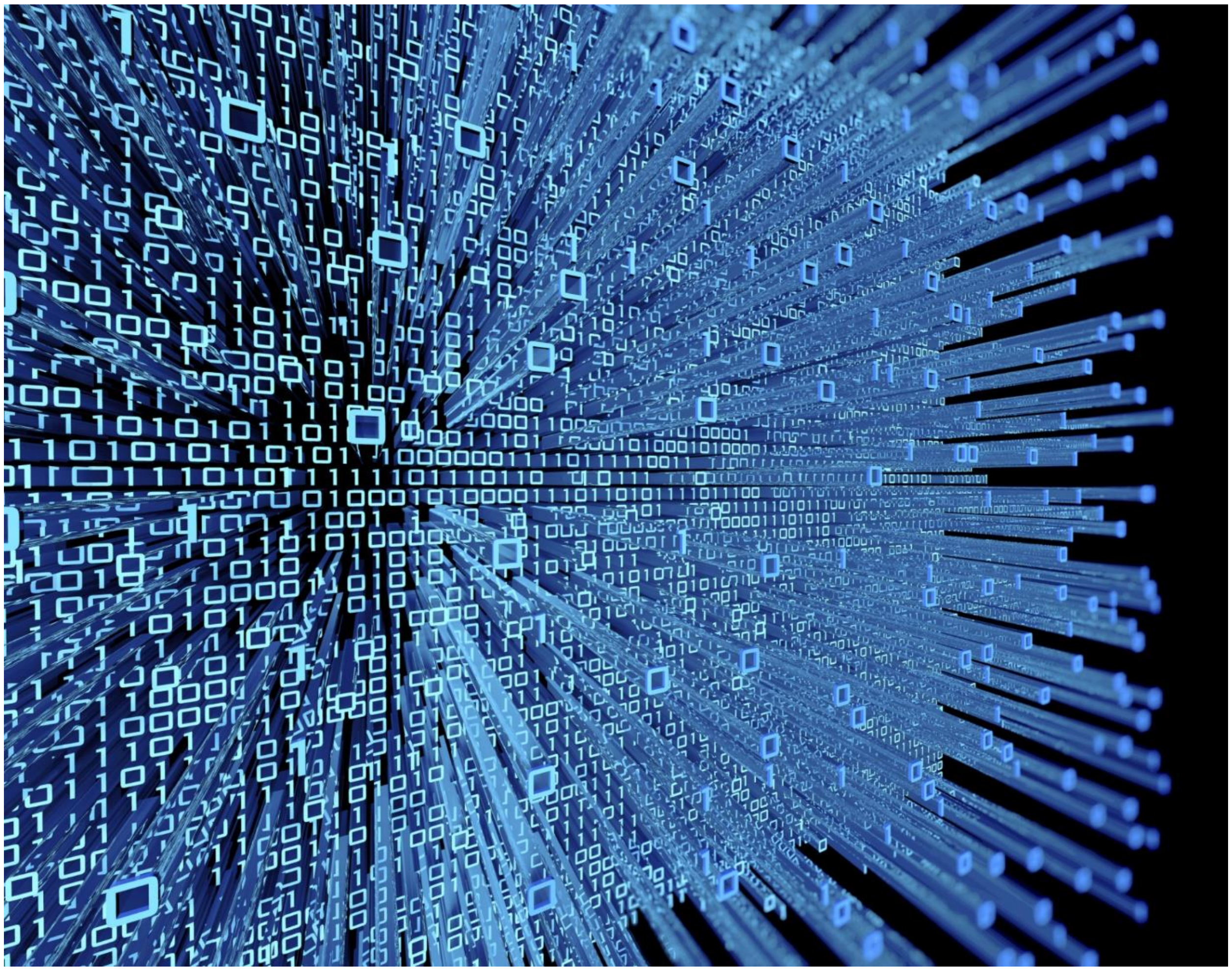
Computer Science Unit 2

School Based Assessment (SBA)



Title: A Banking System for SPCCU of Montserrat

Candidates: Denzil Queeley, Maxwell Ross, Jenalyn Weekes

Centre Number: 110201

Candidate Numbers:

School: Montserrat Community College

Lecturer: Mr. T Jarvis

Table of Contents

Definition of the Problem

The developers of this program would like to propose to the St. Patrick's Cooperative Credit Union (SPCCU) of Montserrat, an improvement to their current version of their banking system. Presently there are several problems with their current system which we see can be improved. These problems include: security of the system, time efficiency and general banking services.

Problems with the current system:

- The SPCCU customers and staff are requesting for a more secure system to prevent mishaps. The customers would like a system that protects their sensitive information such as their bank account number, the total amount of money in their account etc. Also, to prevent other co-workers and unauthorized persons from using their account for fraud-related activities, the staff would like to have usernames and passwords to login to the system.
- The SPCCU customers are complaining about the time in which it takes to do simple transactions at the bank, especially when queues are long. The SPCCU staff are also saying it is very tedious and time consuming to determine the eligibility of a customer to acquire a loan. In addition, finding specific customer documents for updating of accounts, loans and general customer information is an added challenge.
- General banking services such as creation and deletion of bank accounts, withdrawals, deposits and loan requests are still done manually and are prone to human errors.

Overall, the implementation of this program will help solve these problems as well as giving the bank a more efficient system to use.

<u>Data Collection & Analysis</u>

In order to begin the process of checking the project's feasibility, the developers were granted permission from the manager to conduct a series of planned observations and reviewing of internal documents at the SPCCU.

Observations were conducted on both staff and customers while the reviewing of internal documents were used to better develop a more user-familiar system to reduce the time taken to adapt to it.

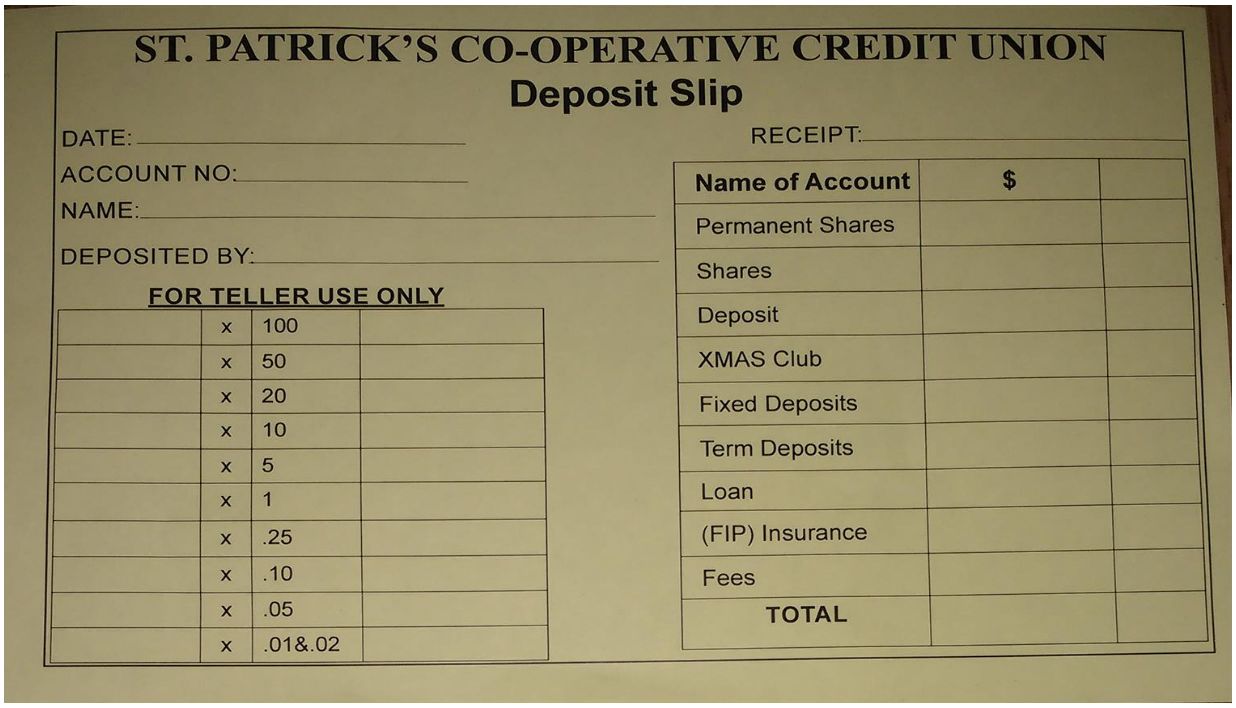
A series of observations were carried out on four (4) separate occasions at the SPCCU with each alternating between the morning and afternoon. The results from these observations were however non-varying and repetitive.

The first and second observation was conducted in the customer area at 8:30am and at 1:00pm on the 8th of December 2017 respectively. During that time, a record was made of how long it took a bank clerk to deal with a customer. From our findings it can be stated that on average it takes about 4 minutes for a customer to be dealt with. This is too long for any business organisation which deals with clients.

According to Investopedia.com, "Automation in banks speeds up the processing time in handling loan applications, because paperwork is reduced. It also permits faster look-up of customer account information and improves the banking service.". Therefore, by implementing our program, the bank can benefit from automation.

On the third and fourth occasion, the observation was carried out in the teller area on the 11th of December 2017 during the morning and afternoon periods respectively. It was observed that bank clerks left their work station unattended, which allows other co-workers to easily access it. This leaves their work station open to malicious acts since there is currently no secured way of properly protecting sensitive bank and customer information.

After these observations were conducted, the developers of this program proceeded to make full use of the opportunity to inspect the non-confidential internal documents which are used by the bank. In addition, the documents were inspected to determine what fields are required to be provided by the system. Below are the deposit and withdrawal documents used.



ST. PATRICK'S CO-OPERATIVE CREDIT UNION

Deposit Slip

DATE:

RECEIPT:

ACCOUNT NO:

NAME:

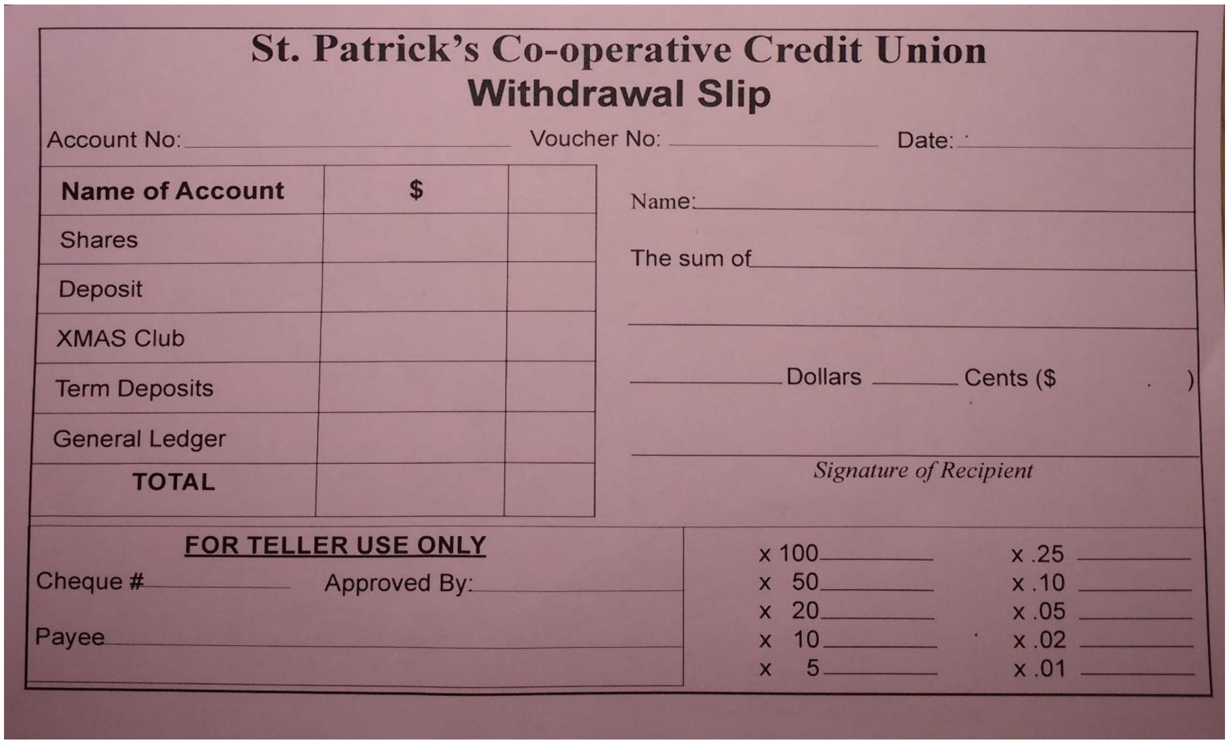
DEPOSITED BY:

FOR TELLER USE ONLY

	x	100	
	x	50	
	x	20	
	x	10	
	x	5	
	x	1	
	x	.25	
	x	.10	
	x	.05	
	x	.01&.02	

Name of Account	$	
Permanent Shares		
Shares		
Deposit		
XMAS Club		
Fixed Deposits		
Term Deposits		
Loan		
(FIP) Insurance		
Fees		
TOTAL		

Figure 1: Showing Deposit Slip



St. Patrick's Co-operative Credit Union

Withdrawal Slip

Account No: Voucher No: Date:

Name of Account	$	
Shares		
Deposit		
XMAS Club		
Term Deposits		
General Ledger		
TOTAL		

Name:

The sum of

Dollars Cents ($.)

Signature of Recipient

FOR TELLER USE ONLY

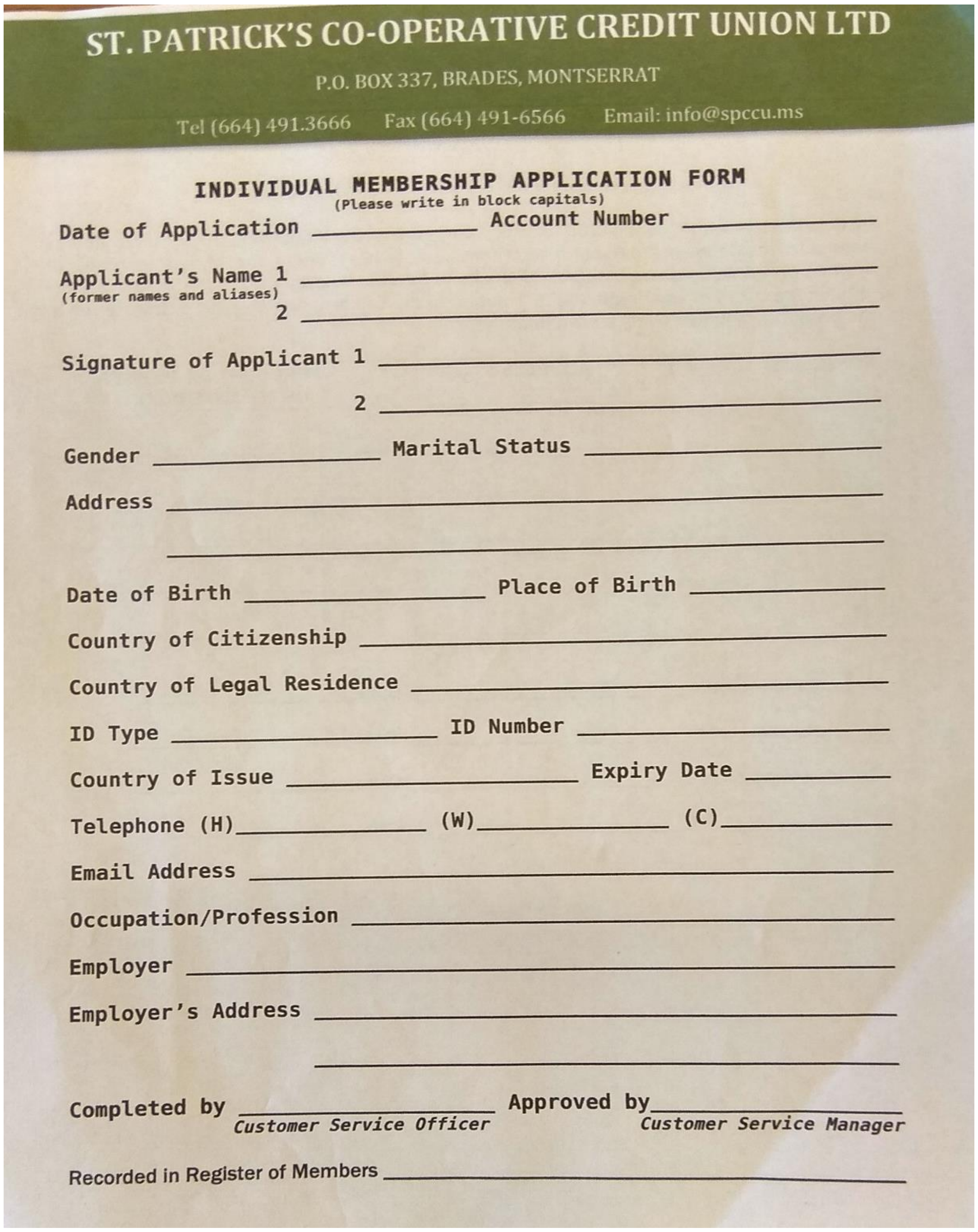
Cheque # Approved By:

Payee

x 100	x .25
x 50	x .10
x 20	x .05
x 10	x .02
x 5	x .01

Figure 2: Showing Withdrawal Slip

After thorough analysis of the above documents, bank tellers and customers both have a lot of data to record manually for simple task such as deposits and withdrawals. This results in long queues due to the time it takes to deal with a customer. Also creation of new accounts is time consuming as new customers will have to manually complete a lengthy form as shown below.



ST. PATRICK'S CO-OPERATIVE CREDIT UNION LTD

P.O. BOX 337, BRADES, MONTSERRAT

Tel (664) 491.3666 Fax (664) 491-6566 Email: info@spccu.ms

INDIVIDUAL MEMBERSHIP APPLICATION FORM

(Please write in block capitals)

Date of Application ____________ Account Number ____________

Applicant's Name 1 ____________
(former names and aliases)
2 ____________

Signature of Applicant 1 ____________
2 ____________

Gender ____________ Marital Status ____________

Address ____________
____

Date of Birth ____________ Place of Birth ____________

Country of Citizenship ____________

Country of Legal Residence ____________

ID Type ____________ ID Number ____________

Country of Issue ____________ Expiry Date ____________

Telephone (H)____________ (W)____________ (C)____________

Email Address ____________

Occupation/Profession ____________

Employer ____________

Employer's Address ____________
____

Completed by ____________ Approved by____________
Customer Service Officer Customer Service Manager

Recorded in Register of Members ____________

Figure 3: Showing Membership Form

Context Level Diagram

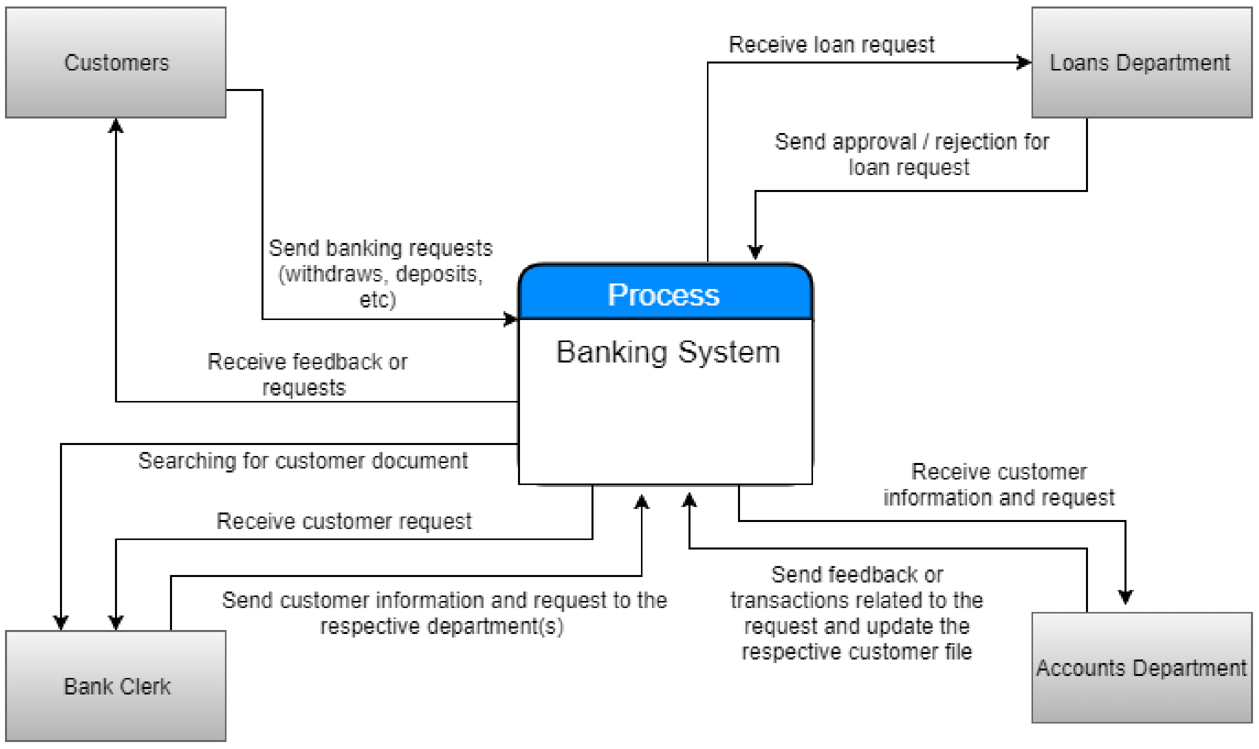

Customers
Loans Department
Receive loan request
Send approval / rejection for loan request
Send banking requests (withdraws, deposits, etc)
Process
Banking System
Receive feedback or requests
Searching for customer document
Receive customer request
Send customer information and request to the respective department(s)
Receive customer information and request
Send feedback or transactions related to the request and update the respective customer file
Bank Clerk
Accounts Department

Figure 4: Showing a Context Level Data Flow Diagram for the SPCCU Banking System. This shows the general banking system process and the various entities connected to it.

Level 1 Data Flow Diagram

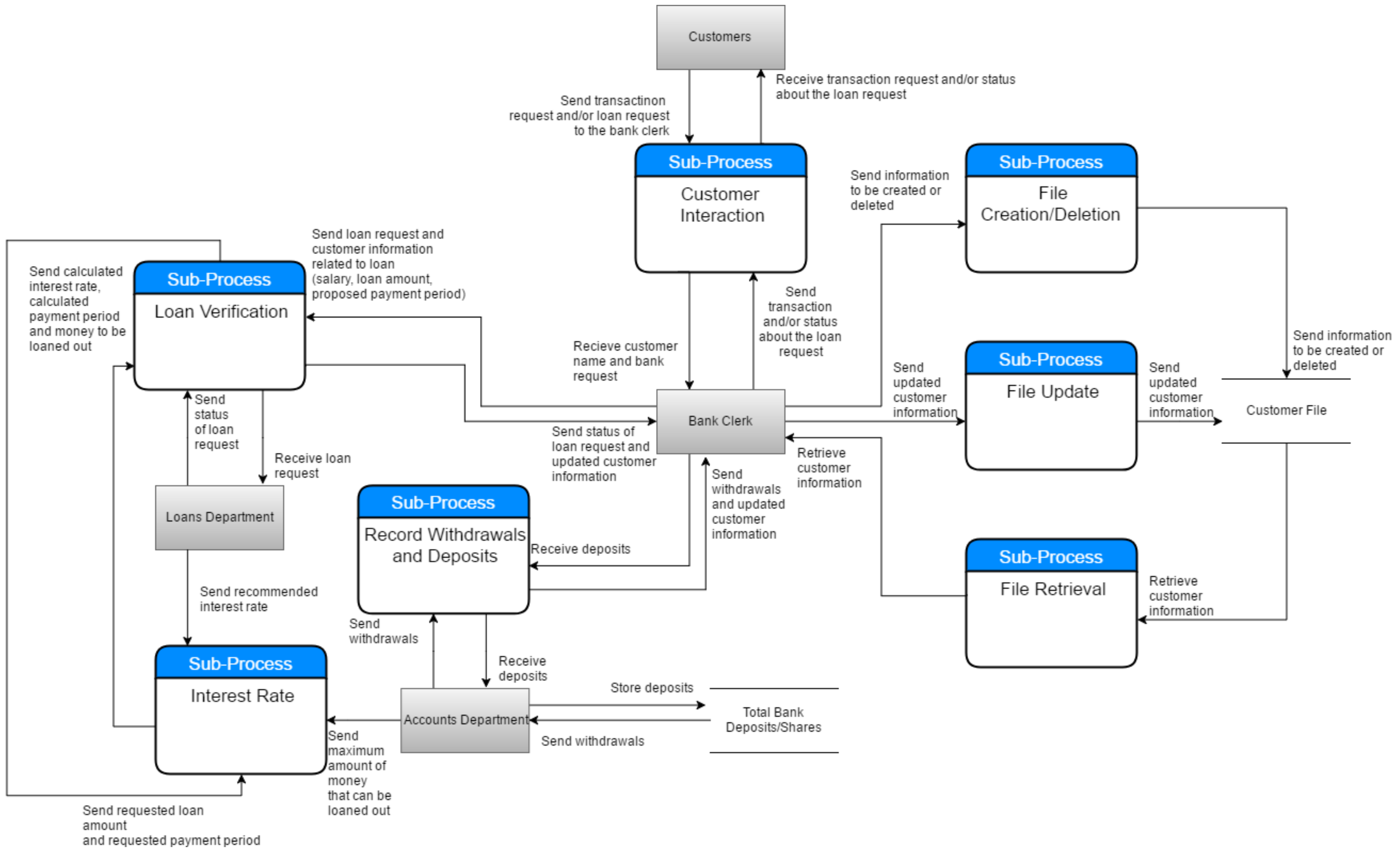

Customers
Send transactinon request and/or loan request to the bank clerk
Receive transaction request and/or status about the loan request
Sub-Process
Customer Interaction
Send information to be created or deleted
Sub-Process
File Creation/Deletion
Send calculated interest rate, calculated payment period and money to be loaned out
Sub-Process
Loan Verification
Send loan request and customer information related to loan (salary, loan amount, proposed payment period)
Send transaction and/or status about the loan request
Recieve customer name and bank request
Send information to be created or deleted
Send updated customer information
Sub-Process
File Update
Send updated customer information
Customer File
Bank Clerk
Send status of loan request
Receive loan request
Send status of loan request and updated customer information
Retrieve customer information
Send withdrawals and updated customer information
Loans Department
Sub-Process
Record Withdrawals and Deposits
Receive deposits
Sub-Process
File Retrieval
Retrieve customer information
Send recommended interest rate
Send withdrawals
Sub-Process
Interest Rate
Receive deposits
Store deposits
Accounts Department
Total Bank Deposits/Shares
Send withdrawals
Send maximum amount of money that can be loaned out
Send requested loan amount and requested payment period

Figure 5: An in-depth view of the banking system process. The banking system is broken down into sub processes which are connected to the various entities. There is also the inclusion of data storages (files) which shows the information being saved by the banking.

Entity Relationship Diagram

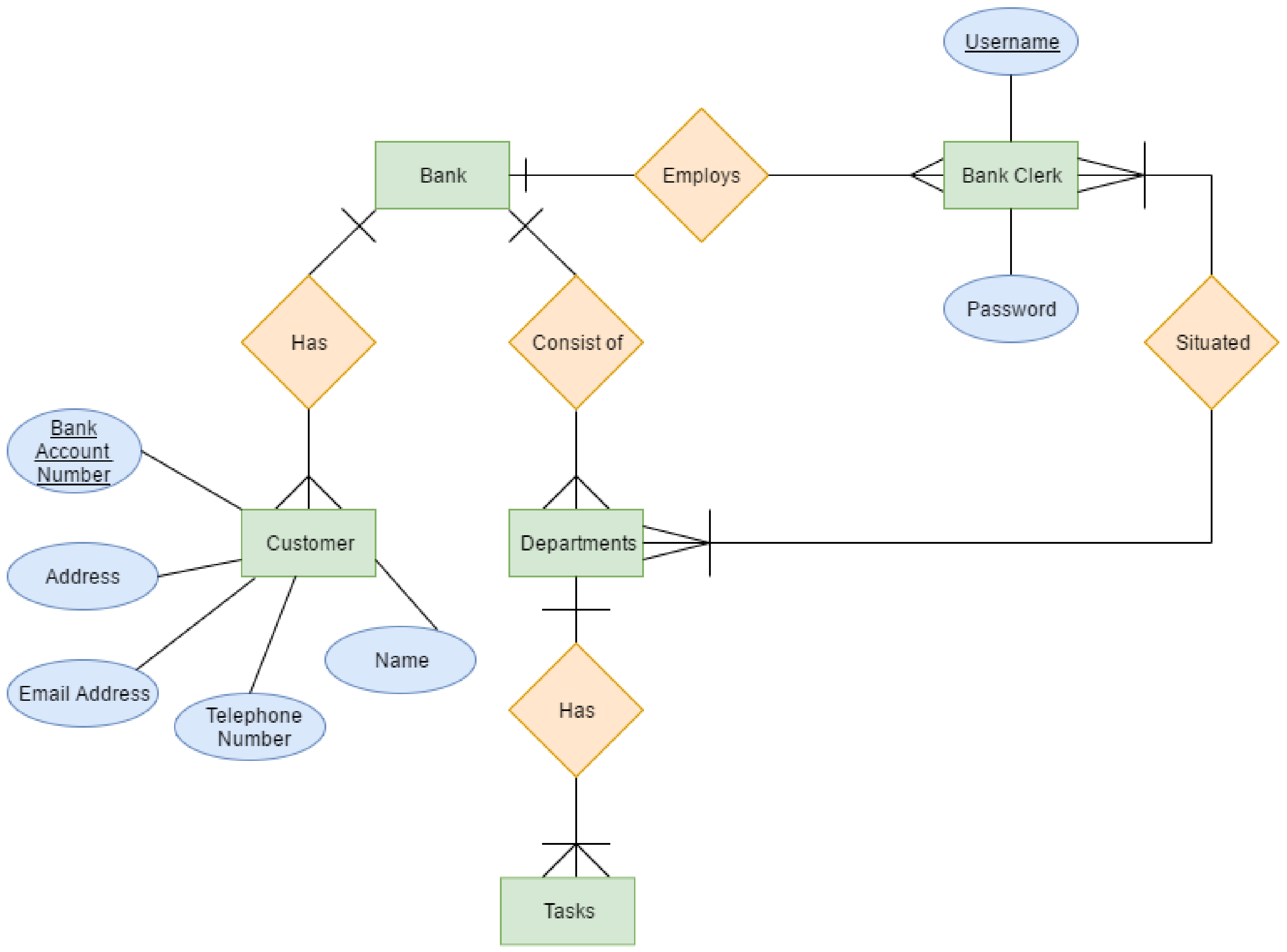


Figure 6: The entity relationship diagram of the banking system. The cardinality relationship of the entities within the banking system, as well as attributes (or characteristics) of entities are shown in this diagram.

Functional and Non-Functional Requirements

Functional Requirements

Transactions (Withdraws and Deposits) – The program must be able to handle the withdrawals and deposits of customers

Authentication login – The program must have a login capability that verifies the user and prevent unwanted access to their account.

File Processing – The program must be able to create, update and perform other operations on customer files.

Searching – The program must be able to find a file or information based on the entered key (E.g. customer name, bank account number) and retrieve information from the file.

Determining eligibility for a loan – The program must be able to determine if a customer meets the requirement to acquire a loan based on predefined criteria.

Non-Functional Requirements

Maintainability – Errors in the program must be easily detected and rectified

Dependability – The program must be able to precisely carry out its intended function over a predetermined period

Effectiveness – The program must be able to do the same task multiple times and consistently produce the same results.

Usability – The program must be efficient in delivering a set of specified results to the user in an efficient and satisfactory manner under specified conditions.

Extensibility – The program must be easily upgradeable based on the changing needs of the bank as well include additional feature

System Structuring

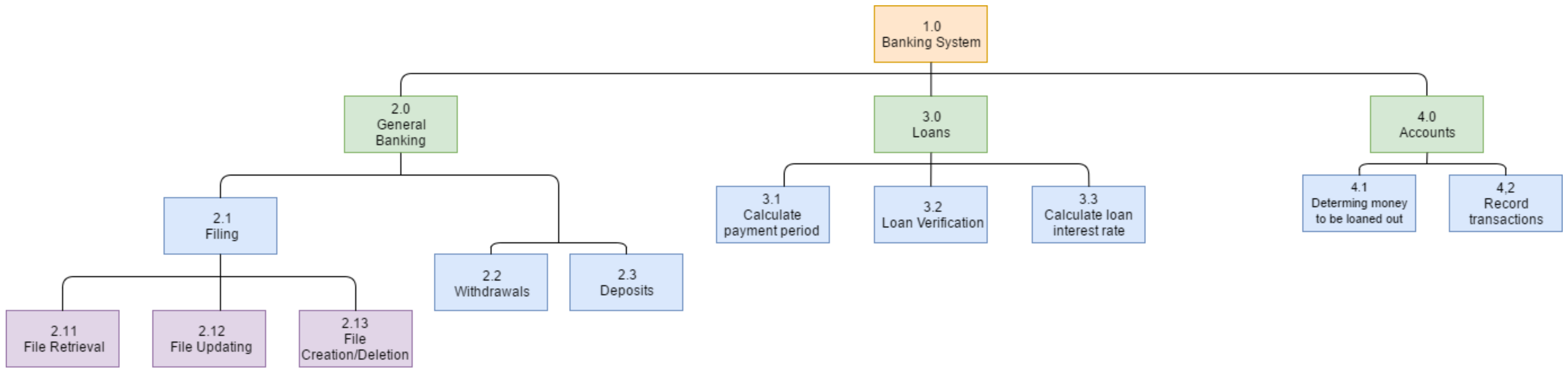


Figure 7: Showing a hierarchical input process output model of the banking system with its associated modules.

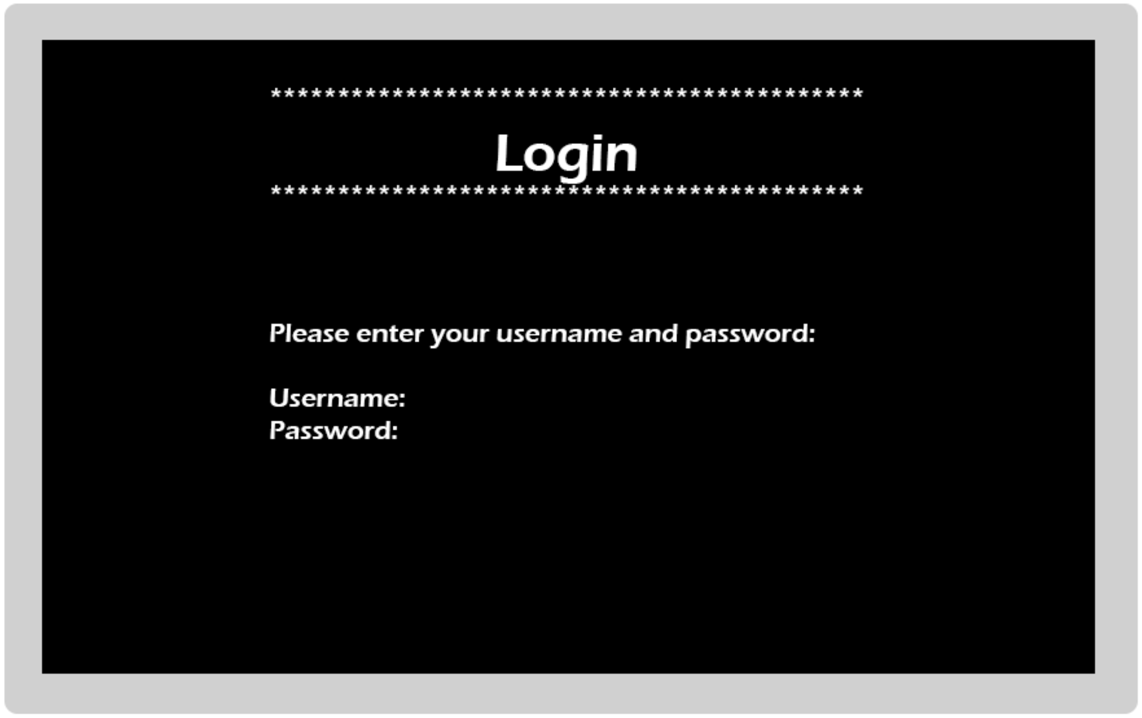
User Interface Design

The command line interface is a text-based user interface used to operate software and the computer's operating system. It allows the user to interact with the computer by using commands (instructions telling a computer to do something). The computer displays a prompt, the user keys in the command and presses enter to execute the command.

This type of interface needs much less memory (RAM) and CPU processing time in order to work. It also only needs a keyboard and a monitor to be operated. Therefore, an inexpensive computer system can be used as it uses less system resources than other interfaces.

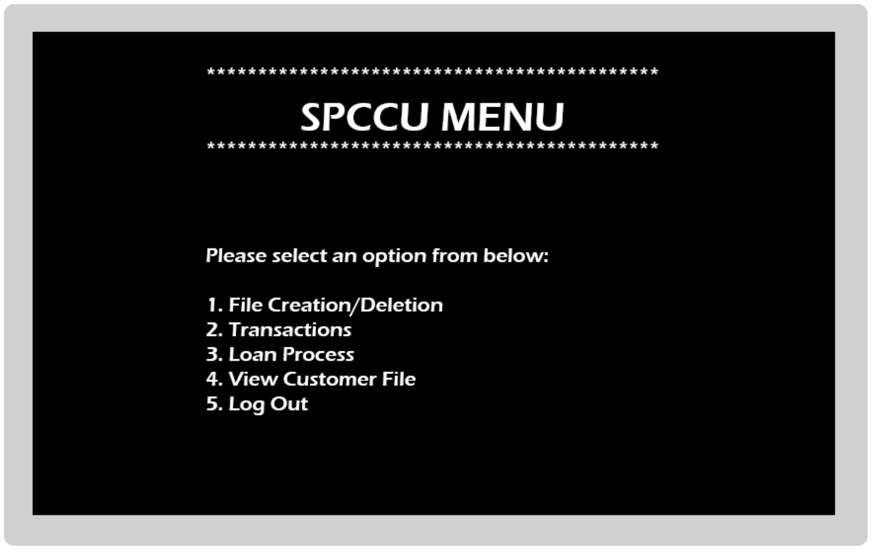
After the user has learnt how to navigate through the command line interface and use its commands, this type of interface can be much faster than using any other interface as a user can quickly key in commands and navigate through. Thus increasing the speed and efficiency of the user and decreasing the time taken to carry out a task.

It is more powerful than other interfaces as the user can simply enter a command and it is executed as soon as the user presses enter. Batch processing such as renaming or moving numerous files can also be quickly done with a command line interface.



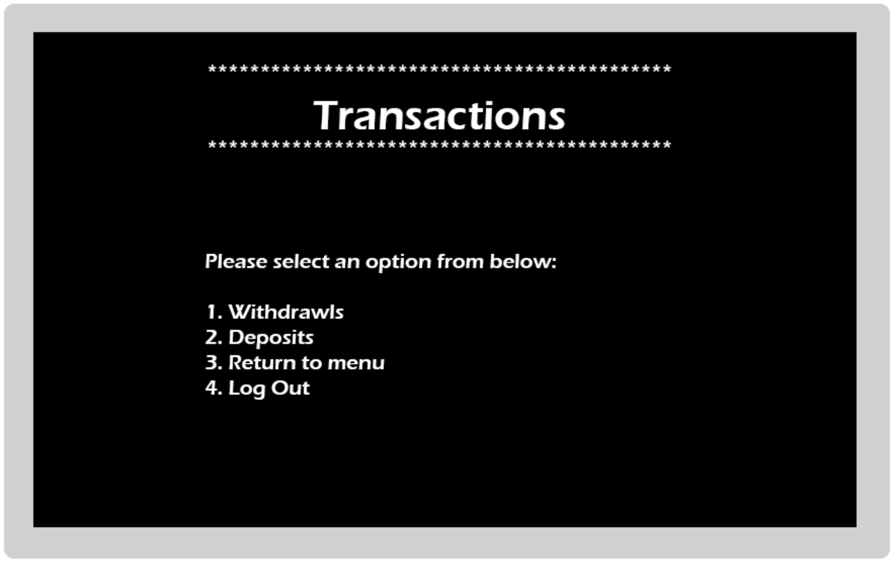
```
*********************************************
                 Login
*********************************************


Please enter your username and password:

Username:
Password:
```

Figure 8: Initial screen displayed to the user requesting that they enter a valid username and corresponding password. If this request is not fulfilled after three trials, the program will terminate itself.



```
*******************************************
SPCCU MENU
*******************************************


Please select an option from below:

1. File Creation/Deletion
2. Transactions
3. Loan Process
4. View Customer File
5. Log Out
```

Figure 9: Screen displayed after successful login giving user access to the menu. The menu displayed gives the user options to carry out any of the following: create or delete a file, conduct a transaction, perform loan process, display a customer file or log out.

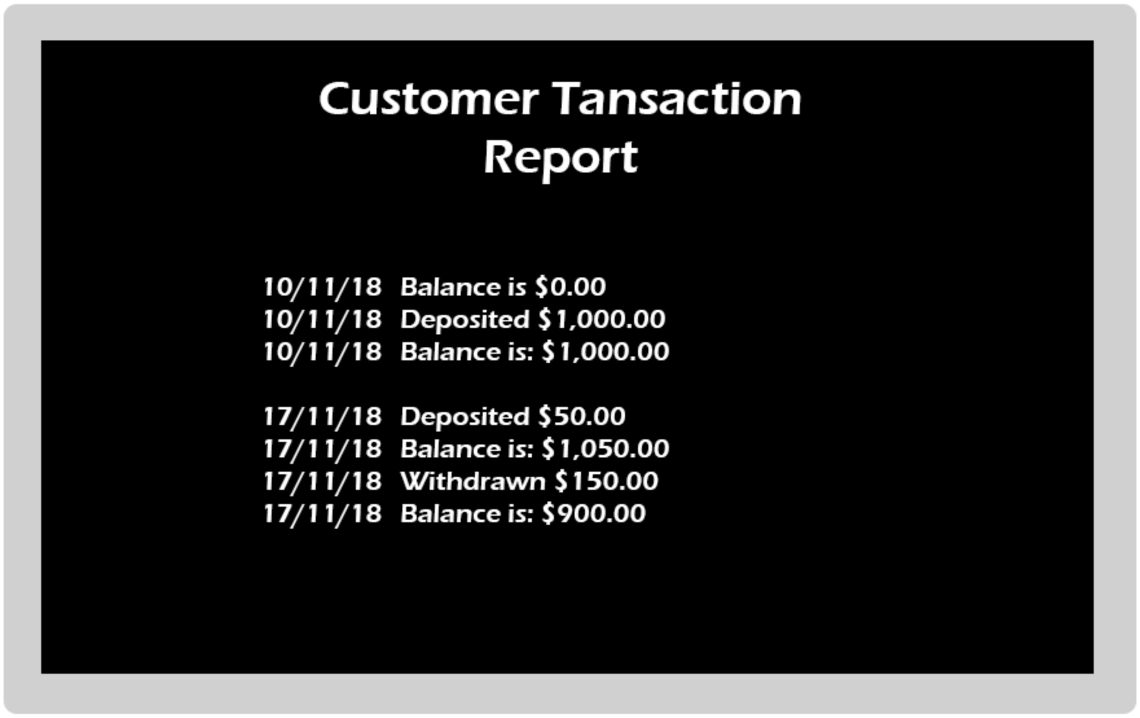


```
*******************************************
Transactions
*******************************************


Please select an option from below:

1. Withdrawls
2. Deposits
3. Return to menu
4. Log Out
```

Figure 10: Screen showcasing transaction options available: to withdraw from the user's account, or to conduct a deposit of funds into a user's account. Also giving options to return to the menu or log out form the user's account.

Report Design

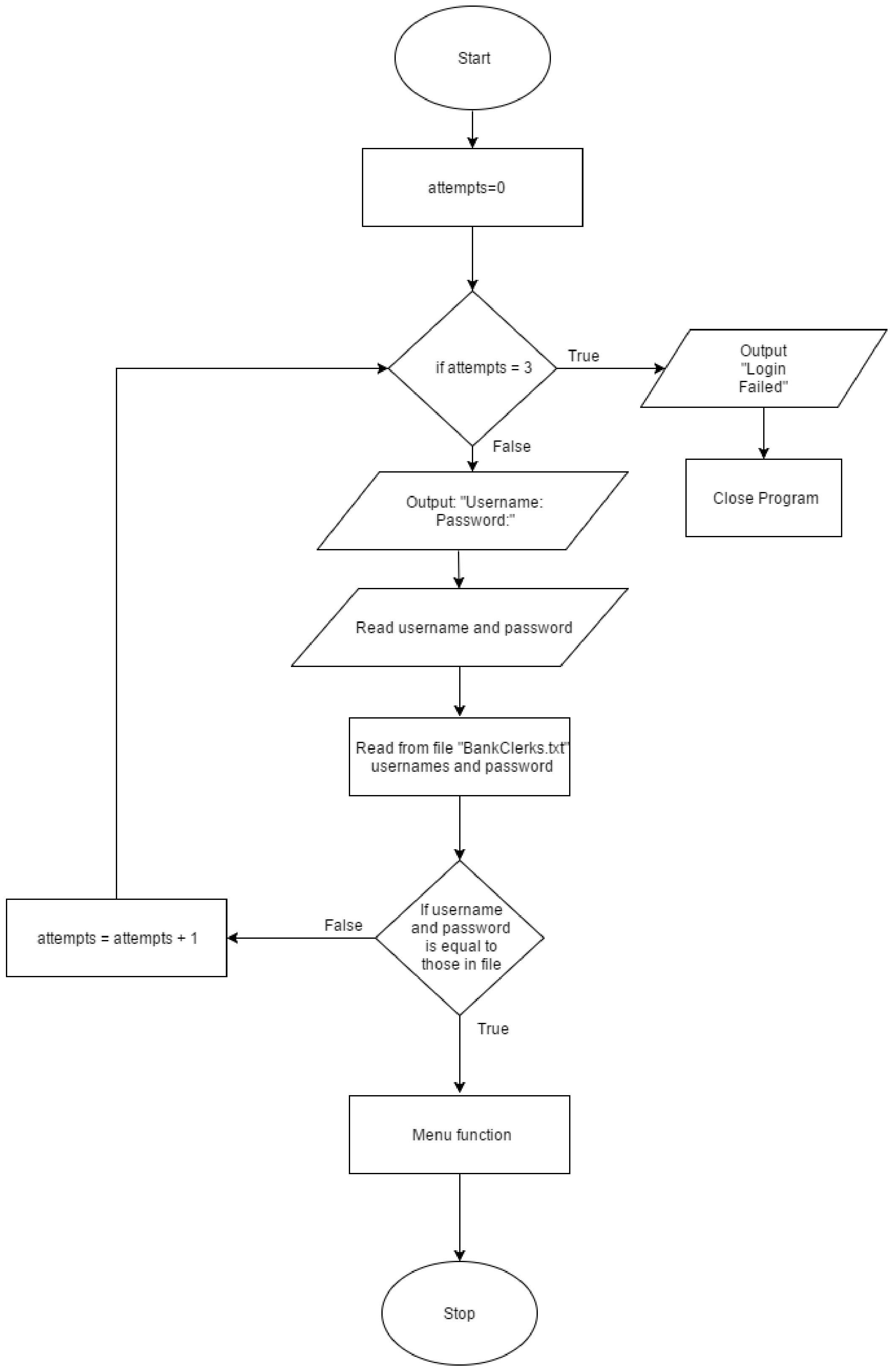
This shows the customer transaction report which contains all transactions done by the customer, the date at which time this transaction occurred and their balance after each consecutive transaction. The document is arranged in a tabular format without lines where the dates are in the left column, the transaction and balance in the middle and the amount of money in the right column. Spaces are inserted between each column to show the tabular layout. For each new date a space is also used to improve the document's readability.



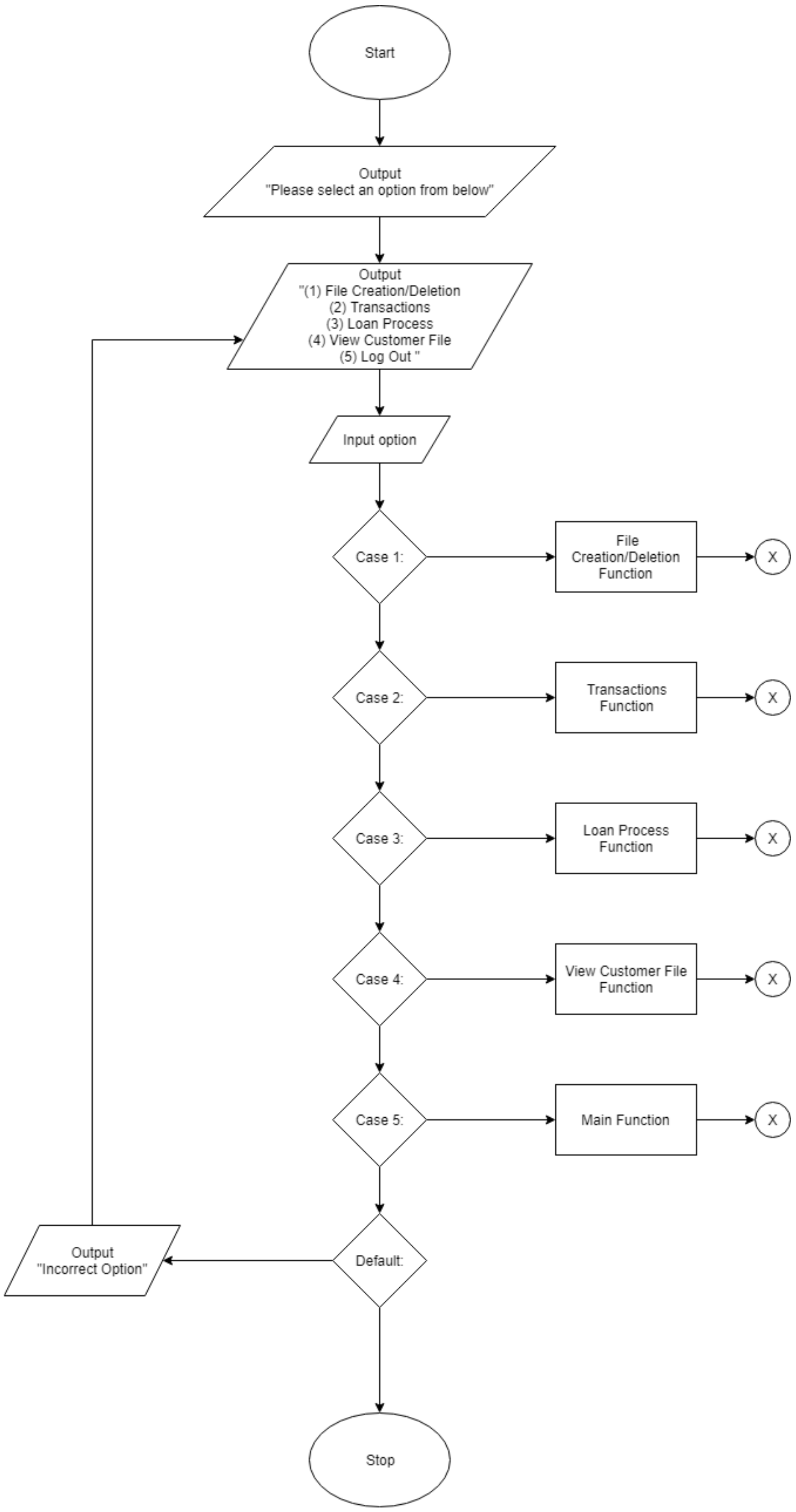
```
Customer Tansaction
Report

10/11/18  Balance is $0.00
10/11/18  Deposited $1,000.00
10/11/18  Balance is: $1,000.00

17/11/18  Deposited $50.00
17/11/18  Balance is: $1,050.00
17/11/18  Withdrawn $150.00
17/11/18  Balance is: $900.00
```

Figure 11: Showing customer transaction report.

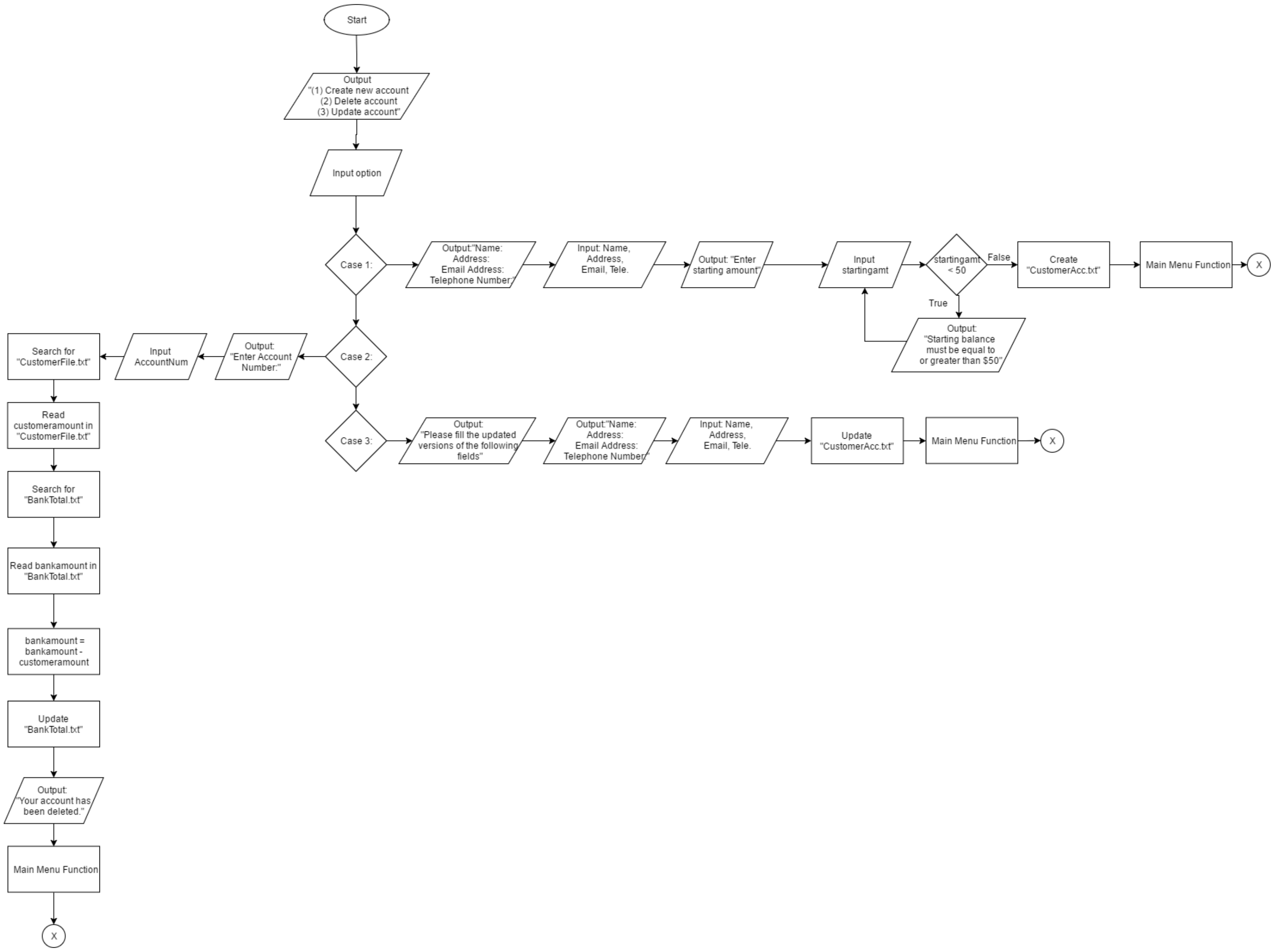
Login Function Flowchart


Start
attempts=0
if attempts = 3
True
Output "Login Failed"
Close Program
False
Output: "Username: Password:"
Read username and password
Read from file "BankClerks.txt" usernames and password
If username and password is equal to those in file
False
attempts = attempts + 1
True
Menu function
Stop

Main Menu Function Flowchart

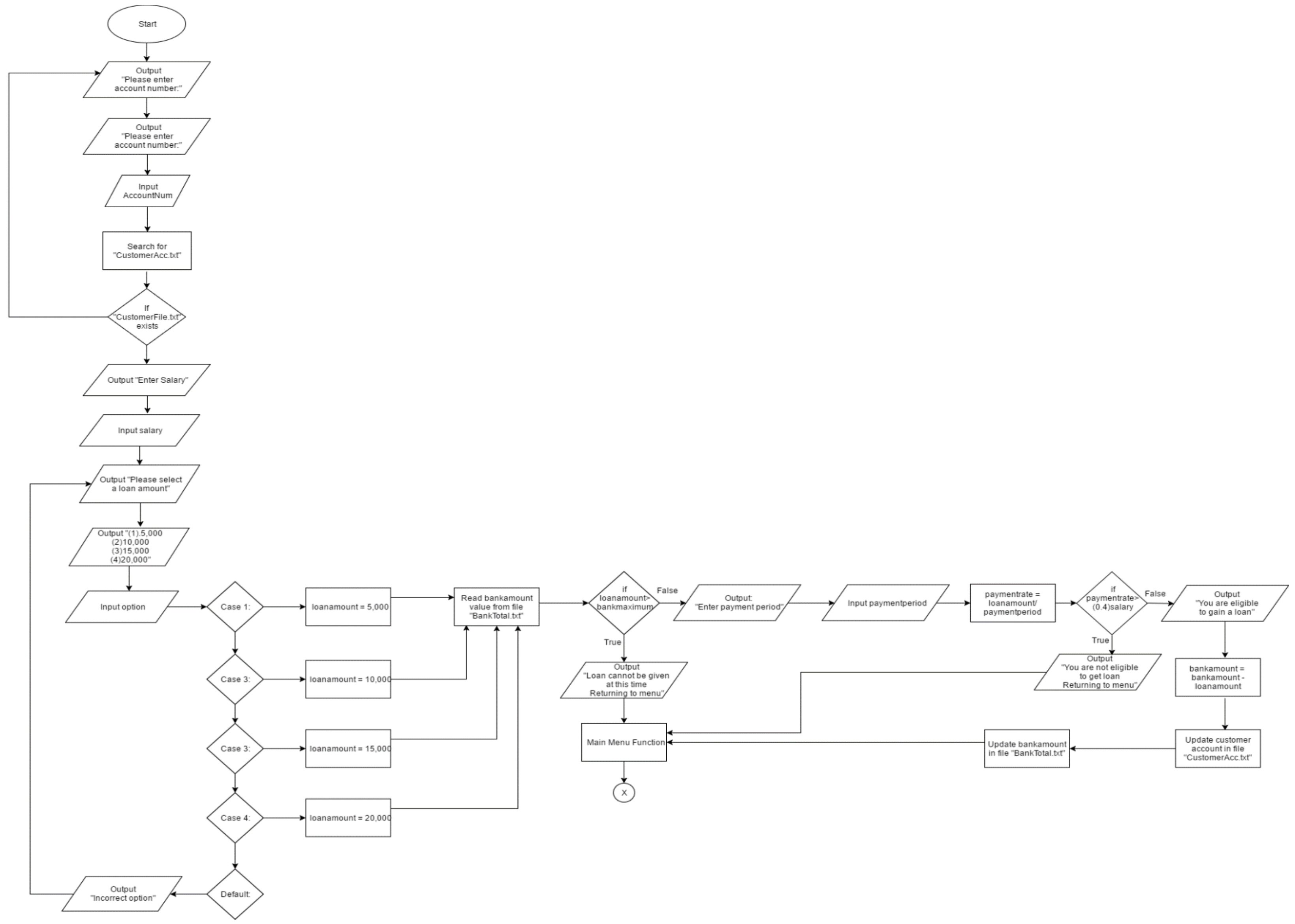

Start
Output "Please select an option from below"
Output "(1) File Creation/Deletion (2) Transactions (3) Loan Process (4) View Customer File (5) Log Out "
Input option
Case 1:
File Creation/Deletion Function
X
Case 2:
Transactions Function
X
Case 3:
Loan Process Function
X
Case 4:
View Customer File Function
X
Case 5:
Main Function
X
Default:
Output "Incorrect Option"
Stop

Customer Function Flowchart

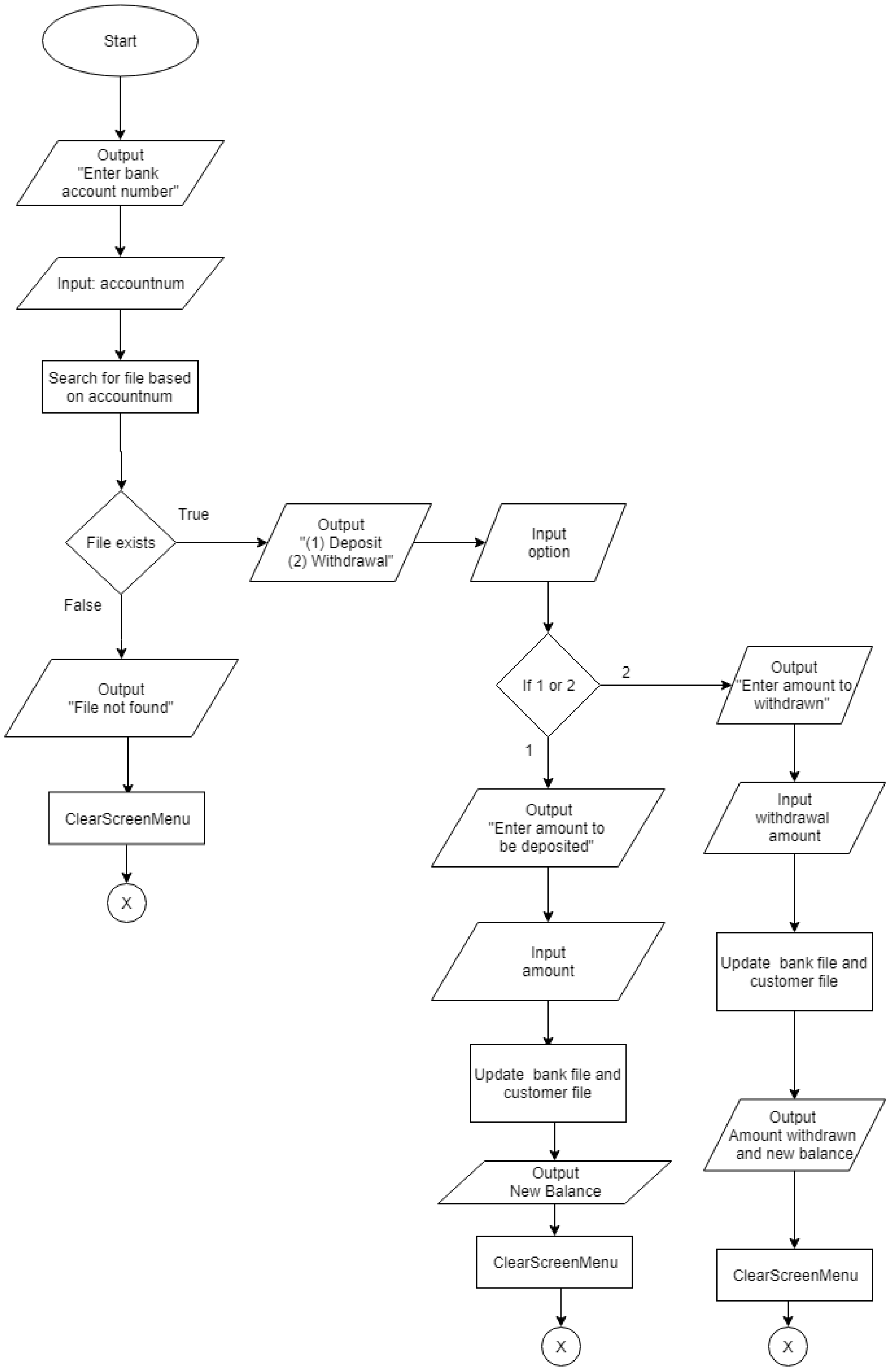


Start
Output "(1) Create new account (2) Delete account (3) Update account"
Input option
Case 1:
Output:"Name: Address: Email Address: Telephone Number:"
Input: Name, Address, Email, Tele.
Output: "Enter starting amount"
Input startingamt
startingamt < 50
False
True
Output: "Starting balance must be equal to or greater than $50"
Create "CustomerAcc.txt"
Main Menu Function
X
Case 2:
Output: "Enter Account Number:"
Input AccountNum
Search for "CustomerFile.txt"
Read customeramount in "CustomerFile.txt"
Search for "BankTotal.txt"
Read bankamount in "BankTotal.txt"
bankamount = bankamount - customeramount
Update "BankTotal.txt"
Output: "Your account has been deleted."
Main Menu Function
X
Case 3:
Output: "Please fill the updated versions of the following fields"
Output:"Name: Address: Email Address: Telephone Number:"
Input: Name, Address, Email, Tele.
Update "CustomerAcc.txt"
Main Menu Function
X

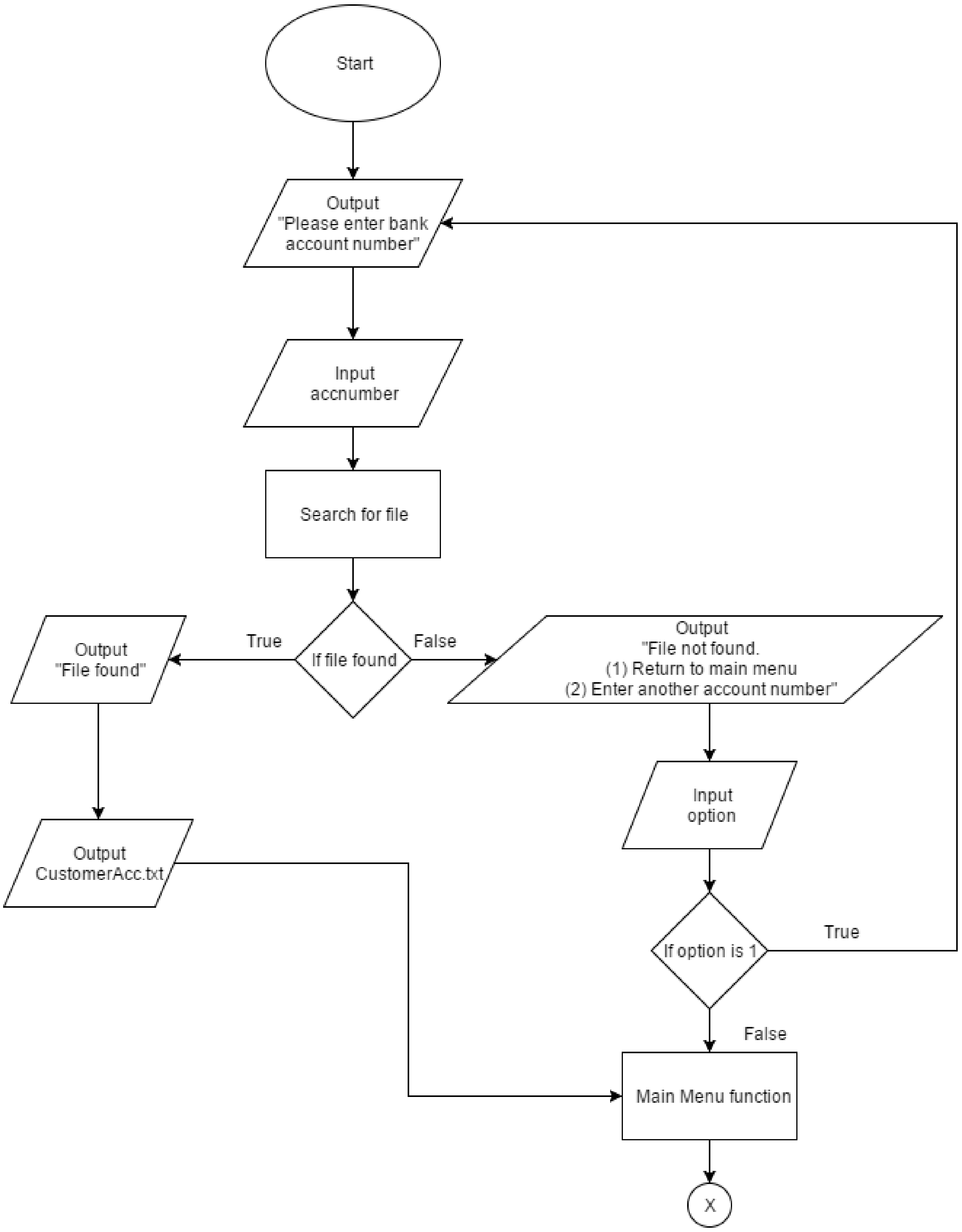
Loans Function Flowchart


Start
Output "Please enter account number:"
Output "Please enter account number:"
Input AccountNum
Search for "CustomerAcc.txt"
If "CustomerFile.txt" exists
Output "Enter Salary"
Input salary
Output "Please select a loan amount"
Output "(1).5,000 (2)10,000 (3)15,000 (4)20,000"
Input option
Case 1:
loanamount = 5,000
Case 3:
loanamount = 10,000
Case 3:
loanamount = 15,000
Case 4:
loanamount = 20,000
Default:
Output "Incorrect option"
Read bankamount value from file "BankTotal.txt"
if loanamount> bankmaximum
False
True
Output "Loan cannot be given at this time Returning to menu"
Main Menu Function
X
Output: "Enter payment period"
Input paymentperiod
paymentrate = loanamount/ paymentperiod
if paymentrate> (0.4)salary
False
True
Output "You are not eligible to get loan Returning to menu"
Output "You are eligible to gain a loan"
bankamount = bankamount - loanamount
Update customer account in file "CustomerAcc.txt"
Update bankamount in file "BankTotal.txt"

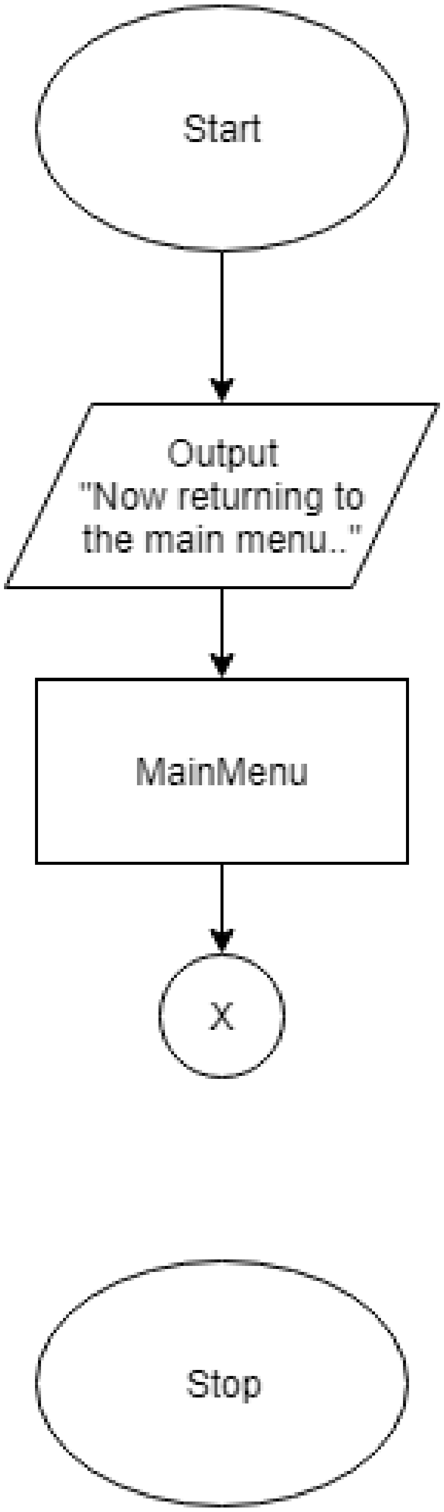
Transactions Function Flowchart


Start
Output "Enter bank account number"
Input: accountnum
Search for file based on accountnum
File exists
True
False
Output "File not found"
ClearScreenMenu
X
Output "(1) Deposit (2) Withdrawal"
Input option
If 1 or 2
1
2
Output "Enter amount to be deposited"
Input amount
Update bank file and customer file
Output New Balance
ClearScreenMenu
X
Output "Enter amount to withdrawn"
Input withdrawal amount
Update bank file and customer file
Output Amount withdrawn and new balance
ClearScreenMenu
X

Display Customer File Function Flowchart



Clear Screen Menu Function Flowchart



Data Structures

The data structures, stack, struct and array, were implemented in the coding for the project.

Evidence of this is prevalent in the file *“BankTotal”* and customer’s *“Balance”,* where the amount present is popped out so that a transaction can be performed -that is withdrawal, deposit, etc.- and then pushed back in once it is completed. Additionally, the stack data structure is used in the file *“Bank Customer”* when a customer’s account is searched for. The account numbers found are popped one by one and compared with the key until they match, after which they are pushed back in.

The record (struct) data structure was used to create accounts for different users as well as to store attributes related to customers, such as, name and phone number.

Arrays were used throughout the code. For example, they were used to store the customers’ bank account numbers when conducting a search.

Code

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
#include <unistd.h>
#include <time.h>
#include <windows.h>
#include <ctype.h>
#include <stdlib.h>
#include <process.h>
typedef struct BankClerks //Declaring a structure called Bank Clerks{
    char password[10], username[15];}clerks;
typedef struct Customers //Delcaring a structure called Customers{
    char name[50],address[50],email[20],phonenum[20];
    int accountnum;
    float balance;}custs;
FILE *Balance; //Declaring files to be used
FILE *BankTotal;
FILE *CustomerFile;
FILE *CustomerDatabase;
float balance,bankamount, customeramount;
char filename1[20],filename2[20],date[100];
int main(){
    int login;
    BankClerksInfo();
    login = BankClerkLogin();
    if (login == 1){
        system("pause");
        system("cls");
          Menu();}
    else //If they fail to log in then the program will close
            printf("The program is now exiting as you failed to login and
have run out of attempts.");
            return(0);}
int BankClerksInfo() //This function holds all the login information for the
bank clerks.{
    //Declaration of file and variables
    FILE *BankClerksFile;
    clerks bankclerks[3];
    int count;
    strcpy(bankclerks[0].username, "JWeekes");//Initializes the members of
the array bank clerks
    strcpy(bankclerks[0].password, "passworda");

    strcpy(bankclerks[1].username, "DQueeley");
    strcpy(bankclerks[1].password, "passwordb");
    strcpy(bankclerks[2].username, "MRoss");
    strcpy(bankclerks[2].password, "passwordc");
    BankClerksFile = fopen("BankClerks.txt", "w");//Creates the file
BankClerks.txt which will be used to store bank clerks information
    if (BankClerksFile != NULL) //Checks to see if the file exists{
        for(count=0; count<3; ++count) //Writes the values of the array
lecturers to the BankClerks.txt file{
```

```
            fprintf(BankClerksFile, "%s %s\n", 
bankclerks[count].username,bankclerks[count].password);};
    fclose(BankClerksFile);}
    else
        //Output this line if the BankClerks.txt file cannot be found
        printf("\nThe file BankClerks.txt cannot be found.");
        return (0);}
int BankClerkLogin()//This function allows the bank clerks to log into the 
program.{
    FILE *BankClerksFile;    //Declaration of file and variables
    char login_name[20], login_password[20];
    int attempts,i,x,j;
    clerks bankclerks[3];
    char login[20]="\tLOGIN";    //Displaying LOGIN with asterisks around it
    printf("\t\t");
    for(j=0;j<20;j++){
        Sleep(20);
        printf("*");}
    printf("   \n \t \t  ");
    for(j=0;j<20;j++){
        Sleep(50);
        printf("%c",login[j]);}
    printf("\n \t \t");
    for(j=0;j<20;j++){
        Sleep(20);
        printf("*");}
    BankClerksFile = fopen("BankClerks.txt", "r");
    //This loop runs three times as the user will have three chances to login
    for (attempts=0; attempts<3; ++attempts){
        //Prompt the user to enter their login information
        printf("\nPlease enter your username and password: ");
        printf("\nUsername: ");
        gets(login_name);
        printf("Password: ");
        gets(login_password);
        //Reads the file to get the bank clerks info and compares them to the 
values entered.
        for(i=0; i<3; ++i){
            fscanf (BankClerksFile, "%s %s\n", bankclerks[i].username, 
bankclerks[i].password);
            //Compares the login name and password to the lecturers info from 
the file
            if ((strcmp(bankclerks[i].username, login_name)== 0) && 
(strcmp(bankclerks[i].password, login_password)==0)){
                printf("You have successfully logged in, 
%s.\n",bankclerks[i].username);
                fclose(BankClerksFile);
                return 1;};};

    //If an incorrect login name or login password is entered this would be 
executed, then the user's chances will decrease
    puts("\nYou have entered an incorrect username and password 
combination.");
    //Checking to see if the user has used all 3 attempts and displaying 
attempts remaining
```

```
    if (attempts == 3){
            fclose(BankClerksFile);
            return(2);}
    else
        x = 2-attempts;
        printf("You now have %d attempt(s) remaining.\n",x);
        system("pause");
        system("cls");
        fflush(stdin);
    };
}
int Menu() //This function has a list of options for the bank clerks to
choose from depending on what the customer wants to do.
{
    int option;
    do
    {
        //Displaying SPCCU MENU with asterisks around it
        char d[20]="  SPCCU MENU ";
        int i=0,j;
        printf("\t\t");
        for(j=0;j<20;j++)
        {
        Sleep(50);
        printf("*");
        }
        printf("   \n \t \t  ");
        for(j=0;j<20;j++)
        {
        Sleep(50);
        printf("%c",d[j]);
        }
        printf("\n \t \t");
        for(j=0;j<20;j++)
        {
        Sleep(50);
        printf("*");
        }

        //Displays the various options and determines which one the user
wants to do
        puts("\nPlease select an option from below. . .");
        puts("(1) Account Creation/Closure/Updating");
        puts("(2) Transactions");
            puts("(3) Loan Process");
        puts("(4) View Customer File");
        puts("(5) Logout");
            printf("Option: ");
        scanf("%d",&option);
        fflush(stdin); //Clears the input stream for command prompt to
prevent spaces and other text to be carried on to a next function

        //Checks to ensure if the value the user entered is in the range of 1
and 5
```

```
        if (option<1 || option>5)
            {
                printf("Invalid option, your option must be either 1,2,3,4 or 
5.\n");
                ClearScreenMenu();
            }
        else

        //Using switch-case function to determine what function to call based 
on the user's entered option
        switch(option)
        {
            case 1:
                system("cls");
                AccountManagement();
                break;
            case 2:
                system("cls");
                Transactions();
                break;
            case 3:
                system("cls");
                Loans();
                break;
            case 4:
                system("cls");
                ViewFile();
                break;
            case 5:
                system("cls");
                main();
                break;
        }
    }
    while(option<1 || option>5);//This loop will continue running until the 
user inputs a number in the range of 1 and 5
}

int Transactions() //This functions allows the user to process withdraws and 
deposits.
{
    custs customer;
    float deposit, withdraw;
    int choice, choice2, accountnum[1000], i=0, count=0;

    //Getting the values for the  current date
    struct tm *tm;
    time_t v;
    v = time(NULL);
    tm = localtime(&v);
    strftime(date, sizeof(date), "%d/%m/%Y", tm);

    char t[20]=" TRANSACTIONS ";
    int j;
    printf("\t\t");
    for(j=0;j<20;j++)
    {
        Sleep(50);
        printf("*");
    }
    printf("   \n \t \t  ");
    for(j=0;j<20;j++)
    {
        Sleep(50);
        printf("%c",t[j]);
    }
    printf("\n \t \t");
    for(j=0;j<20;j++)
    {
        Sleep(50);
        printf("*");
    }


    printf("\nPlease enter your bank account number: ");
    scanf("%d",&customer.accountnum);
    CustomerDatabase = fopen("BankCustomers.txt","r");

    //Reads the customer account numbers from the file into an array 
accountnum
    while(!feof(CustomerDatabase))
    {
        fscanf(CustomerDatabase,"%d\n",&accountnum[i]);
        count++;
        i++;
    }


    //Searches through the array to find entered account number
    for(i=0;i<count;i++)
    {
        if(customer.accountnum == accountnum[i])
        {
            printf("Account number is in database\n");
            break;
        }
    }

    //Checking to see whether the file exists
    sprintf(filename1,"%d.txt",customer.accountnum);
    if ((CustomerFile = fopen(filename1,"r")) == NULL)
    {
        printf("Account file cannot be found or the account is closed. . .");
        ClearScreenMenu();
    }
    else
    {
        puts("Please select an option from below: ");
        puts("(1) Deposit");
        puts("(2) Withdraw");
        printf("Option: ");
        scanf("%d",&choice);
        fflush(stdin);

        if (choice<1 || choice>2)//condition ensuring choice input is valid
        {
            printf("Invalid option, your option must be either 1 or 2.\n");
            ClearScreenMenu();
        }
        else

        //Using switch function to determine what action to do based on what
the user entered
        switch(choice)
        {
            case 1:
                printf("Enter amount to deposited: ");//not done
                scanf("%f",&deposit);

                sprintf(filename2,"%d Balance.txt",customer.accountnum);

                Balance = fopen(filename2,"r");//opens file
                BankTotal = fopen("BankTotal.txt","r");//opens file
                CustomerFile = fopen(filename1,"a");//opens file

                fscanf(Balance,"%f",&customer.balance);//scans file
                fscanf(BankTotal,"%f",&bankamount);//scans file

                fprintf(CustomerFile,"\n\n****Customer Transactions****\n%s
Balance is: $%.2f \n",date,customer.balance);

                customer.balance = customer.balance + deposit;
                bankamount = bankamount + deposit;

                Balance = fopen(filename2,"w");
                BankTotal = fopen("BankTotal.txt","w");


                //Updating the customers balance and the bank total
                fprintf(CustomerFile,"%s Deposited: $%.2f \n",date,deposit);
                fprintf(CustomerFile,"%s New Balance is: $%.2f
\n",date,customer.balance);
                fprintf(Balance,"%.2f",customer.balance);
                fprintf(BankTotal,"%.2f",bankamount);

                fclose(CustomerFile);
                fclose(Balance);
                fclose(BankTotal);

                printf("New Balance is: $%.2f", customer.balance);
                ClearScreenMenu();
           case 2:
                printf("Enter amount to be withdrawn: ");
                scanf("%f",&withdraw);

                sprintf(filename2,"%d Balance.txt",customer.accountnum);

                Balance = fopen(filename2,"r");
                BankTotal = fopen("BankTotal.txt","r");
                CustomerFile = fopen(filename1,"a");

                fscanf(Balance,"%f",&customer.balance);
                fscanf(BankTotal,"%f",&bankamount);

                fprintf(CustomerFile,"\n\n****Customer Transactions****\n%s
Balance is: $%.2f \n",date,customer.balance);

                customer.balance = customer.balance - withdraw;
                bankamount = bankamount - withdraw;

                printf("Amount Withdrawn: %.2f \nNew Balance is: %.2f\n\n",
withdraw, customer.balance);

                system("pause");

                Balance = fopen(filename2,"w");
                BankTotal = fopen("BankTotal.txt","w");

                //Updating the customers balance and the bank total
                fprintf(CustomerFile,"%s Withdrawn: $%.2f \n",date,withdraw);
                fprintf(CustomerFile,"%s New Balance is: $%.2f
\n",date,customer.balance);

                fprintf(Balance,"%.2f",customer.balance);
                fprintf(BankTotal,"%.2f",bankamount);

                fclose(CustomerFile);
                fclose(Balance);
                fclose(BankTotal);
                ClearScreenMenu();
        }
            while(choice<1 || choice>2);
    }
    return 0;
}

int AccountManagement()//This function creates,closes and update customer
files.
{
    custs customer;
    char amount[100000];
    int option, enter = 0, i;

    char d[30]="ACCOUNT MANAGEMENT";
    int j;
    printf("\t\t");
    for(j=0;j<22;j++)
    {
        Sleep(50);
        printf("*");
    }
    printf("   \n \t \t  ");
    for(j=0;j<22;j++)
    {
        Sleep(50);
        printf("%c",d[j]);
    }
    printf("\n \t \t");
    for(j=0;j<22;j++)
    {
        Sleep(50);
        printf("*");
    }


    //Prompting the user for their option
    puts("\nPlease select an option from below. . .");
    puts("(1) Account Creation");
    puts("(2) Account Closure");
    puts("(3) Account Updating");
    printf("Option: ");
    scanf("%d",&option);
    fflush(stdin); //Clears the input stream for CMD

    //Checks to ensure if the value the user entered is in the range of 1 and
3
    if (option<1 || option>3)
        {
            puts("Invalid option, your option must be either 1,2 or 3. . .");
            ClearScreenMenu();
        }

    switch(option)
    {
        case 1:

            //Getting information about the customer so that an account can
be created for them
            system("cls");
            puts("ACCOUNT CREATION");
            puts("Please enter the following information. . .");
            printf("Full Name: ");
            gets(customer.name);
            printf("Address: ");
            gets(customer.address);
            printf("Email Address: ");
            gets(customer.email);
            fflush(stdin);
            printf("Telephone Number: ");
            gets(customer.phonenum);
            printf("Starting Amount: ");
            gets(amount);

            //Checking to see if the starting amount only contains numbers
and that it is in the range of 50 and 1,000,000
            for(i=0;i<strlen(amount);i++)
            {
                if(isdigit(amount[i]) == 0)
                {
                    puts("A customer's starting amount must be between 50 and
1,000,000");
                    ClearScreenMenu();
                }
            };

            customer.balance = atof(amount);
            if (customer.balance>1000000 || customer.balance<50)
            {
                puts("A customer's starting amount must be between 50 and
1,000,000.");
                ClearScreenMenu();
            };

            //Generating a random number for the Customer's bank account
number
            srand(time(NULL));
            customer.accountnum = rand();

            //Storing text in a variable to open them as a file
            sprintf(filename1,"%d.txt",customer.accountnum);
            sprintf(filename2,"%d Balance.txt",customer.accountnum);

            //Creating files needed to be used for an account creation
            BankTotal = fopen("BankTotal.txt","r");
            CustomerFile = fopen(filename1,"w");
            Balance = fopen(filename2,"w");

            fscanf(BankTotal,"%f",&bankamount);
            BankTotal = fopen("BankTotal.txt","w");
            bankamount = bankamount + customer.balance;

            CustomerDatabase = fopen("BankCustomers.txt","a");
            fprintf(CustomerDatabase,"%d\n",customer.accountnum);

            fprintf(BankTotal,"%.2f",bankamount);
            fprintf(CustomerFile,"Name: %s\nAccount Number: %d\nAddress:
%s\nEmail Address: %s\nTelephone Number: %s",customer.name,
customer.accountnum, customer.address, customer.email, customer.phonenum);
            fprintf(Balance,"%.2f",customer.balance);

            fclose(CustomerDatabase);
            fclose(BankTotal);
            fclose(CustomerFile);
            fclose(Balance);
            printf("%s's account number is: 
%d",customer.name,customer.accountnum);
            printf("\n%s account has been successfully created with starting 
amount %.2f" ,customer.name, customer.balance);
            ClearScreenMenu();
            break;

        case 2:

            system("cls");
            puts("ACCOUNT CLOSURE");

            printf("Please enter the bank account number to be closed: ");
            scanf("%d",&customer.accountnum);

            sprintf(filename1,"%d.txt",customer.accountnum);
            sprintf(filename2,"%d Balance.txt",customer.accountnum);

            //Checking to see if the account exists
            if(((CustomerFile = fopen(filename1,"r"))&&(Balance = 
fopen(filename2,"r"))) == NULL)
            {
                printf("\nThe bank account number you entered cannot be 
found. . .");
                ClearScreenMenu();
            }
            else
            {
                do
                {
                    //Verifying that the person wants to close the account
                    fflush(stdin);
                    system("cls");
                    printf("Are you sure you want to close this account?
\nEnter 1 for Yes and 2 for No.");
                    printf("\nOption: ");
                    scanf("%d", &option);
                }
                while(option < 1 || option >2);
                if(option == 1)
                    {
                        //Subtracting the person's balance from the total 
bank file and updating it
                        BankTotal = fopen("BankTotal.txt","r");
                        fscanf(Balance,"%f",&customer.balance);
                        fscanf(BankTotal,"%f",&bankamount);
                        bankamount = bankamount - customer.balance;
                        BankTotal = fopen("BankTotal.txt","w");
                        fprintf(BankTotal,"%.2f",bankamount);

                        fclose(CustomerFile);
                        fclose(Balance);
                        fclose(BankTotal);
                        //Deleting the files associated with the entered bank
account number
                        unlink(filename1);
                        unlink(filename2);

                        printf("Account number %d has been successfully
closed. Balance before closure %.2f. . .",customer.accountnum,
customer.balance);
                        ClearScreenMenu();

                    }
                }
                ClearScreenMenu();
            case 3:

                system("cls");
                puts("ACCOUNT UPDATING");

                printf("Please enter bank account number to be updated: ");
                scanf("%d",&customer.accountnum);

                sprintf(filename1,"%d.txt",customer.accountnum);

                //Checking to see if the account exists
                if ((CustomerFile = fopen(filename1,"r")) == NULL)
                {
                    printf("The account number you entered cannot be found or
the account does not exist. . .");
                    ClearScreenMenu();
                }
                else

                    fflush(stdin);
                    CustomerFile = fopen(filename1,"a");

                    //Getting the updated information about the customer
                    puts("Please fill the updated versions of the following
fields. . .");
                    printf("Name: ");
                    gets(customer.name);
                    printf("Address: ");
                    gets(customer.address);
                    printf("Email address: ");
                    gets(customer.email);
                    printf("Telephone number: ");
                    gets(customer.phonenum);

                    //Getting the values for the current date
                    struct tm *tm;
                    time_t v;
                    v = time(NULL);
                    tm = localtime(&v);
                    strftime(date, sizeof(date), "%d/%m/%Y", tm);

                    //Updating the file
                    fprintf(CustomerFile,"\n\n****Updated Customer
Information****\nDate Updated: %s \nName: %s\nAddress: %s\nEmail Address:
%s\nTelephone Number: %s", date ,customer.name, customer.address,
customer.email, customer.phonenum);
                    fclose(CustomerFile);

                    puts("Account has been successfully updated");
                    ClearScreenMenu();
    }
}

int ClearScreenMenu()//This function pauses the screen,clears it then returns
to the menu.
{
    fflush(stdin);
    puts("\nNow returning to the menu . . .");
    system("pause");
    system("cls");
    Menu();
};

int ViewFile() //This function opens the file of a Customer.
{
    //Declaration of file and variables
    char ans;
    int c;
    custs customer;

    //Displaying CUSTOMER FILE with asterisks around it
    char t[40]=" CUSTOMER FILE ";
    int j;
    printf("\t\t");
    for(j=0;j<20;j++)
    {
        Sleep(50);
        printf("*");
    }
    printf("   \n \t \t  ");
    for(j=0;j<20;j++)
    {
        Sleep(50);
        printf("%c",t[j]);
    }
    printf("\n \t \t");
    for(j=0;j<20;j++)
    {
        Sleep(50);
        printf("*");
    }


    //Entering the customers account number
    printf("\nPlease enter the customer's bank account number: ");
    scanf("%d",&customer.accountnum);
    sprintf(filename1,"%d.txt",customer.accountnum); //Assigns the entered
customer account number and the .txt file extension into the variable
filename

    //Checks to see if the file was successfully opened. If it was not
successful then it returns to the menu
    if((CustomerFile = fopen(filename1 , "r")) == NULL)
    {
        printf("Account number cannot be found.");
        ClearScreenMenu();
    };

    //Reads the entire file and outputs it on the screen
    printf("\n");
    while(1)
    {
        c = fgetc(CustomerFile);
        if(feof(CustomerFile))
        {
            break;
        }
        printf("%c",c);
    };
    fclose(CustomerFile);

    //Determines what the user wants to do next
    printf("\n\nPress ENTER if you want to continue viewing customer
files\nOr ANY OTHER KEY to return to the menu . . .\n\n\n");
    ans = getch();
    if (ans != 13) //Checks the user's option and determines whether to call
the function again or return to the menu
    {
        ClearScreenMenu();
    }
    else
        ViewFile();
}

int Loans()//This function allows customer to take out a loan from the bank.
{
    custs customer;
    int option, accountnum, paymentperiod;
    float salary, loanamount, paymentrate, bankmaximum=21000;
    char date[20];

    system("cls");

    char d[20]="  LOANS PROCESS ";
    int i=0,j;
    printf("\t\t");
    for(j=0;j<20;j++)
    {
        Sleep(50);
        printf("*");
    }

    printf("   \n \t \t  ");
    for(j=0;j<20;j++)
    {
        Sleep(50);
        printf("%c",d[j]);
    }

    printf("\n \t \t");
    for(j=0;j<20;j++)
    {
        Sleep(50);
        printf("*");
    }


    printf("\nPlease enter your bank account number\n");
    scanf("%d",&customer.accountnum);
    sprintf(filename1,"%d.txt",customer.accountnum);
    if ((CustomerFile = fopen(filename1,"r")) == NULL)
        {
            printf("Account does not exist\n\n");
            ClearScreenMenu();
        }
    else
    {
        printf("Account exists\n");
        Sleep(2*1000);
        system("cls");
    }

        puts("Enter your monthly salary: $");
        scanf("%f", &salary);
        printf("Please select a loan amount: ");
        printf("\n(1)$5,000 \n(2)$10,000 \n(3)15,000 \n(4)20,000 \n");
        printf("Option: ");
        scanf("%d", &option);
        if (option<1 || option>4)//This check if the users input is erroneous
            {
                system("cls");
                printf("Invalid option, your option must be either 1, 2, 3 or
4.\n"); //Output displayed if input is erroneous

            }
        else

        switch(option)
        {
            case 1://if option is 1
                loanamount= 5000;
                //reads customer name
                if(loanamount>bankmaximum)
                {
                   printf("Loan cannot be given at this time \n");
                   ClearScreenMenu();
                }
                else
                {
                    printf("Enter payment period in months: ");
                    scanf("%d", &paymentperiod);
                    system("cls");

                    paymentrate=(loanamount/paymentperiod);
                    if(paymentrate>(0.4*salary)) //ensures payment rate can
be satisfied within payment period selected
                    {
                        printf("You are not eligible to receive a loan...");
                        ClearScreenMenu();
                    }
                    else
                    {
                        printf("\nYou are eligible to receive a loan.");
                        sprintf(filename2,"%d
Balance.txt",customer.accountnum);

                        Balance = fopen(filename2,"r");
                        BankTotal = fopen("BankTotal.txt","r");

                        fscanf(Balance,"%f",&customer.balance);
                        fscanf(BankTotal,"%f",&bankamount);

                        customer.balance = customer.balance +
loanamount;//adds loan amount to the customer's bank account
                        bankamount = bankamount - loanamount;//subtracts loan
amount from the total amount in the bank

                        printf("\nA loan of %.2f was entered into your
account. \nYour account balance is now %.2f\n\n", loanamount,
customer.balance);

                        Balance = fopen(filename2,"w");
                        BankTotal = fopen("BankTotal.txt","w");

                        fprintf(Balance,"%.2f",customer.balance);
                        fprintf(BankTotal,"%.2f",bankamount);

                        fclose(Balance);
                        fclose(BankTotal);

                    }
                }
                ClearScreenMenu();
                break;
            case 2:
                 loanamount= 10000;
                    //reads customer name
                    if(loanamount>bankmaximum)
                    {
                       printf("Loan cannot be given at this time \n");
                       ClearScreenMenu();
                    }
                    else
                    {
                        printf("Enter payment period in months: ");
                        scanf("%d", &paymentperiod);
                        system("cls");

                        paymentrate=(loanamount/paymentperiod);
                        if(paymentrate>(0.4*salary)) //ensures payment rate
an be satisfied within payment period selected
                        {
                            printf("You are not eligible to receive a
loan...\n Returning to Menu...");
                            system("pause");
                            Menu();
                        }
                        else
                        {
                            printf("You are eligible to receive a loan.");
                            sprintf(filename2,"%d
Balance.txt",customer.accountnum);

                            Balance = fopen(filename2,"r");
                            BankTotal = fopen("BankTotal.txt","r");

                            fscanf(Balance,"%f",&customer.balance);
                            fscanf(BankTotal,"%f",&bankamount);

                            customer.balance = customer.balance +
loanamount;//adds loan amount to the customer's bank account
                            bankamount = bankamount - loanamount;//subtracts
loan amount from the total amount in the bank

                            printf("\nA loan of %.2f was entered into your
account. \nYour account balance is now %.2f\n\n", loanamount,
customer.balance);

                            Balance = fopen(filename2,"w");
                            BankTotal = fopen("BankTotal.txt","w");

                            fprintf(Balance,"%.2f",customer.balance);
                            fprintf(BankTotal,"%.2f",bankamount);

                            fclose(Balance);
                            fclose(BankTotal);

                        }
                    }
                    ClearScreenMenu();
                break;
            case 3:
                 loanamount= 15000;
                    //reads customer name
                    if(loanamount>bankmaximum)
                    {
                       printf("Loan cannot be given at this time \n");
                       ClearScreenMenu();
                    }
                    else
                    {
                        printf("Enter payment period in months: ");
                        scanf("%d", &paymentperiod);
                        system("cls");

                        paymentrate=(loanamount/paymentperiod);
                        if(paymentrate>(0.4*salary)) //ensures payment rate
an be satisfied within payment period selected
                        {
                            printf("You are not eligible to receive a
loan...\n Returning to Menu...");
                            system("pause");
                            Menu();
                        }
                        else
                        {
                            printf("You are eligible to receive a loan.");
                            sprintf(filename2,"%d
Balance.txt",customer.accountnum);

                            Balance = fopen(filename2,"r");
                            BankTotal = fopen("BankTotal.txt","r");

                            fscanf(Balance,"%f",&customer.balance);
                            fscanf(BankTotal,"%f",&bankamount);

                            customer.balance = customer.balance +
loanamount;//adds loan amount to the customer's bank account
                            bankamount = bankamount - loanamount;//subtracts
loan amount from the total amount in the bank

                            printf("\nA loan of %.2f was entered into your
account. \nYour account balance is now %.2f\n\n", loanamount,
customer.balance);
                            Balance = fopen(filename2,"w");
                            BankTotal = fopen("BankTotal.txt","w");
                            fprintf(Balance,"%.2f",customer.balance);
                            fprintf(BankTotal,"%.2f",bankamount);
                            fclose(Balance);
                            fclose(BankTotal);

                        }
                    }
                    ClearScreenMenu();
                break;
            case 4:
                 loanamount= 20000;
                    //reads customer name
                    if(loanamount>bankmaximum)
                    {
                       printf("Loan cannot be given at this time \n");
                       ClearScreenMenu();
                    }
                    else
                    {
                        printf("Enter payment period in months: ");
                        scanf("%d", &paymentperiod);
                        system("cls");
                        paymentrate=(loanamount/paymentperiod);
                        if(paymentrate>(0.4*salary)) //ensures payment rate
an be satisfied within payment period selected
                        {
                            printf("You are not eligible to receive a
loan...\n Returning to Menu...");
                            system("pause");
                            Menu();}
                        else{
                            printf("You are eligible to receive a loan.");
                            sprintf(filename2,"%d
Balance.txt",customer.accountnum);

                            Balance = fopen(filename2,"r");
                            BankTotal = fopen("BankTotal.txt","r");

                            fscanf(Balance,"%f",&customer.balance);
                            fscanf(BankTotal,"%f",&bankamount);
                            customer.balance = customer.balance +
loanamount;//adds loan amount to the customer's bank account
                            bankamount = bankamount - loanamount;//subtracts
loan amount from the total amount in the bank

                            printf("\nA loan of %.2f was entered into your
account. \nYour account balance is now %.2f\n\n", loanamount,
customer.balance);
                            Balance = fopen(filename2,"w");
                            BankTotal = fopen("BankTotal.txt","w");
                            fprintf(Balance,"%.2f",customer.balance);
                            fprintf(BankTotal,"%.2f",bankamount);
                            fclose(Balance);
                            fclose(BankTotal);

                        }
                    }
                    ClearScreenMenu();
                break;
        }
}
```

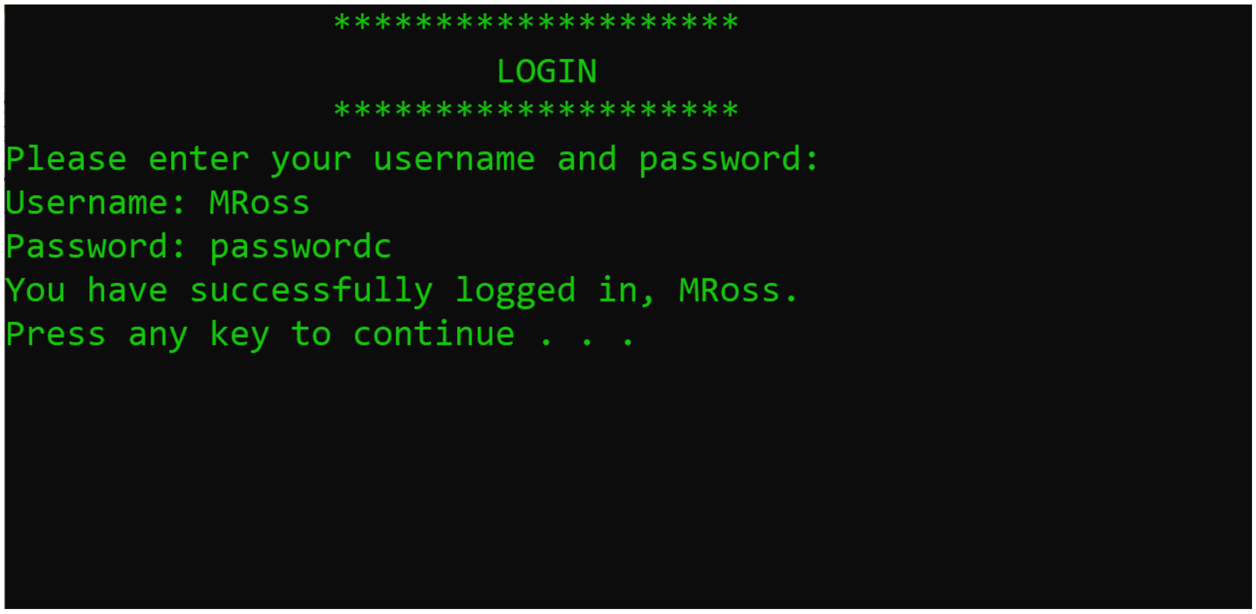
Test Plan

	Test Plan	Description	Expected Result	Actual Result	Success? / Comments
Login	Normal	User entered "MRoss" as username and "passwordc" as password	Program should display message that login was successful.	"You have successfully logged in, MRoss. Press any key to continue …"	Test case was successful
	Extreme	User entered "JWeekes" as the username and "passwordabc" as the password.	Program should display error message that combination is incorrect	"You have entered an incorrect username and password combination."	Test case was successful
	Erroneous	User entered "@@@`" as the username and "1234$" as the password.	Program should display error message that combination is incorrect	"You have entered an incorrect username and password combination."	Test case was successful
Menu	Normal	User entered "1"	Program should direct user to the account management function.	"Account Management Please select an option from below…"	Test case was successful
	Extreme	User entered "7"	Program should display error message.	"Invalid option, your option must be 1, 2, 3, 4 or 5."	Test case was successful
	Erroneous	User entered "a"	Program should display error message.	"Invalid option, your option must be 1, 2, 3, 4 or 5."	Test case was successful
Account Management	Normal	User entered "1"	Program should direct user to Account Creation function.	"ACCOUNT CREATION Please enter the following information …"	Test case was successful
	Extreme	User entered "7"	Program should display error message.	"Invalid option, your option must be 1, 2 or 3."	Test case was successful

	Erroneous	User entered “a”	Program should display error message.	“Invalid option, your option must be 1, 2 or 3.”	Test case was successful
Transactions	Normal	User entered “12694”	Program should alert user that the account is valid.	“Account number is in database. Please select an option from below.”	Test case was successful
	Extreme	User entered “1”	Program should display error message.	“Account file can found or the account is closed”	Test case was successful
	Erroneous	User entered “a”	Program should display error message.	“Account file can found or the account is closed”	Test case was successful
Loans Process	Normal	User entered “12694”	Program should alert user that the account is valid.	“Account exists”	Test case was successful
	Extreme	User entered “1”	Program should display error message.	“Account file can found or the account is closed”	Test case was successful
	Erroneous	User entered “a”	Program should display error message.	“Account file can found or the account is closed”	Test case was successful
View Customer File	Normal	User entered “12694”	Program should Allow customer to view their file.	Program displayed customer’s account	Test case was successful
	Extreme	User entered “3456789”	Program should display error message.	“Account number cannot be found”	Test case was successful
	Erroneous	User entered “f”	Program should display error message.	“Account number cannot be found”	Test case was successful

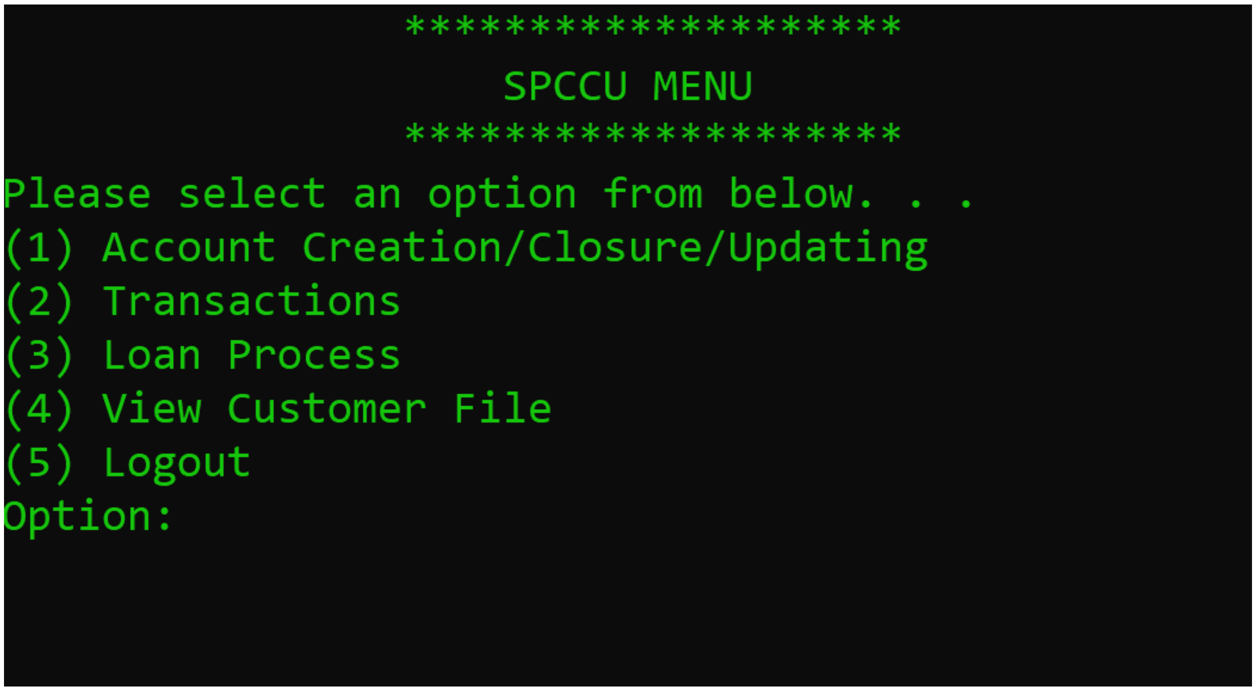
Appendix

Screenshots of the Program



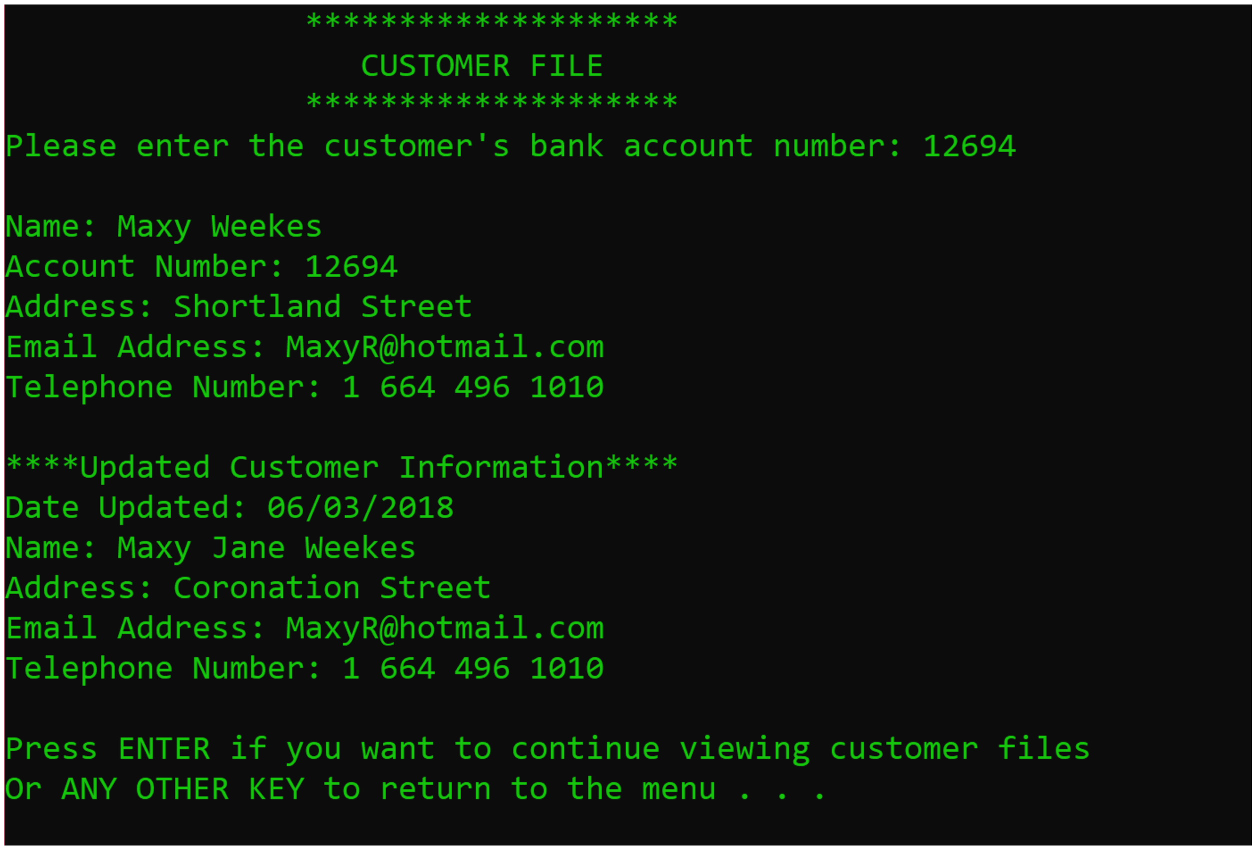
```
                    ********************
                            LOGIN
                    ********************
Please enter your username and password:
Username: MRoss
Password: passwordc
You have successfully logged in, MRoss.
Press any key to continue . . .
```

Figure 12: Showing the Login Function



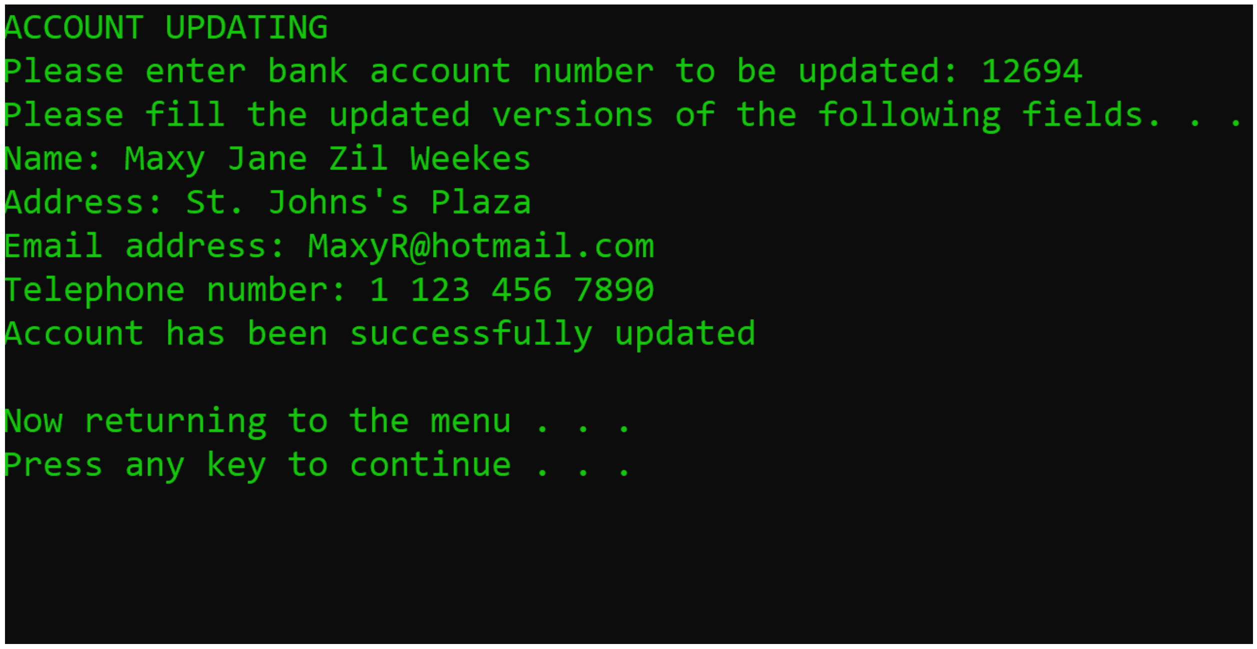
```
                *********************
                      SPCCU MENU
                *********************
Please select an option from below. . .
(1) Account Creation/Closure/Updating
(2) Transactions
(3) Loan Process
(4) View Customer File
(5) Logout
Option:
```

Figure 13: Showing the Menu Function



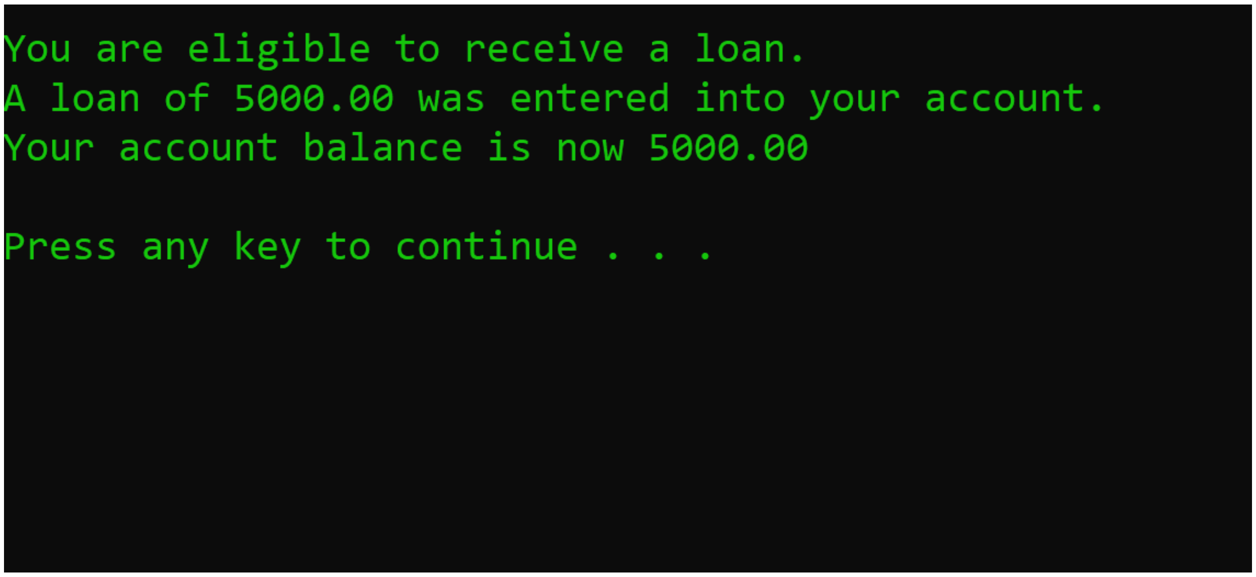
```
                ********************
                   CUSTOMER FILE
                ********************
Please enter the customer's bank account number: 12694

Name: Maxy Weekes
Account Number: 12694
Address: Shortland Street
Email Address: MaxyR@hotmail.com
Telephone Number: 1 664 496 1010

****Updated Customer Information****
Date Updated: 06/03/2018
Name: Maxy Jane Weekes
Address: Coronation Street
Email Address: MaxyR@hotmail.com
Telephone Number: 1 664 496 1010

Press ENTER if you want to continue viewing customer files
Or ANY OTHER KEY to return to the menu . . .
```

Figure 14: Showing the View Customer File Function



```
ACCOUNT UPDATING
Please enter bank account number to be updated: 12694
Please fill the updated versions of the following fields. . .
Name: Maxy Jane Zil Weekes
Address: St. Johns's Plaza
Email address: MaxyR@hotmail.com
Telephone number: 1 123 456 7890
Account has been successfully updated

Now returning to the menu . . .
Press any key to continue . . .
```

Figure 15: Showing the Updating of a Customer's file



```
You are eligible to receive a loan.
A loan of 5000.00 was entered into your account.
Your account balance is now 5000.00

Press any key to continue . . .
```

Figure 16: Showing a Customer receiving a loan

Pictures of Group Members

Figure 17: Maxwell Ross (top) and Jenalyn Weekes(bottom) working on the cover page layout and loans function respectively.

Figure 18: Denzil Queeley debugging the program.

Bibliography

Books

Find bibliography tool to use

West, Jase-Omeileo L. *Computer Science for CAPE Examinations: UNIT 2*. 2016.

Websites

“Automation of Banking.” *Investopedia*, www.investopedia.com/terms/b/branch-automation.asp.

https://www.draw.io/