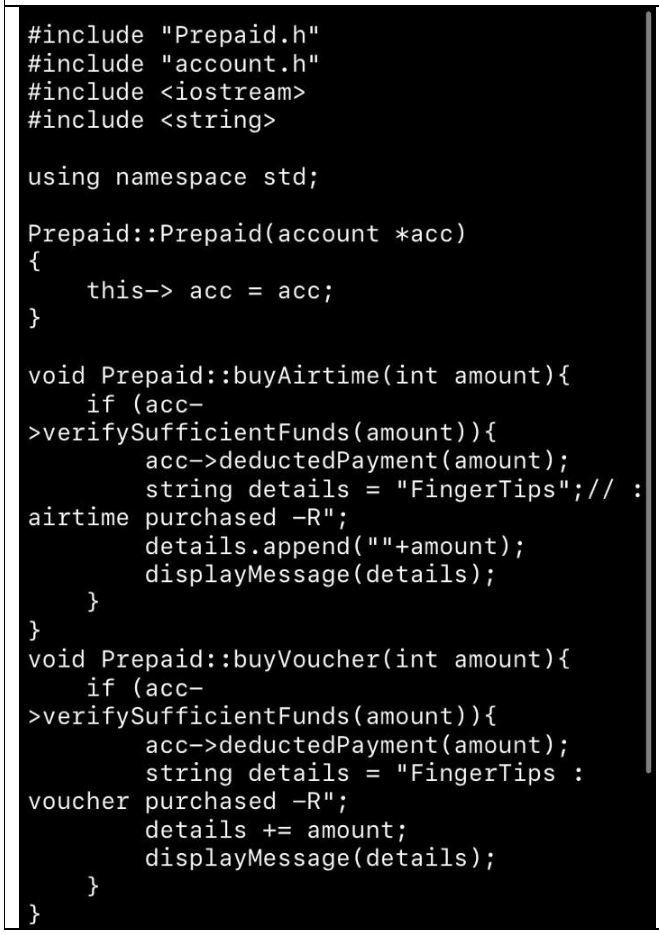
```
#include "Virtual_card.h"

Virtual_card::Virtual_card()
{
    this->cvv = create_cvv+1;
}
int Virtual_card:: create_cvv = 100;
void Virtual_card::display(){
    cout<<"\n"<<endl;
    cout<<"
******************************************
**********"<<endl;
    cout<<"           *
VIRTUAL CARD                *"<<endl;
    cout<<"           *
Tittle                    : "<< this-
>tittle<<"                 *" <<endl;
    cout<<"           *
Name                      : " <<this->name
<<"              *"<<endl;
    cout<<"           *
Surname                   : " <<this-
>surname<<"              *"<<endl;
    cout<<"           *    Account
no              : " <<this-
>account_no<<"               *"<<endl;
    cout<<"           *
CVV                       : " <<this-
>cvv<<"                *"<<endl;
    cout<<"           *    Expire Date(dd/
mm/yy)      : " <<this->expire_date<<"
*"<<endl;
    cout<<"
******************************************
**********"<<endl;
}
```

4. Prepaid

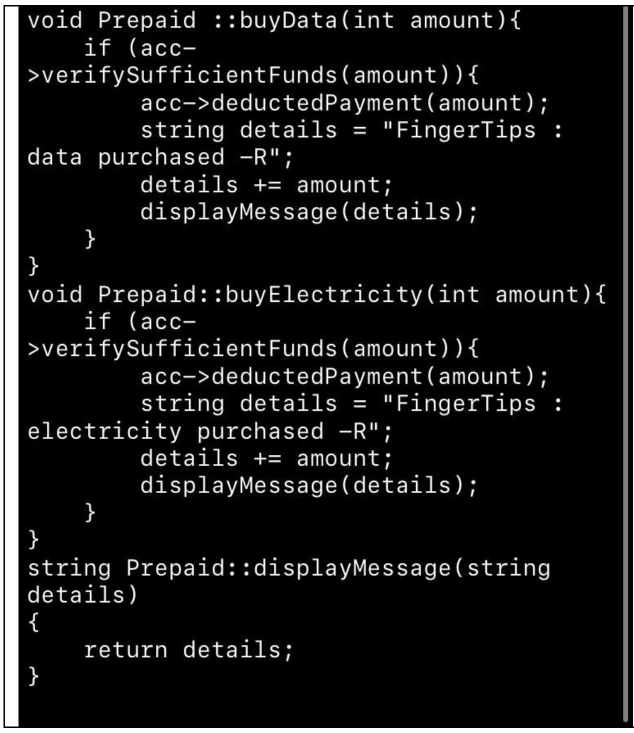
Screenshot	Explanation

	The account allows you to buy prepaid airtime, data, betting voucher and electricity at the tips of your hands. All the transactions made can be stored on the transaction.txt file, which can be used to generate a balance statement.

```
C:\Users\Student\Downloads\Project\Project\BankingApp\bin\Debug\BankingApp.exe
Your Choice : 1
Balance : 4000

0 : Back
1 : Exit
Your Choice : 0

****************************************************FingerTips Bank*****************************************************
*****
*****************************************************Saving Account****************************************************
*********
0 : Back
1 : Balance
2 : Deposit
3 : Make Payment
4 : Transfere
5 : Buy Prepaid
6 : Bank Statement
7 : Update Limit
Your Choice : 5
0 : Back
1 : Voucher
2 : Data
3 : Electricity
Your Choice : 2
Enter amount : 150

0 : Back
1 : Exit
Your Choice :
```

Code Screenshot	
	

```cpp
#include "Prepaid.h"
#include "account.h"
#include <iostream>
#include <string>

using namespace std;

Prepaid::Prepaid(account *acc)
{
    this-> acc = acc;
}

void Prepaid::buyAirtime(int amount){
    if (acc-
>verifySufficientFunds(amount)){
        acc->deductedPayment(amount);
        string details = "FingerTips";// :
airtime purchased -R";
        details.append(""+amount);
        displayMessage(details);
    }
}
void Prepaid::buyVoucher(int amount){
    if (acc-
>verifySufficientFunds(amount)){
        acc->deductedPayment(amount);
        string details = "FingerTips :
voucher purchased -R";
        details += amount;
        displayMessage(details);
    }
}
```

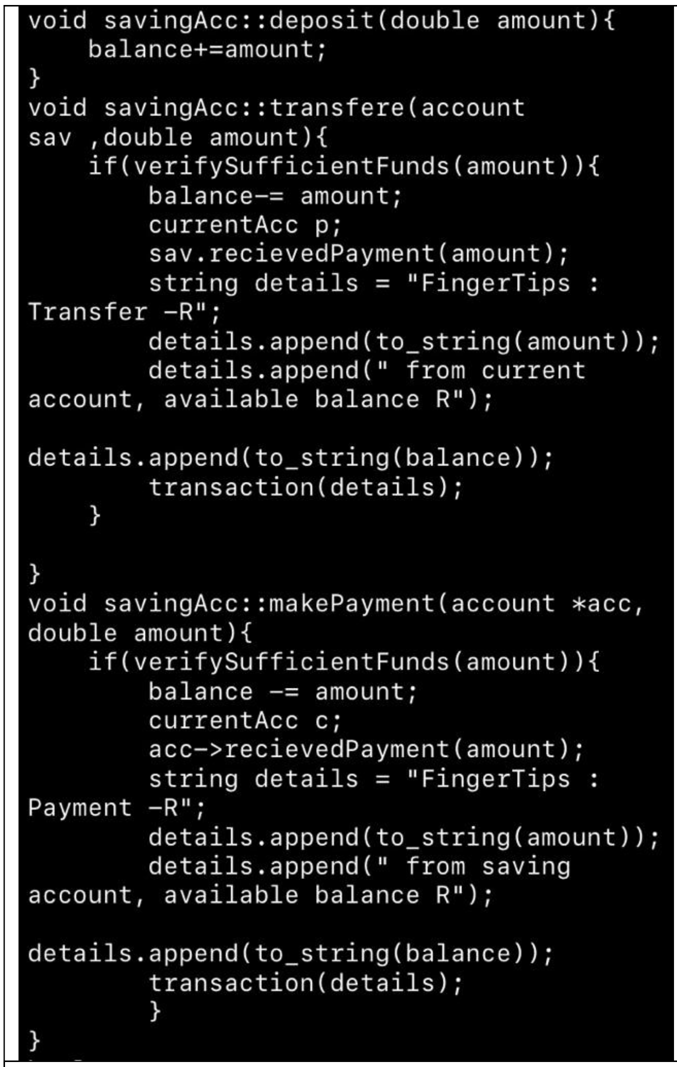


```
void Prepaid ::buyData(int amount){
    if (acc-
>verifySufficientFunds(amount)){
        acc->deductedPayment(amount);
        string details = "FingerTips :
data purchased -R";
        details += amount;
        displayMessage(details);
    }
}
void Prepaid::buyElectricity(int amount){
    if (acc-
>verifySufficientFunds(amount)){
        acc->deductedPayment(amount);
        string details = "FingerTips :
electricity purchased -R";
        details += amount;
        displayMessage(details);
    }
}
string Prepaid::displayMessage(string
details)
{
    return details;
}
```

Programming Techniques

5. Function

<u>**Screenshot:**</u>



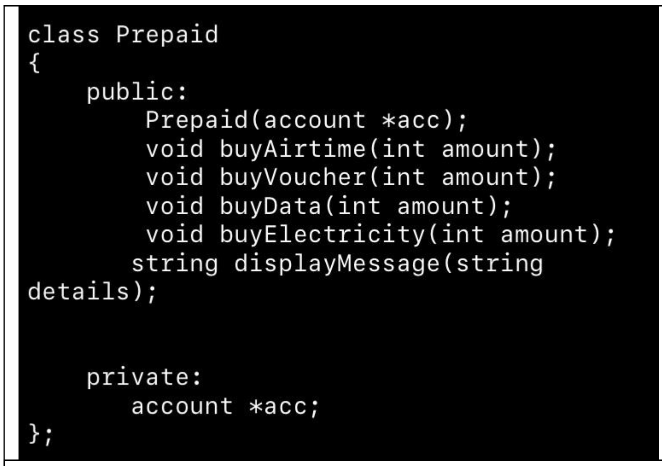
```
void savingAcc::deposit(double amount){
    balance+=amount;
}
void savingAcc::transfere(account
sav ,double amount){
    if(verifySufficientFunds(amount)){
        balance-= amount;
        currentAcc p;
        sav.recievedPayment(amount);
        string details = "FingerTips :
Transfer -R";
        details.append(to_string(amount));
        details.append(" from current
account, available balance R");

details.append(to_string(balance));
        transaction(details);
    }

}
void savingAcc::makePayment(account *acc,
double amount){
    if(verifySufficientFunds(amount)){
        balance -= amount;
        currentAcc c;
        acc->recievedPayment(amount);
        string details = "FingerTips :
Payment -R";
        details.append(to_string(amount));
        details.append(" from saving
account, available balance R");

details.append(to_string(balance));
        transaction(details);
        }
}
```

Motivation:
You can pass data, known as parameters, into a function. Functions are used to perform certain actions, and they are important for reusing code: Define the code once, and use it many times.

How have you met the objectives?	Cross (X) the appropriate box	If you think that you have met the objective completely, provide a short explanation to support the claim
Not met		
Partially		
Completely		

6. Class

Screenshot:



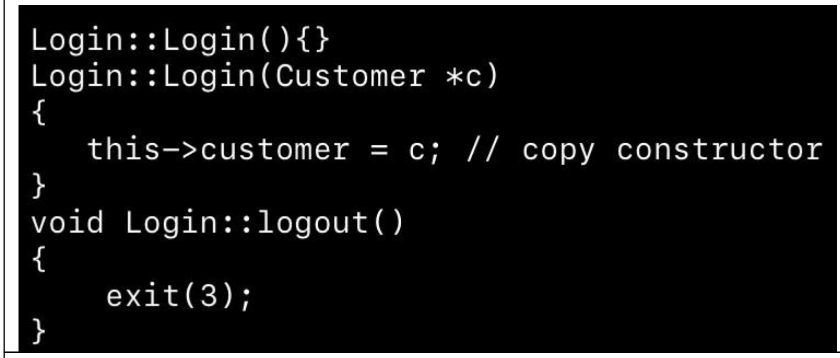
```
class Prepaid
{
    public:
        Prepaid(account *acc);
        void buyAirtime(int amount);
        void buyVoucher(int amount);
        void buyData(int amount);
        void buyElectricity(int amount);
       string displayMessage(string
details);


    private:
       account *acc;
};
```

Motivation:
classes to make it easy to work with objects. Classes and objects became the building blocks C++ uses for creating streamlined and easy-to-read code

How have you met the objectives?	Cross (X) the appropriate box	If you think that you have met the objective completely, provide a short explanation to support the claim
Not met		
Partially		
Completely		

7. Copy Constructor

Screenshot:



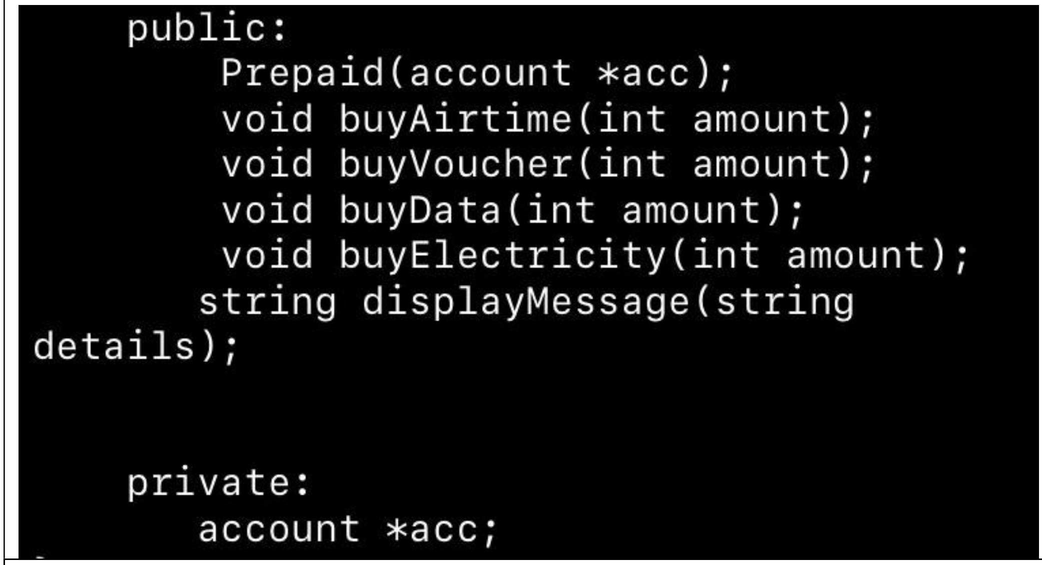
```
Login::Login(){}
Login::Login(Customer *c)
{
   this->customer = c; // copy constructor
}
void Login::logout()
{
    exit(3);
}
```

Motivation:
The copy constructor is a constructor which creates an object by initializing it with an object of the same class, which has been created previously. The copy constructor is used to − Initialize one object from another of the same type. Copy an object to pass it as an argument to a function.

How have you met the objectives?	Cross (X) the appropriate box	If you think that you have met the objective completely, provide a short explanation to support the claim
Not met		
Partially		
Completely		

8. Pointer

Screenshot:



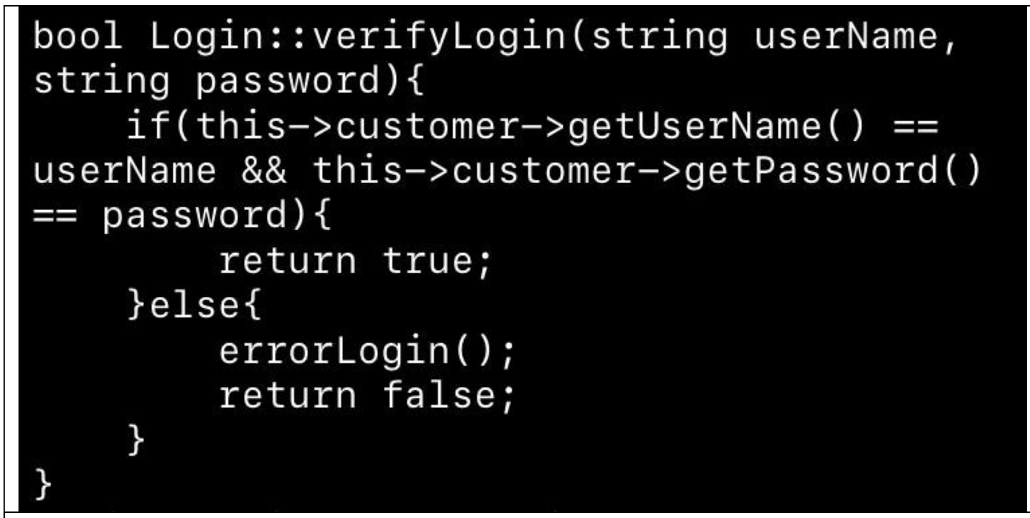
```
    public:
        Prepaid(account *acc);
        void buyAirtime(int amount);
        void buyVoucher(int amount);
        void buyData(int amount);
        void buyElectricity(int amount);
       string displayMessage(string
details);


    private:
       account *acc;
```

Motivation:
Pointers are used to store and manage the addresses of dynamically allocated blocks of memory. Such blocks are used to store data objects or arrays of objects. Most structured and object-oriented languages provide an area of memory, called the heap or free store, from which objects are dynamically allocated.

How have you met the objectives?	Cross (X) the appropriate box	If you think that you have met the objective completely, provide a short explanation to support the claim
Not met		
Partially		
Completely	X	

9. Reference

Screenshot:



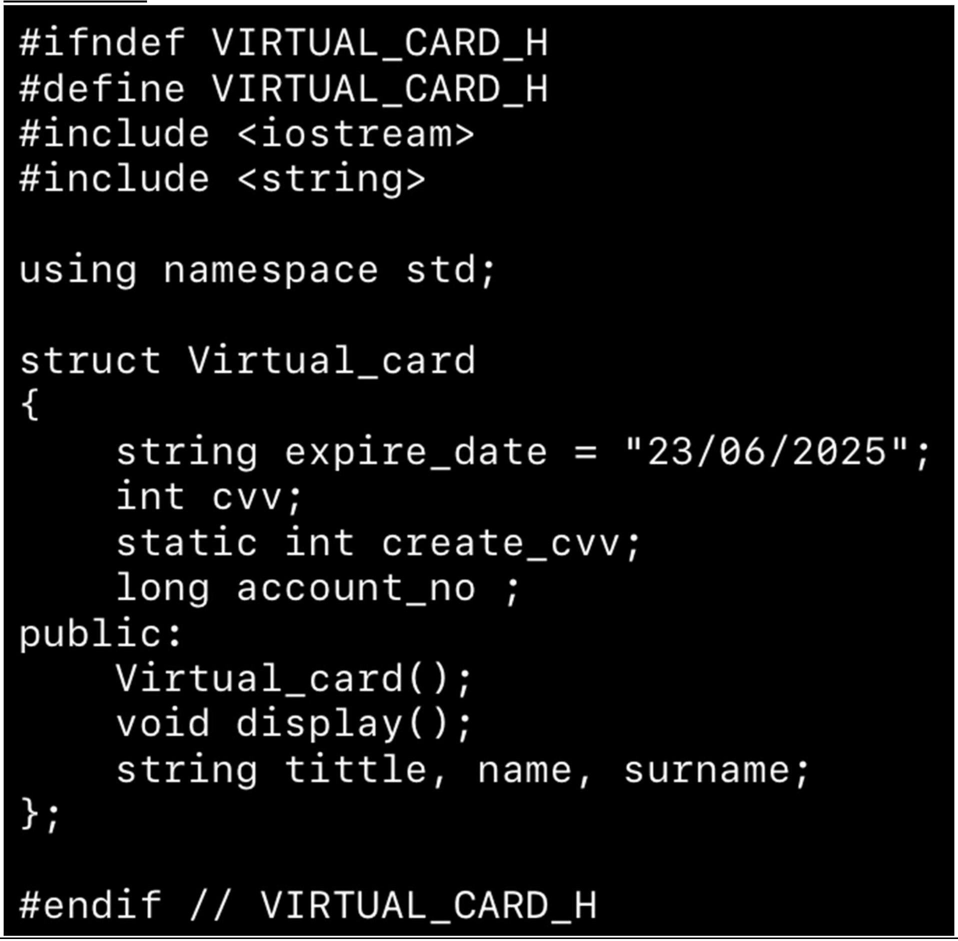
```
bool Login::verifyLogin(string userName,
string password){
    if(this->customer->getUserName() ==
userName && this->customer->getPassword()
== password){
        return true;
    }else{
        errorLogin();
        return false;
    }
}
```

<u>Motivation:</u>
The main use of references is acting as function formal parameters to support pass-by-reference. In a reference variable is passed into a function, the function works on the original copy (instead of a clone copy in pass-by-value). Changes inside the function are reflected outside the function.

How have you met the objectives?	Cross (X) the appropriate box	If you think that you have met the objective completely, provide a short explanation to support the claim
Not met		
Partially		
Completely	X	

10. Struct

Screenshot:



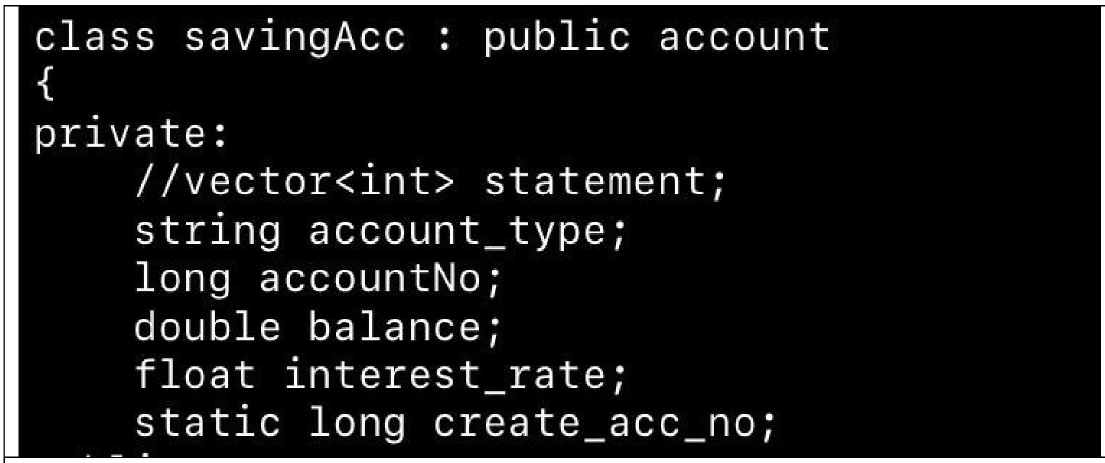
```
#ifndef VIRTUAL_CARD_H
#define VIRTUAL_CARD_H
#include <iostream>
#include <string>

using namespace std;

struct Virtual_card
{
    string expire_date = "23/06/2025";
    int cvv;
    static int create_cvv;
    long account_no ;
public:
    Virtual_card();
    void display();
    string tittle, name, surname;
};

#endif // VIRTUAL_CARD_H
```

Motivation:
Structure (Struct) is a way to group several related variables into one place. The struct in this code grouped cvv, account_no, create_cvv into one place when we were creating the virtual_card class. We also have public variables for the user details such tittle, name, and surname of the user. This will help display these details on the virtual card.

How have you met the objectives?	Cross (X) the appropriate box	If you think that you have met the objective completely, provide a short explanation to support the claim
Not met		
Partially		
Completely	x	

11. Data Structures

Screenshot:



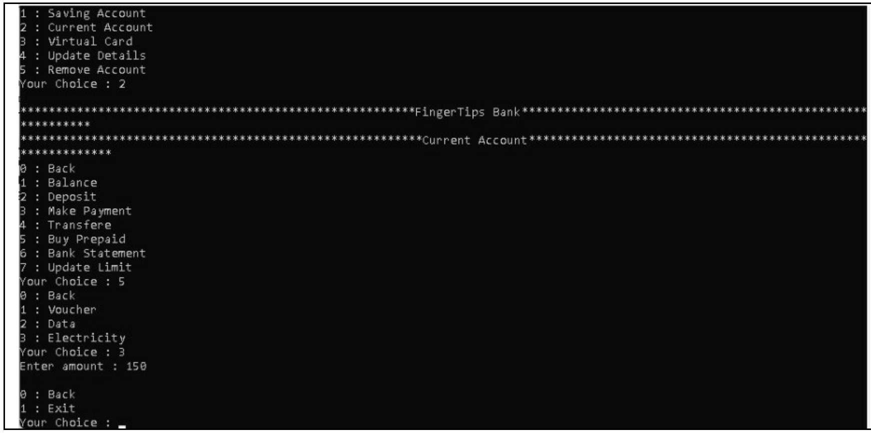
```
class savingAcc : public account
{
private:
    //vector<int> statement;
    string account_type;
    long accountNo;
    double balance;
    float interest_rate;
    static long create_acc_no;
```

Motivation:
The structure is a user-defined data type. It works similarly like arrays. Structures help you in grouping items of different types in a single group. It stores the collection of different data types. Each element of structure is a member. In this case it was used to name different variables (integer was used for account number, balance for double, account type a string was used, interest rate a float was used and a long was used).

How have you met the objectives?	Cross (X) the appropriate box	If you think that you have met the objective completely, provide a short explanation to support the claim
Not met		
Partially		
Completely	X	

Additional Item/s

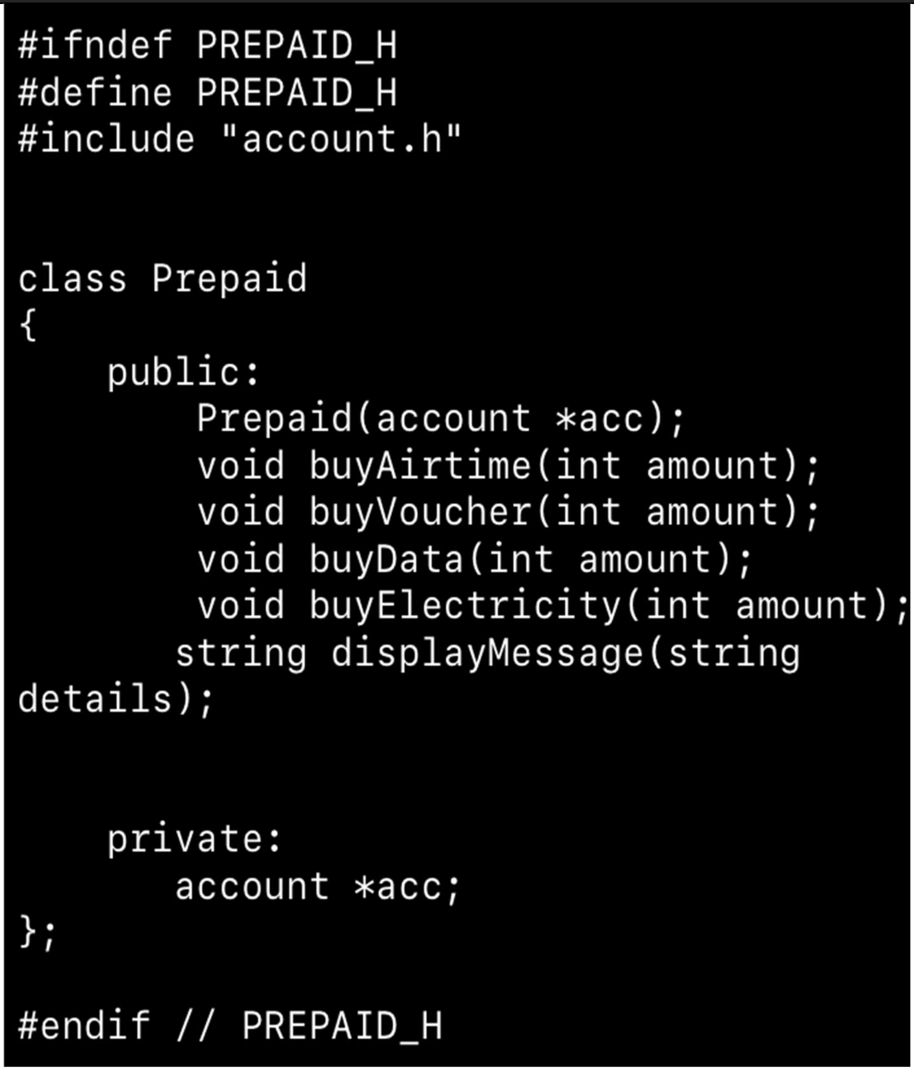
Screenshot	What does your FingerTip include?



```
1 : Saving Account
2 : Current Account
3 : Virtual Card
4 : Update Details
5 : Remove Account
Your Choice : 2

*******************************************************FingerTips Bank***********************************************
**********
*******************************************************Current Account**********************************************
*************
0 : Back
1 : Balance
2 : Deposit
3 : Make Payment
4 : Transfere
5 : Buy Prepaid
6 : Bank Statement
7 : Update Limit
Your Choice : 5
0 : Back
1 : Voucher
2 : Data
3 : Electricity
Your Choice : 3
Enter amount : 150

0 : Back
1 : Exit
Your Choice : _
```

The user has many options as buying airtime, data, voucher, electricity etc. The user can do this at their fingertips. Easy and very fast.

Code Screenshot



```
#ifndef PREPAID_H
#define PREPAID_H
#include "account.h"


class Prepaid
{
    public:
        Prepaid(account *acc);
        void buyAirtime(int amount);
        void buyVoucher(int amount);
        void buyData(int amount);
        void buyElectricity(int amount);
       string displayMessage(string
details);


    private:
       account *acc;
};

#endif // PREPAID_H
```

Appendix

FingerTip allows the user to print Bank Statements, which allows you to transfer money between accounts and print the bank statement with all your transactions.

FingerTip makes it easier for you to do your banking transactions.

FingerTip gives you more control over your finances

FingerTip allows you to manage your finances more efficiently

FingerTip is a good way to manage your accounts

FingerTip is more user-friendly than other existing channels such as bank branches, ATMs, and telephone banking.

FingerTip eliminates time constraints so that anyone can use banking services whenever they want.

FingerTip eliminates geographical limitations and increases the flexibility of mobility, thus allowing you to conduct my operations in any location with an internet connection.

FingerTip is a status symbol

FingerTip have more prestige than those who do not.

FingerTip have a high profile.

Use of updated technology like Fingertips can prevent loss of status.