Group No.4

Bank Management System

SUBMITTED BY
Uzma Fatima (F20CSC03)
Amber Asif (F20CSC02)
Nisha Fatima(F20CSC13)
Bisma Imran(F20CSC29)

SUPERVISED BY

MS. DUAA BAIG

OBJECT ORIENTED PROGRAMMING
PROJECT REPORT SUBMITTED TO THE FACULTY OF COMPUTER SCIENCE

TABLE OF CONTENTS

1. INTRODUCTION 1
1.1. OBJECTIVE 1
1.2. DESCRIPTION 1
1.3. PROJECT FEATURES 1

2. CONCEPTS IMPLEMENTED 2
2.1. METHODOLOGY 2

3. FUNCTIONAL REQUIREMENTS 3
3.1. CRUID FUNCTION 3
3.112. CREATE ACCOUNT 3
3.13. UPDATE RECORD 3
3.14. VIEW YOUR RECORD 3
3.15. DELETE RECORD 3
3.16. DEPOSIT AMOUNT 4
3.17. WITHDRAW AMOUNT 4
3.18. VIEW ALL 4
3.2 pillars of oops........
3.21 inheritence........
3.22 abstraction........
3.23 polymorphism........
3.24 encapsulation........

4. NON-FUNCTIONAL REQUIREMENTS 4
5. SOURCE CODE 5
6. OUTPUTS 6
7. FLOWCHART 7

1 INTRODUCTION

1.1 OBJECTIVE

The utmost objective of our project "Bank Management System", as the name suggests is to store the record of the bank customers securely in a file and to provide ease to users to access them and perform some functionalities such as modify, delete or search their record.

1.2 DESCRIPTION

Considering the sensitivity of the bank customer's data, our project stores each record in a binary file which is humanly unreadable. User can access their data only if they succeed to enter correct username and the corresponding password which will maintain the security and integrity of the data. User can create, modify, delete and view their account and can deposit and withdraw account if they have enough balance.

1.3 PROJECT FEATURES

1 In this project following feature we’re going to implement:

2 Account creation if it doesn't exist, to maintain uniqueness of the data.

3 Account details modification (excluding Account Number and Username which remains unique for each member throughout).

4 Confirmation of password, providing the flexibility to the user.

5 Password hidden on the screen.

6 A firewall between unauthorized users by username and password protected layer.

7 To find desired record simply search through account number.

8 Details about the person will be shown with proper headings.

9 Only admin can view all the records except for the passwords of the users.

10 Delete account after verification through username and password.

2 CONCEPTS IMPLEMENTED

2.1 METHODOLOGY

This program will ask user to create, search, modify, withdraw, deposit and delete their account.

- Searched for the requirements needed for bank management system.
- Made basic code such as creating classes, their data members and functions.
- Looked for how many pillars of OOP could be implemented initially.
- The code was made and executed successfully on Dev C++

- Adjusted error bits and polymorphism to fulfill the course project requirement.
- Tried working on GUI on Dev C++.
- Lastly, we worked on and eventually successfully made our project report.

3 FUNCTIONAL REQUIREMENTS

A functional requirement defines a system or its component. It describes the functions a software must perform. A function is nothing but inputs, its behavior, and outputs. It can be a data manipulation, user interaction, or any other specific functionality which defines what function a system is likely to perform.

3.1 CRUD FUNCTION

Functional software requirements help you to capture the intended behavior of the system. This behavior may be expressed as functions, services or tasks or which system is required to perform. These functional requirements include the following:

3.11 CREATE ACCOUNT

This asks user to enter his account number first, if it exists then it will simply display the record of that user and display a small message that record already exists.

3.12 UPDATE RECORD

Prompts user to enter valid username and password, and after verification provides ease to user to modify his first name, last name, password and date of birth.

3.13 VIEW YOUR RECORD

This function allows user to search desired record by entering the valid Account Number.

3.14 DELETE RECORD

This function provides user a new feature to delete or remove desired record from the database by using just name or ID.

3.15 DEPOSIT AMOUNT

This Function provides user a new feature to deposit amount in his account once he succeeds to verify himself with correct username and password.

3.16 WITHDRAW AMOUNT

This Function provides user once he succeeds to verify himself with correct username and password a new feature to withdraw amount from his account

3.18 VIEW ALL

Only admin can view all the records except the passwords by verifying using the correct pin and username.

3.2 Pillars of OOP:

3.21 Inheritance:

This has been achieved through inheriting Account class from BasicInfo class

3.22 Abstraction:

Validation, and other calling functions has been hidden from the user hence hiding the complexity.

3.23 Polymorphism:

+ and – operator has been overloaded in Deposit and Withdraw functions.

3.24 Encapsulation:

All the data members are kept private. Data members and member function are wrap up in classes.

4 NON-FUNCTIONAL REQUIREMENTS

Non-Functional Requirement (NFR) specifies the quality attribute of a software system. They judge the software system based on Responsiveness, Usability, Security, Portability and other non-functional standards that are critical to the success of the software system. Following functions are used to achieve the nonfunctional requirements.

4.1 Check Password
4.2 Check String
4.3 Check Date
4.4 Check Username
4.5 Validate ID

Besides function we used error bits and hid the passwords on screen to ensure the security of the data.

5 SOURCE CODE

```
#include <iostream>// to perform cin cout
#include <fstream>//use file streams
#include <iomanip>// for error bits
#include <string>//string class for string fucntions
#include <cctype>// for isalpha,ispunct,isdigit,type
check functions
#include <cstring>//string class
#include <sstream>//string stream, convert integer in
string
#include <string.h>//string library
#include <stdio.h>//getch()
#include <conio.h>//console
#include <stdlib.h>//standard library, atoi, string to
integer
#include <windows.h>// showmessages,buttons
using namespace std;
int size;
class BasicInfo{        // base class
  protected:
unsigned long int id;
        char uname[20];
        char password[20];
        char fname[20];
        char lname[20];
        char dob[12];
        unsigned int balance=0;

};
class Account:public BasicInfo{     // inheritance

        public:
                void getfname();//get first name
                void getlname();//get last name
                void getuname();//get user name
                void getpassword();//get password
                void getdata();// get all the basic data
                void getdate();//get date of birth
                void getid(unsigned long int);//get acount
number
                int validateid();//check if id is correct
                void displaydata();//display data of the
user
                int checkstring(string str);//check if it
contains alphabets only
                int checkpassword(string str);//check
alpha numeric requirements
```

```
                int checknumber(string str);//check digits
only
                int checkdate(int,int,int);//check if date is
correct
                void storedata();//store data in file
                void getfromfile();//retrieve data from file
                int searchid (unsigned long int i);//search
id in file
                void viewall();//display all records
                void Delete(char*,char*);//delete account
                void Updatebasicdata(char*,char*);//calls
Modifybasicdata() and store in file
                void Modifybasicdata();//modify data
                void getbalance();//get balance
                void Updateamount(char* p,char*t,char
c);// update balance in file
                        int withdrawamount();//takes amount
and also checks if balance is enough
                                void depositamount();//takes
amount
                                void operator -(int);// operator
overloading function
                                void operator +(int);// operator
overloading function
```

```
                            int getchoice(string);//takes
choice and validates it
};
int Account::checkstring(string str){
int x=1;
   for(int i=0; (i<str.length());){

x=isalpha(str[i]);// if character is alphabet then
returns 2 else returns 0
    if(x!=0){// if character is an alphabet

    i++;// move to next index and check it
   }
  else if (x!=2){// if not an alphabet
  MessageBox(NULL,"Please enter a valid
string","BANK MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);

  break;// break from the loop
  x=0;// set x=0 for later comparison
  break;// finally break
  }

   }
```

```
if(x==0)// if not alphabet
    return 1;// return 1 to take string again
    else return 0;// All set!! Perfect!




}
 int Account::checkpassword(string str){
 int s=1,d=1,u=1,l=1,c,space;// initialization, can be
declared only as well



     for(int i=0; (i<str.length());){
       d=isdigit(str[i]);// checks if digit
       if(d!=0)// if digit
       break;// break from checking futher to another
check
       else
     i++;// move to next index if not a digit
 }
 if(d!=0){// if a digit
   for(int i=0; (i<str.length());){
u=isupper(str[i]);// check if upper case
```

```
    if(u!=0)// yes it's an upper case
    break;// stops from checking furhter and move to
another check
    else
    i++;// move to another character if not an upper
case
 }
 if(u!=0){// if upper case is true
    for(int i=0; (i<str.length());){

l=islower(str[i]);// checks if lower
    if(l!=0)//if character is in lower case
    break;// break from futher checking to another
check
    else
    i++;// else move to another character and keep
checking if lower
 }
 if(l!=0){// if lower is found
    for(int i=0; (i<str.length());){

c=ispunct(str[i]);// checks for special character
    if(c!=0)// special character found!!
    break;// break from checking futher to another
check
    else
    i++;// else check for another character
 }
 if(c!=0){
    for(int i=0; (i<str.length());){

  space=(!str[i]);// if space in the password
  if(space!=0)// if space found then break from futher
checking
    break;
    else
    i++;// else check rest of the characters
 }
 if (space==0)// if space not found then perfect,
return 0, which indicates that password is PERFECT
 return 0;
 else return 1;// EHEM!! Not fulfilling the
requirements,too weak
 }
 else {
```

```
          MessageBox(NULL,"Enter an alpha numeric
password with upper and lower case","BANK
MANAGEMENT SYSTEM",MB_ICONERROR|MB_OK);
// If any of the requirements doesn't get fulfilled,
ERROR MESSAGE
 return 1;// Weak Password
}
 }
 else {
          MessageBox(NULL,"Enter an alpha numeric
password with upper and lower case","BANK
MANAGEMENT SYSTEM",MB_ICONERROR|MB_OK);
 // If any of the requirements doesn't get fulfilled,
ERROR MESSAGE
 return 1;// Weak Password
}
 }
 else {
          MessageBox(NULL,"Enter an alpha numeric
password with upper and lower case","BANK
MANAGEMENT SYSTEM",MB_ICONERROR|MB_OK);
 // If any of the requirements doesn't get fulfilled,
ERROR MESSAGE
 return 1;// Weak Password
```

```
}

 }
 else {
	MessageBox(NULL,"Enter an alpha numeric
password with upper and lower case","BANK
MANAGEMENT SYSTEM",MB_ICONERROR|MB_OK);
 // If any of the requirements doesn't get fulfilled,
ERROR MESSAGE
 return 1;// Weak Password
}
}
int Account::validateid(){// checks if id is 5 digits long
and is an unsigned integer only
	 int n=1;
	unsigned long int i;
   while(n==1){// until correct

	while(1){// until no error bits
	cin.unsetf(ios::skipws);// donot skip empty
inputs and whitespaces
	cout<<"\nEnter Account Number: ";
	cin>> i;
```

```
        if(cin.good()){// if input is correct
                cin.ignore(5,'\n');// takes 5 characters and
ignore the rest, or until user enters the enter key
                n=0;// perfect
                break;// break from the loop
                }

cin.clear();// clear the buffer
        MessageBox(NULL,"INVALID!!!","BANK
MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);// error message
if requirements not fulfill
cin.ignore(5,'\n');

  }
  if(i>99999||i<10000){// must be 5 digits
                n=1;
        MessageBox(NULL,"INVALID!! ENTER 5 DIGIT
ID","BANK MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);
                }
                else n=0;
                cin.clear();
                }
```

```
if(n==0)
return i;

cin.clear();

}
void Account::getid(unsigned long int i){// after
validating id, set id
  id=i;
}
void Account::getfname(){// gets first name
   int n=1;// 1 indicates that string is incorrect, for
futher check it is initialized 1

   while(n==1){// takes first name until n=0
      cout<<"\n\nEnter First name: ";

   cin.getline(fname,10);// takes first 10 characters
only
   string fn=fname;// converting to string to use string
class functions

   n=fn.empty();// checks if user entered an empty
string or not
```

```
   if (n==1){// if empty
         MessageBox(NULL,"CAN'T BE LEFT
BLANK","BANK MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);// display error
message
   }
   else {
   n=checkstring(fn);// if not blank then check if string
doesnt contain digits or special character
}
}

}
void Account::getlname(){// get last name
   int n=1;
   while(n==1){// until input is correct

      cout<<"\n\nEnter last name: ";
   cin.getline(lname,10,'\n');
   string ln=lname;// array to string
   n=ln.empty();// string class member function to
check if empty or not
   if (n==1){// if empty
```

```
        MessageBox(NULL,"CAN'T BE LEFT
BLANK","BANK MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);
   }
   else {
   n=checkstring(ln);// if not empty then check
alphabets only
}
}

}
void Account::getuname(){// get username
   int n=1;



   while(n==1){// until correct input
      cout<<"\n\nEnter username: ";

    cin.getline(uname,10,'\n');
   string un=uname;
   n=un.empty();// checks empty or not
   if(n==1){// if empty
```

```
        MessageBox(NULL,"CAN'T BE LEFT
BLANK","BANK MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);
}
   }

}
void Account::getpassword(){// takes password
   int n=1;

   while(n==1){
   cout<<"\n\t\tPassword Must Be greater than 8";//
8-15 characters long
while(n==1)
{
   cout<<"\n\nPassword: ";
   char p;// passwords character by character
   for(int i=0;i<20;){
        p=getch();// get character
        if((p!='\b')&&(p!='\r')){// if not backspace or
enter key pressed

        password[i]=p;
      ++i;
```

```
        cout<<"*"; }// output stars to hide password
        else if(p=='\b'&&i>=1){// if backspace and
atleast one character on the screen

        cout<<"\b \b";
        --i;}
        else if(p=='\r'){// if enter pressed
              password[i]='\0';// null in the last
character to terminate
              break;
        }
  }
   string pass=password;// array to string
   n=pass.empty();// checks if empty
   if (n==1)
        MessageBox(NULL,"CAN'T BE LEFT
BLANK","BANK MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);
   else {
      if (pass.length()<8){
        MessageBox(NULL,"PASSWORD TOO
SHORT","BANK MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);
      n=1;
   }
   else if(pass.length()>15){// must be less than 16
               MessageBox(NULL,"PASSWORD TOO
LONG","BANK MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);
       n=1;
   }
       else {
       n=checkpassword(password);// checks if an
alpha numeric password with upper and lower case
   }
}
}

if(n==0)
{char pass2[20];
   cout<<"\n\nConfirm your password: ";// password
confirmation
    char p;
   for(int i=0;i<20;){
          p=getch();
          if((p!='\b')&&(p!='\r')){
```

```
        pass2[i]=p;
      ++i;
        cout<<"*"; }
        else if(p=='\b'&&i>=1){

        cout<<"\b \b";
        --i;}
        else if(p=='\r'){
                pass2[i]='\0';
                break;
        }
   }

   if(!strcmp(pass2,password)){// compare both the
passwords

   n=0;}
   else{
        MessageBox(NULL,"PASSWORD DIDN'T
MATCH","BANK MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);
    n=1;
}
}
```

```
}
//Successful
}
void Account::getdata(){

getfname();// get first name
getlname();// get last name
getuname();// get user name
getpassword();// get password name
getdate();// get dob
getbalance();// get initial balance

}
void Account::getbalance(){

        while(1){// until unsigned integer only
        cin.unsetf(ios::skipws);
        cout<<"\n\nEnter Balance: ";
        cin>> balance;

        if(cin.good()){
               break;
               }
```

```
cin.clear();
         MessageBox(NULL,"INVALID!!!","BANK
MANAGEMENT SYSTEM",MB_ICONERROR|MB_OK);
cin.ignore(10,'\n');

   }
}
void Account::getdate(){

char year[5];
char day[3];
char month[3];
char date[12];

int y,m,d;// year, month, day
int n=1;
while (n==1){
       cout<<"\n\nEnter DATE OF BIRTH in yy/mm/dd
format\n";
   while(true){
         cin.unsetf(ios::skipws);
         cout<<"\n\nEnter year:  ";
cin.getline(year,5);
 y= atoi(year);// change string in int
```

```
          if(cin.good()){

                  break;
                  }
cin.clear();
          MessageBox(NULL,"INCORRECT YEAR","BANK
MANAGEMENT SYSTEM",MB_ICONERROR|MB_OK);
cin.ignore(5,'\n');
   }
//cout<<y;
while(true){
          cin.unsetf(ios::skipws);
          cout<<"\n\nEnter month:  ";
cin.getline(month,3,'\n');
 m=atoi(month);// change string in integer
          if(cin.good()){

                  break;
                  }
cin.clear();
          MessageBox(NULL,"INCORRECT
MONTH","BANK MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);
cin.ignore(3,'\n');
```

```
   }

while(true){
          cin.unsetf(ios::skipws);
   cout<<"\n\nEnter day:  ";

cin.getline(day,3,'\n');
 d=atoi(day);
          if(cin.good()){


                 break;
                 }
cin.clear();
          MessageBox(NULL,"INCORRECT DAY","BANK
MANAGEMENT SYSTEM",MB_ICONERROR|MB_OK);
cin.ignore(10,'\n');
   }


n=checkdate(y,m,d);// passing integer values of year,
month and day
}
```

```
if(strlen(month)==1){//  if 1 to 9 entered
   int temp;
temp=month[0];
month[0]='0';
month[1]=temp;//if month is 2 then set 02, just an
example
}


if(strlen(day)==1){//  if 1 to 9 entered
   int temp;
temp=day[0];
day[0]='0';
day[1]=temp;//if day is 2 then set 02 just an example
}
 int i=0;
for( int j=0;j<2;j++){

   date[i++]=day[j];

}

date[i++]='-';// separate day month and year with -
   for(int j=0;j<2;j++){
```

```
   date[i++]=month[j];

}
date[i++]='-';
   for(int j=0;j<4;j++){
   date[i++]=year[j];
}
cout<<"\nDate:  "<<date;
strcpy(dob,date);// copy array to actual data member
of date

}
int Account::checkdate(int year,int month, int day)
{
    int n=1;
   if((1000 <= year )&&(year<= 2015))
     {
       if((month==1 || month==3 || month==5||
month==7|| month==8||month==10||month==12)
&& day>0 && day<=31)// 31 days exist in
1,3,5,7,8,10,12 months only
       n=0;
       else
    if(month==4 || month==6 || month==9||
month==11 && day>0 && day<=30)
                n=0;
        else
          if(month==2)
            {
            if((year%400==0 || (year%100!=0 &&
year%4==0)) && day>0 && day<=29)// checks leap
year
                  n=0;
            else if(day>0 && day<=28)
                  n=0;
            else{
                n=1;
          MessageBox(NULL,"Day,Month,or Year out of
range... Please enter again","BANK MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);


                 }


            }
       else{
```

```
                n=1;
          MessageBox(NULL,"Day,Month,or Year out of
range... Please enter again","BANK MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);
                 }
     }
    else{
                n=1;
          MessageBox(NULL,"Day,Month,or Year out of
range... Please enter again","BANK MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);
                 }

 return n;
}
   void Account::displaydata(){
          cout<<id<<"         "<<uname<<"
"<<fname<<" "<<lname<<"         "<<dob<<"
"<<balance<<endl;
   }
   void Account::storedata(){
       ofstream f;
       f.open("BasicInfo.txt",ios::app|ios::binary);//
opens binary file in append mode
```

```
       f.write((char*)this,sizeof(*this));// open file in
write mode write object, and  size of object indicated
by sizeof(*this)
       f.close();
   }
   int Account::searchid(unsigned long int i){
    int flag=0;
    ifstream in;
       in.open("BasicInfo.txt",ios::in|ios::binary);
   // if(!in)
    //{
     //   cout<<"\nfile not found";
    //}
    //else
    //{
       in.read((char*)this,sizeof(*this));// read from
file, where to read, size?
       while(!in.eof())
       {
          if(i==id)
          {

                flag=1;
cout<<"_____________________________________________
__________________________________________"<<endl;
              cout<<"Acc.Number              "<<"User
Name "<<"Name"<<"    "<<"Balance"<<endl;

  cout<<"________________________________________
___________________________________________"<<endl
;
              displaydata();
              break;

         }
              in.read((char*)this,sizeof(*this));
}
if (flag==0)
{
   return 0;
}
else return 1;
in.close();
  }//}
  void Account::viewall()     {
  Account ac;
```

```
   ifstream inFile;
   inFile.open("BasicInfo.txt",ios::binary);
   if(!inFile)
   {
                      MessageBox(NULL,"File could not be
open !! Press any Key...","BANK MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);
   }
   cout<<"\n\n\t\tACCOUNT HOLDER LIST\n\n";
   cout<<"=====================================
=================================\n";
   cout<<"A/c no.          USER NAME        NAME
DOB      Balance\n";
   cout<<"=====================================
=================================\n";
   while(inFile.read(reinterpret_cast<char *> (&ac),
sizeof(Account)))// type casting, data stores
character wise in file
   {
        ac.displaydata();
   }
   inFile.close();
}
```

```
void Account::Delete(char* p,char*t)// passing array
reference
   {
      int pos, flag = 0;
   ifstream ifs;
   ifs.open("BasicInfo.txt", ios::in | ios::binary);
   ofstream ofs;
   ofs.open("temp.dat", ios::out | ios::binary);
   while (!ifs.eof()){

      ifs.read((char*)this, sizeof(Account));

      // if(ifs)checks the buffer record in the file
      if (ifs) {
         if ((!strcmp(p,password))
&&(!strcmp(t,uname))){// if username and
corresponding password match
            flag = 1;
            cout << "The deleted record is \n";
            // display the record
            displaydata();
         }
         else {
            // copy the record of "BasicInfo" file to
"temp" file
            ofs.write((char*)this, sizeof(Account));
         }
      }
   }
   ofs.close();
   ifs.close();
   // delete the old file
   remove("BasicInfo.txt");
   // rename new file to the older file
   rename("temp.dat", "BasicInfo.txt");
   if (flag == 1)
      cout << "\n\n\t\tRecord Successfully Deleted
\n";
   else
                    MessageBox(NULL,"RECORD NOT
FOUND","BANK MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);
}
int Account::getchoice(string str){// parameter for
displaying later in the prog what to update
        int choice;
   int n=1;
   while(n==1){

   cout<<str;// Give choice to user whether to update
or not
   while(true){
          cin.unsetf(ios::skipws);
          cout<<"\nChoice: ";
          cin>> choice;
          if(cin.good()){
                 cin.ignore(1,'\n');
                 break;
                 }
cin.clear();
                        MessageBox(NULL,"INVALID
CHOICE","BANK MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);
cin.ignore(1,'\n');
   }
   if((choice!=2)&&(choice!=1)){
                        MessageBox(NULL,"INVALID
CHOICE","BANK MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);
   n=1;
   getch();
}
else
n=0;
cin.clear();
}
if (choice==1)
return 1;
else return 0;
}
  void Account:: Modifybasicdata(){
   int choice;
   choice=getchoice("\n\n\t\tUpdate First Name??
1-Yes  2-No");
   if (choice==1)
   {
          getfname();
     }
   choice=getchoice("\n\n\t\tUpdate Last Name??
1-Yes  2-No");
   if (choice==1)
   {
          getlname();
     }
   choice=getchoice("\n\n\t\tUpdate Password??    1-
Yes  2-No");
   if (choice==1)
   {
         getpassword();
     }
   choice=getchoice("\n\n\t\tUpdate DOB??    1-Yes
2-No");
   if (choice==1)
   {
         getdate();
     }
  }
void Account::Updatebasicdata(char*p,char*t)
{
   bool found=false;
   fstream File;
   File.open("BasicInfo.txt",ios::binary|ios::in|ios::out
);
   if(!File)
   {
                                 MessageBox(NULL,"File could
not be open !! Press any Key...","BANK
MANAGEMENT SYSTEM",MB_ICONERROR|MB_OK);
```

```
        return;
    }
    while(!File.eof() && found==false)// until record
found or the file end
    {
        File.read(reinterpret_cast<char *> (this),
sizeof(*this));
            if ((!strcmp(p,password))
&&(!strcmp(t,uname)))
        {
            displaydata();
            cout<<"\n\n\t\tEnter The New Details of
account"<<endl;
            cin.ignore(0);
            Modifybasicdata();
            int pos=(-
1)*static_cast<int>(sizeof(*this));// file pointer
moves to next record so position of pointer-1
            File.seekp(pos,ios::cur);
            File.write(reinterpret_cast<char *> (this),
sizeof(*this));
            system("cls");// clear the screen
            cout<<"\n\n\t\t Record Updated\n\n";
            found=true;
            displaydata();
            getch();// pause screen until user presses
any key
        }
  }
  File.close();
  if(found==false)
                        MessageBox(NULL,"RECORD
NOT FOUND","BANK MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);
}
void Account::operator +(int amount){
  balance+=amount;// object.balance+integer---------
operator overloading definition
       }
       void Account::operator -(int amount){
            balance-=amount;// object.balance-
integer--------- operator overloading definition
       }
  void Account:: Updateamount(char* p,char*t,char
c){
            bool found=false;
            int n=1;
  fstream File;
  File.open("BasicInfo.txt",ios::binary|ios::in|ios::out
);
  if(!File)
  {
                          MessageBox(NULL,"File could
not be open !! Press any Key...","BANK
MANAGEMENT SYSTEM",MB_ICONERROR|MB_OK);
       return;
  }
  while(!File.eof() && found==false)
  {
       File.read(reinterpret_cast<char *> (this),
sizeof(*this));
       if ((!strcmp(p,password))
&&(!strcmp(t,uname)))
       {found=true;
            if(c=='d'){// if deposit
            depositamount();
       }
       else
       n= withdrawamount();
       if (n==1)
{
            int pos=(-1)*static_cast<int>(sizeof(*this));
            File.seekp(pos,ios::cur);
            File.write(reinterpret_cast<char *> (this),
sizeof(*this));
            cout<<"\n\n\t\t Record Updated\n\n";
  cout<<"
==================================================
===========================\n";
  cout<<"A/c no.        USER NAME        NAME
DOB      Balance\n";
  cout<<"=============================================
=================================\n";
            displaydata();
        }  }
  }
  File.close();
  if(found==false)
                            MessageBox(NULL,"RECORD
NOT FOUND","BANK MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);
        }
        void Account:: depositamount(){
            unsigned int amount;
  while(1){
        cin.unsetf(ios::skipws);
        cout<<"\n\nEnter Amount: ";
        cin>> amount;
        if(cin.good()){
              break;
              }
cin.clear();

  MessageBox(NULL,"INVALID!!","BANK
MANAGEMENT SYSTEM",MB_ICONERROR|MB_OK);
cin.ignore(8,'\n');
  }
  *this+amount;
  }
  int Account:: withdrawamount(){
        unsigned int amount;
        while(1){
        cin.unsetf(ios::skipws);
        cout<<"\n\nEnter Amount: ";
        cin>> amount;
        if(cin.good()){
              break;
              }
cin.clear();
```

```
   MessageBox(NULL,"INVALID","BANK
MANAGEMENT SYSTEM",MB_ICONERROR|MB_OK);
cin.ignore(8,'\n');
   }
   if (balance>=amount){
   *this-amount;
   return 1;
   }
   else{
                                   MessageBox(NULL,"NOT
ENOUGH BALANCE","BANK MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);
          return 0;
   }
          }
int Menu();
int CreateAccount();
void UpdateAccount();
void DepositAmount();
void WithdrawAmount();
void ViewYourAccount();
void DeleteAccount();
void ViewAll();
```

```
   int main(){
          int c;
          while(1){
                 system("cls");
                 switch(Menu())
                 {
                        case 1:

                               cin.clear();
                                c=CreateAccount();
                               system("cls");
                               if(c==1){
                               cout<<"\n\n\n\n\t\tAccount
Created Successfully";
                               getch();
                               cin.clear();}
                               break;
                        case 2:

                               UpdateAccount();
                               break;
                        case 3:

                               DepositAmount();
                               break;
                        case 4:

                                WithdrawAmount();

                                break;
                        case 5:

                               ViewYourAccount();
                               break;
                        case 6:

                        DeleteAccount();
                               break;
                                   case 7:

                        ViewAll();
                               break;
                               case 8:

                                   cout<<"\n\n\tThank you for
using this application";
                                   cout<<"\n\n\tPress any key
to exit";
```

```
                                        getch();
                                        exit(0);
                                        default:
                                        MessageBox(NULL,"INVALID
CHOICE","BANK MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);
                                             getch();
                                             cin.clear();
                                                  }
       }

  }
int Menu(){
  int choice=0;
cin.clear();
cout<<"\n\n\t\t\t\t\tBANK MANAGEMENT
SYSTEM\n";
  cout<<"\n\n\n\t\t\t\t1-Create Account";
  cout<<"\n\n\t\t\t\t2-Update Account";
  cout<<"\n\n\t\t\t\t3-Deposit Amount";
  cout<<"\n\n\t\t\t\t4-Withdraw Amount";
  cout<<"\n\n\t\t\t\t5-View Your Account";
  cout<<"\n\n\t\t\t\t6-Delete Account";
  cout<<"\n\n\t\t\t\t7-View All";
```

```
   cout<<"\n\n\t\t\t\t8-Exit";
   cout<<"\n\n\n\t\t\t\tEnter Your Choice";
   int n=1;

   while(n==1){

   while(true){
          cin.clear();
          cin.unsetf(ios::skipws);
          cin.clear();
          cout<<"\n\t\t\t\tChoice: ";
          cin>> choice;
          if(cin.good()){
                 cin.ignore(1);
                 break;
                 }
cin.clear();

   MessageBox(NULL,"INCORRECT INPUT","BANK
MANAGEMENT SYSTEM",MB_ICONERROR|MB_OK);
cin.ignore(1);

   }
   n=0;
cin.clear();
}
return choice;
cin.clear();
}
int CreateAccount(){
   system("cls");
   cout<<"\n\t\t\tCreate Account\n";
    int found;
   unsigned long int checkid;
   Account ac;
cin.clear();
    checkid=ac.validateid();
    found=  ac.searchid(checkid);// checks if id exists
or not
   if (found==0){ // if doesnt exist
          ac.getid(checkid);
                 ac.getdata();
   ac.storedata();

   }
```

```
                              MessageBox(NULL,"RECORD
EXIST","BANK MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);
  if(found==0){
       return 1;
  }

cin.clear();
  getch();

}
void UpdateAccount(){
  system("cls");
  cout<<"\n\t\t\tUpdate Account\n";
  Account a;
  char password[20];
       char user[20];

  cout<<"\nEnter username: ";
  //   cin.ignore();
       cin.getline(user,10,'\n');
  //   cout<<"\nEntered user name is"<<user;
  //   cin.ignore();
       cout<<"\nEnter password: ";
```

```
 char p;
 for(int i=0;i<20;){
       p=getch();
       if((p!='\b')&&(p!='\r')){

       password[i]=p;
    ++i;
       cout<<"*"; }
       else if(p=='\b'&&i>=1){

       cout<<"\b \b";
       --i;}
       else if(p=='\r'){
              password[i]='\0';
              break;
       }
 }
 //cout<<"\nPassword was..."<<password<<endl;
 system("cls");
 cout<<"\n\n\n\t\t\t\tYour Record\n\n\n";
              a.Updatebasicdata(password,user);
                     getch();

}
```

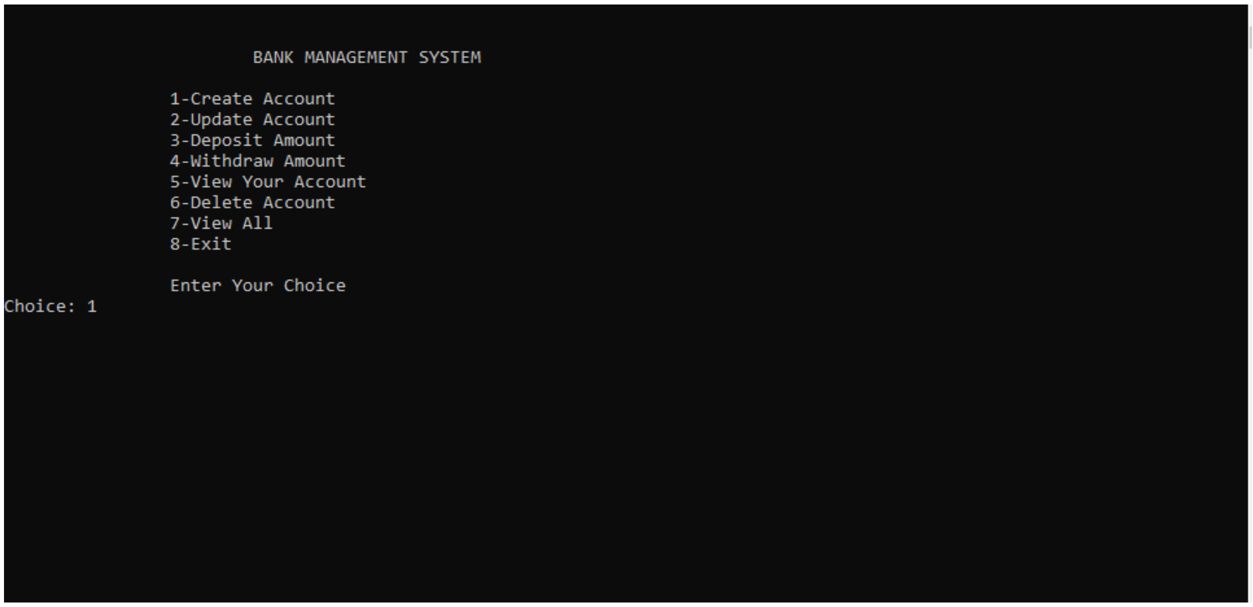
```
void DepositAmount(){
   system("cls");
   cout<<"\n\t\t\tDeposit Amount\n";
   Account d;
char password[20];
char user[20];
char p;

   cout<<"\nEnter username: ";

          cin.getline(user,10,'\n');

          cout<<"\nEnter password: ";
   for(int i=0;i<20;){
          p=getch();
          if((p!='\b')&&(p!='\r')){

          password[i]=p;
      ++i;
          cout<<"*"; }
          else if(p=='\b'&&i>=1){

          cout<<"\b \b";
          --i;}
          else if(p=='\r'){
                 password[i]='\0';
                 break;
          }
   }

   d.Updateamount(password,user,'d');
          getch();
   cin.clear();
}
void WithdrawAmount(){
   system("cls");
   cout<<"\n\t\t\t Withdraw Amount\n";
   Account w;
char password[20];
char user[20];
char p;

   cout<<"\nEnter username: ";

          cin.getline(user,10,'\n');

          cout<<"\nEnter password: ";
   for(int i=0;i<20;){
          p=getch();
          if((p!='\b')&&(p!='\r')){

          password[i]=p;
        ++i;
          cout<<"*"; }
          else if(p=='\b'&&i>=1){

          cout<<"\b \b";
          --i;}
          else if(p=='\r'){
               password[i]='\0';
               break;
          }
     }

     w.Updateamount(password,user,'w');
   getch();
     cin.clear();

   }
   void ViewYourAccount(){

     system("cls");
   cout<<"\n\t\t\tView Your account\n\n";
   int found=0;
   unsigned long int checkid;
   Account v;

    checkid=v.validateid();
    found=  v.searchid(checkid);
    if(found==0)
                MessageBox(NULL,"RECORD NOT
FOUND","BANK MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);

         getch();
}
void DeleteAccount(){
   system("cls");
   cout<<"\n\t\t\tDelete Account\n\n";
   Account da;
char password[20];
char user[20];
char p;

   cout<<"\nEnter username: ";
```

```
}



        cin.getline(user,10,'\n');

        cout<<"\nEnter password: ";
   for(int i=0;i<20;){
        p=getch();
        if((p!='\b')&&(p!='\r')){

        password[i]=p;
      ++i;
        cout<<"*"; }
        else if(p=='\b'&&i>=1){

        cout<<"\b \b";
        --i;}
        else if(p=='\r'){
                password[i]='\0';
                break;
        }
   }

   da.Delete(password,user);
        getch();


}
```

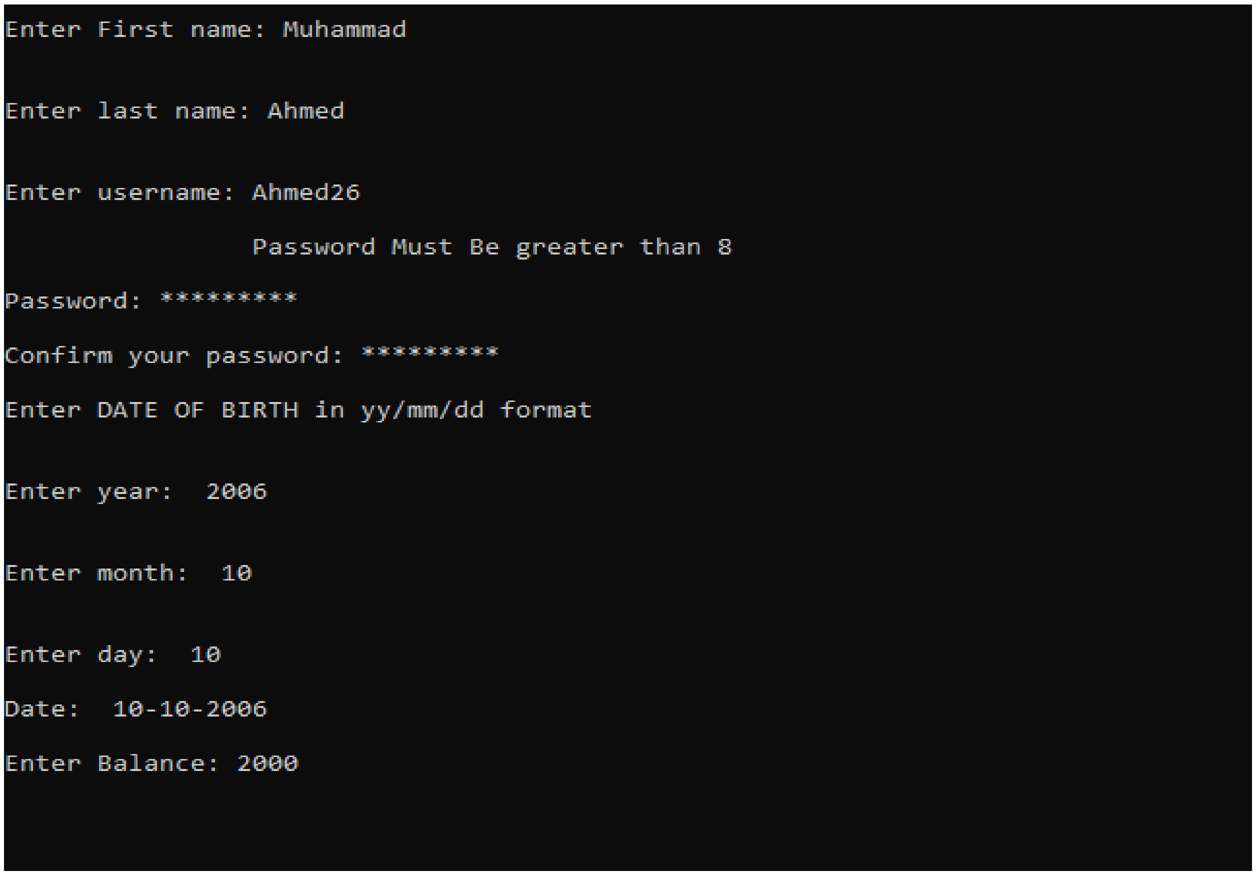
```
void ViewAll(){
  system("cls");
        cout<<"\n\t\t\tView All\n\n";
  Account va;
  char pin[]="99113322";
  char username[]="AdminBBMM23";
  char u[13];
  char pinc[9];
  char p;

cout<<"\nEnter user name: ";
  cin.getline(u,13);
cout<<"\nEnter pin";
  for(int i=0;i<10;){
        p=getch();
        if((p!='\b')&&(p!='\r')){

        pinc[i]=p;
     ++i;
        cout<<"*"; }
        else if(p=='\b'&&i>=1){

        cout<<"\b \b";
        --i;}
        else if(p=='\r'){
                pinc[i]='\0';
                break;
        }
   }

   if((!strcmp(u,username)&&(!strcmp(pin,pinc)))){//
checking if pin and username match no not
        va.viewall();
        getch();
        cin.clear();
   }
   else{


   MessageBox(NULL,"INCORRECT PIN OR
USERNAME","BANK MANAGEMENT
SYSTEM",MB_ICONERROR|MB_OK);
   cin.clear();

}

   getch();
   cin.clear();
```

}

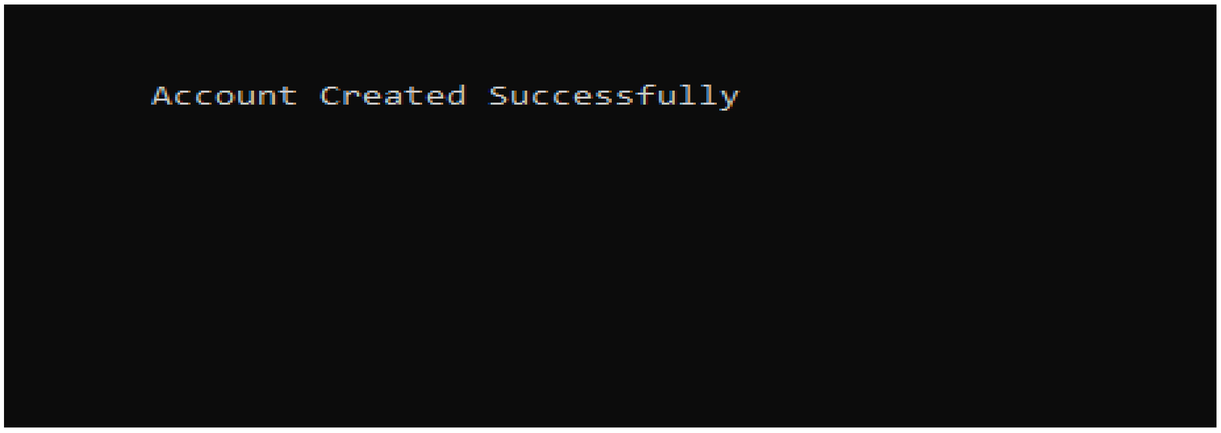
6 OUTPUTS



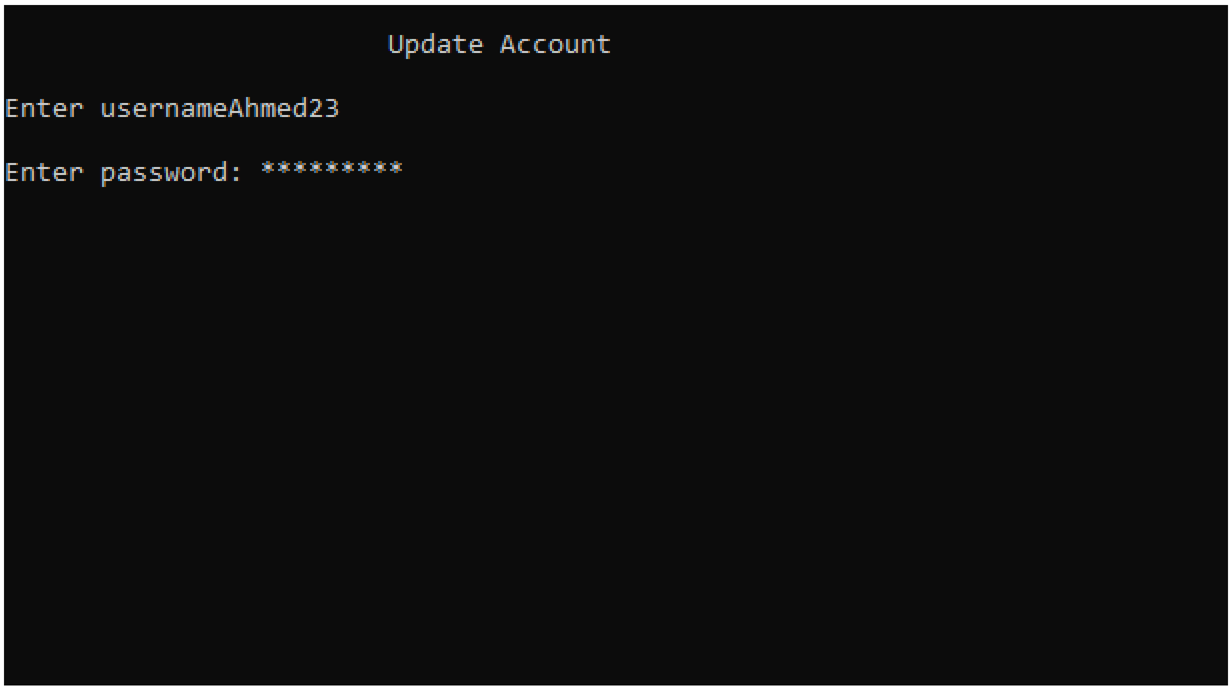
```
                        BANK MANAGEMENT SYSTEM

                1-Create Account
                2-Update Account
                3-Deposit Amount
                4-Withdraw Amount
                5-View Your Account
                6-Delete Account
                7-View All
                8-Exit

                Enter Your Choice
Choice: 1
```



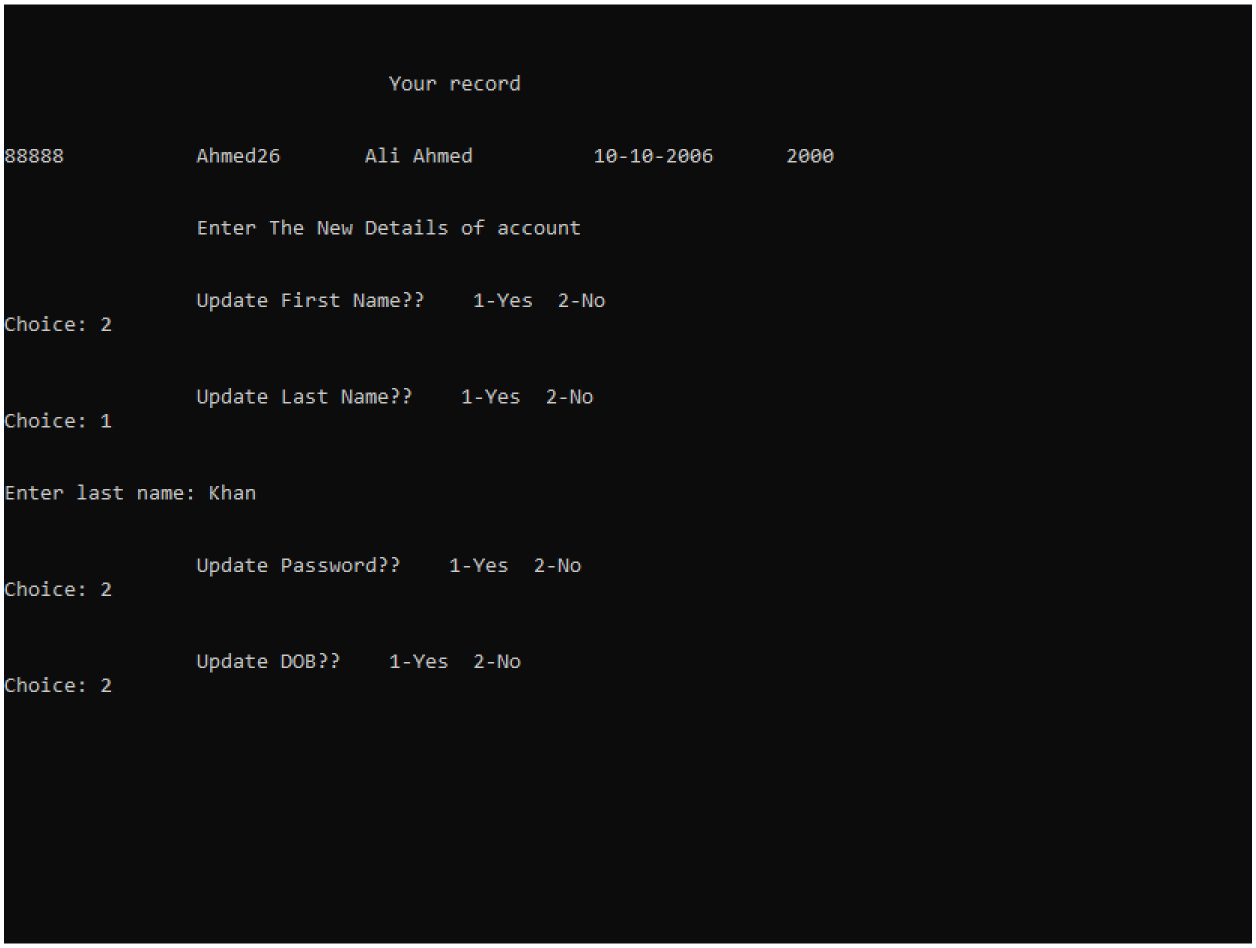
```
Enter First name: Muhammad

Enter last name: Ahmed

Enter username: Ahmed26
                Password Must Be greater than 8
Password: *********
Confirm your password: *********
Enter DATE OF BIRTH in yy/mm/dd format

Enter year:  2006

Enter month:  10

Enter day:  10
Date:  10-10-2006
Enter Balance: 2000
```



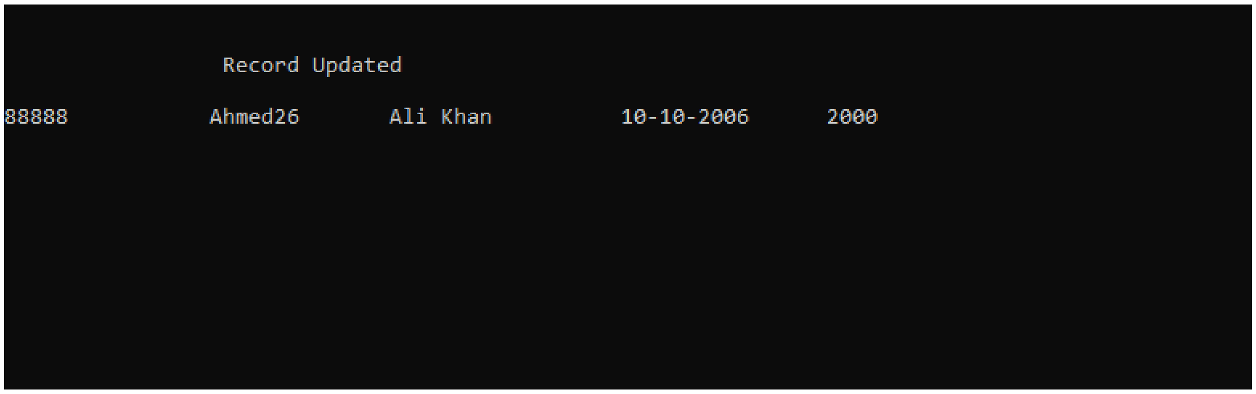
```
         Account Created Successfully
```



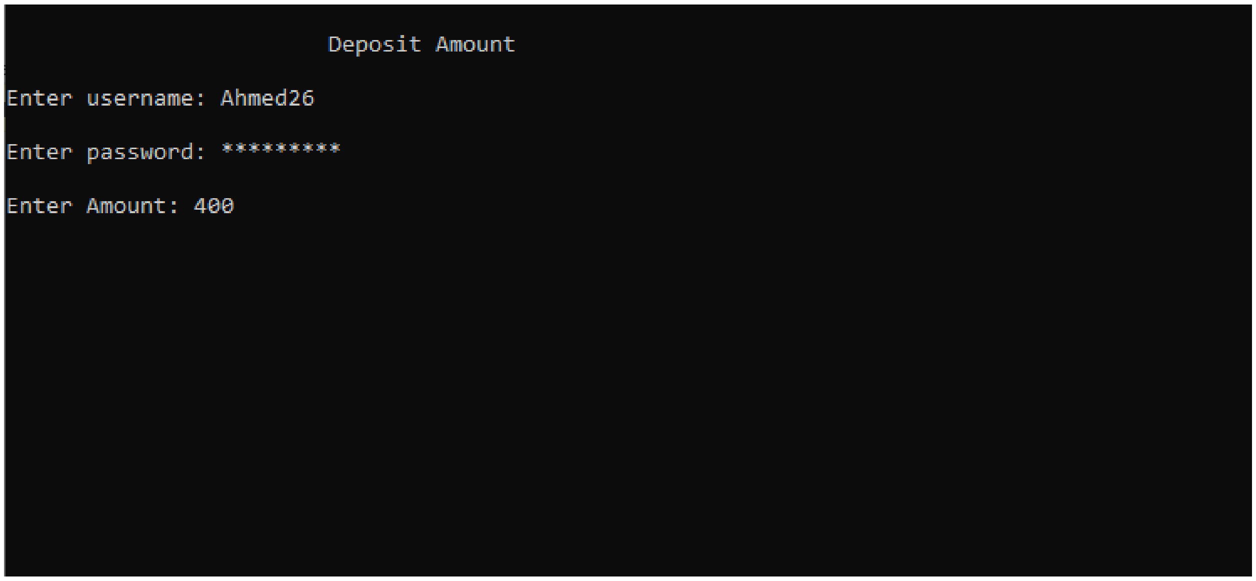
```
                          Update Account

Enter usernameAhmed23

Enter password: *********
```



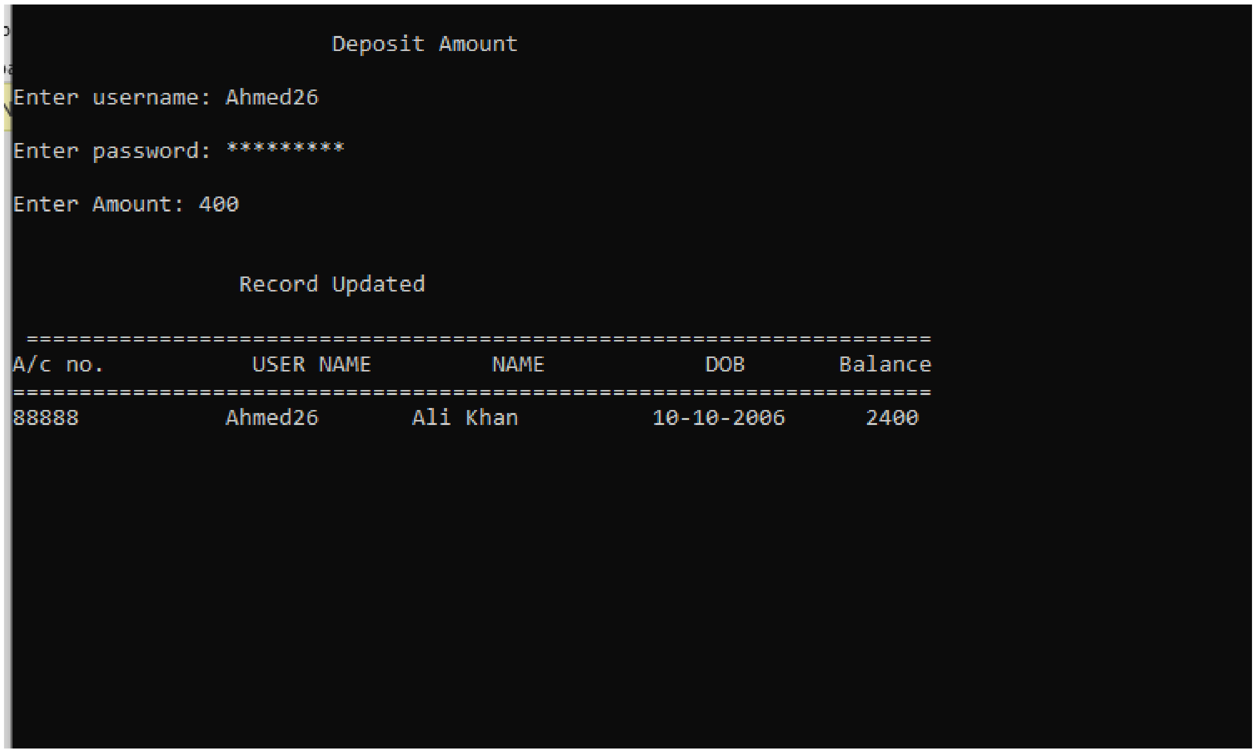
```
                                Your record

88888           Ahmed26       Ali Ahmed          10-10-2006      2000

                Enter The New Details of account

                Update First Name??    1-Yes   2-No
Choice: 2

                Update Last Name??    1-Yes   2-No
Choice: 1

Enter last name: Khan

                Update Password??    1-Yes   2-No
Choice: 2

                Update DOB??    1-Yes   2-No
Choice: 2
```



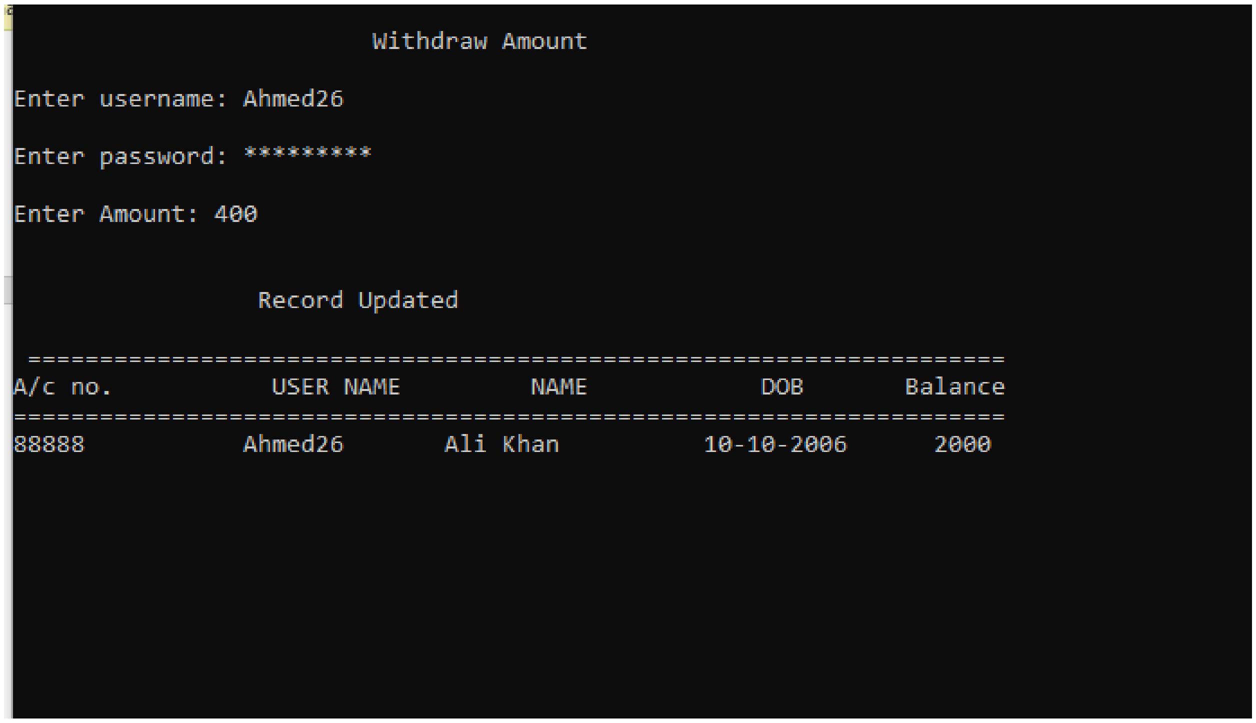
```
                 Record Updated

88888           Ahmed26       Ali Khan          10-10-2006      2000
```



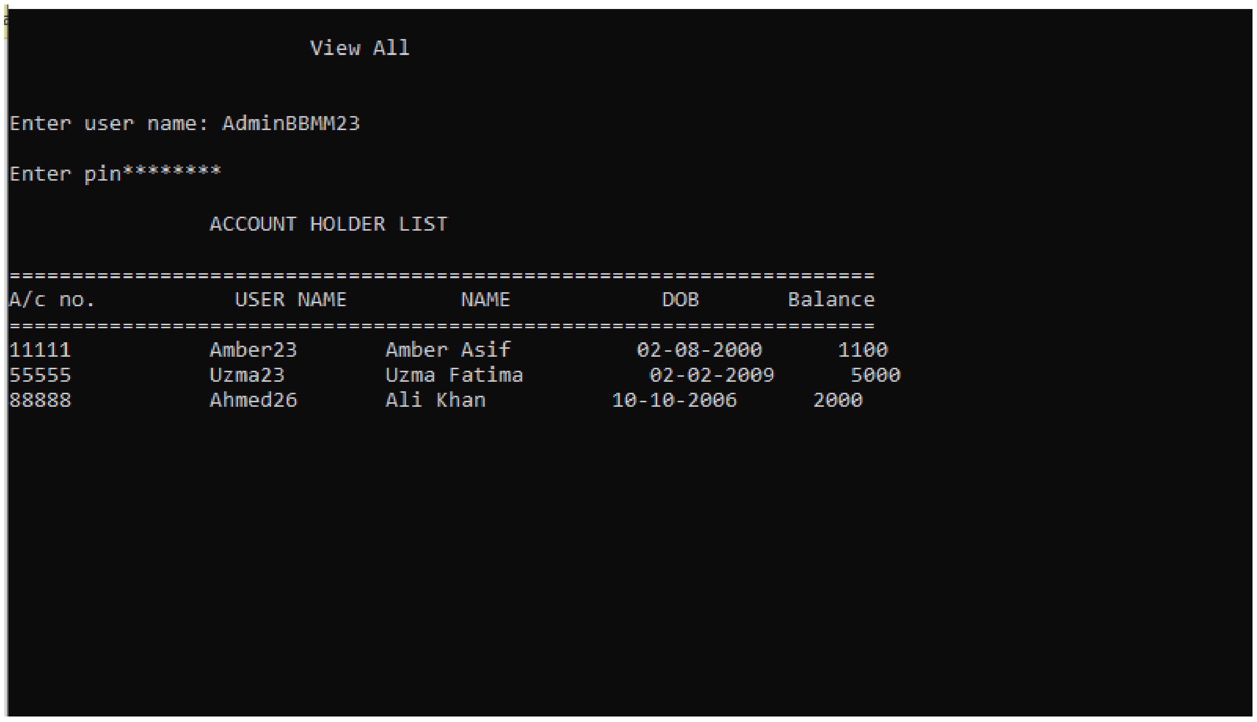
```
                         Deposit Amount

Enter username: Ahmed26

Enter password: *********

Enter Amount: 400
```



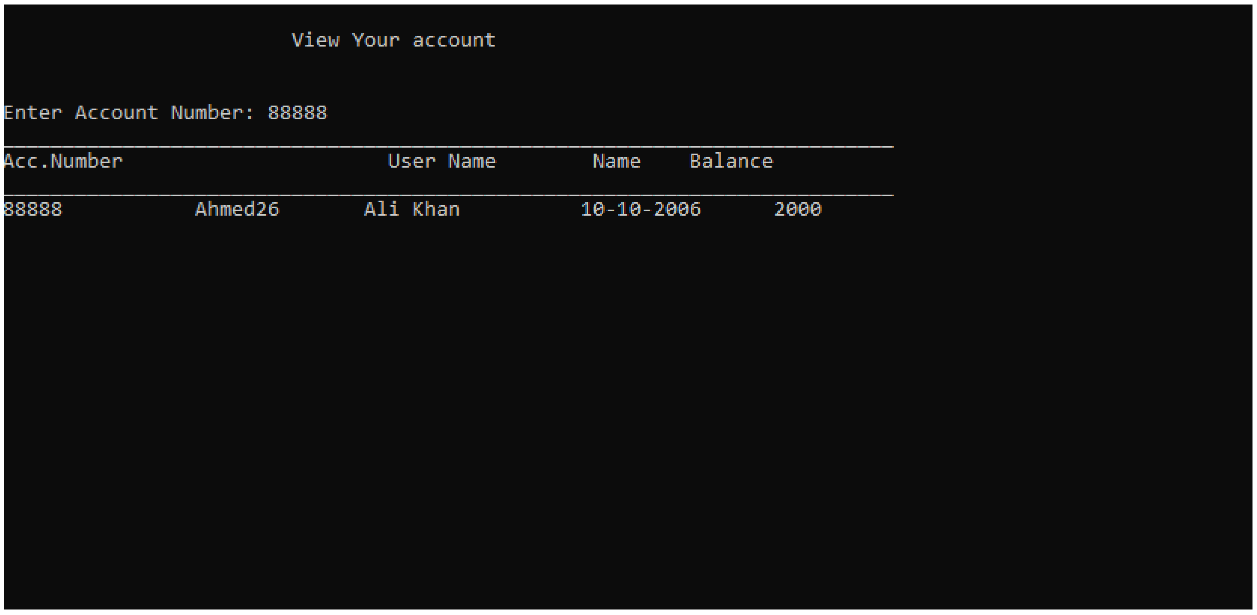
```
                        Deposit Amount

Enter username: Ahmed26

Enter password: *********

Enter Amount: 400


                 Record Updated

 ======================================================================
A/c no.            USER NAME         NAME            DOB       Balance
======================================================================
88888           Ahmed26       Ali Khan          10-10-2006      2400
```



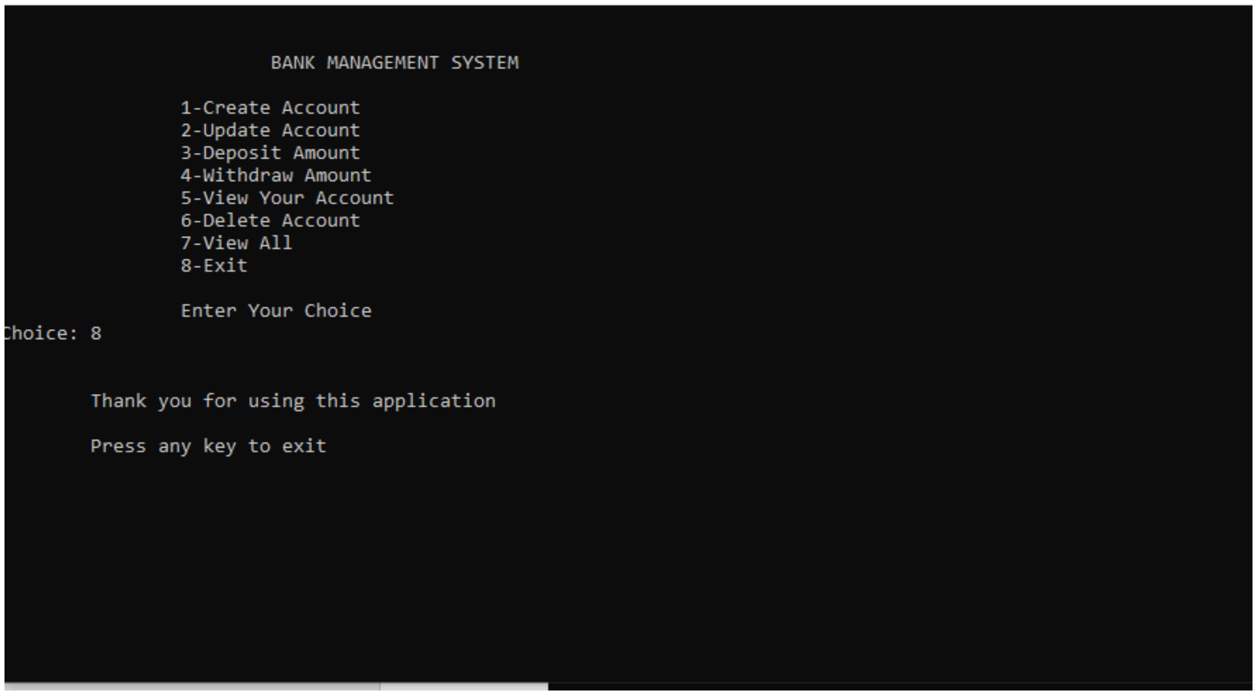
```
                          Withdraw Amount

Enter username: Ahmed26

Enter password: *********

Enter Amount: 400


                 Record Updated

 ======================================================================
A/c no.            USER NAME         NAME            DOB       Balance
======================================================================
88888           Ahmed26       Ali Khan          10-10-2006      2000
```



```
                        View All


Enter user name: AdminBBMM23

Enter pin********

                ACCOUNT HOLDER LIST

=====================================================================
A/c no.           USER NAME         NAME            DOB       Balance
=====================================================================
11111           Amber23       Amber Asif          02-08-2000      1100
55555           Uzma23        Uzma Fatima          02-02-2009      5000
88888           Ahmed26       Ali Khan          10-10-2006      2000
```

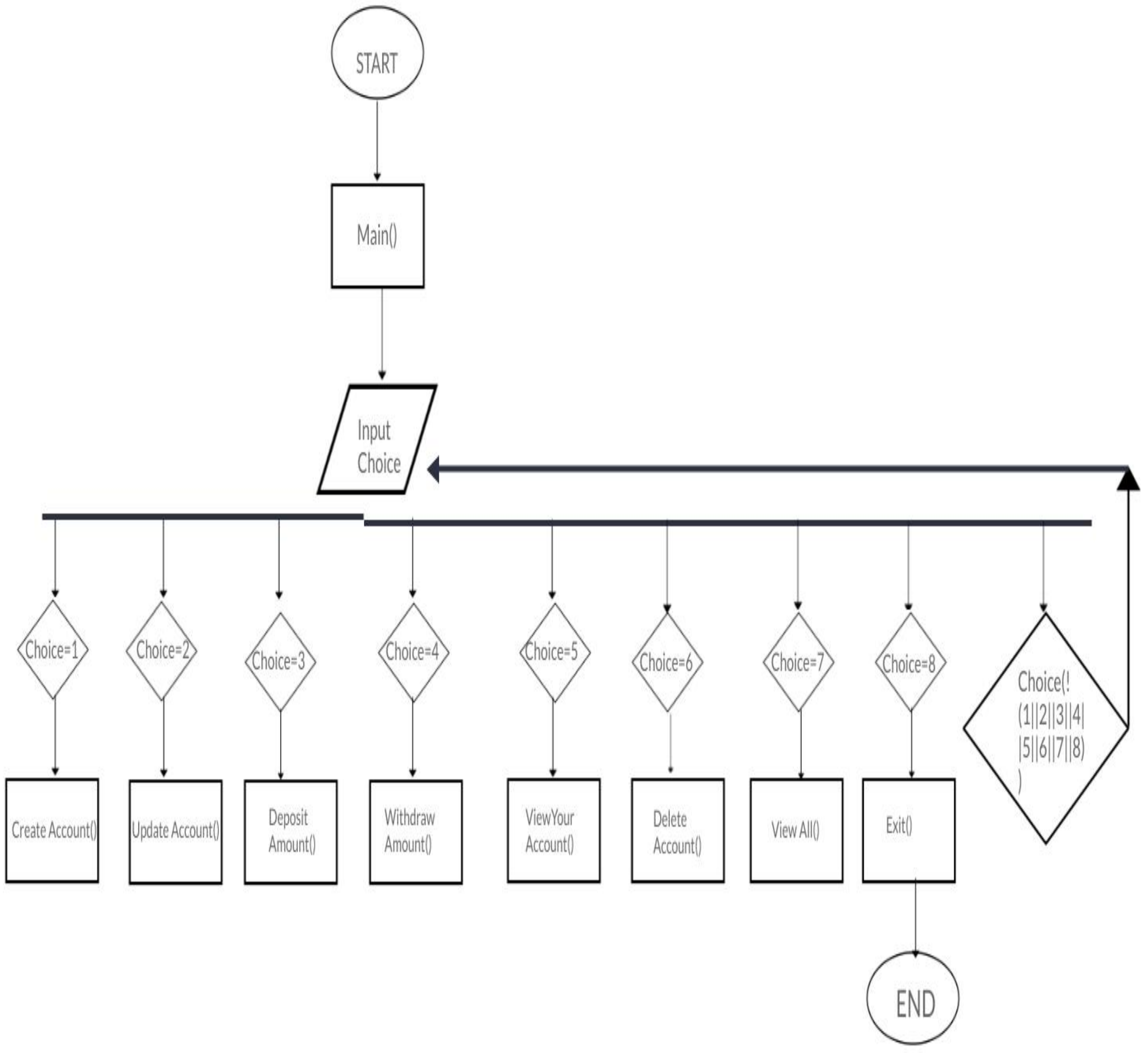


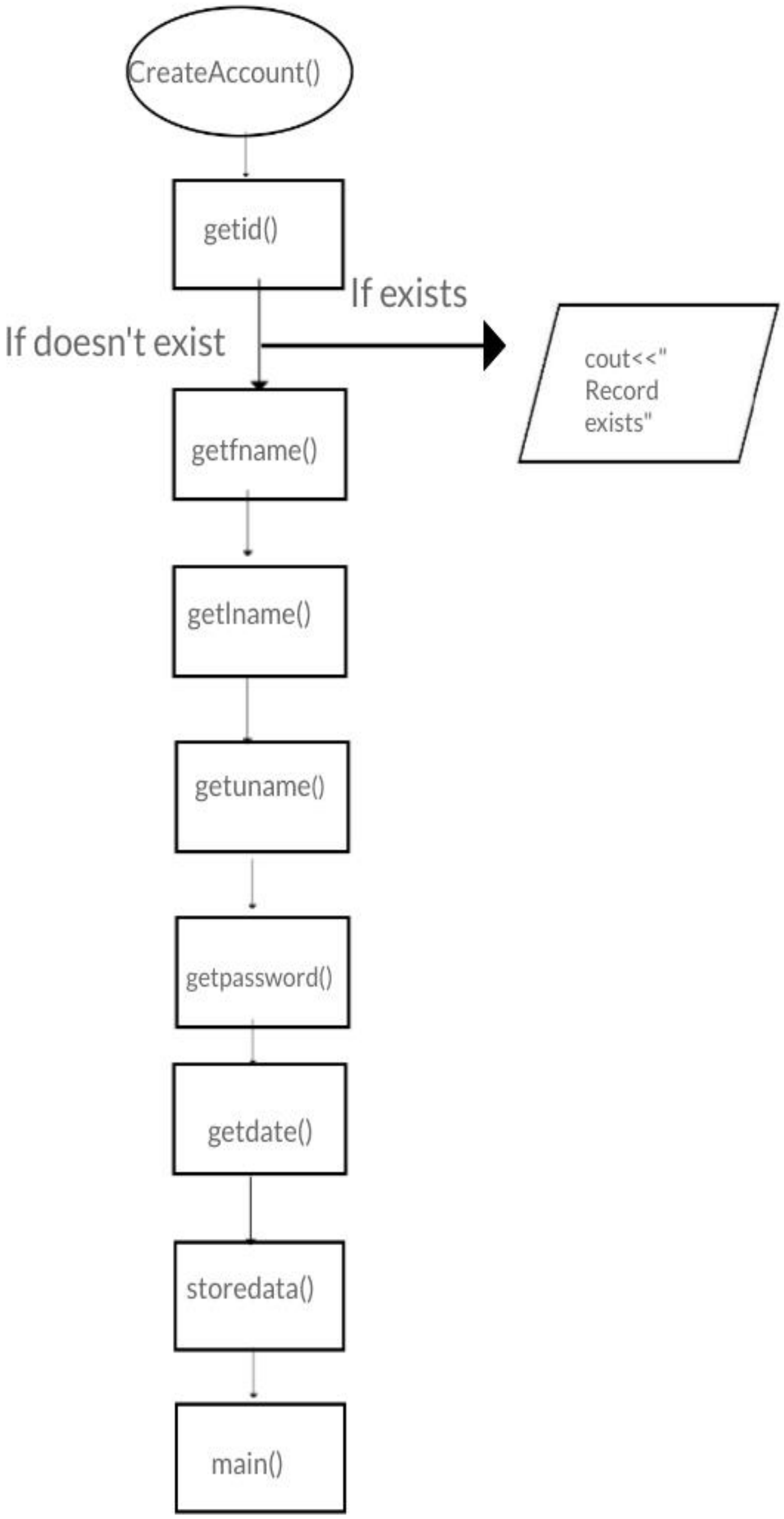
```
                        View Your account


Enter Account Number: 88888
__________________________________________________________________________
Acc.Number                      User Name        Name    Balance
__________________________________________________________________________
88888          Ahmed26       Ali Khan          10-10-2006      2000
```

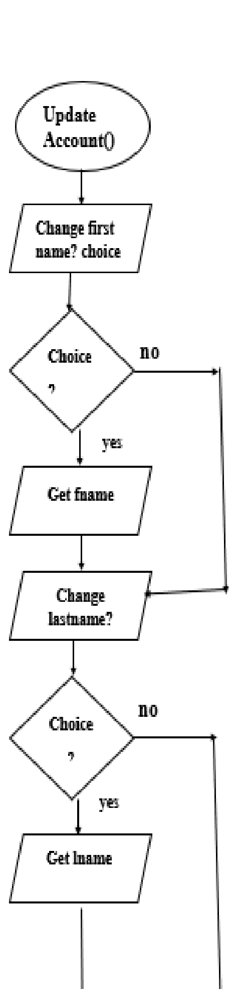


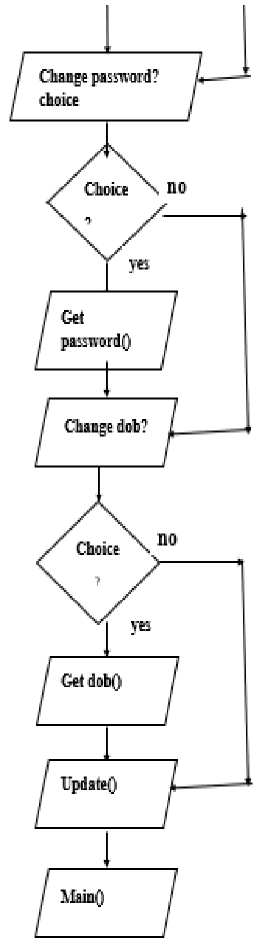
```
                        BANK MANAGEMENT SYSTEM

                1-Create Account
                2-Update Account
                3-Deposit Amount
                4-Withdraw Amount
                5-View Your Account
                6-Delete Account
                7-View All
                8-Exit

                Enter Your Choice
hoice: 8


        Thank you for using this application

        Press any key to exit
```

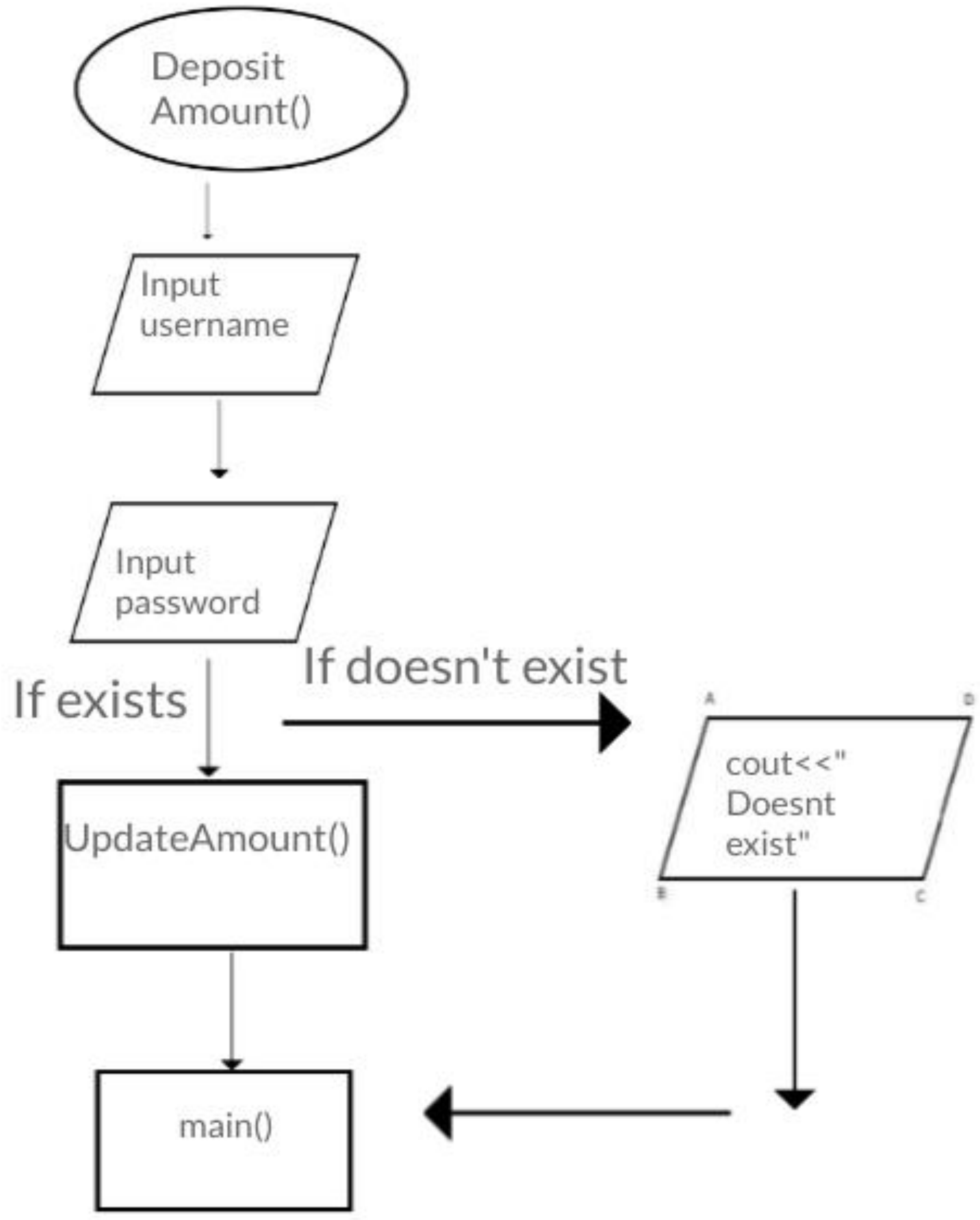
- **FLOWCHART**

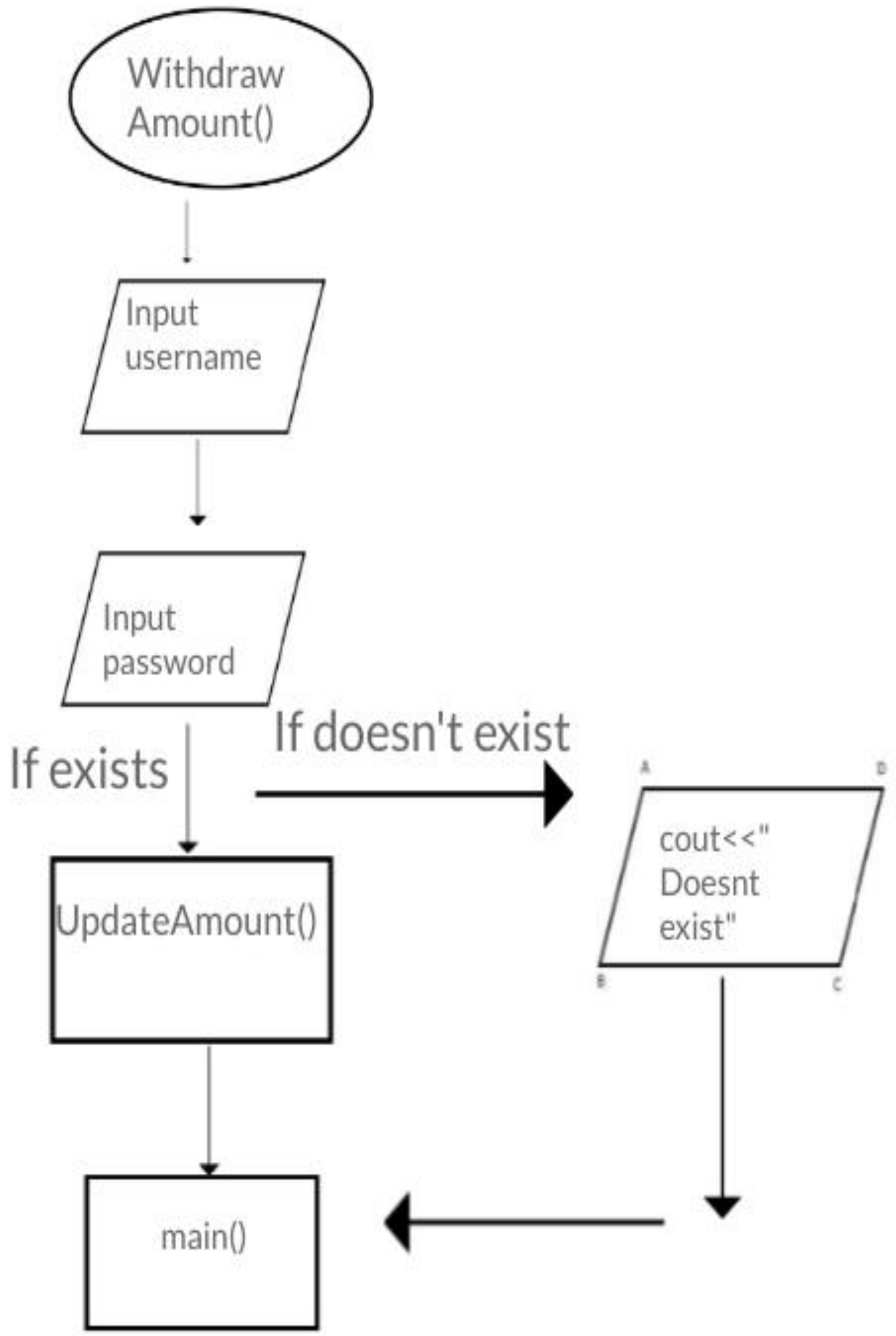

START
Main()
Input Choice
Choice=1
Choice=2
Choice=3
Choice=4
Choice=5
Choice=6
Choice=7
Choice=8
Choice(!(1||2||3||4||5||6||7||8))
Create Account()
Update Account()
Deposit Amount()
Withdraw Amount()
ViewYour Account()
Delete Account()
View All()
Exit()
END

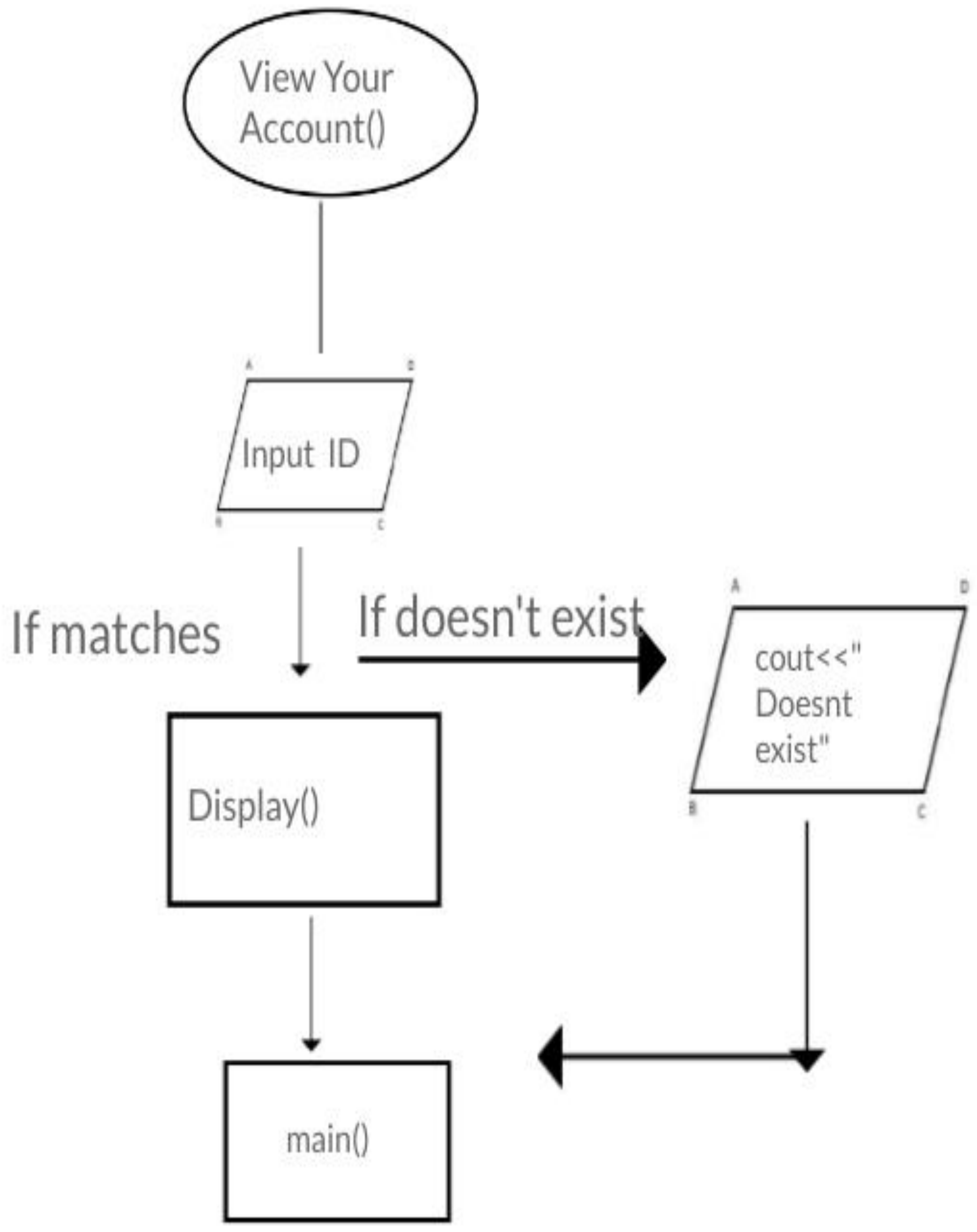

CreateAccount()
getid()
If exists
If doesn't exist
cout<<" Record exists"
getfname()
getlname()
getuname()
getpassword()
getdate()
storedata()
main()

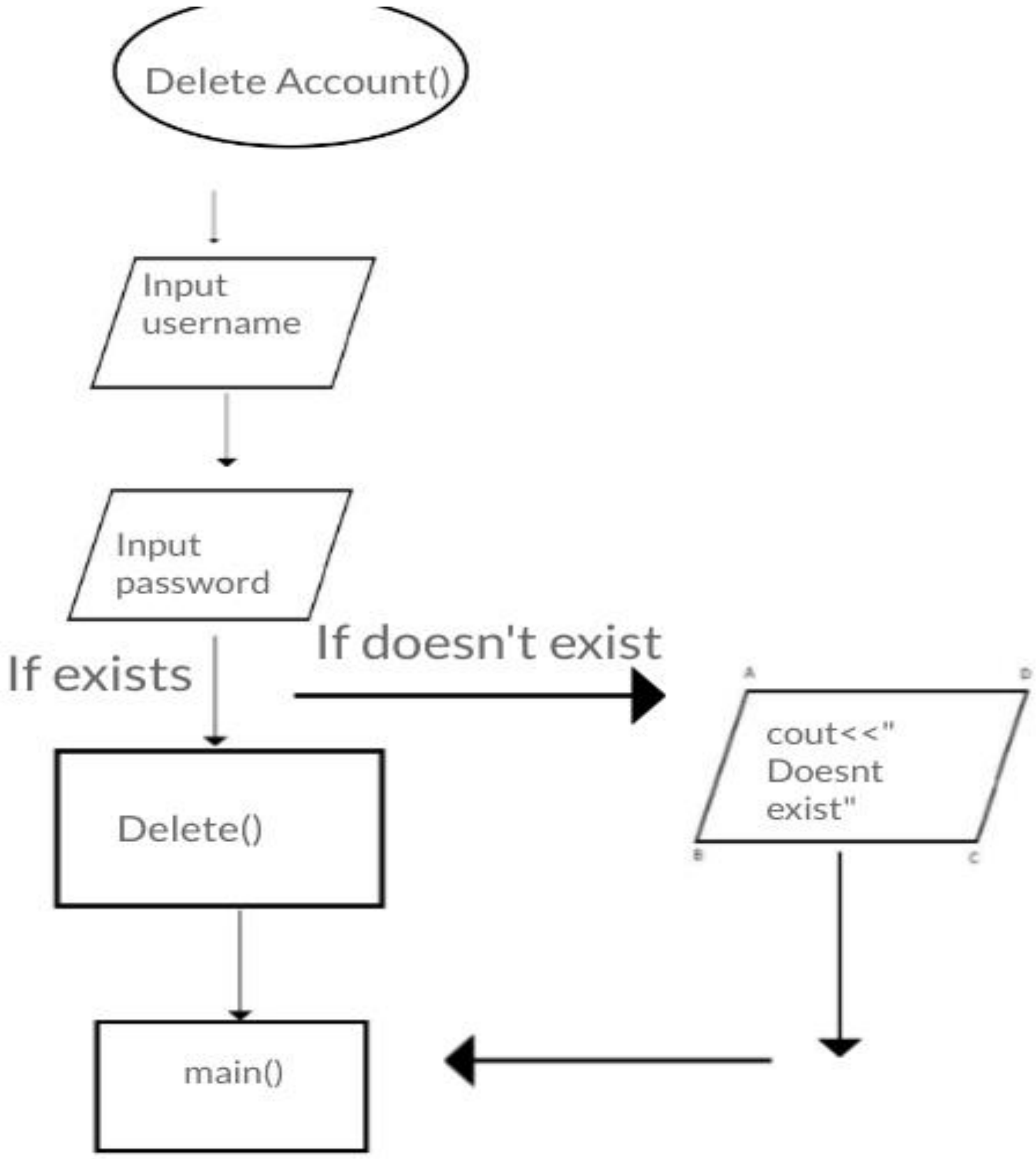

Update Account()
Change first name? choice
Choice ?
no
yes
Get fname
Change lastname?
Choice ?
no
yes
Get lname

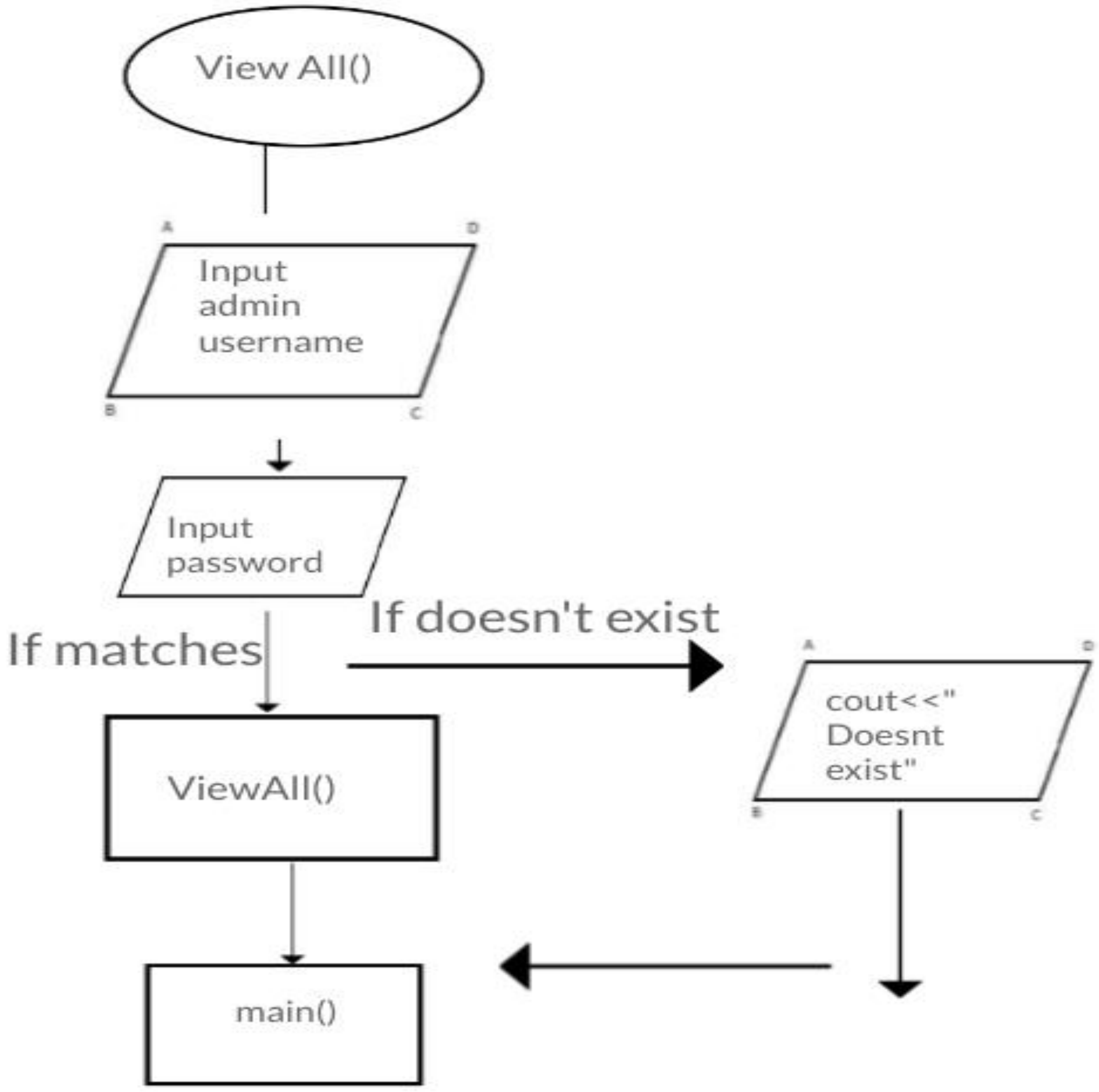

Change password? choice
Choice ?
no
yes
Get password()
Change dob?
Choice ?
no
yes
Get dob()
Update()
Main()


Deposit Amount()
Input username
Input password
If exists
If doesn't exist
A
D
cout<<" Doesnt exist"
B
C
UpdateAmount()
main()


Withdraw Amount()
Input username
Input password
If exists
If doesn't exist
A
D
cout<<" Doesnt exist"
B
C
UpdateAmount()
main()


View Your Account()
Input ID
If matches
If doesn't exist
cout<<" Doesnt exist"
Display()
main()


Delete Account()
Input username
Input password
If exists
If doesn't exist
Delete()
cout<<" Doesnt exist"
main()


View All()
Input admin username
Input password
If matches
If doesn't exist
ViewAll()
cout<<" Doesnt exist"
main()