GOVERNMENT AUTONOMOUS COLLEGE, ROURKELA

(SUNDARGARH, RAGHUNATHPALI, ROURKELA, ODISHA)

A PROJECT REPORT ON

BANK MANAGEMENT SYSTEM

Submitted for practical fulfillment of award of
BACHELOR IN SCIENCE
IN
Computer Science (Honours)
TO

Sambalpur University, Jyotibihar, Burla

Submitted By:-

SEEMA EKKA (19CSC016)

EKTA BHENGRA (19CSC024)

LIPIKA MAHAPATRA (19CSC050)

SANKARSHAN PRADHAN (19CSC047)

ACKNOWLEDGEMENT

We would express our sincere gratitude to the principal of the Government Autonomous College, Rourkela. **Mr. Bijaya Kumar Behera** for giving us an opportunity to be a part of this college. It would be never possible without his support and encouragement. We are grateful to you sir. Thank you, sir. We thankful to**Dr. Abeg Jaiswal**, Head of the Self-Financing department, our project guide, who always encourage us not only for studies but also for extra activities. We thankful to **Mr. Jayant Kumar Samantary**, our project guide for helping us in every step of our project, and also thankful for their splendid suggestion that they gave us to improve our project. Their friendly and humorous nature help us to feel easy in new environment. And lastly, we thank everyone that has caused us to suffer, without you we would have no reason to express ourselves.

Thanks a lot, to all who helped us to make this project possible.

SEEMA EKKA

EKTA BHENGRA

LIPIKA MAHAPATRA

SANKARSHAN PRADHAN

CERTIFICATE

This is to be certify that project work entitled **"BANK MANAGEMENT SYSTEM"** has been carried by:-

NAME: SANKARSHAN PRADHAN Roll No. 19CSC047

Under the guidance for the partial fulfillment of Bachelor in Science. The Embodies result of original work and studies carried out by student himself and the contents of the thesis do not form the basis for the award of any other degree to the candidate or to anybody else.

Guided By : **Mr. Jayanta Kumar Samantary**

Place : Rourkela DATE :

EXTERNAL:

CONTENTS

ABSTRACT

The adoption of Electronic Banking by commercial enterprises has been in existence since the mid 90s, much greater in number due to lower operating costs associated with it. Electronic banking has initially been in the form of automatic teller machines and telephone transactions. More recently, it has been transfer made by the Internet, a new delivery channel for banking services that benefits both customers and banks. Internet banking system services can include: Open an account, Balance enquiry, Request for Cheque book, Beneficiary payments (EFT), Viewing monthly. Furthermore, customer's application for electronic banking facilities is expanding as the cost savings on transactions over the Internet are significant.

OBJECTIVE

- Coordinate elements of banking activity with a view to generating profits
- Ensures an optimal and organic system of interaction those elements.
- Keep money safe for customers.
- Offer customers interest on deposits, helping to protect against money losing value against inflation.
- Lending money to firms, customers and homebuyers.
- Offering financial advice and related financial services, such as insurance.

INTRODUCTION

General Overview of the Problem:

A bank is a commercial or state institution that provides financial services , including issuing money in form of coins, banknotes or debit cards, receiving deposits of money, lending money and processing transactions. A commercial bank accepts deposits from customers and in turn makes loans based on those deposits. Some banks (called Banks of issue) issue banknotes as legal tender. Many banks offer ancillary financial services to make additional profit; for example: selling insurance products, investment products or stock broking .Currently in most jurisdictions commercial banks are regulated and require permission to operate. Operational authority is granted by bank regulatory authorities and provides right to conduct the most fundamental banking services such as accepting deposits and making loans. A commercial bank is usually defined as an institution that provides selected banking services without meeting the legal definition of bank .Banks have a long history, and have influenced economy and politics for centuries. In history, the primary purpose of a bank was to provide liquidity to trading companies. Banks advanced funds to allow business to purchase inventory, and collected those funds back with interest when the goods are sold. For centuries, the banking industry only dealt with business not customers. Commercial lending today is a very intense activity, with banks carefully analyzing the financial condition of its business clients to determine the level of risk in each loan transaction. Banking services have expanded to include services directed at individuals and risks in these much smaller transactions are pooled.

In today's world, the way of functioning and managing the system has been totally changed. There is a sudden and adrupt changes in the structure, maintenance and modification, handling, leveling inside every system. Without *managing system through computer applications and programming, the development of infrastructures are unfinished. There are many errors and drawbacks without use of computer programming and applications.*

As we know that, "necessity is the mother of invention", so in today's challenging world, every system is developed and launched by the use of computer software and programming.

Aim for this application:

The project that we have undertaken aims to develop a banking system that is clean, user-friendly and multi-functional. Development of this application includes a number of fields such that user feels comfortable and the system appears as dynamic to him. The project "Banking System" includes the following functionalities:

- *Transactions can be done with minimum user events.*
- *All transactional details and accounts are stored in files on stable storage.*
- *Customers can view their own account details and can use them as necessary*
- *Customer can inquire an account and can inquire about interest*
- *All customer's data are stored in files on a stable storage*
- *Account holders have to pass through a login system to enter their accounts*
- *This system possess password-protected administrative access; thus preventing the whole management system from unauthorized access*

- *To provide flexibility for secure and save transaction.*
- *For better performance.*
- *Reducing man power.*
- *For doing work more accurately.*
- *Faster performance.*

Proposed System And Objectives:

The application will be extremely beneficial for the Customers intending to use and operate their bank account and will get various benefits in the field of management of accounts on a clean and user-friendly platform.

"Bank Account System", is a simple application, which is especially generated and designed for the bank in order to enter the applicant information about his or her bank account and can perform other function like currency change. It is user name and ID protected as well.

Following are the major objectives behind the new proposed system:

- *It creates a user friendly environment, where a normal user can access through all the benefits of the system.*

- *It provides security from unauthorized access, only admin or authorized users are access granted to the system.*
- *It increases efficiency and saves the time.*
- *No any danger and obstacles from external entities.*
- *Easy access of saved data inside the system.*
- *Complex Banking operations and Transaction operations are efficiently handled by the application*
- *It is cost effective*
- *It has ease of use along with complete reference*
- *It is highly secured and less time consuming; hence time wastage can be avoided*
- *Up to date records of the customers are maintained by the authority.*

Problem description:

The Bank Management System is an application for maintaining a person's account in a bank. The system provides the access to the costumer to create an account, deposit/withdraw the cash from his account, also to convert currency. The following documentation provides the specification of the system.

We are mainly concerned with developing a banking system where a Customer can submit his/her deposit amount to bank if he/she has an account or can create a new account in this bank. Customer can also view the status and change currency of his/her bank account, can view account balance. One can easily maintain the above things if he/she has an account by login through his unique account number.

SCOPE

- ***This project is developed to make the work of bank employee much easier in the process of maintaining the records of the customer.***
- ***This helps the customer to provide easy deposit /withdrawal and make the process user friendly to the employer.***
- ***In this way,it can provide bst security ever.the employer can execute the process of deposit /withdraw of the amount with maximum security.***
- ***This system also provides accuracy.***
- ***This also reduces the time taken for the customer to complete his transaction.***

HARDWARE AND SOFTWARE SPECIFICATIONS

Configuration for hardware and software

It doesn’t need any additional hardware or software to operate the program, but the following requirements should be strongly maintained.

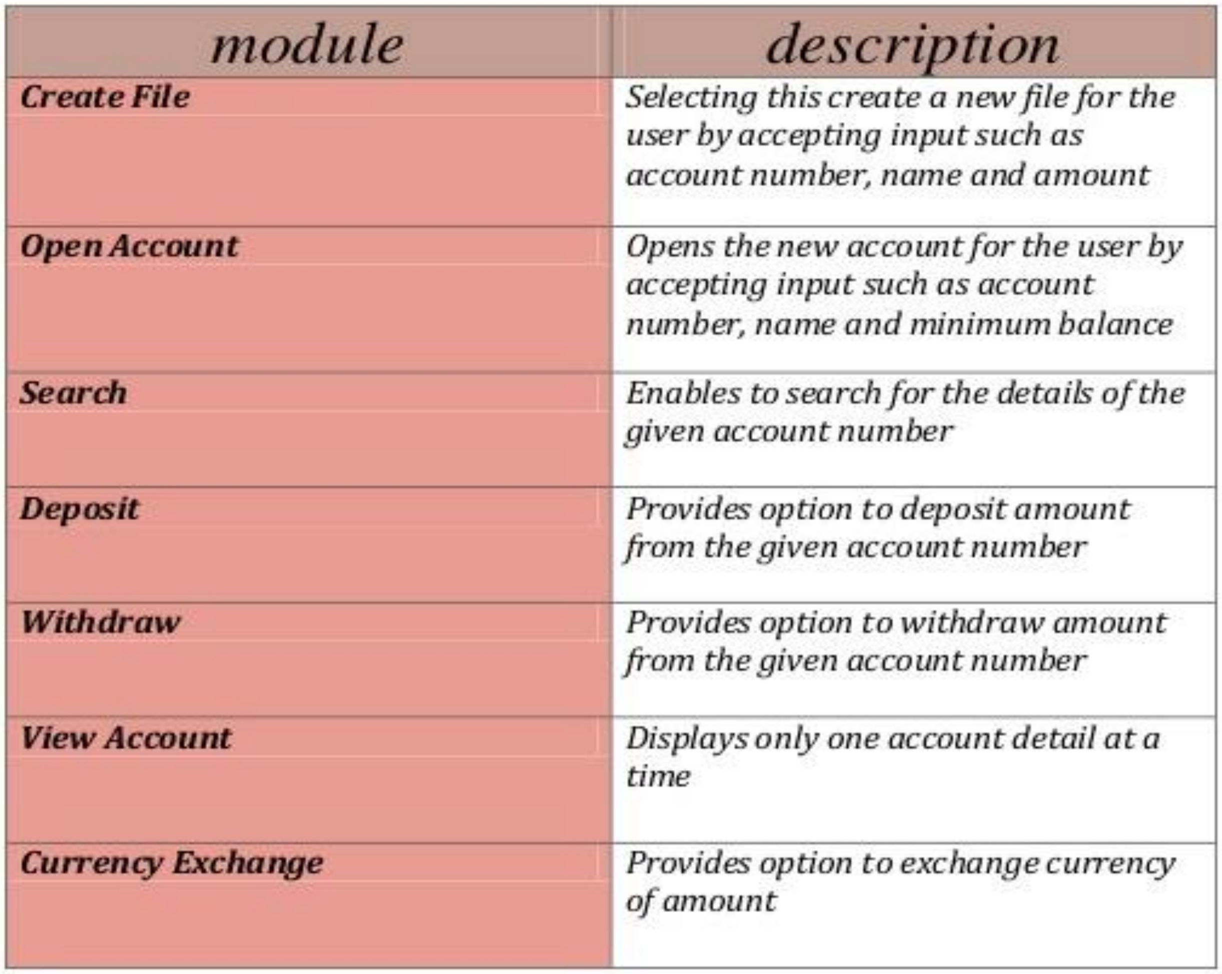
Requirements for hardware:

- 512 MB of RAM or higher
- 800 hz processor or above
- CD ROM
- 20 MB of hard-disk space

Requirements for software:

- Operating system WINDOWS 10
- PROGRAM TUBROC3 needs to be installed.
- The content of BGI files in the folder TC needs to be copied in the BIN folder for functioning of graphical attributes.

Module description: -



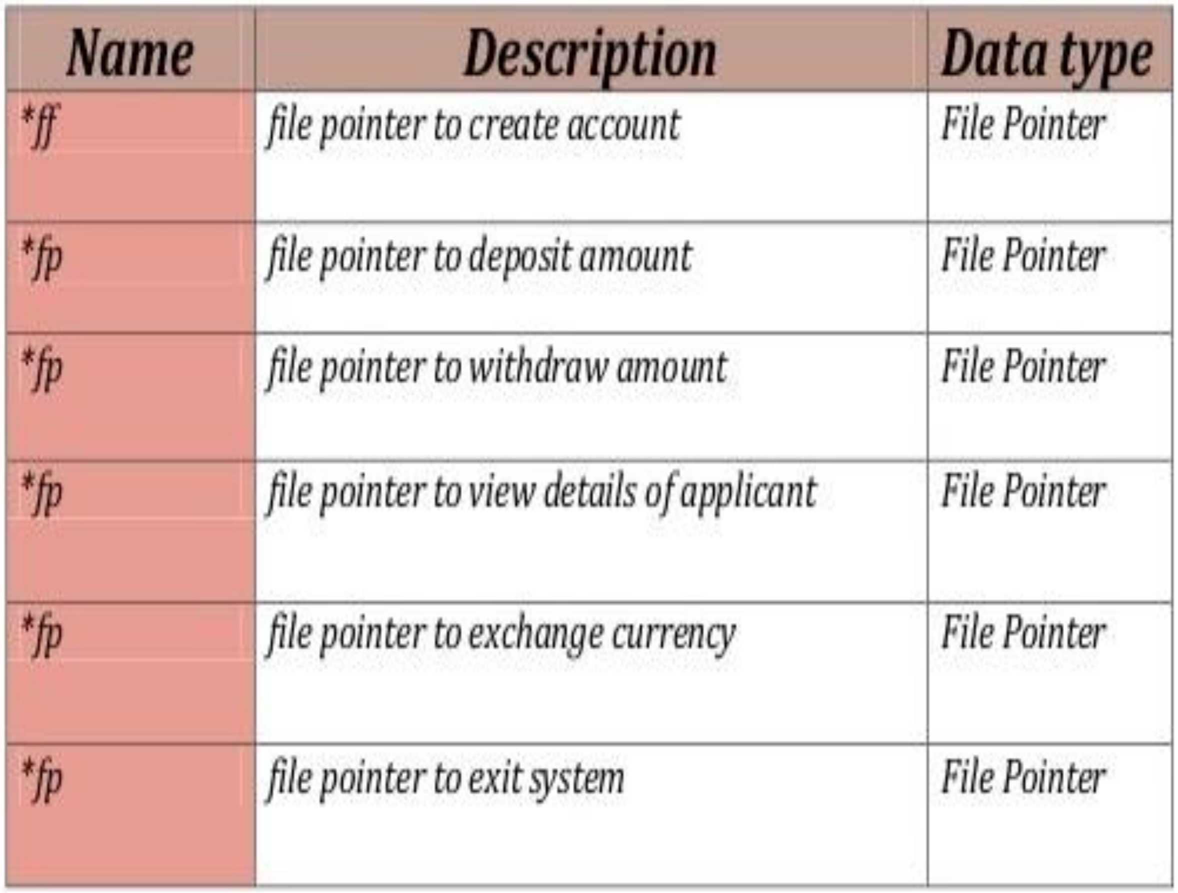
module	description
Create File	Selecting this create a new file for the user by accepting input such as account number, name and amount
Open Account	Opens the new account for the user by accepting input such as account number, name and minimum balance
Search	Enables to search for the details of the given account number
Deposit	Provides option to deposit amount from the given account number
Withdraw	Provides option to withdraw amount from the given account number
View Account	Displays only one account detail at a time
Currency Exchange	Provides option to exchange currency of amount

Input specification: -

Struct Information

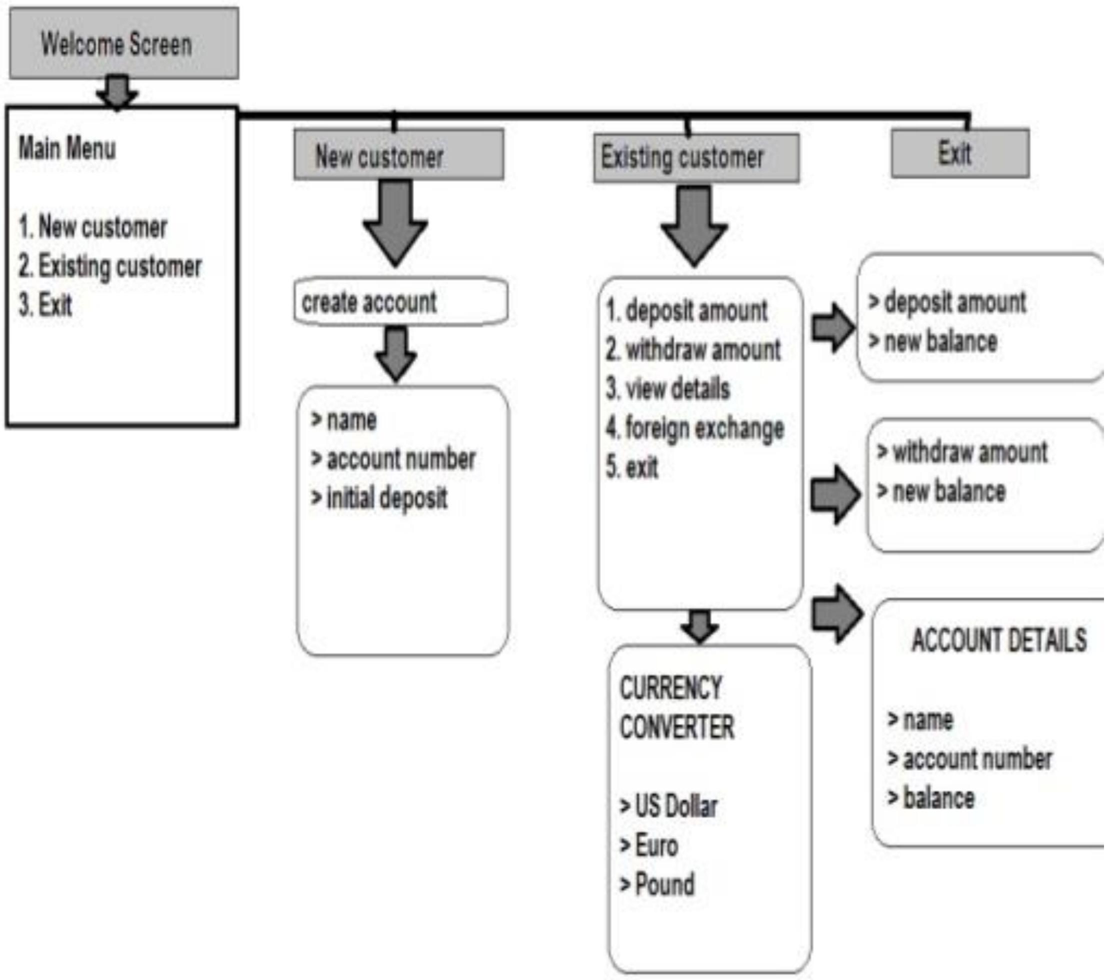
S.no	***Variable Name***	***Description***	***Data Type***	***Size***
1.	*acc_no*	*Applicant account no.*	*int*	*2*
2.	*name*	*Applicant name*	*char*	*20*
3.	*bal*	*Applicant balance*	*float*	*4*

File specification: -



Name	Description	Data type
*ff	file pointer to create account	File Pointer
*fp	file pointer to deposit amount	File Pointer
*fp	file pointer to withdraw amount	File Pointer
*fp	file pointer to view details of applicant	File Pointer
*fp	file pointer to exchange currency	File Pointer
*fp	file pointer to exit system	File Pointer

Output screen design



Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.
It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in a maize of instant. Thus the objective of input design is to create an input layout that is easy to follow.

A quality output is one, which meets the requirements of the end user and presents the information clearly. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively.

Database design:

A database is an organized mechanism that has the capability of storing information through which a user can retrieve stored information in an effective and efficient manner. The data is the purpose of any database and must be protected. The database design is a two level process. In the first step, user requirements are gathered together and a database is designed which will meet these requirements as clearly as possible. This step is called Information Level Design and it is taken independent of any individual DBMS. In the second step, this Information level design is transferred into a design for the specific DBMS that will be used to implement the system in question. This step is called Physical Level Design, concerned with the characteristics of the specific DBMS that will be used. A database design runs parallel with the system design. The organization of the data in the database is aimed to achieve the following two major objectives.

ϖ Data Integrity

ϖ Data independence

Normalization is the process of decomposing the attributes in an application, which results in a set of tables with very simple structure. The purpose of normalization is to make tables as simple as possible. Normalization is carried out in this system for the following reasons.

- To structure the data so that there is no repetition of data , this helps in saving.
- To permit simple retrieval of data in response to query and report request.
- To simplify the maintenance of the data through updates, insertions, deletions.
- To reduce the need to restructure or reorganize data which new application requirements arise.

Processing and validation: -

Main menu:

There are number of options regarding users to creating a new account or logging in as existing customer to access the facilities system providing or exit the system after accessing the user requirements.

Creating new account:

All the requirements should be filled with proper with correct identification such as account number, name and initial deposit.

Logging in as existing customer:

Correct account number should be required to access through existing customer otherwise this facility is not executable. There are also number of options regarding users to avail the facilities system is giving to the user.

Depositing and Withdrawing account:

This required a amount customer want to deposit or withdraw and show the current balance after executing customer's requirement.

Viewing details:

Correct account number should be required in order to view details of customer's account .If not entered correct account number, then it can't be accessed.

Converting currency:

This required a number of money user want to convert and also required the name of currency user wish to convert in.

Uniqueness:

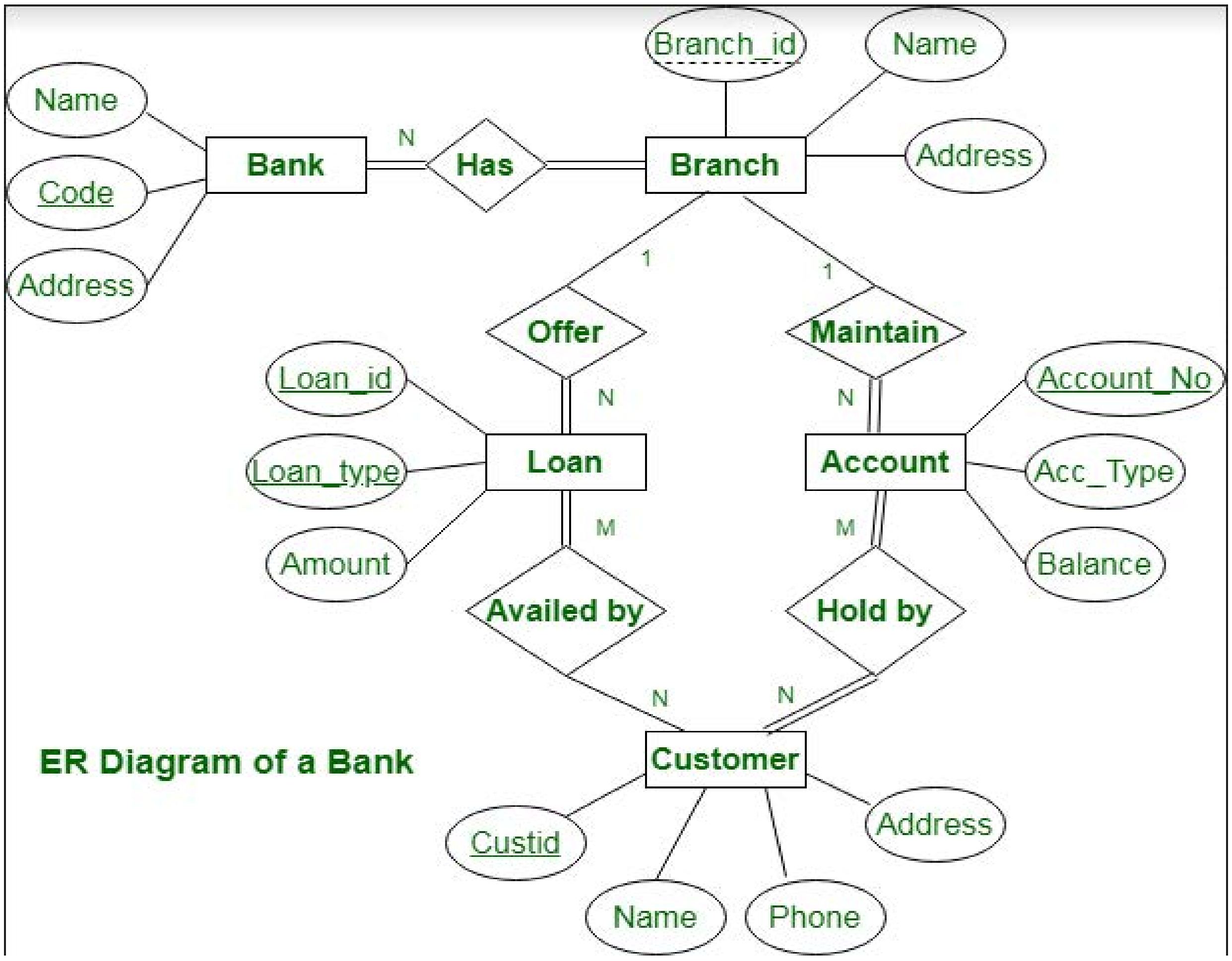
Each account number should be unique throughout all the existing records. Name should be of character type and account number should be of digit. Character and digits are not allowed.

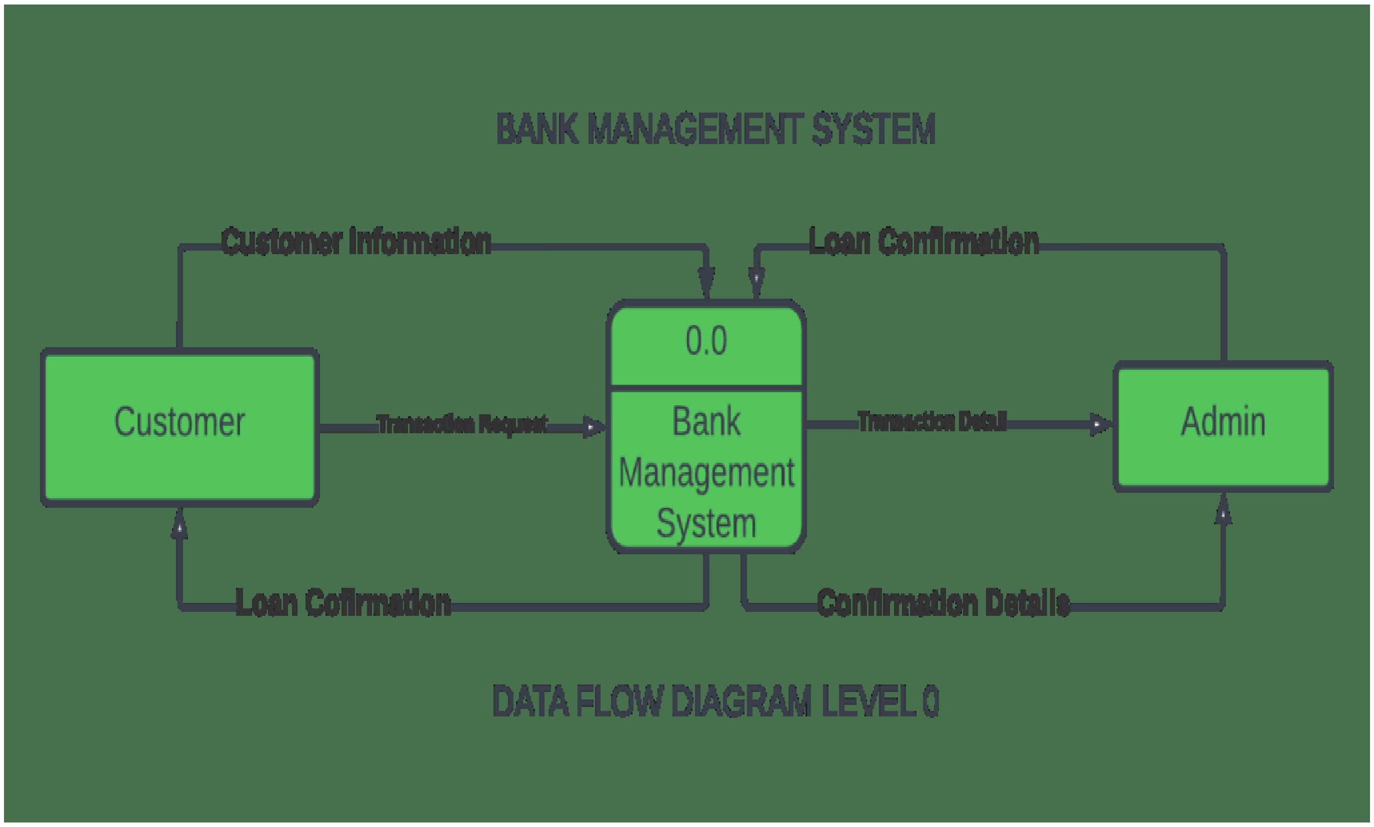
Checking of size:

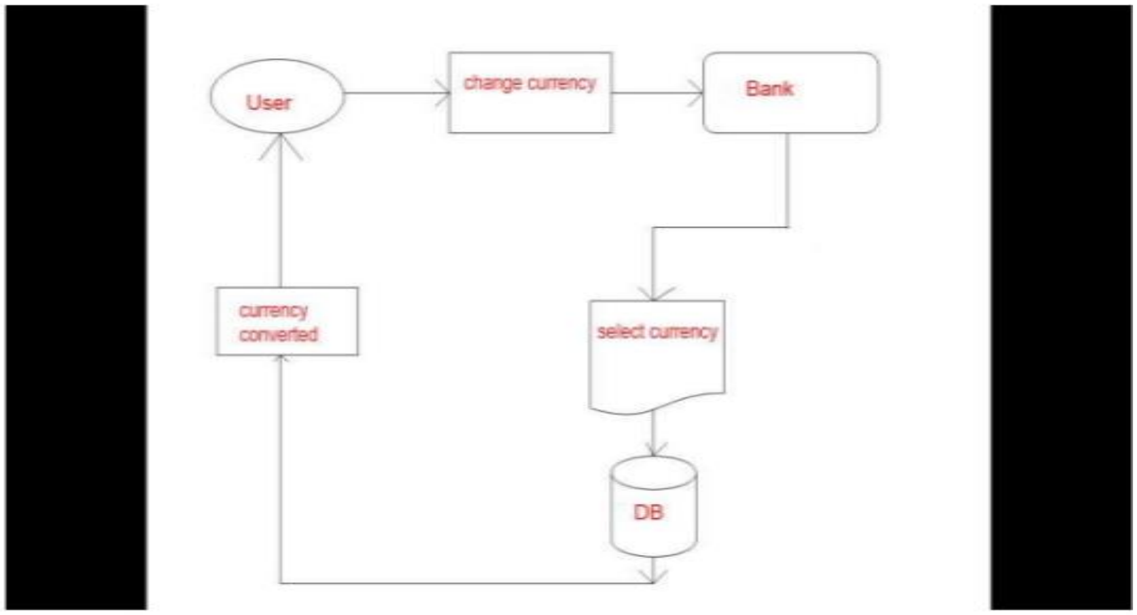
Account number cannot be greater than 2 of size.

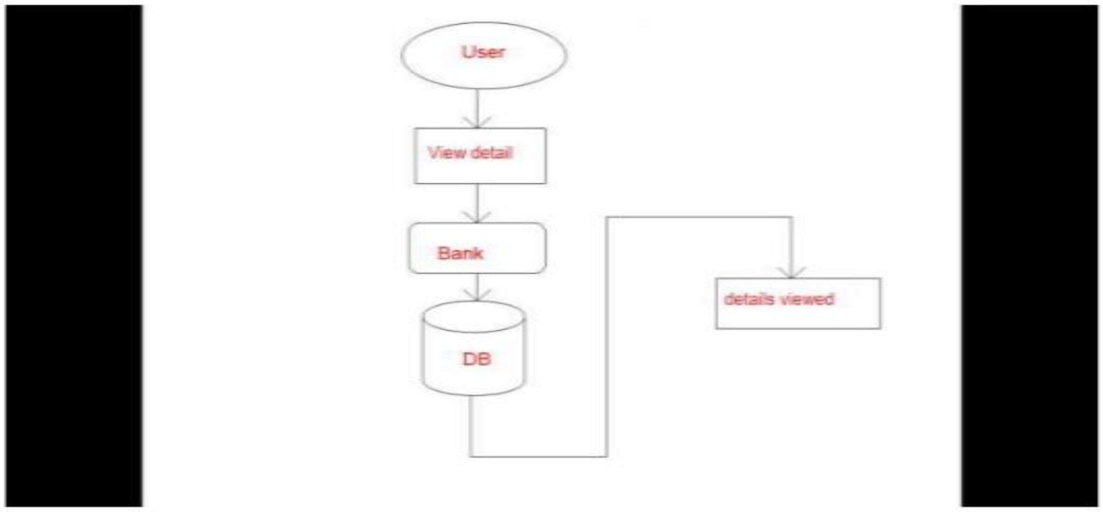
Name can't be greater than 20 characters.

E -R DIAGRAMS


Branch_id
Name
Name
Bank
N
Has
Branch
Address
Code
Address
1
1
Offer
Maintain
Loan_id
N
N
Account_No
Loan
Account
Loan_type
Acc_Type
M
M
Amount
Balance
Availed by
Hold by
N
N
ER Diagram of a Bank
Customer
Address
Custid
Name
Phone

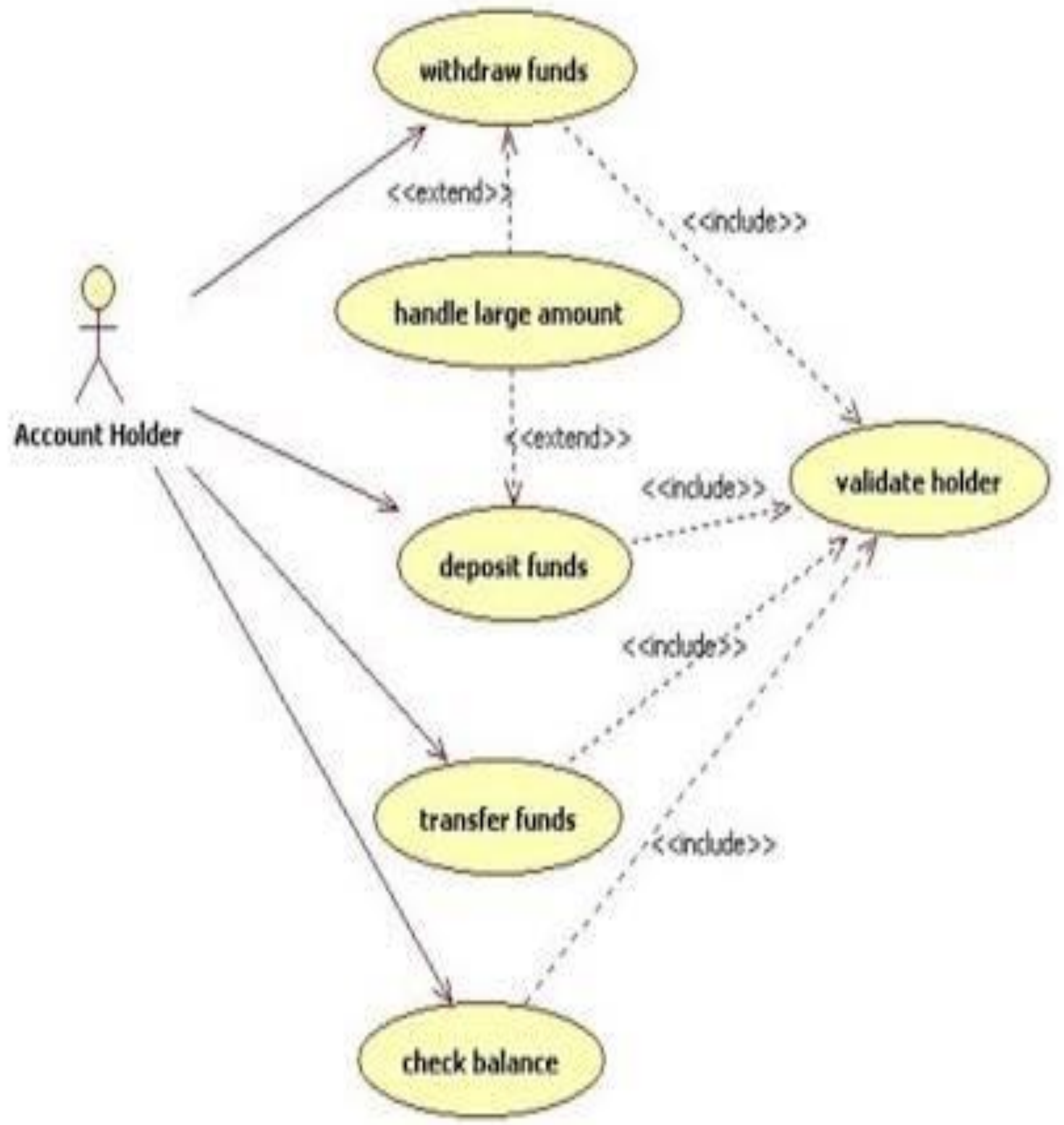

BANK MANAGEMENT SYSTEM
Customer Information
Loan Confirmation
0.0
Bank Management System
Customer
Transaction Request
Transaction Detail
Admin
Loan Cofirmation
Confirmation Details
DATA FLOW DIAGRAM LEVEL 0





E R Diagram for currency converter

UML diagram:


withdraw funds
<<extend>>
<<include>>
handle large amount
Account Holder
<<extend>>
<<include>>
validate holder
deposit funds
<<include>>
transfer funds
<<include>>
check balance

PROGRAMMING TEST: -

TEST PLAN:

S.no.	Test Objective
1.	To check whether program runs or not.
2.	To check if the program menu displays all menu options or not.
3.	To check all options are right or not.
4.	To check account was created or not.
5.	To check if the program add amount after depositing.
6.	To check if the program subtract amount after withdrawing.
7.	To check if 'view detail' option shows the info or not.
8.	To check if currency converter works properly or not.
9.	To check if the exit option works or not.
10.	To check whether menu options return back or not.
11.	To check whether menu is opened after entering correct account number.
12.	To check whether msg of "account no. is not valid" prompt or not when incorrect account no. is entered.
13.	To check if the color text are displayed or not.

TESTS: -

Test Case	*1*
Test Objective	*To check whether program runs or not.*
Test Data	*Running the program.*
Expected Result	*Main screen should display successfully.*
Test Result	*Main screen appears correctly.*
Conclusion	*Expected result matches actual result.*

Test Case	*2*
Test Objective	*To check if the program menu displays all menu options or not.*
Test Data	*Open the main menu*
Expected Result	*All the options should be displayed correctly with serial order*
Test Result	*Menu options are displayed in right order*
Conclusion	*Expected result matches actual result.*

Test Case	3
Test Objective	To check all options are right or not.
Test Data	Open menus
Expected Result	All options are right
Test Result	All options are right and displayed correctly.
Conclusion	Expected result matches actual result.

Test Case	4
Test Objective	To check account was created or not.
Test Data	Enter account number.
Expected Result	Bank account must be created.
Test Result	Account was successfully created
Conclusion	Expected result matches actual result.

Test Case	*5*
Test Objective	*To check if the program add amount after depositing.*
Test Data	*Deposit amount*
Expected Result	*Amount must be added in bank balance*
Test Result	*Bank balance was updated successfully.*
Conclusion	*Expected result matches actual result.*

Test Case	*6*
Test Objective	*To check if the program subtract amount after withdrawing.*
Test Data	*Withdraw amount.*
Expected Result	*Amount must be subtracted from bank balance.*
Test Result	*Bank balance was updated successfully.*
Conclusion	*Expected result matches actual result.*

Test Case	*7*
Test Objective	*To check if 'view detail' option shows the info or not.*
Test Data	*View detail.*
Expected Result	*Details must be show*
Test Result	*Details appear correctly.*
Conclusion	*Expected result matches actual result.*

Test Case	*8*
Test Objective	*To check if currency converter works properly or not.*
Test Data	*Enter currency.*
Expected Result	*Amount must be converted according to desired currency.*
Test Result	*Currency converted successfully.*
Conclusion	*Expected result matches actual result.*

Test Case	*9*
Test Objective	*To check if the exit option works or not.*
Test Data	*Run program.*
Expected Result	*Exit program using option.*
Test Result	*Program exits.*
Conclusion	*Expected result matches actual result.*

Test Case	*10*
Test Objective	*To check whether menu options return back or not.*
Test Data	*Run program.*
Expected Result	*Menu must be returns.*
Test Result	*Menu returns.*
Conclusion	*Expected result matches actual result.*

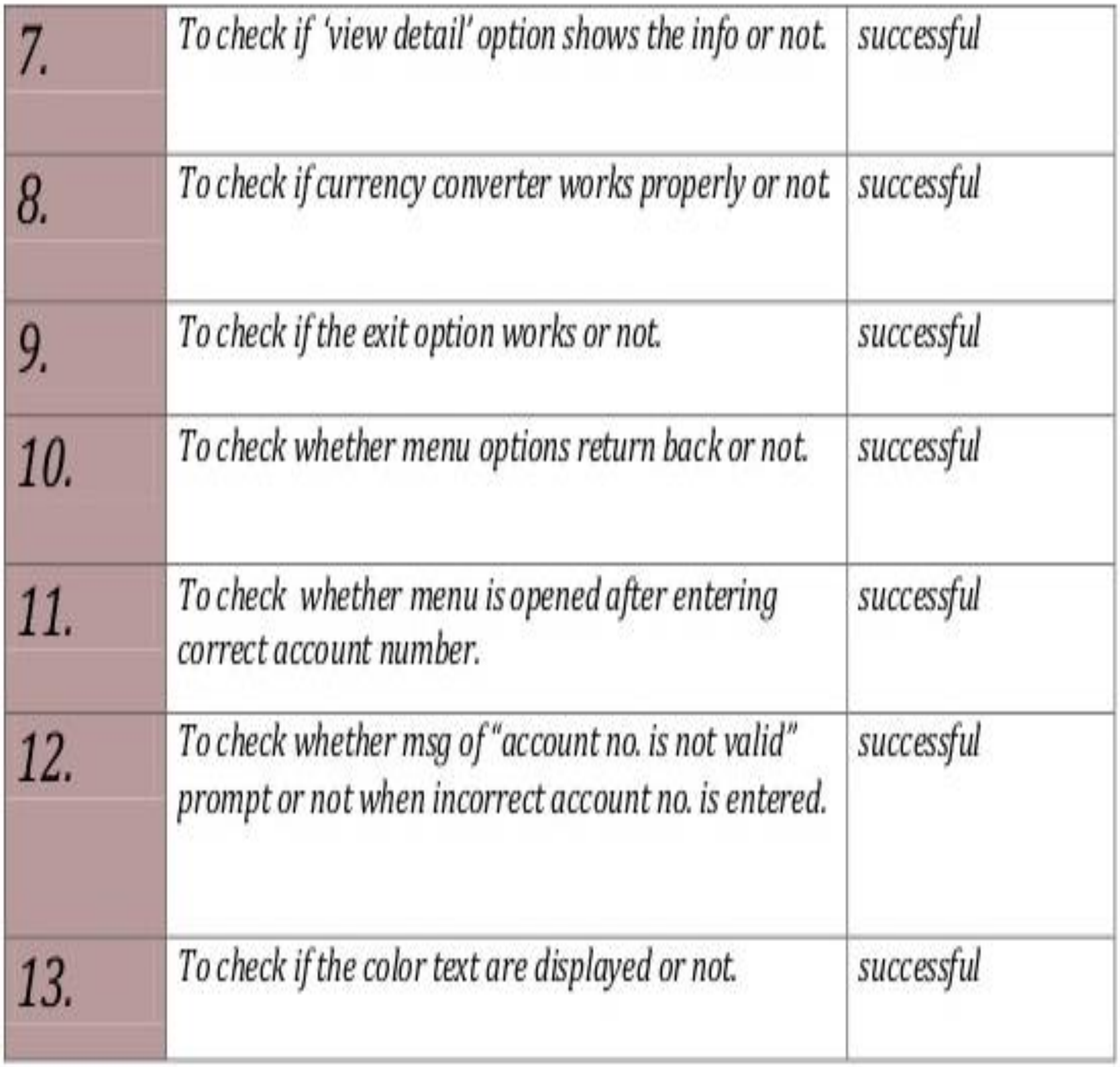
Test Case	*11*
Test Objective	*To check whether menu is opened after entering correct account number.*
Test Data	*Enter account number.*
Expected Result	*Menu must be open.*
Test Result	*Menu opens.*
Conclusion	*Expected result matches actual result.*

Test Case	*12*
Test Objective	*To check whether msg of "account no. is not valid" prompt or not when incorrect account no. is entered.*
Test Data	*Enter wrong account number.*
Expected Result	*Message must appear.*
Test Result	*Message appears.*
Conclusion	*Expected result matches actual result.*

Test Case	*13*
Test Objective	*To check if the color text are displayed or not.*
Test Data	*Run program.*
Expected Result	*Color text should be seen.*
Test Result	*Color text seen.*
Conclusion	*Expected result matches actual result.*

Results:

S.no.	*Test Objective*	*Results*
1.	*To check whether program runs or not.*	*successful*
2.	*To check if the program menu displays all menu options or not.*	*successful*
3.	*To check all options are right or not.*	*successful*
4.	*To check account was created or not.*	*successful*
5.	*To check if the program add amount after depositing.*	*successful*
6.	*To check if the program subtract amount after withdrawing.*	*successful*



7.	*To check if 'view detail' option shows the info or not.*	*successful*
8.	*To check if currency converter works properly or not.*	*successful*
9.	*To check if the exit option works or not.*	*successful*
10.	*To check whether menu options return back or not.*	*successful*
11.	*To check whether menu is opened after entering correct account number.*	*successful*
12.	*To check whether msg of "account no. is not valid" prompt or not when incorrect account no. is entered.*	*successful*
13.	*To check if the color text are displayed or not.*	*successful*

coding

```
#include<fstream.h>
#include<ctype.h>
#include<iomanip.h>
#include<conio.h>
#include<stdio.h>
//                  CLASS USED IN PROJECT
 class account
 {         int acno;
          char name[50];
          int deposit;
          char type;
 public:
          void create_account();           //function to get data from user
          void show_account(); //function to show data on screen
          void modify();   //function to get new data from user
          void dep(int);  //function to accept amount and add to balance amount
          void draw(int);//function to accept amount and subtract from balance amount
          void report();  //function to show data in tabular format
          int retacno();   //function to return account number
          int retdeposit();          //function to return balance amount
          char rettype(); //function to return type of account
 };        //class ends here
void account::create_account()
 {      cout<<"\nEnter The account No. :";
          cin>>acno;
cout<<"\n\nEnter The Name of The account Holder : ";
          gets(name);
          cout<<"\nEnter Type of The account (C/S) : ";
          cin>>type;
          type=toupper(type);
          cout<<"\nEnter The Initial amount(>=500 for Saving and >=1000 for current ) : ";
          cin>>deposit;
          cout<<"\n\n\nAccount Created..";
 }
 void account::show_account()
 {         cout<<"\nAccount No. : "<<acno;
          cout<<"\nAccount Holder Name : ";
          cout<<name;
          cout<<"\nType of Account : "<<type;
          cout<<"\nBalance amount : "<<deposit;
 }    void account::modify()
 {        cout<<"\nThe account No."<<acno;
          cout<<"\n\nEnter The Name of The account Holder : ";
          gets(name);
          cout<<"\nEnter Type of The account (C/S) : ";
          cin>>type;
          type=toupper(type);
          cout<<"\nEnter The amount : ";
          cin>>deposit;
} void account::dep(int x)
 {       deposit+=x;
 }
```

```
void account::draw(int x)
  {   deposit-=x;
  }   void account::report()
  {  Cout<<acno<<setw(10)<<" "<<name<<setw(10)<<" "<<type<<setw(6)<<deposit<<endl;
  }    int account::retacno()
  {  return acno;
  }     int account::retdeposit()
  {      return deposit;
  }      char account::rettype()
  {      return type;
  }
```

// **________function declaration**

```
  void write_account();                    //function to write record in binary file
  void display_sp(int);  //function to display account details given by user
  void modify_account(int);     //function to modify record of file
  void delete_account(int);      //function to delete record of file
  void display_all();                  //function to display all account details
  void deposit_withdraw(int, int);   // function to desposit/withdraw amount for given account
  void intro();           //introductory screen function
```

//________THE MAIN FUNCTION OF PROGRAM

```
            int main()
  {         clrscr();
            char ch;
            int num;
            intro();
            do
```

```
{           clrscr();
            cout<<"\n\n\n\tMAIN MENU";
            cout<<"\n\n\t01. NEW ACCOUNT";
            cout<<"\n\n\t02. DEPOSIT AMOUNT";
            cout<<"\n\n\t03. WITHDRAW AMOUNT";
            cout<<"\n\n\t04. BALANCE ENQUIRY";
            cout<<"\n\n\t05. ALL ACCOUNT HOLDER LIST";
            cout<<"\n\n\t06. CLOSE AN ACCOUNT";
            cout<<"\n\n\t07. MODIFY AN ACCOUNT";
            cout<<"\n\n\t08. EXIT";
            cout<<"\n\n\tSelect Your Option (1-8) ";
            cin>>ch;
            clrscr();
            switch(ch)
            {
            case '1':
                        write_account();
                        break;
            case '2':
                        cout<<"\n\n\tEnter The account No. : "; cin>>num;
                        deposit_withdraw(num, 1);
                        break;
            case '3':
                        cout<<"\n\n\tEnter The account No. : "; cin>>num;
                        deposit_withdraw(num, 2);
                        break;
            case '4':
```

```
                        cout<<"\n\n\tEnter The account No. : "; cin>>num;
                        display_sp(num);
                        break;
                case '5':
                        display_all();
                        break;
                case '6':
                        cout<<"\n\n\tEnter The account No. : "; cin>>num;
                        delete_account(num);
                        break;
                 case '7':
                        cout<<"\n\n\tEnter The account No. : "; cin>>num;
                        modify_account(num);
                        break;
                 case '8':
                        cout<<"\n\n\tThanks for using bank managemnt system...";
                        break;
                 default :cout<<"\a";
                }
                getch();
   }while(ch!='8');
        return 0;
  }
```

//   **<u>function to write in file</u>**

```
            void write_account()
  {         account ac;
            ofstream outFile;
            outFile.open("account.dat",ios::binary|ios::app);
            ac.create_account();
            outFile.write((char *) &ac, sizeof(account));
            outFile.close();
  }
```

//   function to read specific record from file

```
  void display_sp(int n)
  {        account ac;
            int flag=0;
            ifstream inFile;
   inFile.open("account.dat",ios::binary);
            if(!inFile)
{           cout<<"File could not be open !! Press any Key...";
            return;
            }
            cout<<"\nBALANCE DETAILS\n";
   while(inFile.read((char *) &ac, sizeof(account)))
            {
            if(ac.retacno()==n)
                        {         ac.show_account();
                                   flag=1;        }    }
            inFile.close();
            if(flag==0)
                        cout<<"\n\nAccount number does not exist";            }
```

```
//          function to modify record of file
             void modify_account(int n)
{
         int found=0;
          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.read((char *) &ac, sizeof(account)) && found==0)
          {         (ac.retacno()==n)
                    {         ac.show_account();
                               cout<<"\n\nEnter The New Details of account"<<endl;
                               ac.modify();
                               int pos=(-1)*sizeof(account);
                               File.seekp(pos,ios::cur);
                        File.write((char *) &ac, sizeof(account));
                        cout<<"\n\n\t Record Updated";
                        found=1;
                      }           }
          File.close();
          if(found==0)
        cout<<"\n\n Record Not Found ";
 }
```

// function to delete record of file

```
            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((char *) &ac, sizeof(account)))
        {        if(ac.retacno()!=n)
                 {      outFile.write((char *) &ac, sizeof(account));
                 }    }
        inFile.close();
        outFile.close();
        remove("account.dat");
        rename("Temp.dat","account.dat");
        cout<<"\n\n\tRecord Deleted ..";
 }
```

// function to display all accounts deposit list

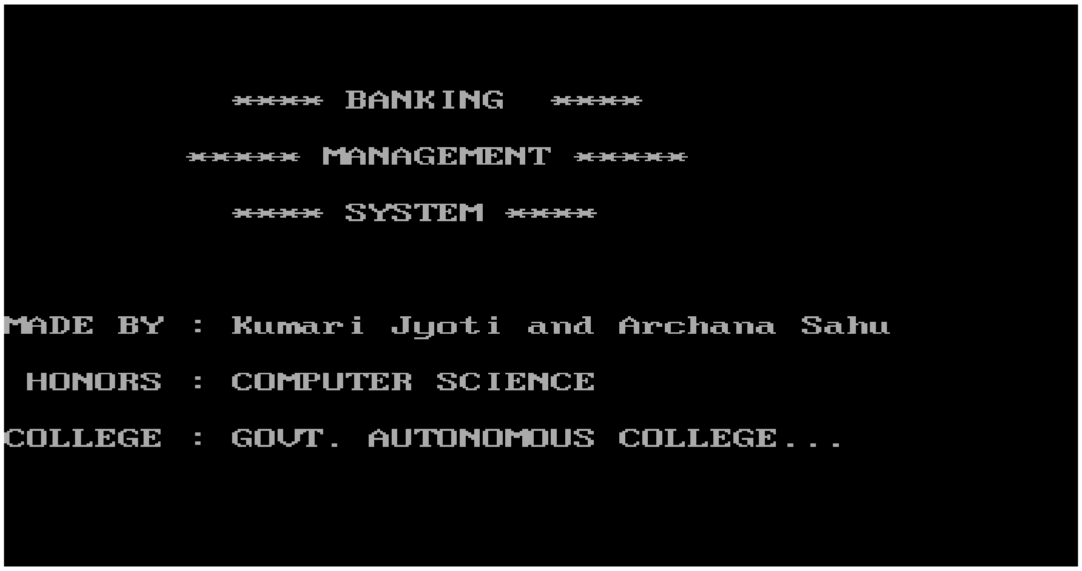
```
           void display_all()
 {       clrscr();
        account ac;
        ifstream inFile;
        inFile.open("account.dat",ios::binary);
        if(!inFile)
        {       cout<<"File could not be open !! Press any Key...";
                return;        }
        cout<<"\n\n\t\tACCOUNT HOLDER LIST\n\n";
        cout<<"A/c no.     NAME          Type  Balance\n";
        while(inFile.read((char *) &ac, sizeof(account)))
        {      ac.report();
        }      inFile.close();     }
```

//   function to deposit and withdraw amounts

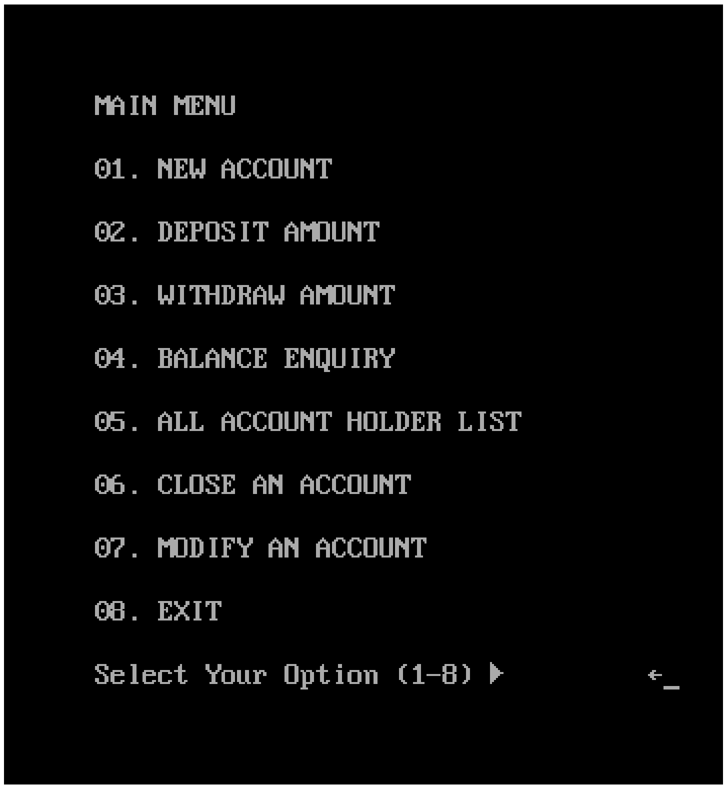
```
  void deposit_withdraw(int n, int option)
  {          int amt;
        int found=0;
        account ac;
        fstream File;
   File.open("account.dat", ios::binary|ios::in|ios::out);
        if(!File)
        {         cout<<"File could not be open !! Press any Key...";
                return;
        }  while(File.read((char *) &ac, sizeof(account)) && found==0)
        {       if(ac.retacno()==n)
                {               ac.show_account();
                        if(option==1)
                        {         cout<<"\n\n\tTO DEPOSITE AMOUNT ";
                                cout<<"\n\nEnter The amount to be deposited";
                                cin>>amt;
                                ac.dep(amt);
                        }  if(option==2)
```

```
{
                                    cout<<"\n\n\tTO WITHDRAW AMOUNT ";
                                     cout<<"\n\nEnter The amount to be withdraw";
                                     cin>>amt;
                                     int bal=ac.retdeposit()-amt;
                                     if((bal<500 && ac.rettype()=='S') || (bal<1000 &&
ac.rettype()=='C'))
                                   cout<<"Insufficience balance";
                                     else
                                     ac.draw(amt);                    }
                              int pos=(-1)* sizeof(ac);
                              File.seekp(pos,ios::cur);
                              File.write((char *) &ac, sizeof(account));
                              cout<<"\n\n\t Record Updated";
                              found=1;      }        }   File.close();
        if(found==0)
                cout<<"\n\n Record Not Found ";            }
//                 INTRODUCTION FUNCTION
 void intro()
  {     cout<<"\n\n\n\t  **** BANK  ****";
        cout<<"\n\n\t***** MANAGEMENT *****";
        cout<<"\n\n\t  **** SYSTEM ****";
        cout<<"\n\n\n\nMADE BY :  KUMARI JYOTI AND ARCHANA SAHU ";
        cout<<"\n\n FINAL YEAR STUDENT (BSC);
        cout<<"\n\nCOLLEGE : GOVERNMENT AUTONOMOUS COLLEGE";
        getch();            }
  //                 END OF THE PROJECT
```

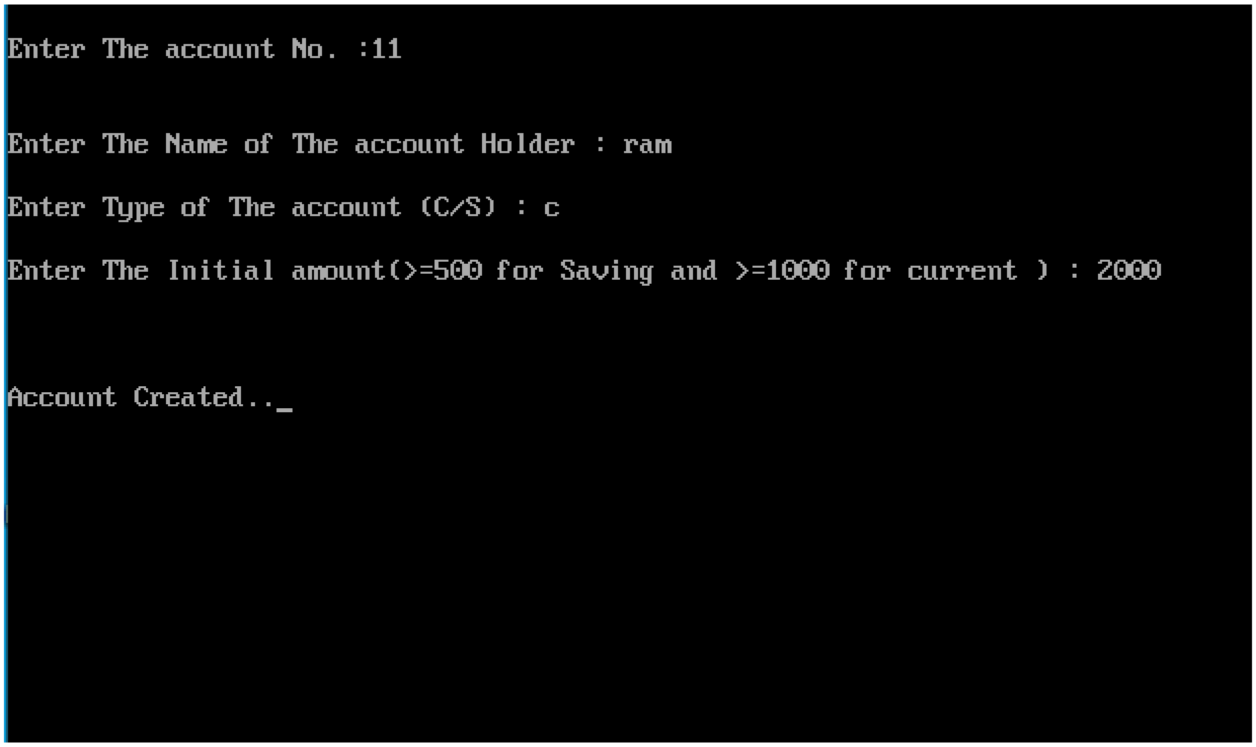
OUTPUT:-



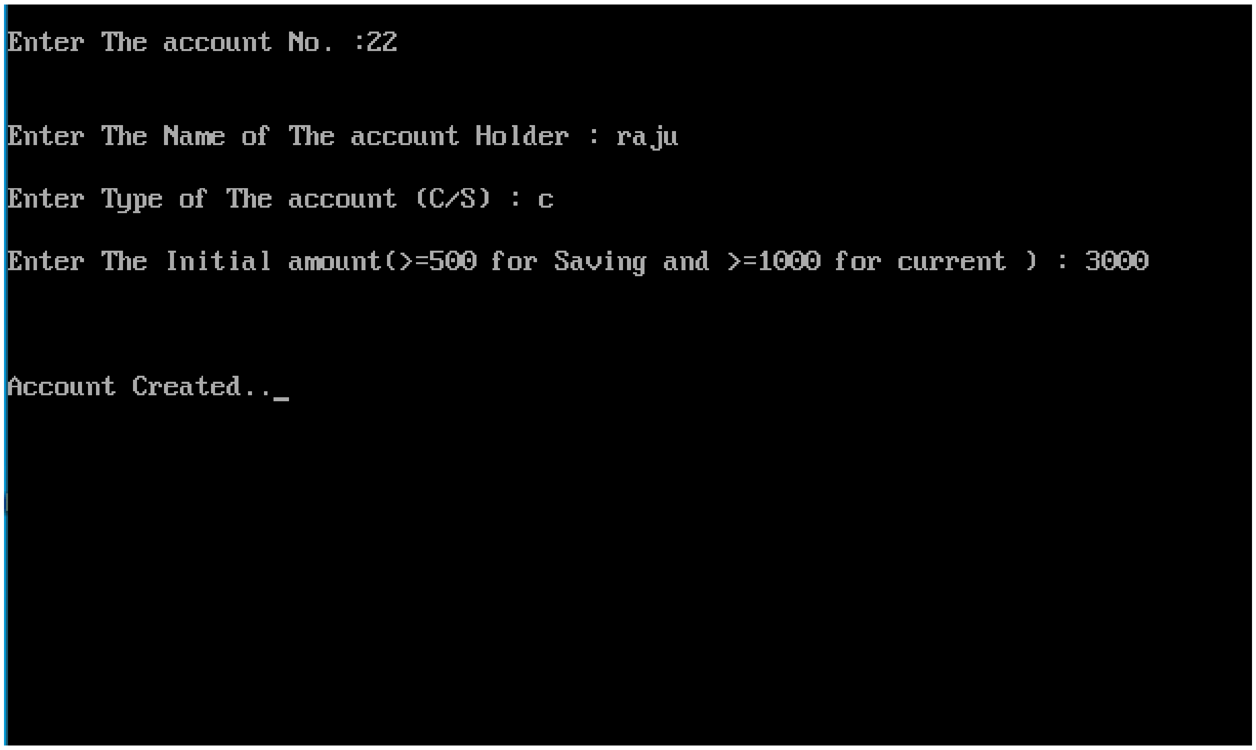
```
          ***** BANKING  *****
        ****** MANAGEMENT ******
          ***** SYSTEM *****

MADE BY : Kumari Jyoti and Archana Sahu
 HONORS : COMPUTER SCIENCE
COLLEGE : GOVT. AUTONOMOUS COLLEGE...
```



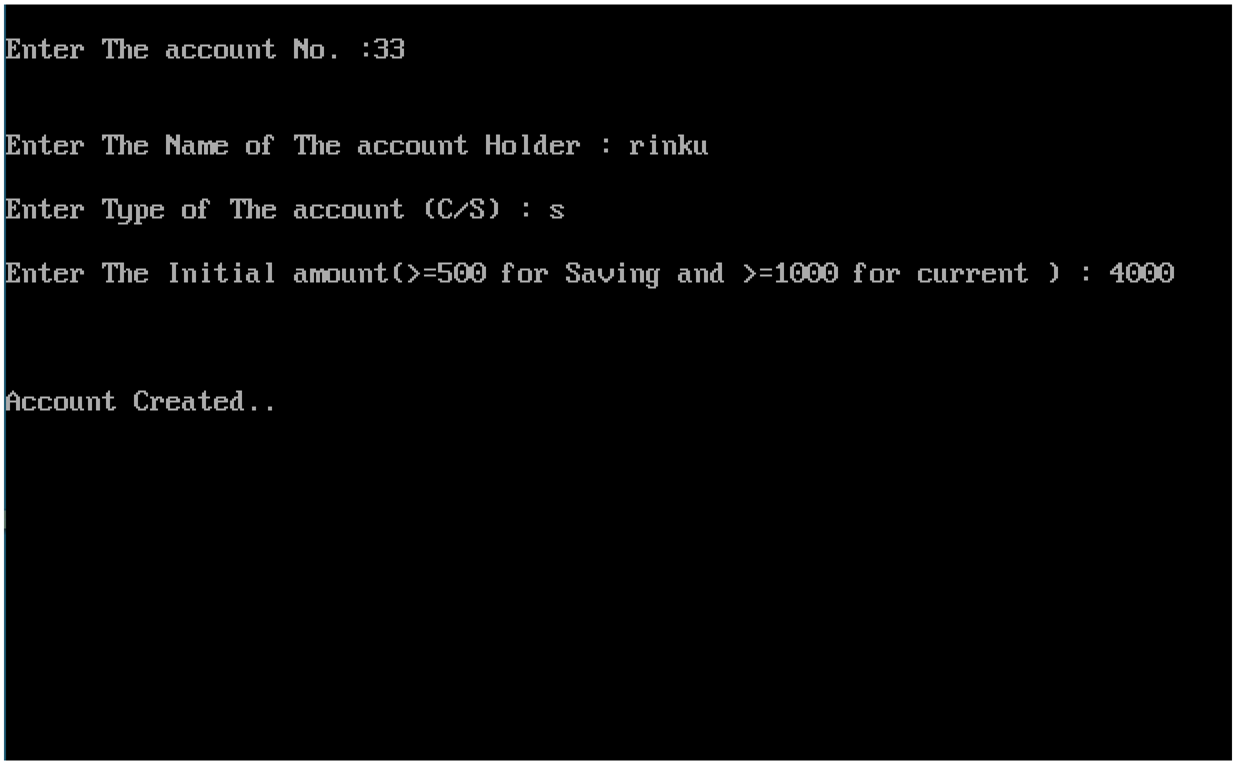
```
MAIN MENU
01. NEW ACCOUNT
02. DEPOSIT AMOUNT
03. WITHDRAW AMOUNT
04. BALANCE ENQUIRY
05. ALL ACCOUNT HOLDER LIST
06. CLOSE AN ACCOUNT
07. MODIFY AN ACCOUNT
08. EXIT
Select Your Option (1-8) ▶        ←_
```



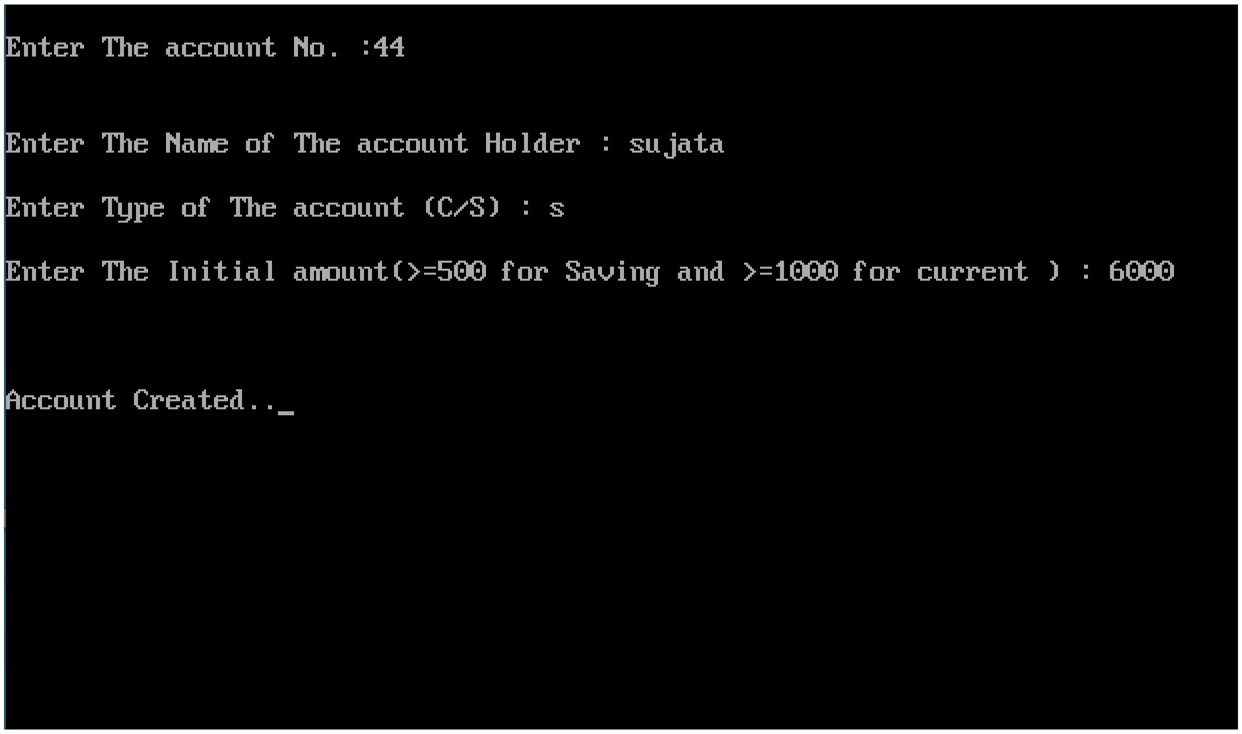
```
Enter The account No. :11


Enter The Name of The account Holder : ram

Enter Type of The account (C/S) : c

Enter The Initial amount(>=500 for Saving and >=1000 for current ) : 2000


Account Created.._
```



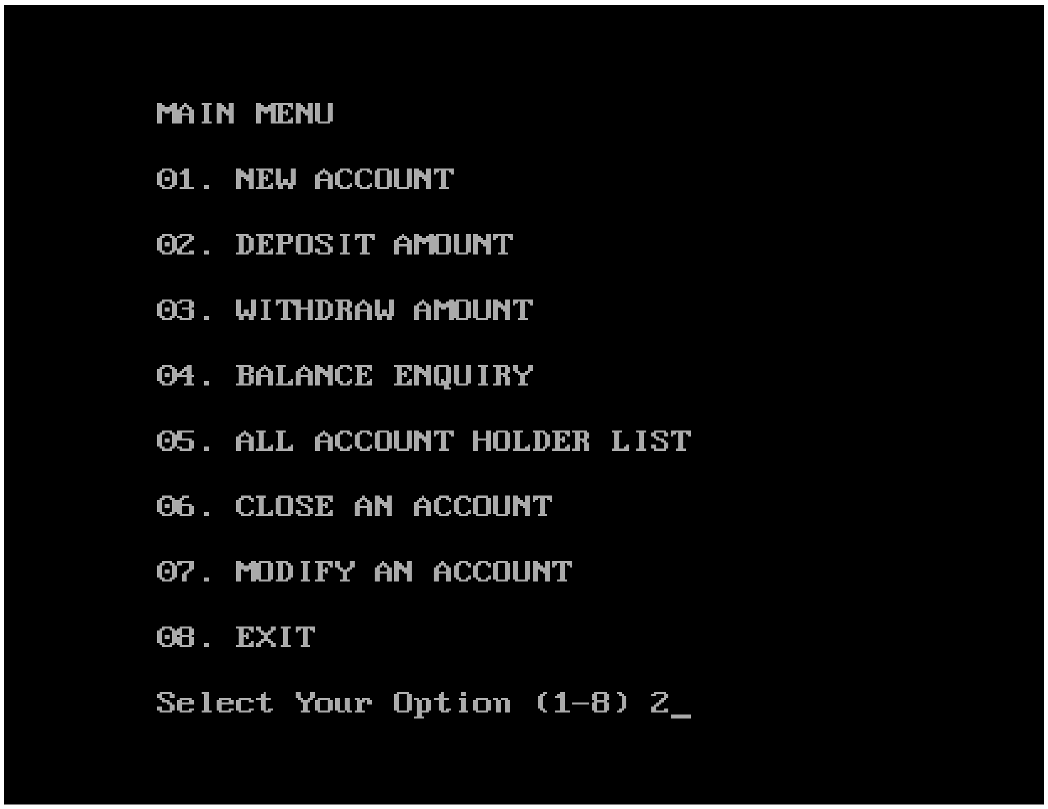
```
Enter The account No. :22


Enter The Name of The account Holder : raju

Enter Type of The account (C/S) : c

Enter The Initial amount(>=500 for Saving and >=1000 for current ) : 3000


Account Created.._
```



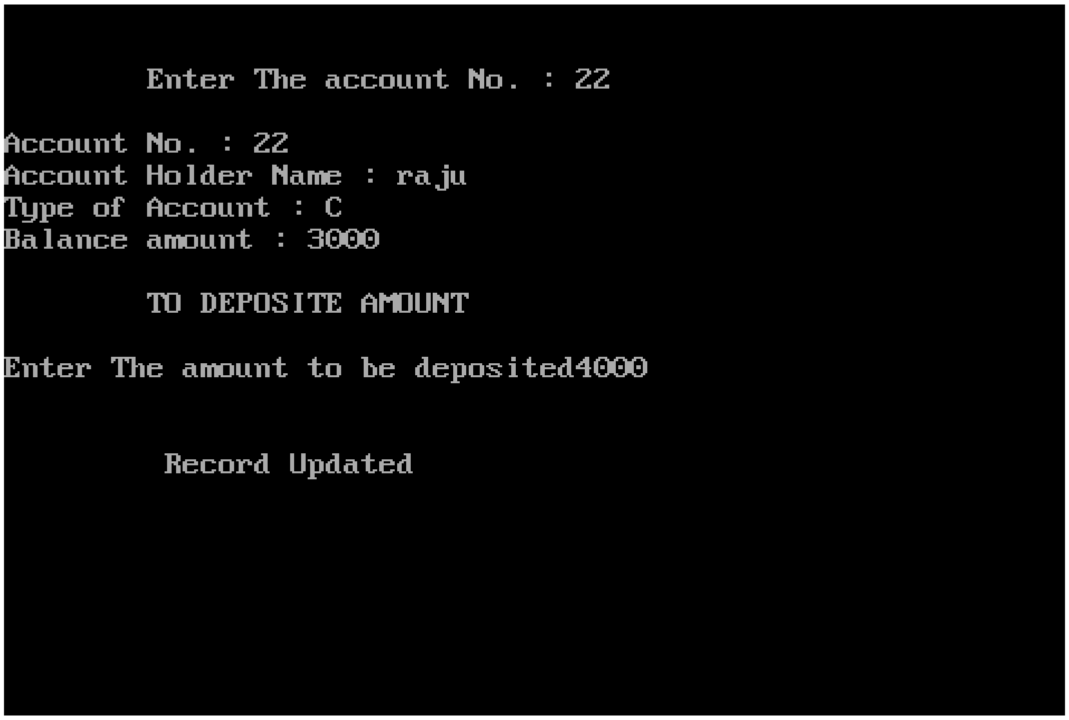
```
Enter The account No. :33


Enter The Name of The account Holder : rinku

Enter Type of The account (C/S) : s

Enter The Initial amount(>=500 for Saving and >=1000 for current ) : 4000


Account Created..
```



```
Enter The account No. :44


Enter The Name of The account Holder : sujata

Enter Type of The account (C/S) : s

Enter The Initial amount(>=500 for Saving and >=1000 for current ) : 6000


Account Created.._
```

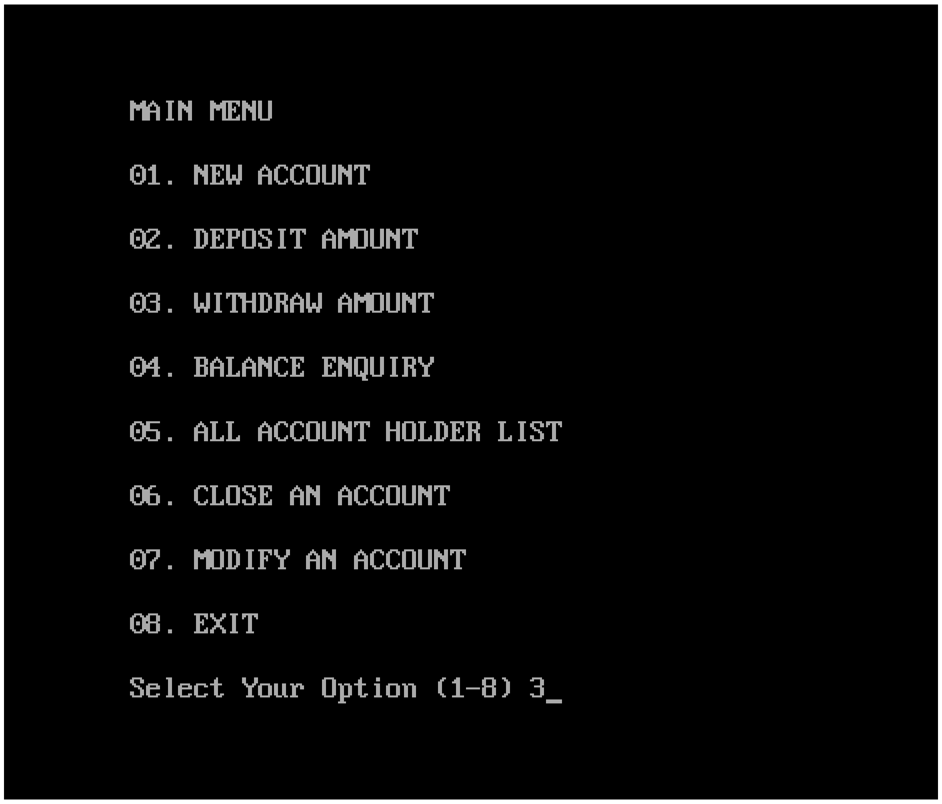


```
MAIN MENU

01. NEW ACCOUNT

02. DEPOSIT AMOUNT

03. WITHDRAW AMOUNT

04. BALANCE ENQUIRY

05. ALL ACCOUNT HOLDER LIST

06. CLOSE AN ACCOUNT

07. MODIFY AN ACCOUNT

08. EXIT

Select Your Option (1-8) 2_
```

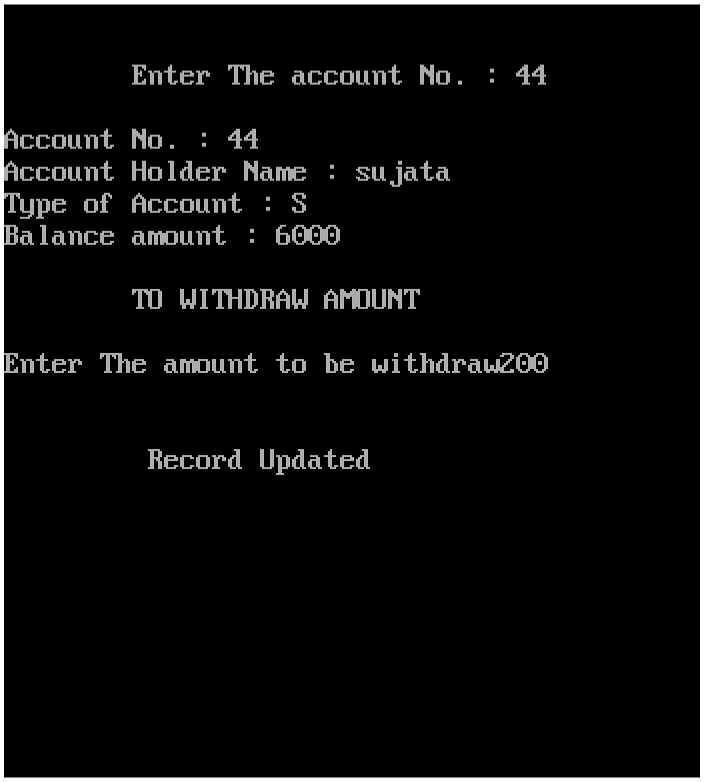


```
        Enter The account No. : 22

Account No. : 22
Account Holder Name : raju
Type of Account : C
Balance amount : 3000

        TO DEPOSITE AMOUNT

Enter The amount to be deposited4000


         Record Updated
```

WITHDRAWL:

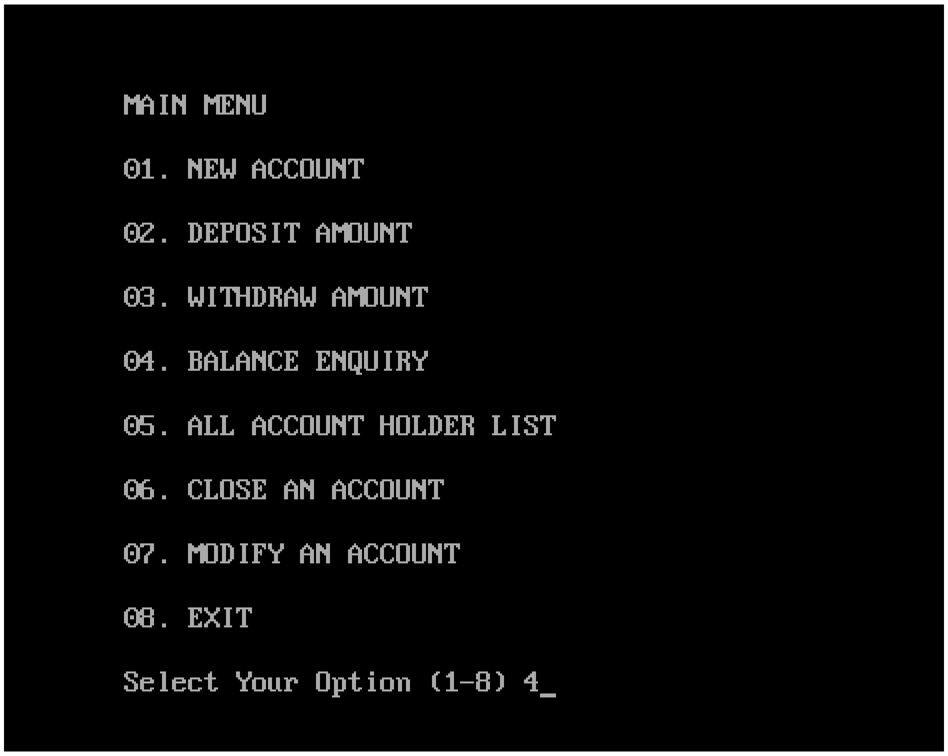


```
MAIN MENU

01. NEW ACCOUNT

02. DEPOSIT AMOUNT

03. WITHDRAW AMOUNT

04. BALANCE ENQUIRY

05. ALL ACCOUNT HOLDER LIST

06. CLOSE AN ACCOUNT

07. MODIFY AN ACCOUNT

08. EXIT

Select Your Option (1-8) 3_
```

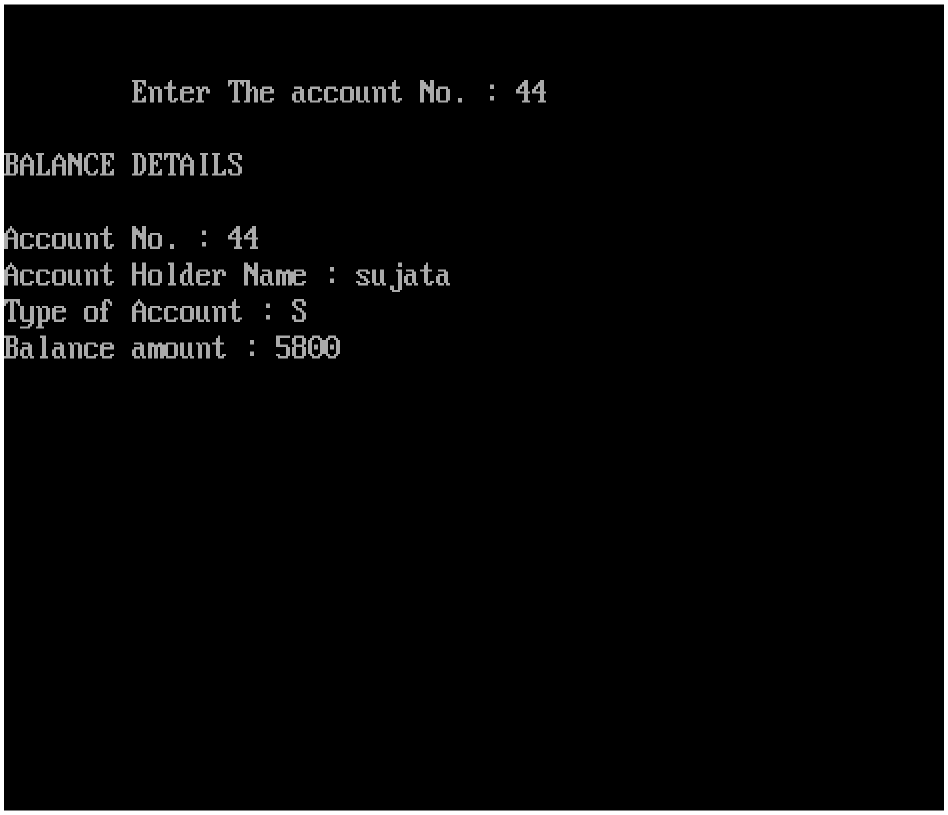


```
        Enter The account No. : 44

Account No. : 44
Account Holder Name : sujata
Type of Account : S
Balance amount : 6000

        TO WITHDRAW AMOUNT

Enter The amount to be withdraw200


         Record Updated
```

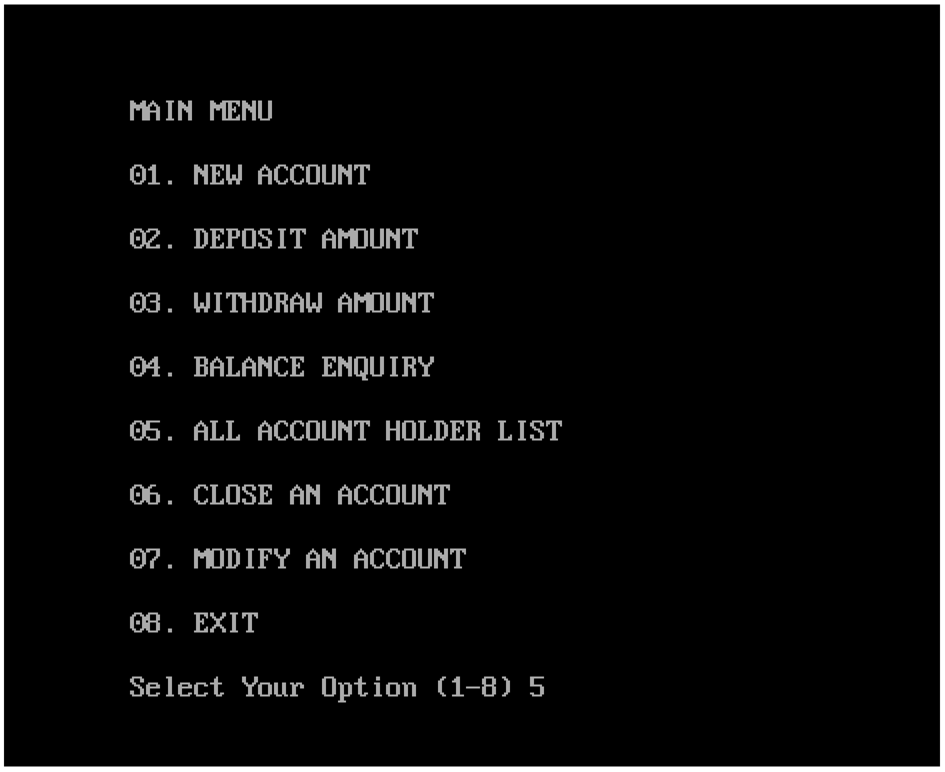
BALANCE ENQUIRY:



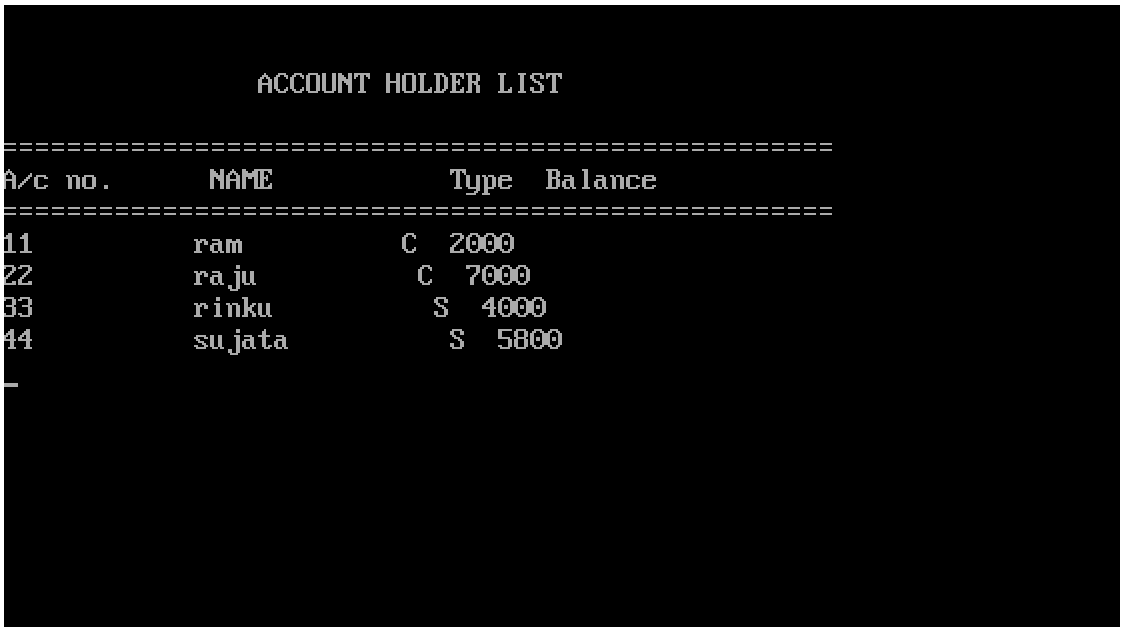
```
MAIN MENU

01. NEW ACCOUNT

02. DEPOSIT AMOUNT

03. WITHDRAW AMOUNT

04. BALANCE ENQUIRY

05. ALL ACCOUNT HOLDER LIST

06. CLOSE AN ACCOUNT

07. MODIFY AN ACCOUNT

08. EXIT

Select Your Option (1-8) 4_
```



```
        Enter The account No. : 44

BALANCE DETAILS

Account No. : 44
Account Holder Name : sujata
Type of Account : S
Balance amount : 5800
```

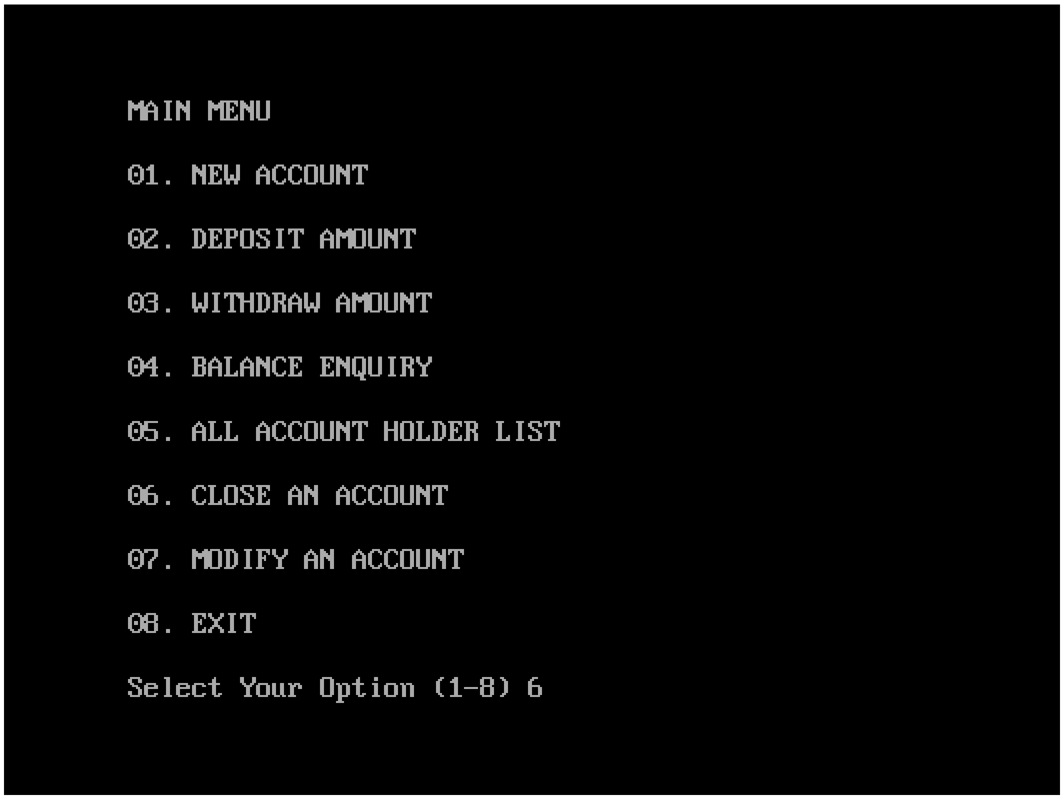


```
MAIN MENU

01. NEW ACCOUNT

02. DEPOSIT AMOUNT

03. WITHDRAW AMOUNT

04. BALANCE ENQUIRY

05. ALL ACCOUNT HOLDER LIST

06. CLOSE AN ACCOUNT

07. MODIFY AN ACCOUNT

08. EXIT

Select Your Option (1-8) 5
```

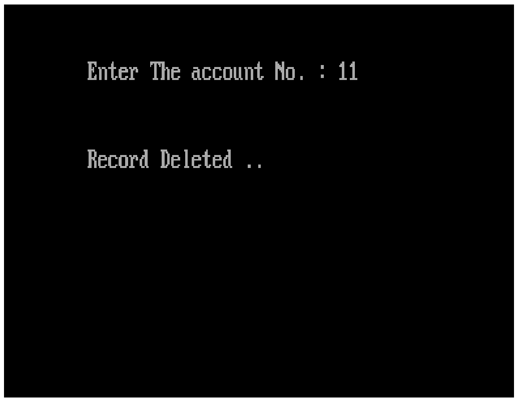


```
               ACCOUNT HOLDER LIST

====================================================
A/c no.      NAME           Type  Balance
====================================================
11          ram          C  2000
22          raju          C  7000
33          rinku          S  4000
44          sujata          S  5800
_
```

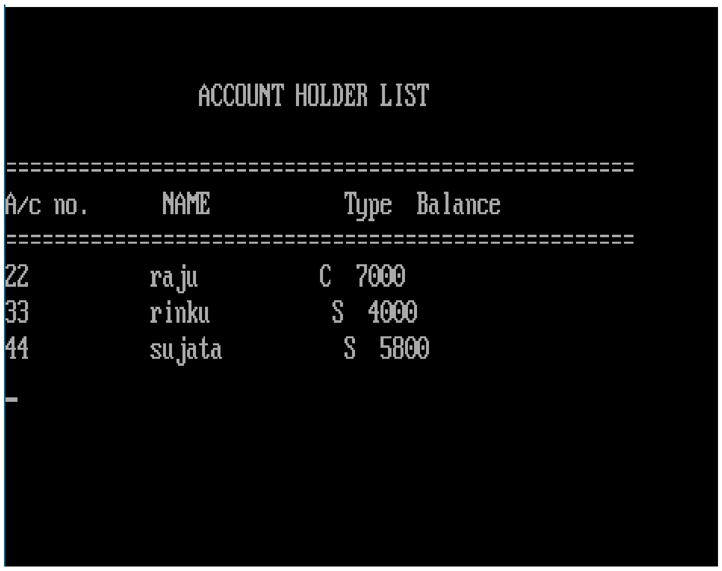
CLOSING THE ACCOUNT:-



```
MAIN MENU

01. NEW ACCOUNT

02. DEPOSIT AMOUNT

03. WITHDRAW AMOUNT

04. BALANCE ENQUIRY

05. ALL ACCOUNT HOLDER LIST

06. CLOSE AN ACCOUNT

07. MODIFY AN ACCOUNT

08. EXIT

Select Your Option (1-8) 6
```

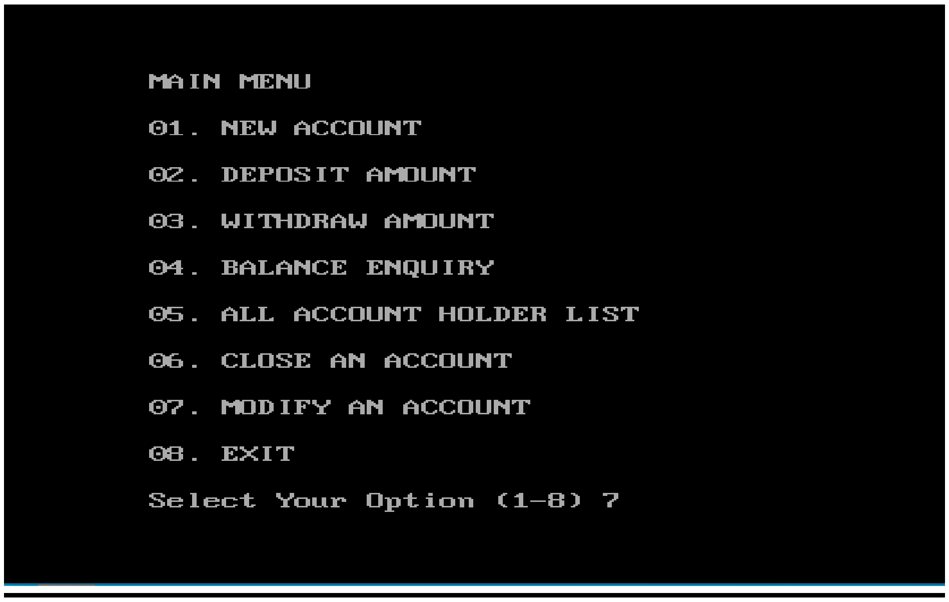


```
Enter The account No. : 11


Record Deleted ..
```

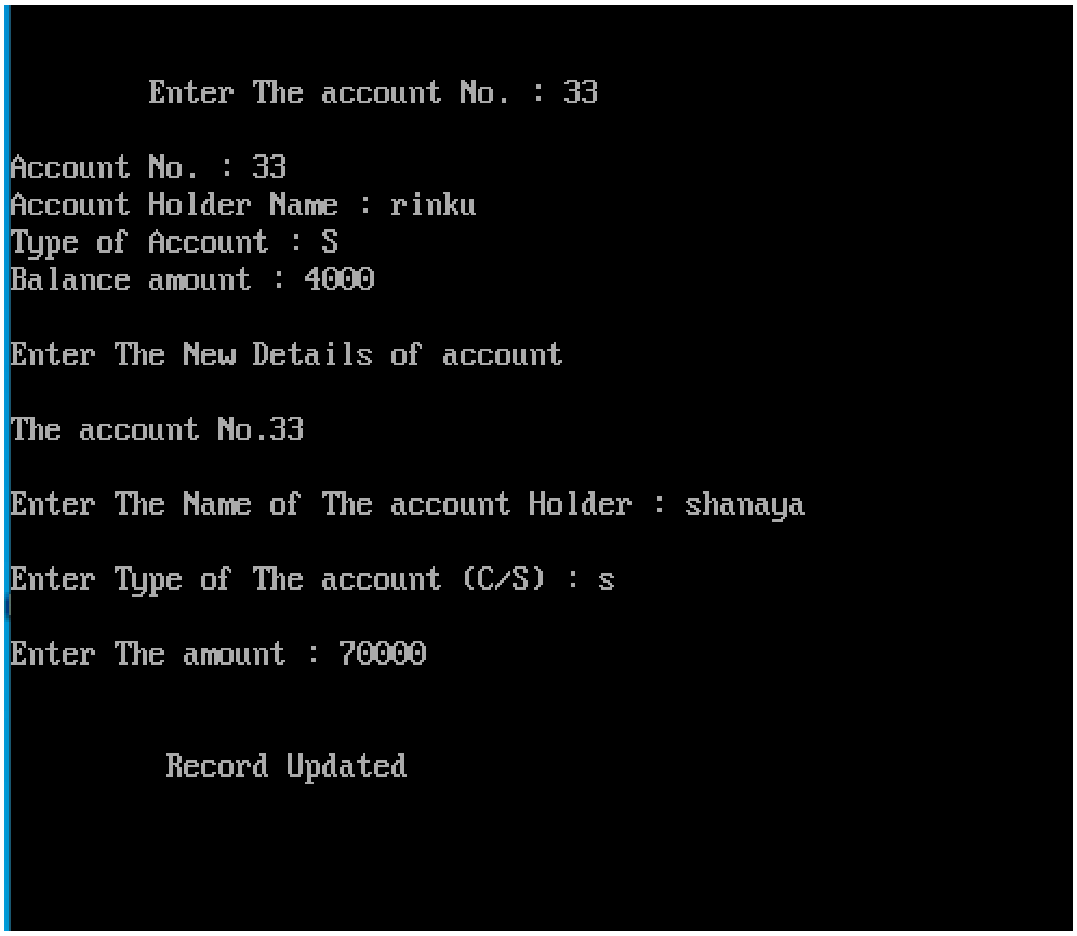


```
                ACCOUNT HOLDER LIST

====================================================
A/c no.      NAME           Type  Balance
====================================================
22          raju          C  7000
33          rinku          S  4000
44          sujata          S  5800
```

MODIFYING THE ACCOUNT:-

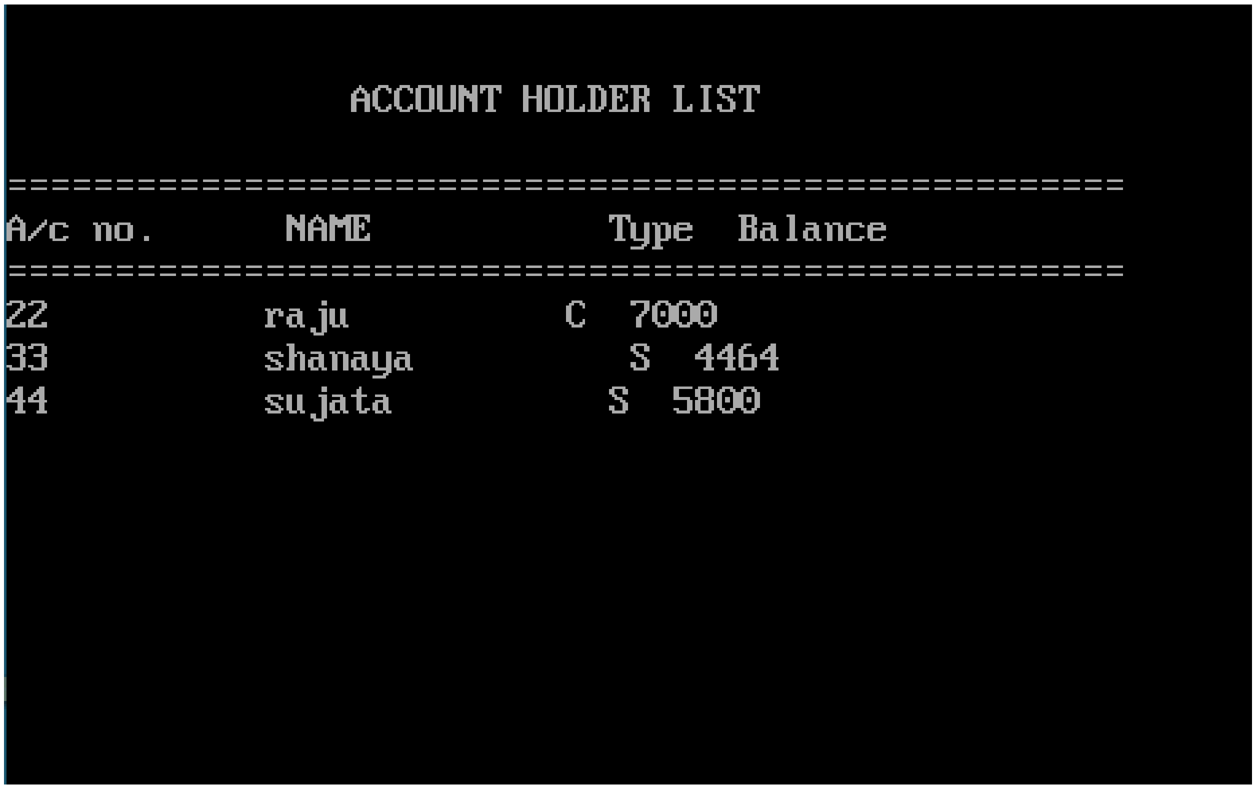


```
MAIN MENU

01. NEW ACCOUNT

02. DEPOSIT AMOUNT

03. WITHDRAW AMOUNT

04. BALANCE ENQUIRY

05. ALL ACCOUNT HOLDER LIST

06. CLOSE AN ACCOUNT

07. MODIFY AN ACCOUNT

08. EXIT

Select Your Option (1-8) 7
```

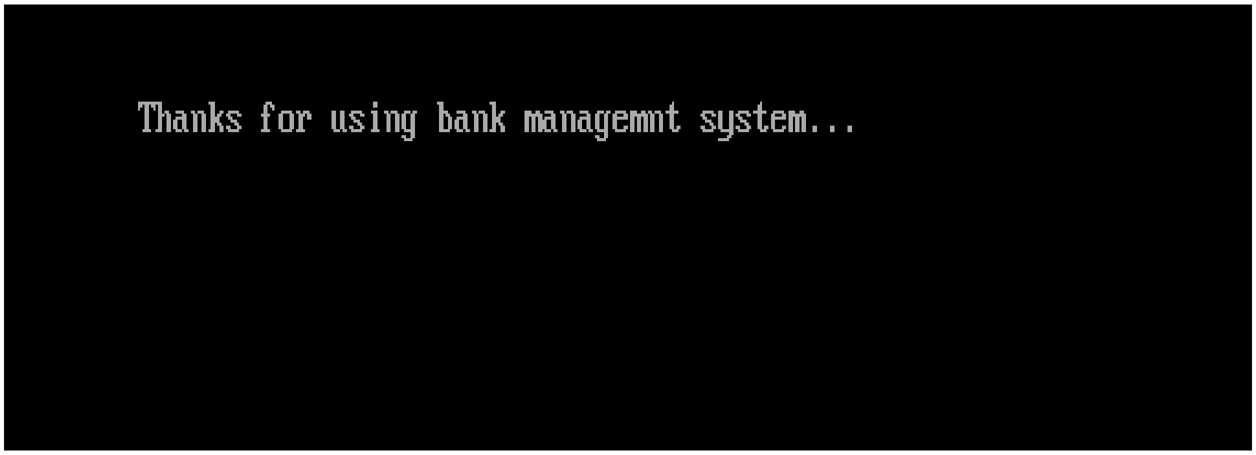


```
        Enter The account No. : 33

Account No. : 33
Account Holder Name : rinku
Type of Account : S
Balance amount : 4000

Enter The New Details of account

The account No.33

Enter The Name of The account Holder : shanaya

Enter Type of The account (C/S) : s

Enter The amount : 70000


         Record Updated
```

New updated record:-



```
                 ACCOUNT HOLDER LIST

==========================================================
A/c no.       NAME            Type  Balance
==========================================================
22           raju           C  7000
33           shanaya           S  4464
44           sujata           S  5800
```



```
      Thanks for using bank managemnt system...
```

CONCLUSION

There are many advantages of using this program as it contains various features like

It is actually a user-friendly software as it is just easy to use by just following the instruction which are appeared on the screen.

This program needs user account no to access user information, so that only authorized users are only allowed to accessed through the internal main system.

Once a record has been saved, duplicate record can't be made. All the record has different account so that there will be not be any misplace of the records entered.

REFERENCES

WWW.RESEARCHGATE.NET

WWW.W3SCHOOL.COM

THANKYOU