Introduction

- **Introduction**

Here is a project we developed as mini project in C bank management system during our first semester; it is complete and totally error-free. This project is focused on customer account services in bank, so it is named "Customer Account Bank Management System".

Here, you can create a new account, view and manage transactions, check the details of an existing account,. Overall, with this project, you can perform banking activities like in a REAL bank. Bank management mini project in C is a console application without graphics

- **Problem Definition:**

Although the basic type of services offered by a bank depends upon the type of bank and the country, services provided usually include: Taking deposits from their customers and issuing current or checking accounts and savings accounts to individuals and business. Extending loans to individuals and business, Cashing cheque. Facilitating money transactions such as wire transfer and cashiers cheque, Consumer & commercial financial advisory services, financial transaction can be performed through many different channels.

- **Scope:**

- This Bank Management System will provide the transaction going inside the bank without manual processing. All information will be updated automatically by using the information stored in the system files

- This system has lots of benefit in day to day life

- **Problem Identification:**

1. Managing data of all the user manually
2. Improper money transfer system.
3. Loan default risk
4. Less data security manually.

Objectives

- To allow only authorized user to access various function and processed available in the system.
- Locate any A/C wanted by the user.
- Provide greater speed & reduced time consumption.
- To maintain the user details neatly.
- To save the time of user
- To provide top management a single point of control.
- To simplify process of banking management system

System Requirements Specification

- **Software Requirement:**

- Turbo C
- Microsoft Visual Studio Code
- Dev C++

- **Hardware Requirement**

  - Computer or laptop
  - Intel(R) Core(TM) i3-Processor
  - RAM-1 GB Minimum
  - Storage-100GB

Methodology

- **Algorithm:**

Step1

It gives firstly menu in that it gives following

1.CREATE A BANK ACCOUNT

2. ALREADY A USER? SIGN IN

3.EXIT

And then it ask for enter your choice

Step2

When you enter the choice then it asks first name, last name, father's name, address, account type, date of birth, phone number, username and then password.

When user will fill the all the detail then it can asks for username name and password.

User fill correct information then it shows all details of user and shows one menu. In that menu it gives following options :

1.CHECK BALANCE
2.TRANSFER MONEY
3.LOG OUT
4.EXIT

When user choose option 1 then it shows TRANSACTION ID, AMOUNT, TOTAL AMOUNT.

When user can choose option 2
In this user can enter the username of the who can transfer money form whom.
And then enter the amount of transferring money.

When user can choose option 3 he can logout from program.

When user can choose option 4 he can exit the program.

Step 3

User fill correct username and password then it shows all details of user and shows one menu. In that menu it gives following options :

1.CHECK BALANCE
2.TRANSFER MONEY
3.LOG OUT
4.EXIT

when user choose option 1 then it shows TRANSACTION ID, AMOUNT TOTAL AMOUNT.

When user can choose option 2
In this user can enter the username of the who can transfer money form whom.
And then enter the amount of transferring money.

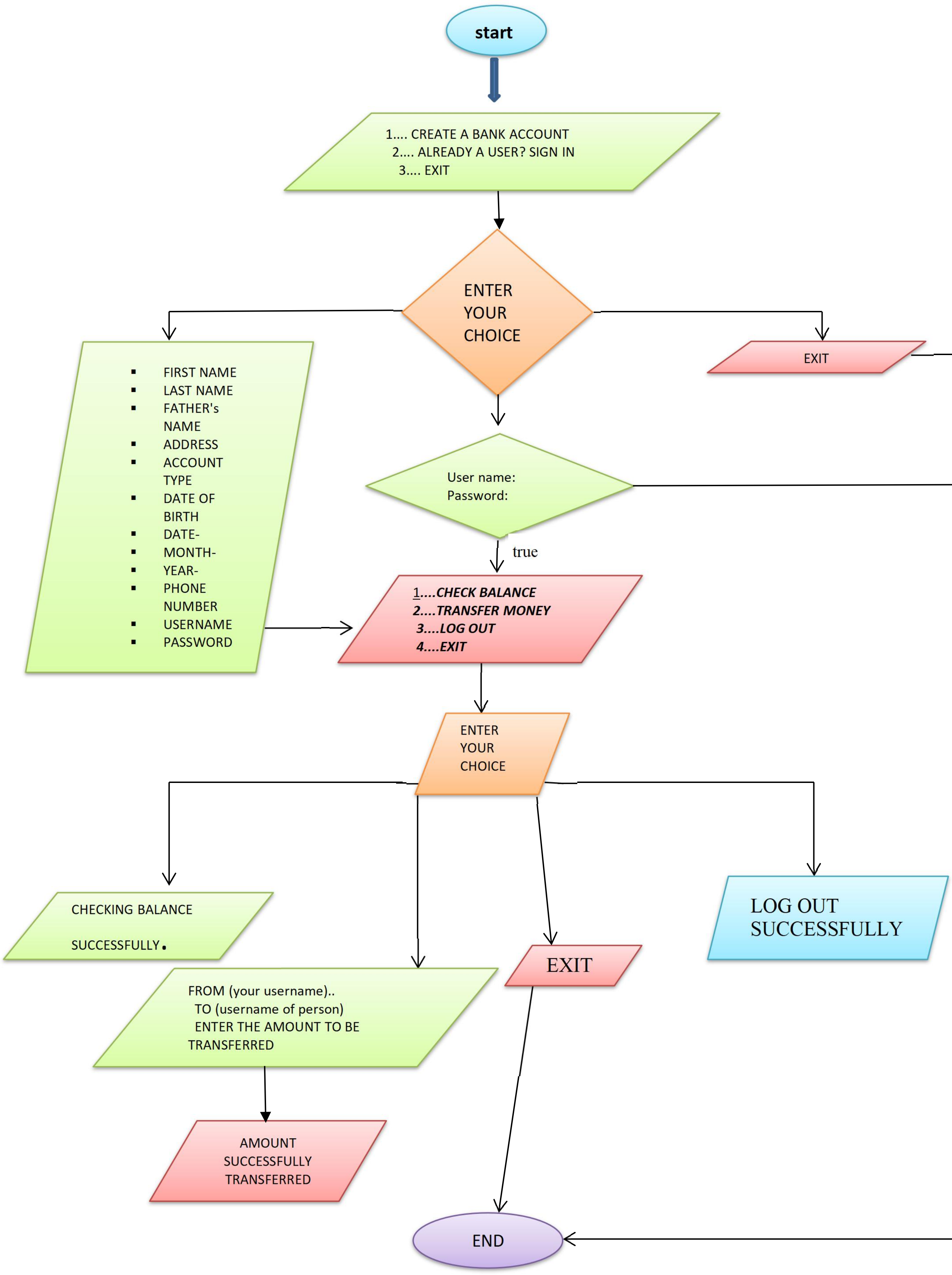
When user can choose option 3 he can logout from program.

When user can choose option 4 he can exit the program.

Step 4

When user can choose option 4 he can exit the program

- **Flow Diagram(Flow Chart):**



Implementation

Let’s look at the C implementation of each of the modules of the program and finally consolidate all the modules together and create a full working program.

1. Create a Bank Account-
Take all the input from the user and make a structure for it to store the data in a file.

2. Transfer Money-
Take the username of another user to whom we want to transfer the money and open his record in the file and write the amount to the file.

3. Check Balance-
Opening a file in which all the transfer records are written and read them one by one and match the username passed in the function to fetch the correct transfer records.

4. Login Functionality-
To add Login functionality, we are opening the file and matching the username provided by the user at the time of registration, and logging in to him if the username is correct and matches with the record present in our file.

“Introduction of C Programming”

C is a very popular language throughout the world and is ideal for a programmer to learn at the beginning of their career. There is little vocabulary to learn, the syntax is simple, and the modular structure of the language is easier to learn

Advantages of learning C:

- It is easy to understand:

  One of the main reasons why people choose C over other programming languages is its simplicity. C is a highly portable language as programs coded in it are far more fast and efficient. This makes learning C easier than any other programming language. You can easily grasp the concepts behind C because there aren't many keywords or symbols involved. In addition, you don't need to be an expert in computer science to get started with C programming. All you have to do is read through some tutorials online and start writing your own codes. Also, there are system-generated functions and user-defined functions in C Language.

- Presence of many Libraries:

  C Language provides lots of built-in functions which consist of system-generated functions and user-defined functions. Many general functions can be used to develop a program, while the programmer can also create a function as per their requirements, which is called a usergenerated/defined function, in C Compiler

- Easy to write:

  Another reason why C is so popular as an efficient language among programmers is that it allows them to create their own software without having to worry about syntax errors. If you're not familiar with coding, then using structured language C will help you develop better skills. With C, you'll find yourself creating more efficient and effective solutions compared to those created by other programming languages.

Source Code:

```
// C program to implement
// the above approach
#include <conio.h>
#include<stdio.h>
#include <stdlib.h>
#include <string.h>
#include <windows.h>

// Declaring all the functions
void checkbalance(char*);
void transfermoney(void);
void display(char*);
void person(char*);
void login(void);
void loginsu(void);
void account(void);
void accountcreated(void);
void afterlogin(void);
void logout(void);

// Declaring gotoxy
// function for setting
// cursor position
void gotoxy(int x, int y)
{
        COORD c;
        c.X = x;
        c.Y = y;

        SetConsoleCursorPosition(
                GetStdHandle(STD_OUTPUT_HANDLE), c);
}

// Creating a structure to store
// data of the user
struct pass {
        char username[50];
        int date, month, year;
        char pnumber[15];
        char adharnum[20];
        char fname[20];
        char lname[20];
        char fathname[20];
        char mothname[20];
        char address[50];
        char typeaccount[20];
};

// Structure to keep track
// of amount transfer
struct money {
	char usernameto[50];
	char userpersonfrom[50];
	long int money1;
};

struct userpass {
	char password[50];
};

// Driver Code
int main()
{
	int i, a, b, choice;
	int passwordlength;

	gotoxy(20, 3);

	// Creating a Main
	// menu for the user
	printf("WELCOME TO BANK ACCOUNT SYSTEM\n\n");
	gotoxy(18, 5);

	printf("************");
	gotoxy(25, 7);

	gotoxy(20, 10);
	printf("1.... CREATE A BANK ACCOUNT");

	gotoxy(20, 12);
	printf("2.... ALREADY A USER? SIGN IN");
	gotoxy(20, 14);
	printf("3.... EXIT\n\n");

	printf("\n\nENTER YOUR CHOICE..");

	scanf("%d", &choice);

	switch (choice) {
	case 1:
		system("cls");
		printf("\n\n USERNAME 50 CHARACTERS MAX!!");
		printf("\n\n PASSWORD 50 CHARACTERS MAX!!");
		account();
		break;

	case 2:
		login();
		break;

	case 3:
		exit(0);
		break;
```

```
			getch();
		}
}

// Function to create accounts
// of users
void account(void)
{
		char password[20];
		int passwordlength, i, seek = 0;
		char ch;
		FILE *fp, *fu;
		struct pass u1;
		struct userpass p1;

		struct userpass u2;

		// Opening file to
		// write data of a user
		fp = fopen("username.txt", "ab");

		// Inputs
		system("cls");
		printf("\n\n!!!!!CREATE ACCOUNT!!!!!");

		printf("\n\nFIRST NAME..");
		scanf("%s", &u1.fname);

		printf("\n\n\nLAST NAME..");
		scanf("%s", &u1.lname);

		printf("\n\nFATHER's NAME..");
		scanf("%s", &u1.fathname);

		printf("\n\nMOTHER's NAME..");
		scanf("%s", &u1.mothname);

		printf("\n\nADDRESS..");
		scanf("%s", &u1.address);

		printf("\n\nACCOUNT TYPE");
		scanf("%s", &u1.typeaccount);

		printf("\n\nDATE OF BIRTH..");
		printf("\nDATE-");
		scanf("%d", &u1.date);
		printf("\nMONTH-");
		scanf("%d", &u1.month);
		printf("\nYEAR-");
		scanf("%d", &u1.year);
```

```
	printf("\n\nADHAR NUMBER");
	scanf("%s", u1.adharnum);

	printf("\n\nPHONE NUMBER");
	scanf("%s", u1.pnumber);

	printf("\n\nUSERNAME.. ");
	scanf("%s", &u1.username);

	printf("\n\nPASSWORD..");

	// Taking password in the form of
	// stars
	for (i = 0; i < 50; i++) {
		ch = getch();
		if (ch != 13) {
			password[i] = ch;
			ch = '*';
			printf("%c", ch);
		}
		else
			break;
	}

	// Writing to the file
	fwrite(&u1, sizeof(u1),
		1, fp);

	// Closing file
	fclose(fp);

	// Calling another function
	// after successful creation
	// of account
	accountcreated();
}

// Successful account creation
void accountcreated(void)
{
	int i;
	char ch;
	system("cls");
	printf(
		"PLEASE WAIT....\n\nYOUR DATA IS PROCESSING....");
	for (i = 0; i < 200000000; i++) {
		i++;
		i--;
	}

	gotoxy(30, 10);
```

```
        printf("ACCOUNT CREATED SUCCESSFULLY....");
        gotoxy(0, 20);

        printf("Press enter to login");

        getch();
        login();
}

// Login function to check
// the username of the user
void login(void)
{
        system("cls");

        char username[50];
        char password[50];

        int i, j, k;
        char ch;
        FILE *fp, *fu;
        struct pass u1;
        struct userpass u2;

        // Opening file of
        // user data
        fp = fopen("username.txt",
                        "rb");

        if (fp == NULL) {
                printf("ERROR IN OPENING FILE");
        }
        gotoxy(34, 2);
        printf(" ACCOUNT LOGIN ");
        gotoxy(7, 5);
        printf("*****************"
                "************");

        gotoxy(35, 10);
        printf("==== LOG IN ====");

        // Take input
        gotoxy(35, 12);
        printf("USERNAME.. ");
        scanf("%s", &username);

        gotoxy(35, 14);
        printf("PASSWORD..");

        // Input the password
        for (i = 0; i < 50; i++) {
                ch = getch();
                if (ch != 13) {
                        password[i] = ch;
                        ch = '*';
                        printf("%c", ch);
                }

                else
                        break;
        }

        // Checking if username
        // exists in the file or not
        while (fread(&u1, sizeof(u1),
                                1, fp)) {
                if (strcmp(username,
                                u1.username)
                        == 0) {
                        loginsu();
                        display(username);
                }
        }

        // Closing the file
        fclose(fp);
}

// Redirect after
// successful login
void loginsu(void)
{
        int i;
        FILE* fp;
        struct pass u1;
        system("cls");
        printf("Fetching account details.....\n");
        for (i = 0; i < 20000; i++) {
                i++;
                i--;
        }

        gotoxy(30, 10);
        printf("LOGIN SUCCESSFUL....");
        gotoxy(0, 20);
        printf("Press enter to continue");

        getch();
}

// Display function to show the
// data of the user on screen
void display(char username1[])
{
```

```
system("cls");
FILE* fp;
int choice, i;
fp = fopen("username.txt", "rb");
struct pass u1;

if (fp == NULL) {
        printf("error in opening file");
}

while (fread(&u1, sizeof(u1),
                        1, fp)) {
        if (strcmp(username1,
                        u1.username)
                == 0) {
                gotoxy(30, 1);
                printf("WELCOME, %s %s",
                        u1.fname, u1.lname);
                gotoxy(28, 2);
                printf("..........................");
                gotoxy(55, 6);
                printf("==== YOUR ACCOUNT INFO ====");
                gotoxy(55, 8);
                printf("*********");
                gotoxy(55, 10);
                printf("NAME..%s %s", u1.fname,
                        u1.lname);

                gotoxy(55, 12);
                printf("FATHER's NAME..%s %s",
                        u1.fathname,
                        u1.lname);

                gotoxy(55, 14);
                printf("MOTHER's NAME..%s",
                        u1.mothname);

                gotoxy(55, 16);
                printf("ADHAR CARD NUMBER..%s",
                        u1.adharnum);

                gotoxy(55, 18);
                printf("MOBILE NUMBER..%s",
                        u1.pnumber);

                gotoxy(55, 20);
                printf("DATE OF BIRTH.. %d-%d-%d",
                        u1.date, u1.month, u1.year);

                gotoxy(55, 22);
                printf("ADDRESS..%s", u1.address);
```

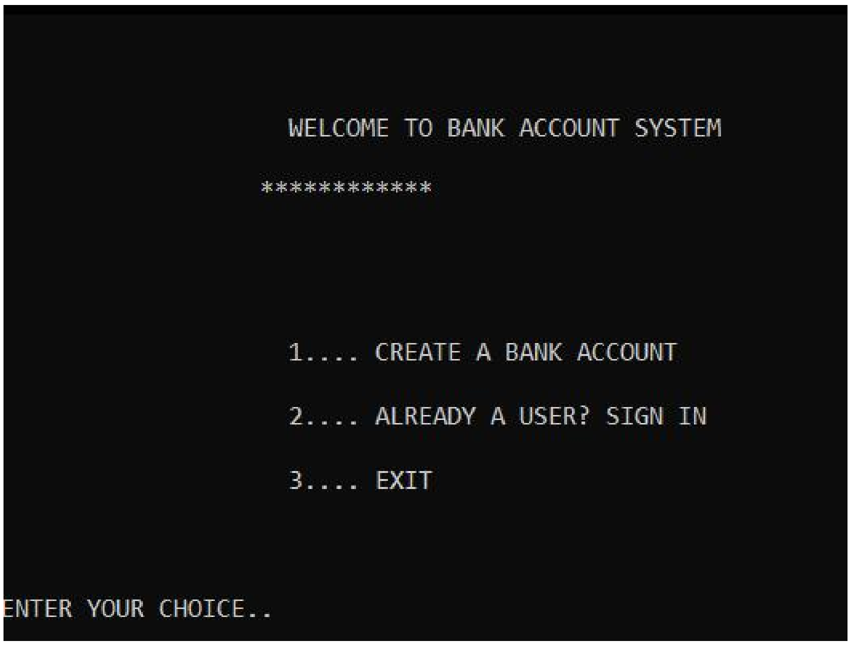
```
                    gotoxy(55, 24);
                    printf("ACCOUNT TYPE..%s",
                              u1.typeaccount);
          }
     }

     fclose(fp);

     gotoxy(0, 6);

     // Menu to perform different
     // actions by user
     printf(" HOME ");
     gotoxy(0, 7);
     printf("**");
     gotoxy(0, 9);
     printf(" 1....CHECK BALANCE");
     gotoxy(0, 11);
     printf(" 2....TRANSFER MONEY");
     gotoxy(0, 13);
     printf(" 3....LOG OUT\n\n");
     gotoxy(0, 15);
     printf(" 4....EXIT\n\n");

     printf(" ENTER YOUR CHOICES..");
     scanf("%d", &choice);

     switch (choice) {
     case 1:
          checkbalance(username1);
          break;

     case 2:
          transfermoney();
          break;

     case 3:
          logout();
          login();
          break;

     case 4:
          exit(0);
          break;
     }
}

// Function to transfer
// money from one user to
// another
void transfermoney(void)
{
```

```
int i, j;
FILE *fm, *fp;
struct pass u1;
struct money m1;
char usernamet[20];
char usernamep[20];
system("cls");

// Opening file in read mode to
// read user's username
fp = fopen("username.txt", "rb");

// Creating a another file
// to write amount along with
// username to which amount
// is going to be transferred
fm = fopen("mon.txt", "ab");

gotoxy(33, 4);
printf("---- TRANSFER MONEY ----");
gotoxy(33, 5);
printf("========================");

gotoxy(33, 11);
printf("FROM (your username).. ");
scanf("%s", &usernamet);

gotoxy(33, 13);
printf(" TO (username of person)..");
scanf("%s", &usernamep);

// Checking for username if it
// is present in file or not
while (fread(&u1, sizeof(u1),
             1, fp))

{
    if (strcmp(usernamep,
               u1.username)
        == 0) {
        strcpy(m1.usernameto,
               u1.username);
        strcpy(m1.userpersonfrom,
               usernamet);
    }
}
gotoxy(33, 16);

// Taking amount input
printf("ENTER THE AMOUNT TO BE TRANSFERRED..");
scanf("%d", &m1.money1);
```

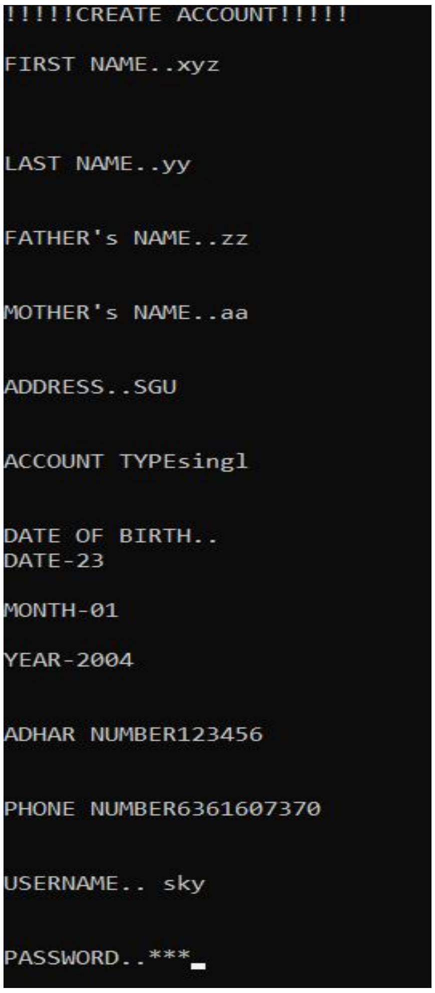
```
        // Writing to the file
        fwrite(&m1, sizeof(m1),
                1, fm);

        gotoxy(0, 26);
        printf(
                "--------------------------------------------------"
                "--------------------------------------------");

        gotoxy(0, 28);
        printf(
                "--------------------------------------------------"
                "--------------------------------------------");

        gotoxy(0, 29);
        printf("transferring amount, Please wait..");

        gotoxy(10, 27);
        for (i = 0; i < 70; i++) {
                for (j = 0; j < 1200000; j++) {
                        j++;
                        j--;
                }
                printf("*");
        }

        gotoxy(33, 40);
        printf("AMOUNT SUCCESSFULLY TRANSFERRED....");
        getch();

        // Close the files
        fclose(fp);
        fclose(fm);

        // Function to return
        // to the home screen
        display(usernamet);
}

// Function to check balance
// in users account
void checkbalance(char username2[])
{
        system("cls");
        FILE* fm;
        struct money m1;
        char ch;
        int i = 1, summoney = 0;

        // Opening amount file record
        fm = fopen("mon.txt", "rb");
        int k = 5, l = 10;
        int m = 30, n = 10;
        int u = 60, v = 10;

        gotoxy(30, 2);
        printf("==== BALANCE DASHBOARD ====");
        gotoxy(30, 3);
        printf("*********");
        gotoxy(k, l);
        printf("S no.");
        gotoxy(m, n);
        printf("TRANSACTION ID");
        gotoxy(u, v);
        printf("AMOUNT");

        // Reading username to
        // fetch the correct record
        while (fread(&m1, sizeof(m1),
                        1, fm)) {
                if (strcmp(username2,
                        m1.usernameto)
                        == 0) {
                        gotoxy(k, ++l);
                        printf("%d", i);
                        i++;
                        gotoxy(m, ++n);
                        printf("%s", m1.userpersonfrom);

                        gotoxy(u, ++v);
                        printf("%d", m1.money1);
                        // Adding and
                        // finding total money
                        summoney = summoney + m1.money1;
                }
        }

        gotoxy(80, 10);
        printf("TOTAL AMOUNT");

        gotoxy(80, 12);
        printf("%d", summoney);

        getch();

        // Closing file after
        // reading it
        fclose(fm);
        display(username2);
}

// Logout function to bring
// user to the login screen
```

```
void logout(void)
{
        int i, j;
        system("cls");
        printf("please wait, logging out");

        for (i = 0; i < 10; i++) {
                for (j = 0; j < 25000000; j++) {
                        i++;
                        i--;
                }
                printf(".");
        }

        gotoxy(30, 10);
        printf("Sign out successfully..\n");

        gotoxy(0, 20);
        printf("press any key to continue..");

        getch();
}
```

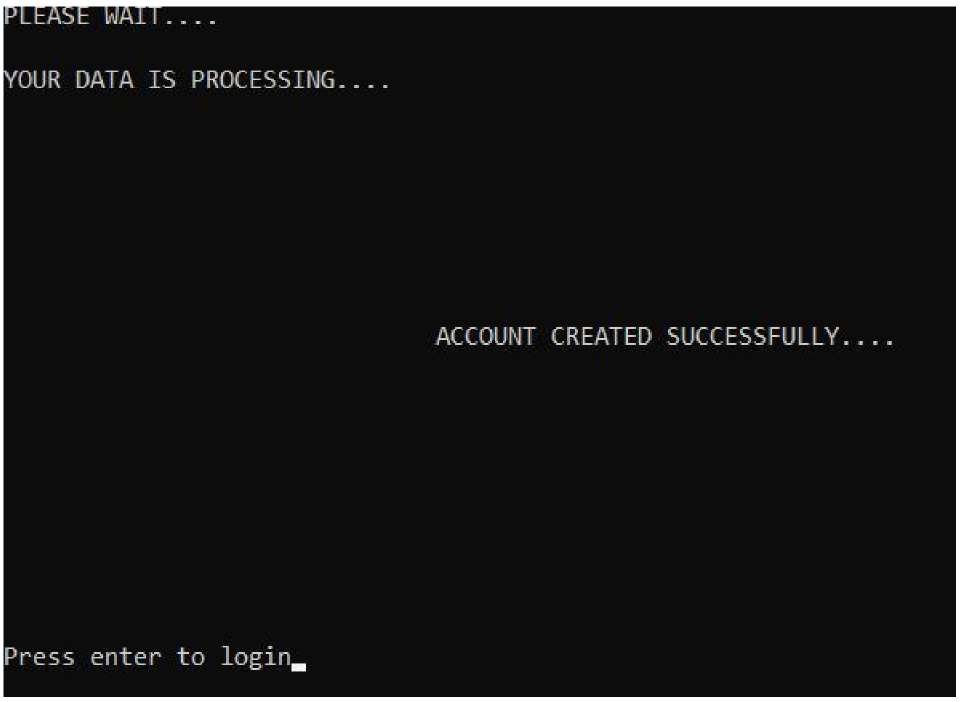
Result



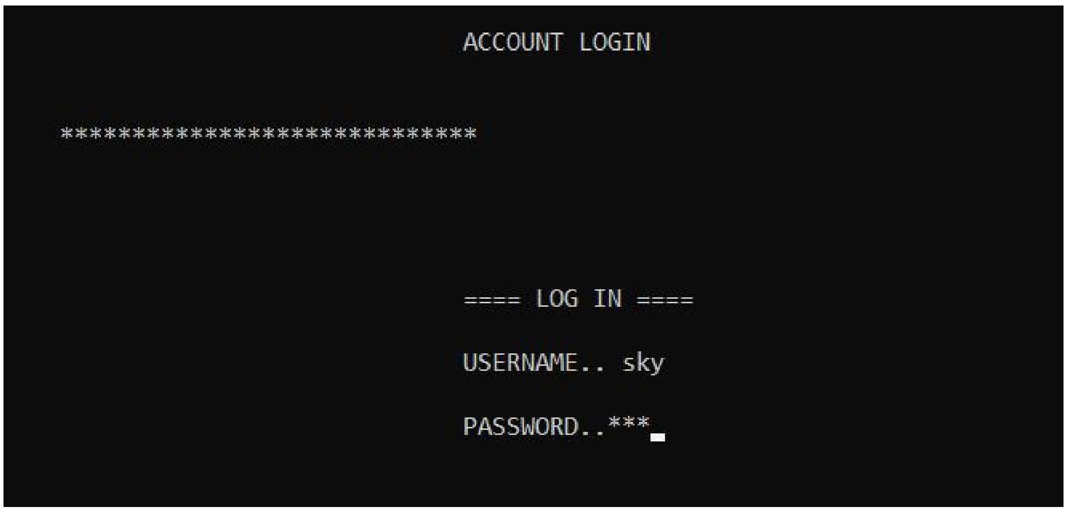
```
                    WELCOME TO BANK ACCOUNT SYSTEM

                  ************



                    1.... CREATE A BANK ACCOUNT

                    2.... ALREADY A USER? SIGN IN

                    3.... EXIT


ENTER YOUR CHOICE..
```



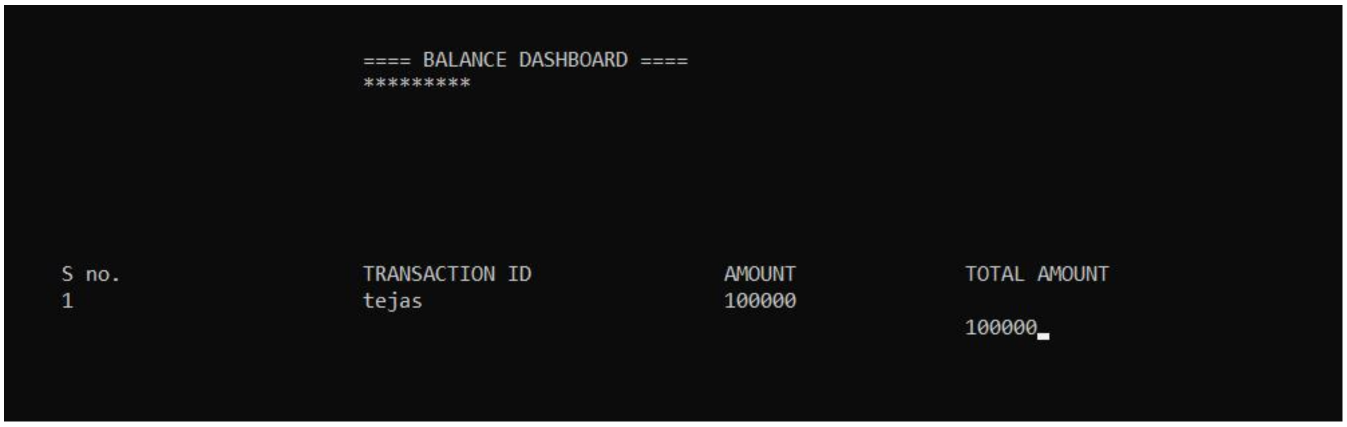
```
!!!!!CREATE ACCOUNT!!!!!

FIRST NAME..xyz


LAST NAME..yy


FATHER's NAME..zz


MOTHER's NAME..aa


ADDRESS..SGU


ACCOUNT TYPEsingl


DATE OF BIRTH..
DATE-23

MONTH-01

YEAR-2004


ADHAR NUMBER123456


PHONE NUMBER6361607370


USERNAME.. sky


PASSWORD..***
```



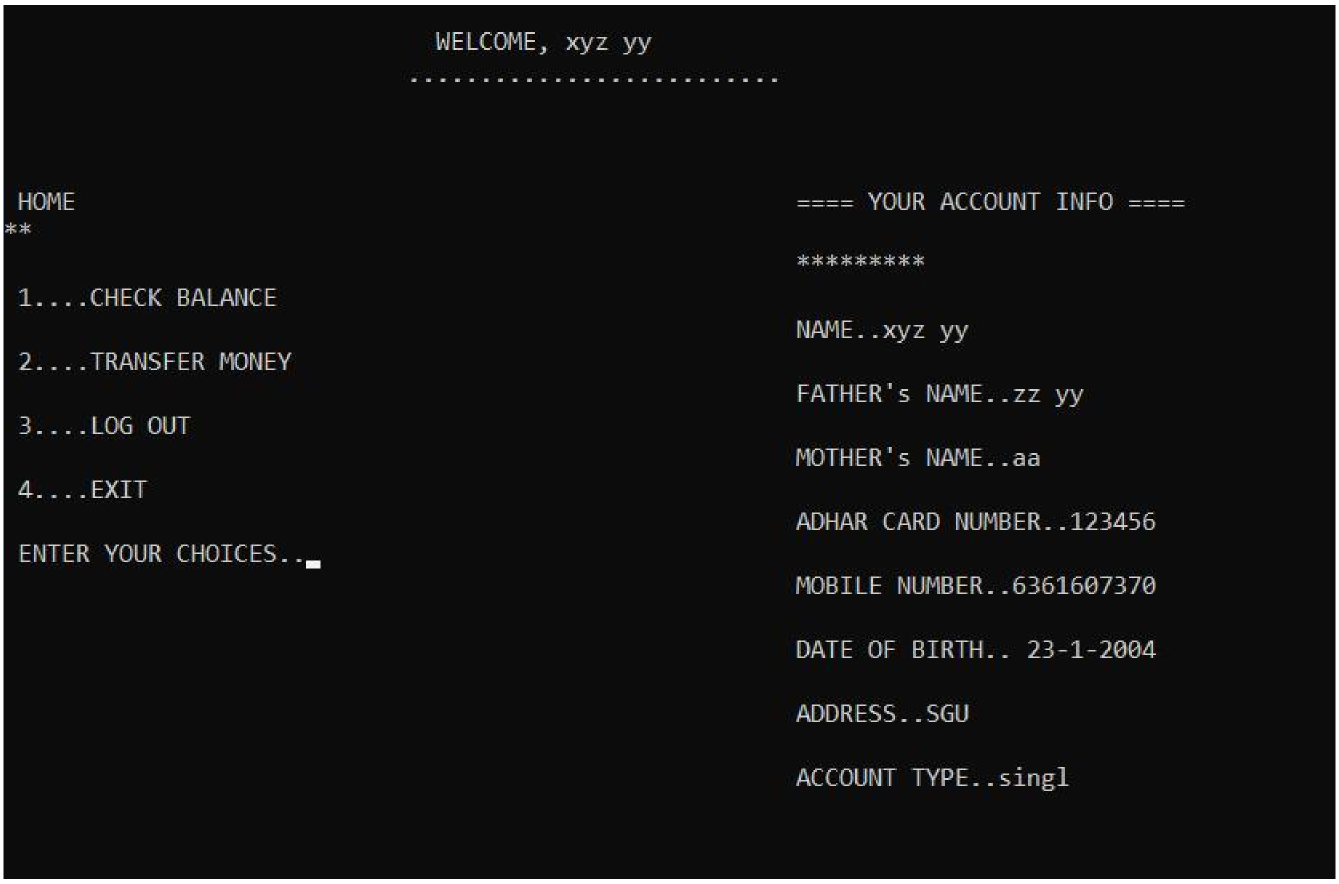
```
PLEASE WAIT....

YOUR DATA IS PROCESSING....




                              ACCOUNT CREATED SUCCESSFULLY....




Press enter to login
```



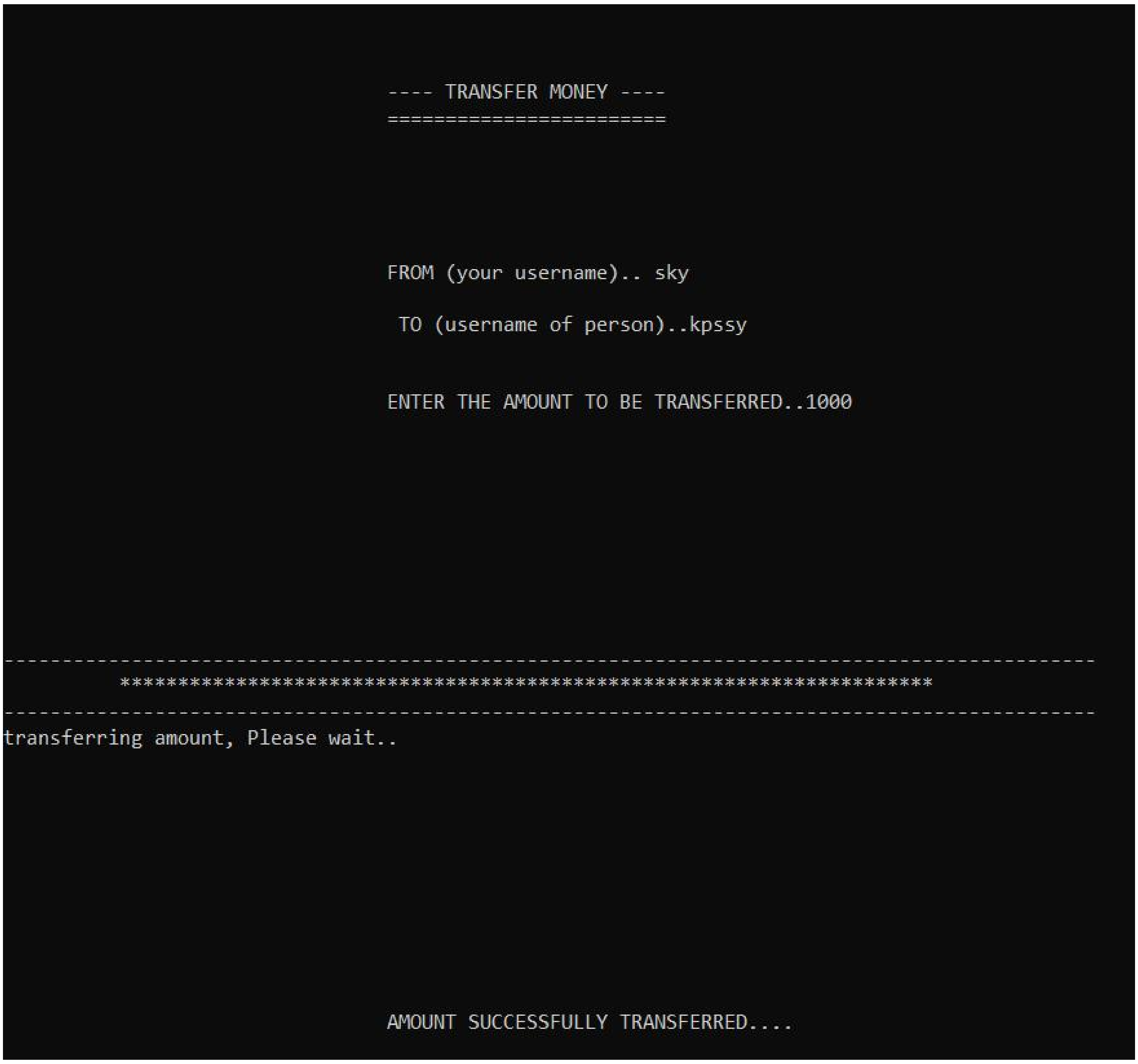
```
                                ACCOUNT LOGIN

 *****************************



                                ==== LOG IN ====

                                USERNAME.. sky

                                PASSWORD..***
```



```
                    ==== BALANCE DASHBOARD ====
                    *********



S no.               TRANSACTION ID          AMOUNT          TOTAL AMOUNT
1                   tejas                   100000
                                                            100000
```



```
                              WELCOME, xyz yy
                            ..........................


 HOME                                                        ==== YOUR ACCOUNT INFO ====
**
                                                             *********
 1....CHECK BALANCE
                                                             NAME..xyz yy
 2....TRANSFER MONEY
                                                             FATHER's NAME..zz yy
 3....LOG OUT
                                                             MOTHER's NAME..aa
 4....EXIT
                                                             ADHAR CARD NUMBER..123456
 ENTER YOUR CHOICES..
                                                             MOBILE NUMBER..6361607370

                                                             DATE OF BIRTH.. 23-1-2004

                                                             ADDRESS..SGU

                                                             ACCOUNT TYPE..singl
```



```
                                  ---- TRANSFER MONEY ----
                                  ========================




                                  FROM (your username).. sky

                                   TO (username of person)..kpssy


                                  ENTER THE AMOUNT TO BE TRANSFERRED..1000







-----------------------------------------------------------------------------------------------
          **********************************************************************
-----------------------------------------------------------------------------------------------
transferring amount, Please wait..






                                  AMOUNT SUCCESSFULLY TRANSFERRED....
```

Conclusion & Future Scope

- **Future Scope:**

  1. Creating New Accounts- The application can be used to create two different types of accounts by the customers, which are Savings Account and Current Account.
  2. Depositing Money- As the world is moving towards the limited use of paper currency, depositing or transferring money from one bank to the other will become as easy as clicking a few buttons using this application.
  3. Withdrawing Money- Requests can be sent through the application to ask for money transfer as well.
  4. Account Holder List- This is a feature for the admin. The admin can view the list of all the account holders.

- **Conclusion:**

Bank management system is a virtualization of transactions in banking system. The banking system are used manual working but when we used online banking system it is totally virtualization process which avoid manual process and converts it in automatic process .