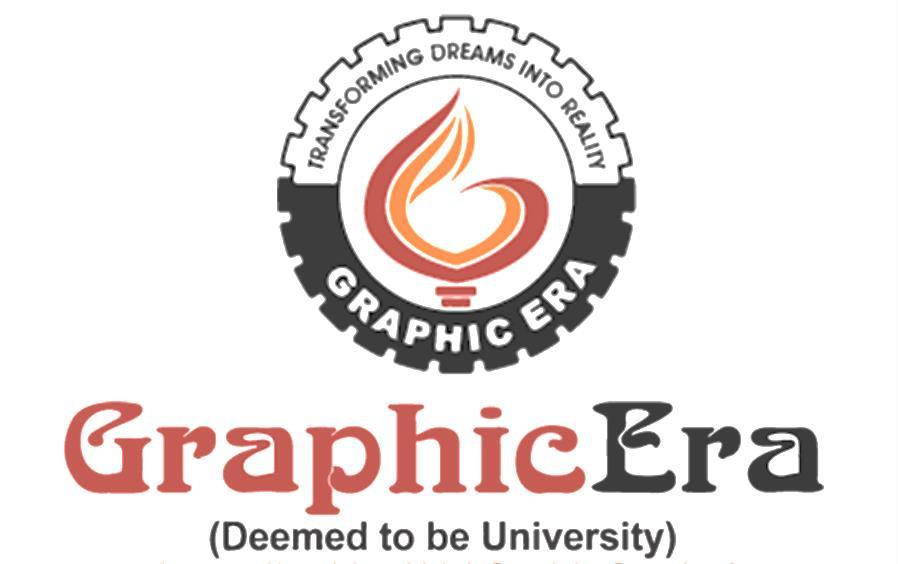
**3rd Sem Project Report**

on

# Banking Management System

(IT III Semester Mini Project) 2020-2021



Guided by: Submitted by:

Mr. Pankaj Kumar(Resource Person) Sujal Agarwal Roll. No.: 2015555 (IT-III-Sem)

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

## GRAPHIC ERA DEEMED TO BE UNVERSITY, DEHRADUN

### 1.1 About Project

#### 1.1.1 Overview

The Mini Project Bank Management System in C++ is a consoled based application and created using c++ programming language. This system is a simple mini project and compiled in Code::Blocks IDE using GCC compiler.This Bank Management System in utilizes classes and record dealing with highlights of C++. This System depends on an idea of client’s record information..

The program runs as a GUI Application, for an easy and user-friendly experience, developed in C++ using the Code Blocks IDE. The backend is supported by CPP file which keeps the code of project like: account details , total account holders list . A login system is also provided in the application for keeping the details secured by allowing only registered users to access the application.

#### 1.1.2 Background and Motivation

The client can likewise check/inquire for he/her balance which shows the record holder’s name with account number and amounts. The client can likewise see all the record holder’s listing and client can likewise close the record by giving the account number.

The motivation behind making this project was create an application that achieves the basic functionality of online Banking System for data because it provides Quick Access, Easy Handling and reduces our time.

### 1.2 Requirements of Project

#### 1.2.1 Hardware Requirement

CPU : x86 64-bit CPU (Intel or AMD Architecture)

RAM : 4 GB (minimum)

Storage : 1.5 MB (without .exe file) Monitor Resolution : 1920x1080

#### 1.2.2 Software Requirement

OS : Windows 10

Language : Code Blocks (IDE: Code Blocks/Atom)

**1.3 Modules of Project**.

#### 1.3.1 User Experience

The program is very user friendly and intuitive and uses a GUI interface implemented in C++ to communicate with the user. Various features are self explanatory. The user first needs to choose an option with provided options to use the application. Proper errors messages are displayed in case of any invalid selection while entering in (or any error as of that matter). After entering , the user have to enter the details according to its option and after that press enter key to go back to the dashboard.

The backend of the program is supported by a CPP program file which is locally saved in the system and contains the code of project. The data is easily available to the user on a single button.

**1.3.2**

**Add**

**New Account**

Th

e

programs a

llows

the

u

ser to add

a

new account

(

info

like A

ccount

No.,

Account type,

Account Holder

Name

,

Open

ing amount etc.

)

by filling a form which

appears on the same w

indow

after

Add

in

g

New

Account press enter

button

to go

the Dashboard. The

data

has

inserted

if there are no errors)

(

successfully

press enter

button

to exit from this windo

w

**1.3.3**

**Deposit**

**And Withdraw Money**

**From**

**Account**

The “

Deposit

Amount

and

W

ithdraw

A

mount

”

button

s

on

Dashboard allows the user to

deposit and

withdraw

money

from the user

account in

a

form filling

way

. The total

of

balance amount

also

calc

ulated

automatically

by

the program after filling the form

,

And

t

he user can also enq

u

ire his

Transaction

det

ai

ls by

pressing appr

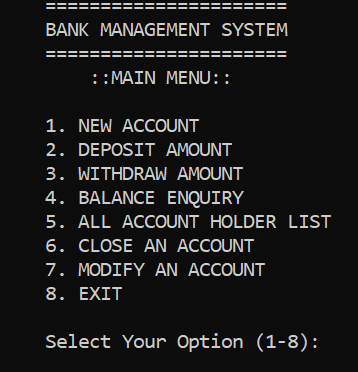
op

riate key

from

dashb

oard.



#### 1.3.4 All Account Holders List

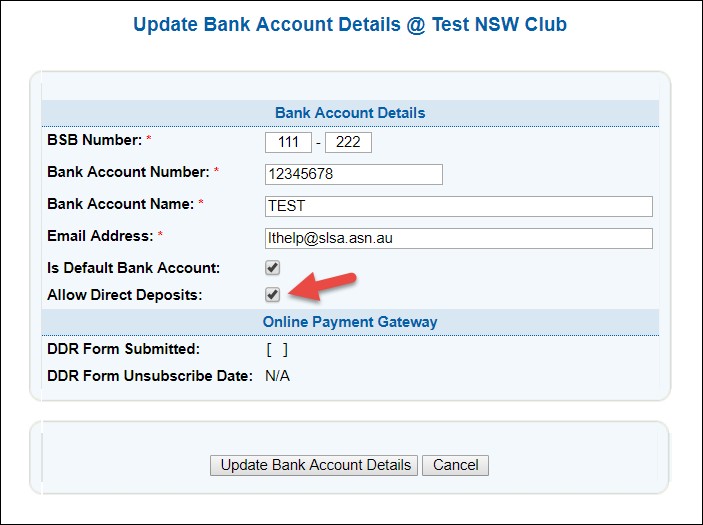
The user can also see the details of an individual account holder by entering his account number or can see the whole list of account holders of the bank by selecting a specific key on dashboard. This key will open a new window over the existing window with a form pre-filled with the data of the all account holders of bank in the form of a list.



#### 1.3.5 Close An Account

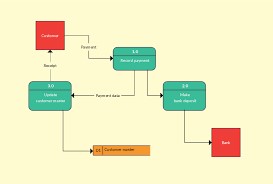
The user can also delete any or all details of account holders by choosing “Close Account” option after selecting it then enter Account Number one by one and pressing enter key to delete all the record.

#### 1.3.6 Update The Account Details

The user is also provided with a feature to Modify the account details if necessary. The user will have to select an option Modify the account after selecting user will have to enter Account Number for which he wants to modify the details of that Customer. After that user can change account holder’s name ,type of account, opening money etc from selected details of Account Holder. After editing all the info user can select “All Account Holder list” option to check the updated details.

### 1.4 Diagrams

#### 1.4.1 Data Flow Diagram



# **REFERENCE**

1. <https://www.w3schools.com/cpp/>
2. <https://www.tutorialspoint.com/cplusplus/index.htm>**3.** Book-Programming in C++ (Deitel and deitel)

#include<iostream>

#include<fstream>

#include<cctype>

#include<iomanip>

using namespace std;

class account

{

    int acno;

    char name[50];

    int deposit;

    char type;

public:

    void create\_account();

    void show\_account() const;

    void modify();

    void dep(int);

    void draw(int);

    void report() const;

    int retacno() const;

    int retdeposit() const;

    char rettype() const;

};

void account::create\_account()

{

    system("CLS");

    cout<<"\n\t\t\tEnter the Account No. : ";

    cin>>acno;

    cout<<"\n\n\t\t\tEnter the Name of the Account holder : ";

    cin.ignore();

    cin.getline(name,50);

    cout<<"\n\t\t\tEnter Type of the Account (C/S) : ";

    cin>>type;

    type=toupper(type);

    cout<<"\n\t\t\tEnter The Initial amount : ";

    cin>>deposit;

    cout<<"\n\n\t\t\tAccount Created..";

}

void account::show\_account() const

{

    cout<<"\n\t\t\tAccount No. : "<<acno;

    cout<<"\n\t\t\tAccount Holder Name : ";

    cout<<name;

    cout<<"\n\t\t\tType of Account : "<<type;

    cout<<"\n\t\t\tBalance amount : "<<deposit;

}

void account::modify()

{

    cout<<"\n\t\t\tAccount No. : "<<acno;

    cout<<"\n\t\t\tModify Account Holder Name : ";

    cin.ignore();

    cin.getline(name,50);

    cout<<"\n\t\t\tModify Type of Account : ";

    cin>>type;

    type=toupper(type);

    cout<<"\n\t\t\tModify Balance amount : ";

    cin>>deposit;

}

void account::dep(int x)

{

    deposit+=x;

}

void account::draw(int x)

{

    deposit-=x;

}

void account::report() const

{

    cout<<acno<<setw(10)<<" "<<name<<setw(10)<<" "<<type<<setw(6)<<deposit<<endl;

}

int account::retacno() const

{

    return acno;

}

int account::retdeposit() const

{

    return deposit;

}

char account::rettype() const

{

    return type;

}

void write\_account();

void display\_sp(int);

void modify\_account(int);

void delete\_account(int);

void display\_all();

void deposit\_withdraw(int, int);

int main()

{

    char ch;

    int num;

    do

    {

    system("CLS");

    cout<<"\n\n\t\t\t\t======================\n";

    cout<<"\t\t\t\tBANK MANAGEMENT SYSTEM";

    cout<<"\n\t\t\t\t======================\n";

        cout<<"\t\t\t\t    ::MAIN MENU::\n";

        cout<<"\n\t\t\t\t1. NEW ACCOUNT";

        cout<<"\n\t\t\t\t2. DEPOSIT AMOUNT";

        cout<<"\n\t\t\t\t3. WITHDRAW AMOUNT";

        cout<<"\n\t\t\t\t4. BALANCE ENQUIRY";

        cout<<"\n\t\t\t\t5. ALL ACCOUNT HOLDER LIST";

        cout<<"\n\t\t\t\t6. CLOSE AN ACCOUNT";

        cout<<"\n\t\t\t\t7. MODIFY AN ACCOUNT";

        cout<<"\n\t\t\t\t8. EXIT";

        cout<<"\n\n\t\t\t\tSelect Your Option (1-8): ";

        cin>>ch;

        switch(ch)

        {

        case '1':

            write\_account();

            break;

        case '2':

            system("CLS");

            cout<<"\n\n\t\t\tEnter The account No. : "; cin>>num;

            deposit\_withdraw(num, 1);

            break;

        case '3':

            system("CLS");

            cout<<"\n\n\t\t\tEnter The account No. : "; cin>>num;

            deposit\_withdraw(num, 2);

            break;

        case '4':

            system("CLS");

            cout<<"\n\n\t\t\tEnter The account No. : "; cin>>num;

            display\_sp(num);

            break;

        case '5':

            display\_all();

            break;

        case '6':

            system("CLS");

            cout<<"\n\n\t\t\tEnter The account No. : "; cin>>num;

            delete\_account(num);

            break;

         case '7':

            system("CLS");

            cout<<"\n\n\t\t\tEnter The account No. : "; cin>>num;

            modify\_account(num);

            break;

         case '8':

            system("CLS");

            cout<<"\n\n\t\t\tBrought To You By code-projects.org";

            break;

         default :cout<<"\a";

        }

        cin.ignore();

        cin.get();

    }while(ch!='8');

    return 0;

}

void write\_account()

{

    account ac;

    ofstream outFile;

    outFile.open("account.dat",ios::binary|ios::app);

    ac.create\_account();

    outFile.write(reinterpret\_cast<char \*> (&ac), sizeof(account));

    outFile.close();

}

void display\_sp(int n)

{

    account ac;

    bool flag=false;

    ifstream inFile;

    inFile.open("account.dat",ios::binary);

    if(!inFile)

    {

        cout<<"File could not be open !! Press any Key...";

        return;

    }

    cout<<"\n\t\t\tBALANCE DETAILS\n";

        while(inFile.read(reinterpret\_cast<char \*> (&ac), sizeof(account)))

    {

        if(ac.retacno()==n)

        {

            ac.show\_account();

            flag=true;

        }

    }

    inFile.close();

    if(flag==false)

        cout<<"\n\n\t\t\tAccount number does not exist";

}

void modify\_account(int n)

{

    bool found=false;

    account ac;

    fstream File;

    File.open("account.dat",ios::binary|ios::in|ios::out);

    if(!File)

    {

        cout<<"File could not be open !! Press any Key...";

        return;

    }

    while(!File.eof() && found==false)

    {

        File.read(reinterpret\_cast<char \*> (&ac), sizeof(account));

        if(ac.retacno()==n)

        {

            ac.show\_account();

            cout<<"\n\n\t\t\tEnter The New Details of account"<<endl;

            ac.modify();

            int pos=(-1)\*static\_cast<int>(sizeof(account));

            File.seekp(pos,ios::cur); //fncallat1353

            File.write(reinterpret\_cast<char \*> (&ac), sizeof(account));

            cout<<"\n\n\t\t\tRecord Updated";

            found=true;

          }

    }

    File.close();

    if(found==false)

        cout<<"\n\n\t\t\tRecord Not Found ";

}

void delete\_account(int n)

{

    account ac;

    ifstream inFile;

    ofstream outFile;

    inFile.open("account.dat",ios::binary);

    if(!inFile)

    {

        cout<<"File could not be open !! Press any Key...";

        return;

    }

    outFile.open("Temp.dat",ios::binary);

    inFile.seekg(0,ios::beg);

    while(inFile.read(reinterpret\_cast<char \*> (&ac), sizeof(account)))

    {

        if(ac.retacno()!=n)

        {

            outFile.write(reinterpret\_cast<char \*> (&ac), sizeof(account));

        }

    }

    inFile.close();

    outFile.close();

    remove("account.dat");

    rename("Temp.dat","account.dat");

    cout<<"\n\nRecord Deleted ..";

}

void display\_all()

{

    system("CLS");

    account ac;

    ifstream inFile;

    inFile.open("account.dat",ios::binary);

    if(!inFile)

    {

        cout<<"File could not be open !! Press any Key...";

        return;

    }

    cout<<"\n\n\t\tACCOUNT HOLDER LIST\n\n";

    cout<<"====================================================\n";

    cout<<"A/c no.      NAME           Type  Balance\n";

    cout<<"====================================================\n";

    while(inFile.read(reinterpret\_cast<char \*> (&ac), sizeof(account)))

    {

        ac.report();

    }

    inFile.close();

}

void deposit\_withdraw(int n, int option)

{

    int amt;

    bool found=false;

    account ac;

    fstream File;

    File.open("account.dat", ios::binary|ios::in|ios::out);

    if(!File)

    {

        cout<<"File could not be open !! Press any Key...";

        return;

    }

    while(!File.eof() && found==false)

    {

        File.read(reinterpret\_cast<char \*> (&ac), sizeof(account));

        if(ac.retacno()==n)

        {

            ac.show\_account();

            if(option==1)

            {

                cout<<"\n\n\t\t\tTO DEPOSITSS AMOUNT";

                cout<<"\n\n\t\t\tEnter The amount to be deposited: ";

                cin>>amt;

                ac.dep(amt);

            }

            if(option==2)

            {

                cout<<"\n\n\t\t\tTO WITHDRAW AMOUNT";

                cout<<"\n\n\t\t\tEnter The amount to be withdraw: ";

                cin>>amt;

                int bal=ac.retdeposit()-amt;

                if(bal<0)

                    cout<<"Insufficience balance";

                else

                    ac.draw(amt);

              }

            int pos=(-1)\*static\_cast<int>(sizeof(ac));

            File.seekp(pos,ios::cur);//fn1353

            File.write(reinterpret\_cast<char \*> (&ac), sizeof(account));

            cout<<"\n\n\t\t\tRecord Updated";

            found=true;

           }

         }

    File.close();

    if(found==false)

        cout<<"\n\n\t\t\tRecord Not Found ";

}