Capgemini

High Level Design & Low Level Design

Document Control :

Project Revision History

Date	Version	Author	Brief Description of Changes	Approver Signature
30.08.2022	1.0	Group 5		

Index

1. Introduction: -

1.1 Intended Audience: -

The audience set for this project would include employees working in banking organisations as well as for their clients which are the customers that use services from these organisation.

1.2 Project Purpose: -

The banking system is specifically developed for banking application with facilities of account opening, deposit & withdrawal. Secondly, the application also allows only the authorized bank personnel to transfer money from one account in the same bank to another. This system also allows banking personnel to edit and delete customer details as well as generate customer report & transaction report. Whereas the customers can use this system to create an account , deposit or withdraw amount as well as view balance.

1.3 Key Project Objectives: -

- Allow user to create account.
- Allow user to do transactions (Withdraw & Deposit Amount).
- View Balance using only Account number.
- Allow banker to edit and delete record.
- Allow banker to do transfer among respective accounts.
- Allow banker to get customer report as well as get transaction report.

1.4Project scope and limitation: -

Primarily, the scope of the banking application features to ensure smooth banking operations. The main aim of the application is to automate records on the system. It provides primary functions which are required by the bank in order to run a stable system. In addition to that it also helps to manually check the records of the pre-existing system like transactions that are made in the past. The application also changes or manipulates the new data that is being added and is then re-recorded. One can also check their present transactions that are in process and keep a check on their accounts via this application. It’s not only useful for the customers but also for the admin.

2.Design Overview: -

Banking Application comprises of the following modules:

Name of the Module	Create account and Do transaction
Handled by	Anuja Nikam
Description	It will create account and do transfer from one account to another.

Name of the Module	Edit and Do Transfer
Handled by	Nitika Mhatre
Description	This will edit the previous details and will do withdraw and debit the amount.

Name of the Module	Password Functions ,View Balance & Delete Record.
Handled by	Twinkle Jain
Description	It will give password to bankers and customer login. To view balance and delete the record.

Name of the Module	Design and Menu and Get Customer Details
Handled by	Vaidehee Dalvi
Description	Designing and creating a menu and also it will get customer details.

Name of the Module	Get Transaction report,list_to_file,file_to_list
Handled by	Vaastav Talwar
Description	It will get transaction report and and file_to_list and list_to_file will read and write from file.

2.1 Design Objectives: -

- Allow user to create a new account
- Do Transaction
- View Balance
- Allow banker to edit and delete record
- Allow banker to do transfer
- Allow banker to get customer report and get transaction report

2.2 Design Alternative: -

We have used linked list instead of stack & queue as Insertion and Deletions operations are fast and easier in linked list. Memory allocation is done during run-time. (i.e., no need to allocate any fixed memory.

2.3 User Interface Paradigms: -

The Banking System gives a user an option to have its personal banking application stored on a system file. A system always works faster than a person can. User is given an interface to create a new account in bank, an option to deposit and transfer amount in the account & view balance. A specific set of users are given interface to edit details of the accounts & delete the account, to transfer an amount among respective accounts, to get transaction history of specific account and to get overall customer report.

2.4 Error Detection / Exceptional Handling: -

- If the user doesn’t have any pre-existing account , the user has to create one else it won’t perform any functions and would give “not found” or “Invalid entry” error.
- While creating the account ,user should first enter the name followed by Aadhar number else it will display “Already exist” and “Invalid length ” error for the respective cases. We check the validity of the name & account number entered with the help of exception handling .If the name entered has the length less than 5 or greater than 15 or the aadhar card number entered is either already existing or of not length 6 digit , an error message will be flashed.
- Next the user has to enter the account type that is either SA or CA .Any entry other than SA or CA will flash the “Invalid account type error”

2.5 Performance: -

The system will work on the user’s terminal. The performance shall depend upon hardware components of the banker/customer and the internet connection

3.SYSTEM ARCHITECTURE: -

3.1 Functions

3.1.1 Login

- Customer : Customer logins by entering customer's account number
- Banker: Banker logins by entering banker's id & a login password.

3.1.2 Customer's Corner

3.1.2.1 Create Account

The customer can create account by entering aadhar number and account type (either SA or CA).Every Aadhar number entered should be unique. According to the type of account, it is mandatory to deposit amount more than MAB to the balance of created account and balance should be maintained thereafter.
MAB for following account type :

- SA : Rs 5000
- CA : Rs 10000

3.1.2.2 Do_Transaction

A. Deposit: This function will add the deposited amount to the current balance.
B. Withdraw: This function will deduct the withdrawal amount from the current balance.

Primary Constraint: The withdrawal amount as well as deposited amount cannot be greater than Rs 50000 for SA and Rs 100000 for CA.

3.1.2.3 View_Balance

This function will display the details from customer file using account number.

3.1.3 Banker's Corner

3.1.3.1 Edit_Customer_Details

The banker can edit the customer's name , account type and balance.

3.1.3.2 Delete_Customer_Details

The entire customer record is deleted from database.

3.1.3.3 Do_Transfer

The banker transfers the amount from source account to destination account.

3.1.3.4 Get_Transaction_Report

The bank statement showing credit and debit information of corresponding account must be displayed on the screen.

3.1.3.5 Get_Customer_Report

The bank statement showing customer details, credit and debit information of corresponding customer account must be displayed on the screen.

3.2 Structure Details:

The system consists of two structures :

- Customer

  This structure contains all the definition of all the variables that are present in the Customer Corner Submenu.

  The customer_id, name ,password & account_type have the char data type whereas balance has double and aadhar_no with int data type.

- Transaction

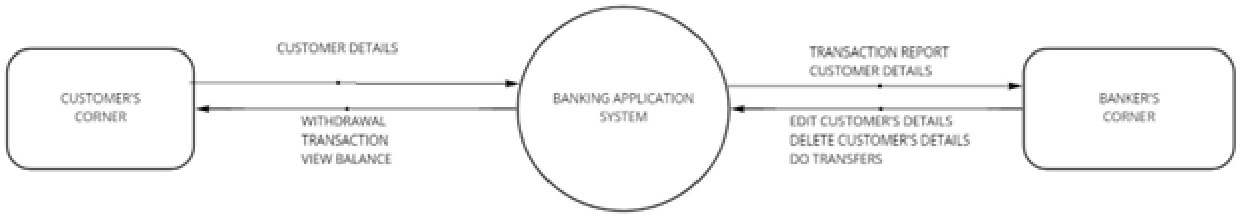
  This structure contains all the definition of variables need in Transaction report.

  The amount variable has the double data type while saccount & daccount has the char data type.

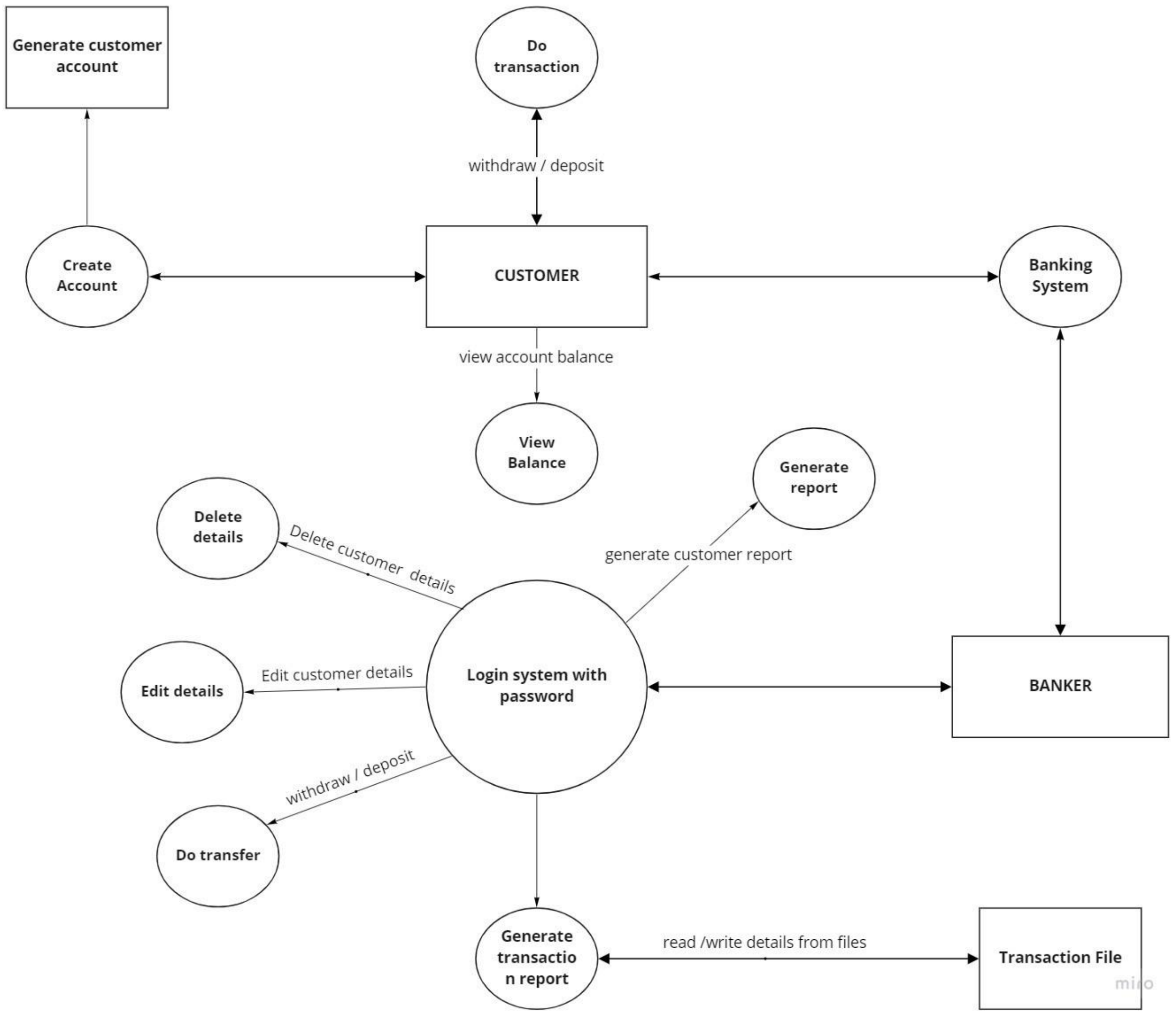
4. Detailed System Design

4.1. Data Flow Diagram

Level 0 DFD

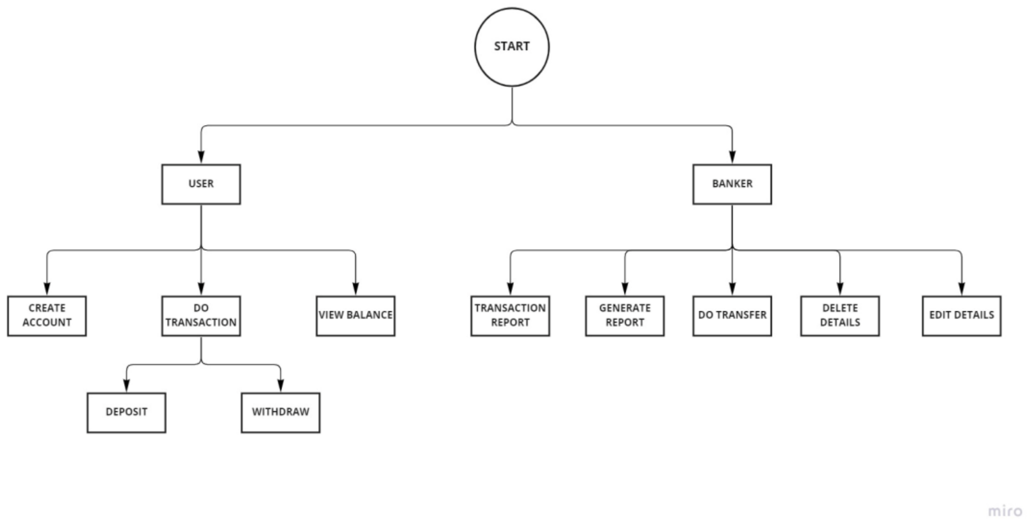


Level 1 DFD

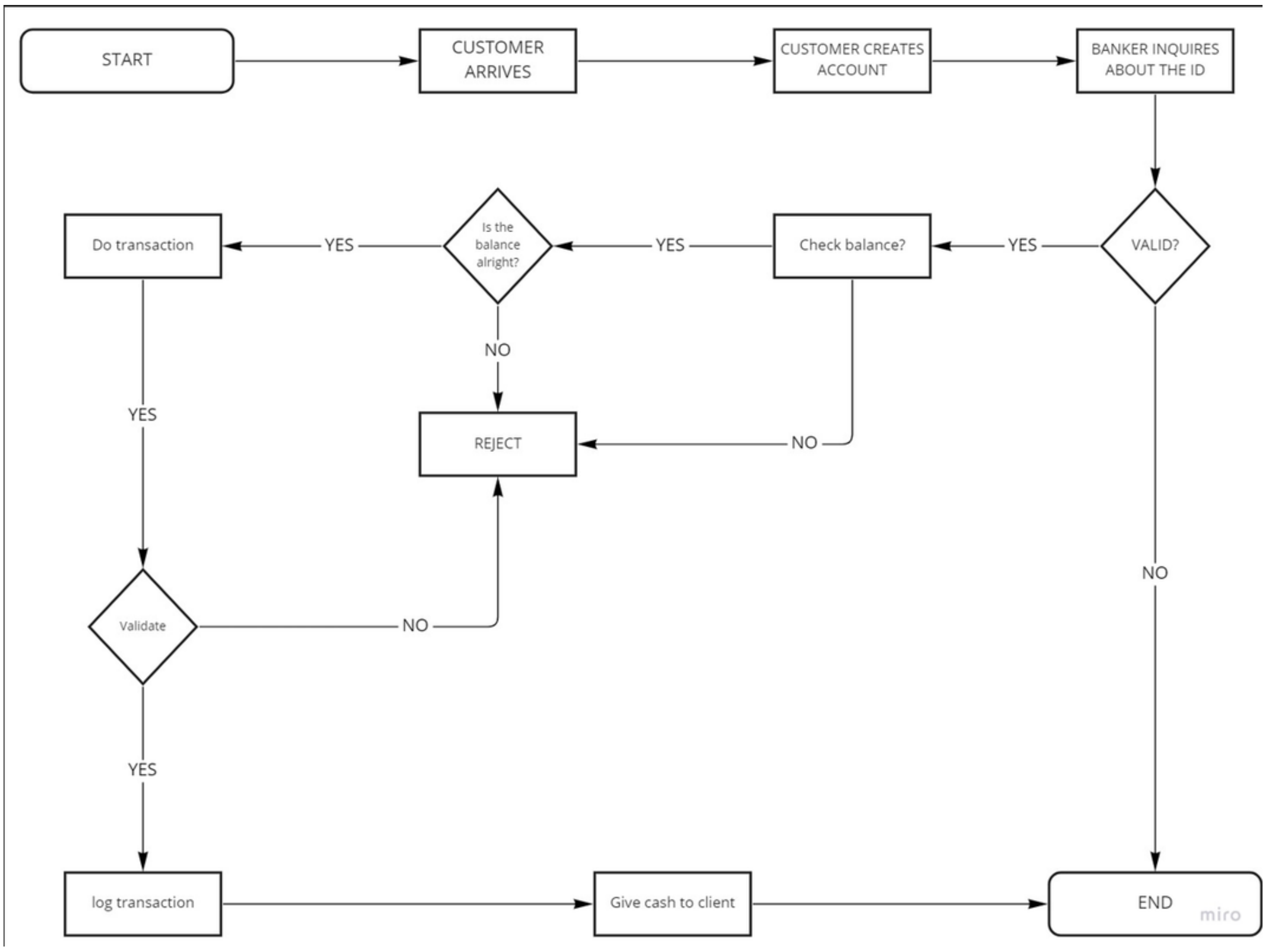


4.2.Flowcharts

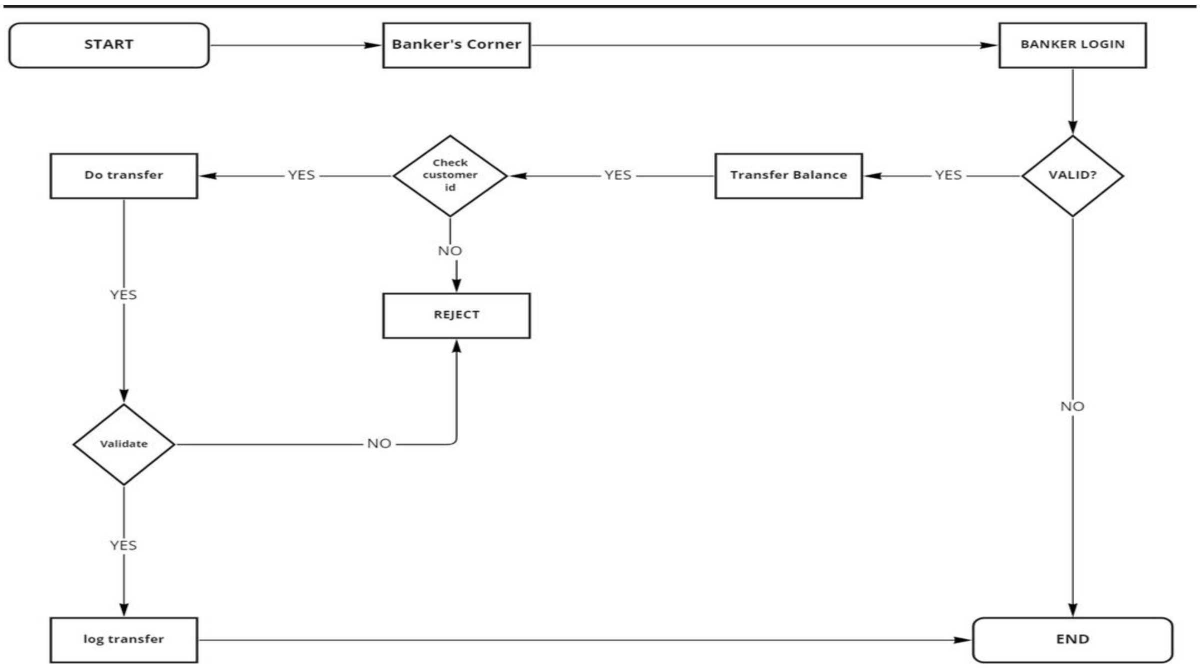
4.2.1.Main Menu Flowchart



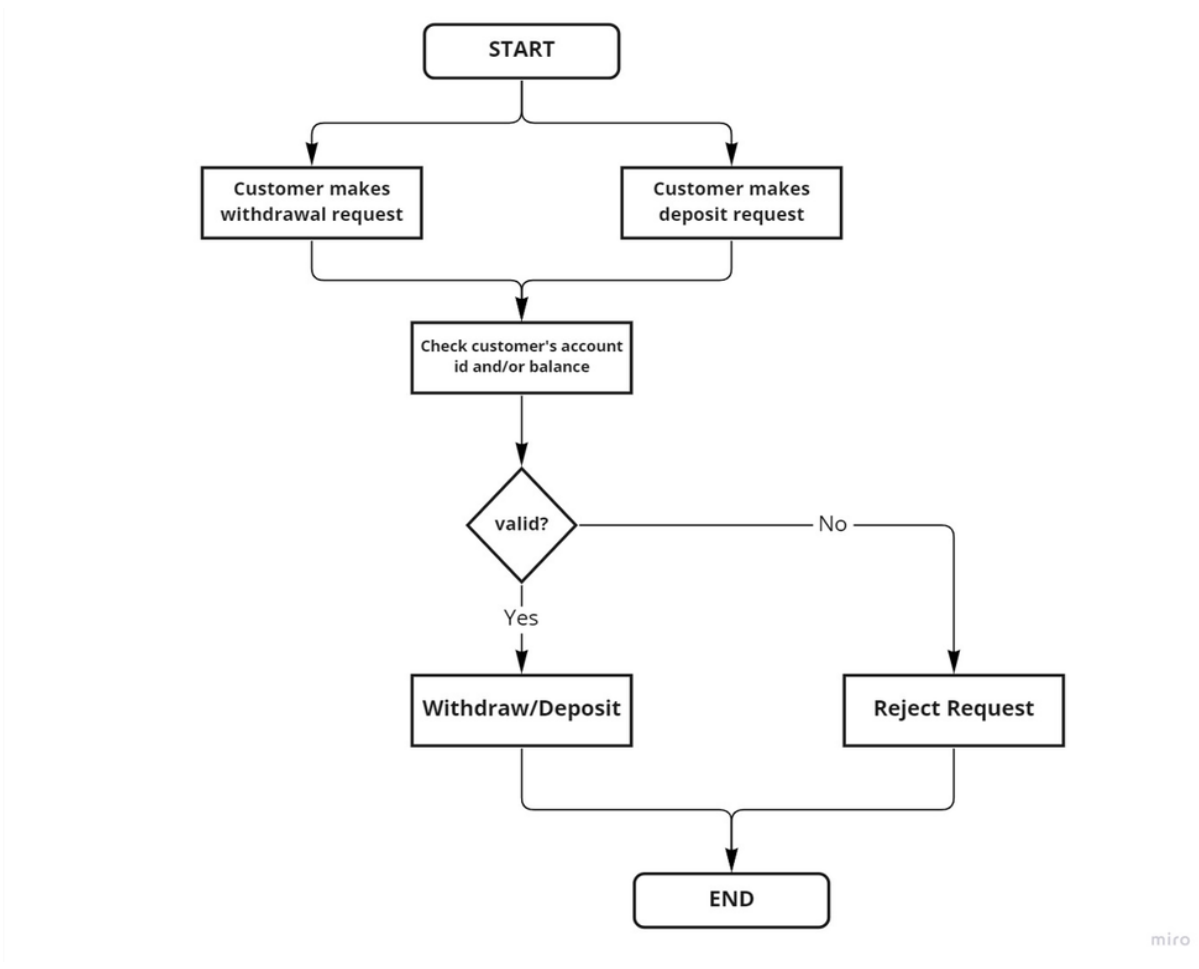
4.2.2 Customer Flowchart



4.2.3 Banker Flowchart



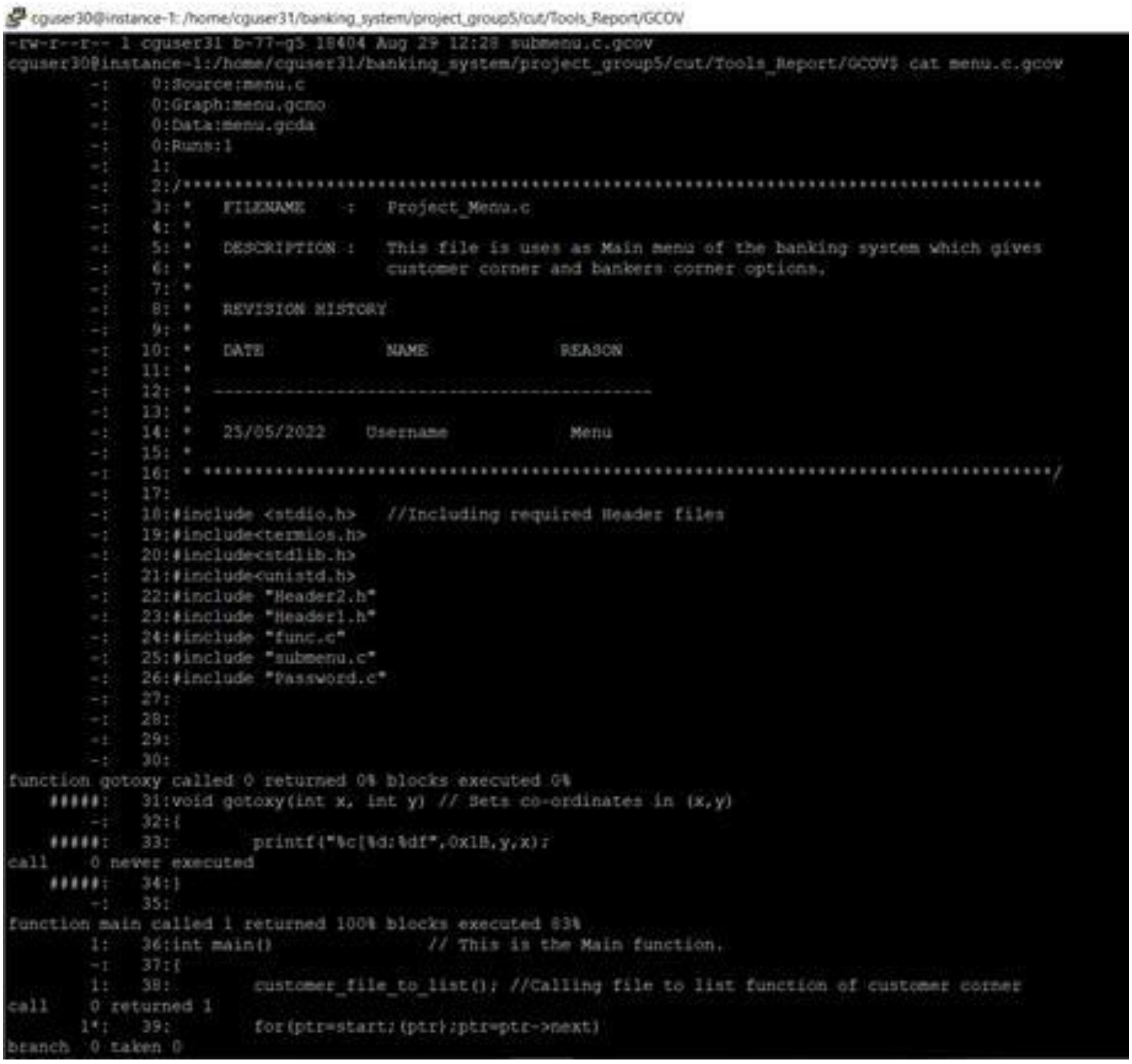
4.2.4 Transaction Flowchart


START
Customer makes withdrawal request
Customer makes deposit request
Check customer's account id and/or balance
valid?
No
Yes
Withdraw/Deposit
Reject Request
END
miro

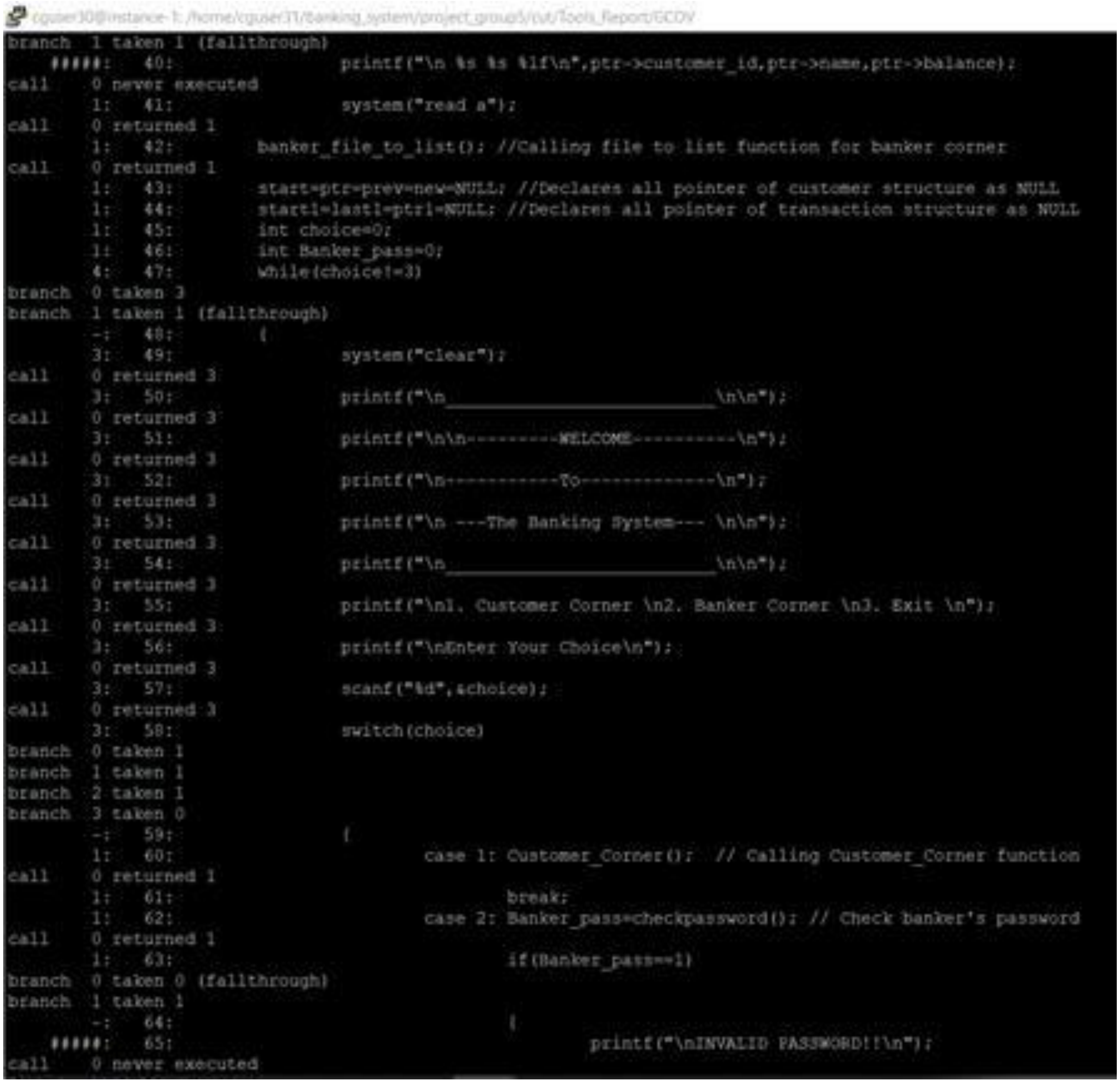
5 Tools Report

5.1 gcov Report:

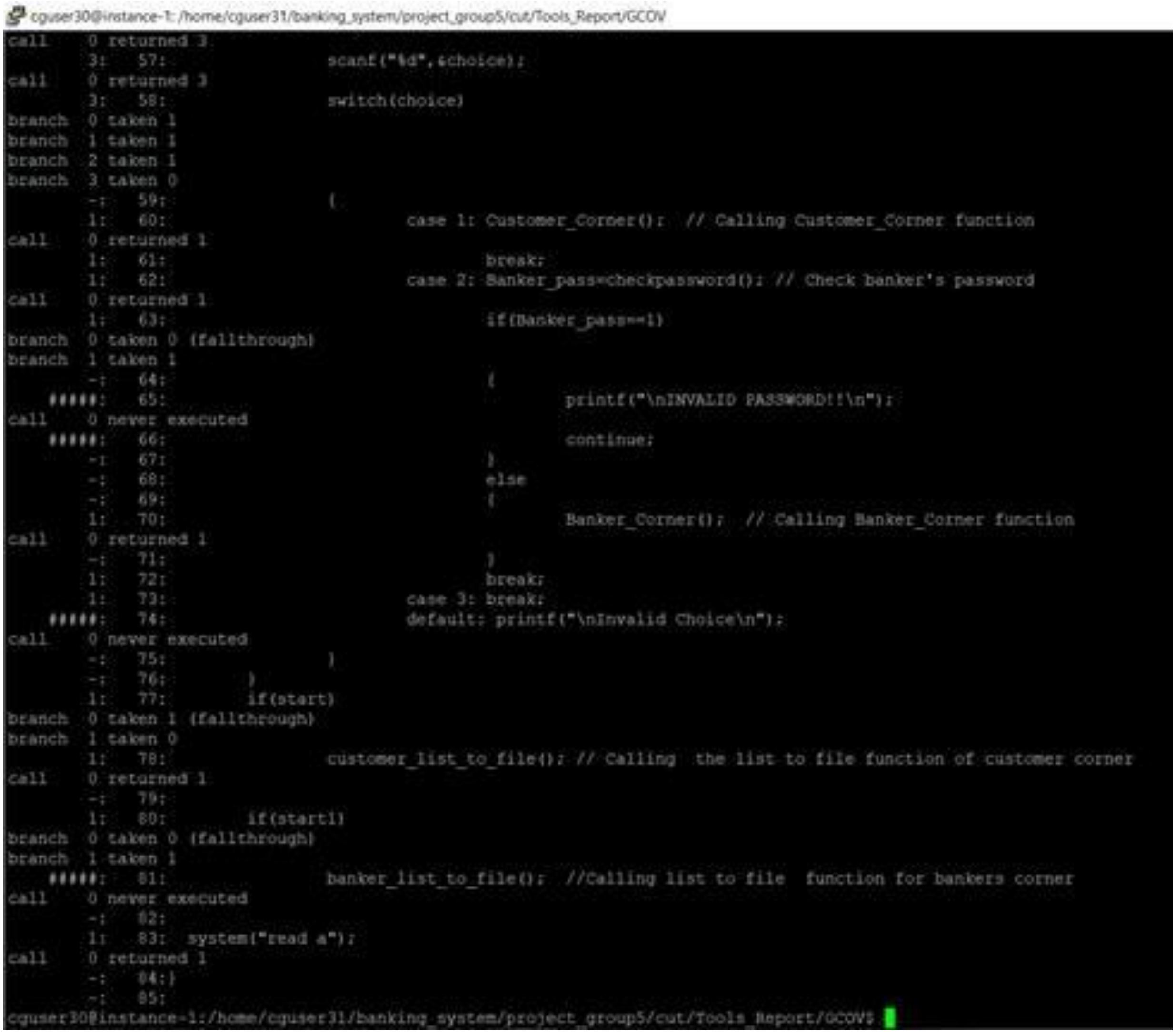
- **func.c**



```
cguser30@instance-1: /home/cguser31/banking_system/project_group5/cut/Tools_Report/GCOV
-rw-r--r-- 1 cguser31 b-77-g5 18404 Aug 29 12:28 submenu.c.gcov
cguser30@instance-1:/home/cguser31/banking_system/project_group5/cut/Tools_Report/GCOV$ cat menu.c.gcov
        -:    0:Source:menu.c
        -:    0:Graph:menu.gcno
        -:    0:Data:menu.gcda
        -:    0:Runs:1
        -:    1:
        -:    2:/**********************************************************************************
        -:    3: *    FILENAME    :   Project_Menu.c
        -:    4: *
        -:    5: *    DESCRIPTION :   This file is uses as Main menu of the banking system which gives
        -:    6: *                    customer corner and bankers corner options.
        -:    7: *
        -:    8: *    REVISION HISTORY
        -:    9: *
        -:   10: *    DATE              NAME              REASON
        -:   11: *
        -:   12: *  ------------------------------------------
        -:   13: *
        -:   14: *    25/05/2022    Username            Menu
        -:   15: *
        -:   16: * ***********************************************************************************/
        -:   17:
        -:   18:#include <stdio.h>   //Including required Header files
        -:   19:#include<termios.h>
        -:   20:#include<stdlib.h>
        -:   21:#include<unistd.h>
        -:   22:#include "Header2.h"
        -:   23:#include "Header1.h"
        -:   24:#include "func.c"
        -:   25:#include "submenu.c"
        -:   26:#include "Password.c"
        -:   27:
        -:   28:
        -:   29:
        -:   30:
function gotoxy called 0 returned 0% blocks executed 0%
    #####:   31:void gotoxy(int x, int y) // Sets co-ordinates in (x,y)
        -:   32:{
    #####:   33:        printf("%c[%d;%df",0x1B,y,x);
call    0 never executed
    #####:   34:}
        -:   35:
function main called 1 returned 100% blocks executed 83%
        1:   36:int main()                // This is the Main function.
        -:   37:{
        1:   38:        customer_file_to_list(); //Calling file to list function of customer corner
call    0 returned 1
       1*:   39:        for(ptr=start;(ptr);ptr=ptr->next)
branch  0 taken 0
```



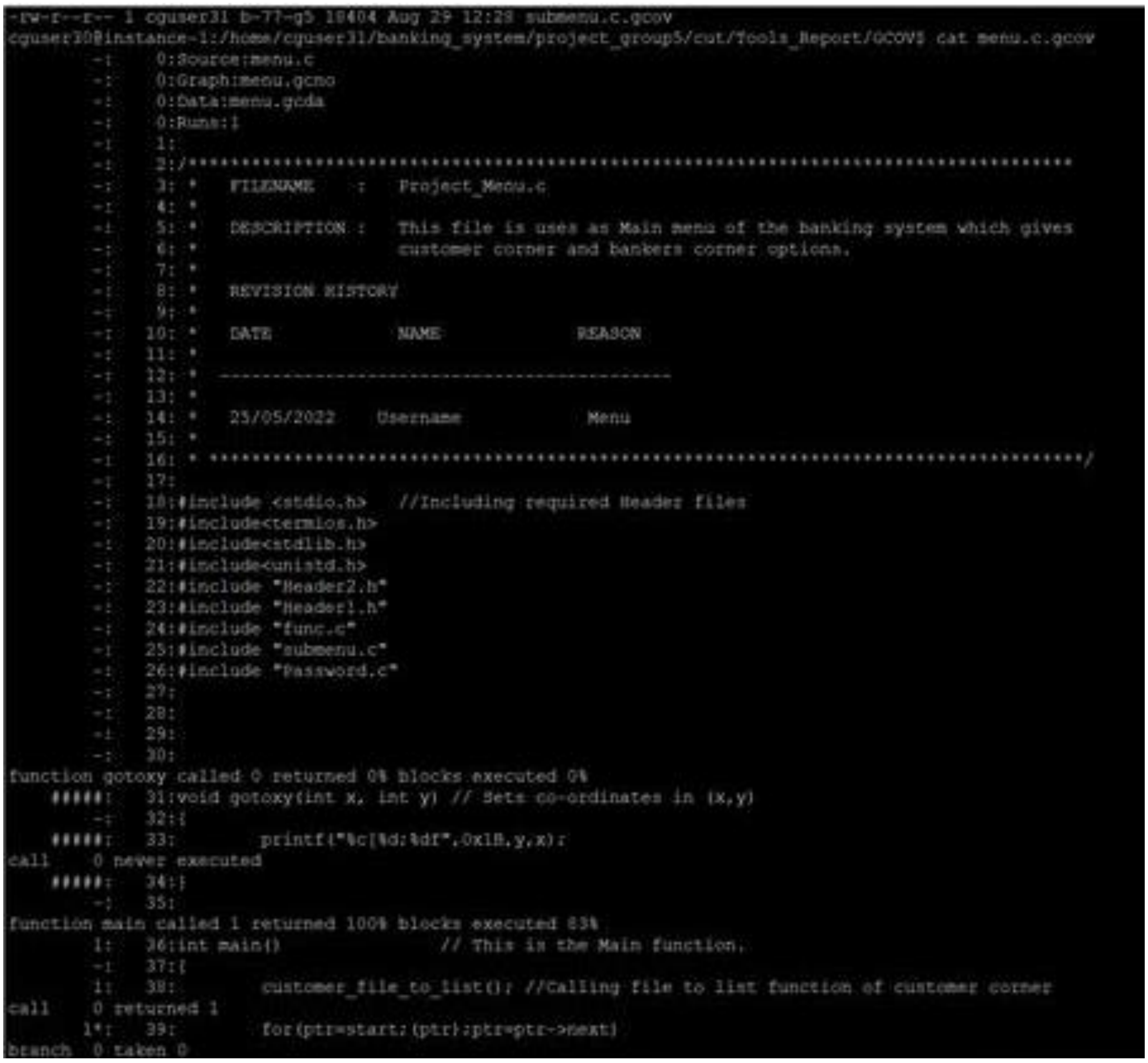
```
cguser30@instance-1: /home/cguser31/banking_system/project_group5/cut/Tools_Report/GCOV
branch  1 taken 1 (fallthrough)
    #####:   40:                printf("\n %s %s %lf\n",ptr->customer_id,ptr->name,ptr->balance);
call    0 never executed
        1:   41:                system("read a");
call    0 returned 1
        1:   42:        banker_file_to_list(); //Calling file to list function for banker corner
call    0 returned 1
        1:   43:        start=ptr=prev=new=NULL; //Declares all pointer of customer structure as NULL
        1:   44:        start1=last1=ptr1=NULL; //Declares all pointer of transaction structure as NULL
        1:   45:        int choice=0;
        1:   46:        int Banker_pass=0;
        4:   47:        while(choice!=3)
branch  0 taken 3
branch  1 taken 1 (fallthrough)
        -:   48:        {
        3:   49:                system("clear");
call    0 returned 3
        3:   50:                printf("\n________________________\n\n");
call    0 returned 3
        3:   51:                printf("\n\n---------WELCOME----------\n");
call    0 returned 3
        3:   52:                printf("\n-----------To-------------\n");
call    0 returned 3
        3:   53:                printf("\n ---The Banking System--- \n\n");
call    0 returned 3
        3:   54:                printf("\n________________________\n\n");
call    0 returned 3
        3:   55:                printf("\n1. Customer Corner \n2. Banker Corner \n3. Exit \n");
call    0 returned 3
        3:   56:                printf("\nEnter Your Choice\n");
call    0 returned 3
        3:   57:                scanf("%d",&choice);
call    0 returned 3
        3:   58:                switch(choice)
branch  0 taken 1
branch  1 taken 1
branch  2 taken 1
branch  3 taken 0
        -:   59:                {
        1:   60:                        case 1: Customer_Corner();  // Calling Customer_Corner function
call    0 returned 1
        1:   61:                                break;
        1:   62:                        case 2: Banker_pass=checkpassword(); // Check banker's password
call    0 returned 1
        1:   63:                                if(Banker_pass==1)
branch  0 taken 0 (fallthrough)
branch  1 taken 1
        -:   64:                                {
    #####:   65:                                        printf("\nINVALID PASSWORD!!\n");
call    0 never executed
```



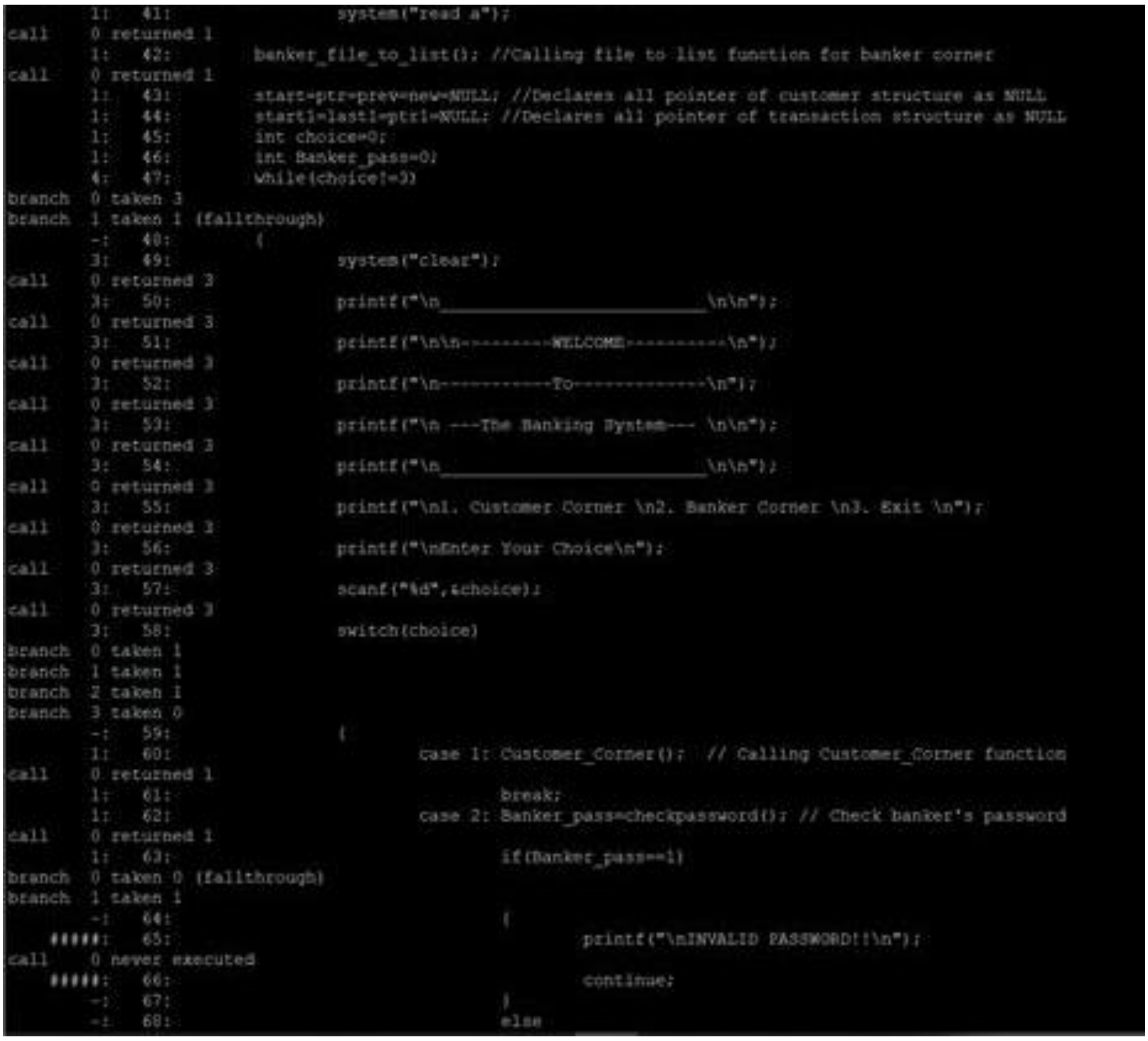
cguser30@instance-1: /home/cguser31/banking_system/project_group5/cut/Tools_Report/GCOV

```
call    0 returned 3
        3:   57:                scanf("%d",&choice);
call    0 returned 3
        3:   58:                switch(choice)
branch  0 taken 1
branch  1 taken 1
branch  2 taken 1
branch  3 taken 0
        -:   59:                {
        1:   60:                        case 1: Customer_Corner();  // Calling Customer_Corner function
call    0 returned 1
        1:   61:                                break;
        1:   62:                        case 2: Banker_pass=checkpassword(); // Check banker's password
call    0 returned 1
        1:   63:                                if(Banker_pass==1)
branch  0 taken 0 (fallthrough)
branch  1 taken 1
        -:   64:                                {
    #####:   65:                                        printf("\nINVALID PASSWORD!!\n");
call    0 never executed
    #####:   66:                                        continue;
        -:   67:                                }
        -:   68:                                else
        -:   69:                                {
        1:   70:                                        Banker_Corner();  // Calling Banker_Corner function
call    0 returned 1
        -:   71:                                }
        1:   72:                                break;
        1:   73:                        case 3: break;
    #####:   74:                        default: printf("\nInvalid Choice\n");
call    0 never executed
        -:   75:                }
        -:   76:        }
        1:   77:        if(start)
branch  0 taken 1 (fallthrough)
branch  1 taken 0
        1:   78:                customer_list_to_file(); // Calling  the list to file function of customer corner
call    0 returned 1
        -:   79:
        1:   80:        if(start1)
branch  0 taken 0 (fallthrough)
branch  1 taken 1
    #####:   81:                banker_list_to_file();  //Calling list to file  function for bankers corner
call    0 never executed
        -:   82:
        1:   83:  system("read a");
call    0 returned 1
        -:   84:}
        -:   85:
cguser30@instance-1:/home/cguser31/banking_system/project_group5/cut/Tools_Report/GCOV$
```

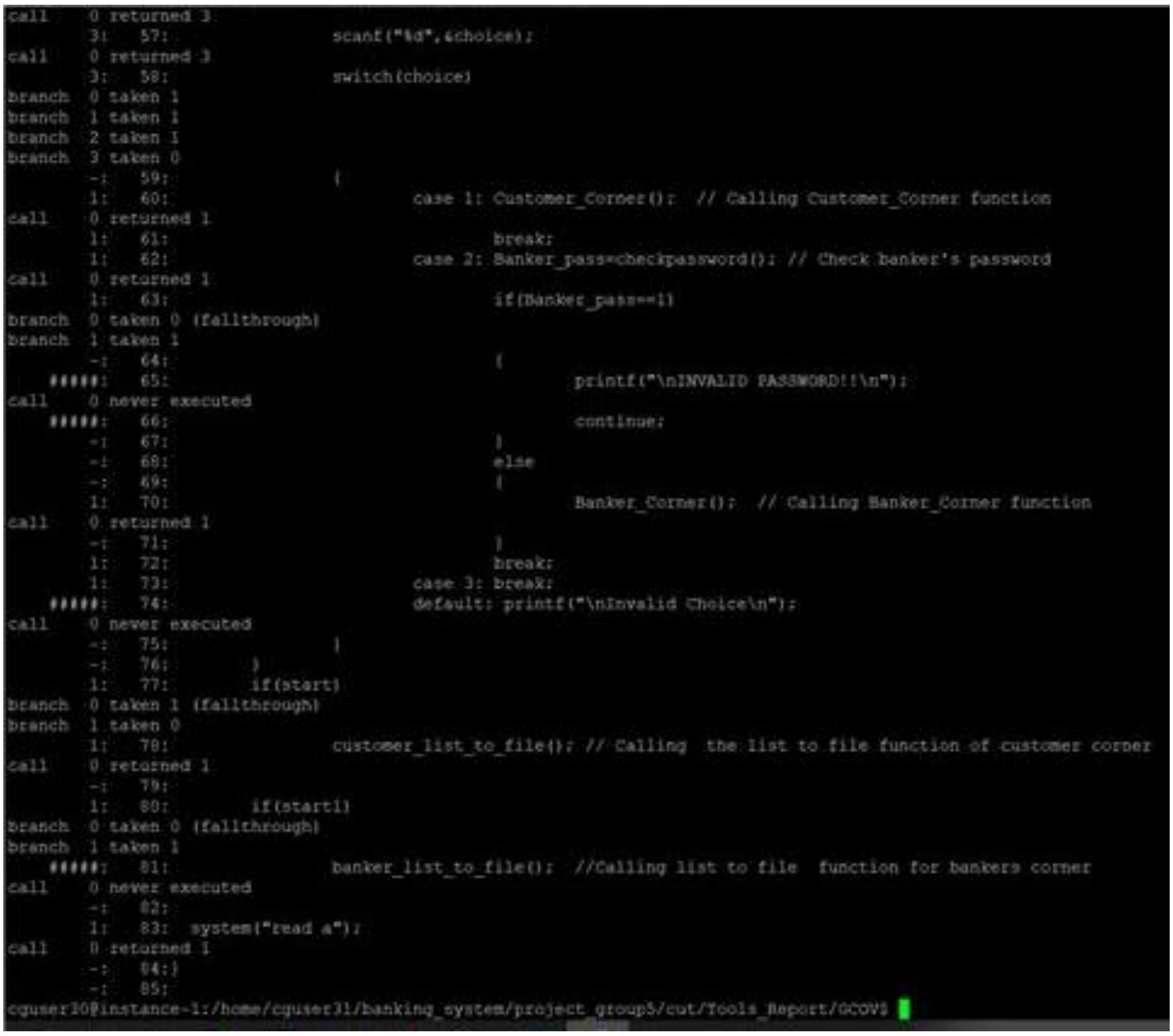
- **menu.c**



```
-rw-r--r-- 1 cguser31 b-77-g5 18404 Aug 29 12:28 submenu.c.gcov
cguser30@instance-1:/home/cguser31/banking_system/project_group5/cut/Tools_Report/GCOV$ cat menu.c.gcov
        -:    0:Source:menu.c
        -:    0:Graph:menu.gcno
        -:    0:Data:menu.gcda
        -:    0:Runs:1
        -:    1:
        -:    2:/****************************************************************************************
        -:    3: *    FILENAME     :    Project_Menu.c
        -:    4: *
        -:    5: *    DESCRIPTION :    This file is uses as Main menu of the banking system which gives
        -:    6: *                     customer corner and bankers corner options.
        -:    7: *
        -:    8: *    REVISION HISTORY
        -:    9: *
        -:   10: *    DATE            NAME              REASON
        -:   11: *
        -:   12: *  -------------------------------------------
        -:   13: *
        -:   14: *    23/05/2022    Username            Menu
        -:   15: *
        -:   16: * ****************************************************************************************/
        -:   17:
        -:   18:#include <stdio.h>    //Including required Header files
        -:   19:#include<termios.h>
        -:   20:#include<stdlib.h>
        -:   21:#include<unistd.h>
        -:   22:#include "Header2.h"
        -:   23:#include "Header1.h"
        -:   24:#include "func.c"
        -:   25:#include "submenu.c"
        -:   26:#include "Password.c"
        -:   27:
        -:   28:
        -:   29:
        -:   30:
function gotoxy called 0 returned 0% blocks executed 0%
    #####:   31:void gotoxy(int x, int y) // Sets co-ordinates in (x,y)
        -:   32:{
    #####:   33:        printf("%c[%d;%df",0x1B,y,x);
call    0 never executed
    #####:   34:}
        -:   35:
function main called 1 returned 100% blocks executed 83%
        1:   36:int main()                // This is the Main function.
        -:   37:{
        1:   38:        customer_file_to_list(); //Calling file to list function of customer corner
call    0 returned 1
       1*:   39:        for(ptr=start;(ptr);ptr=ptr->next)
branch  0 taken 0
```

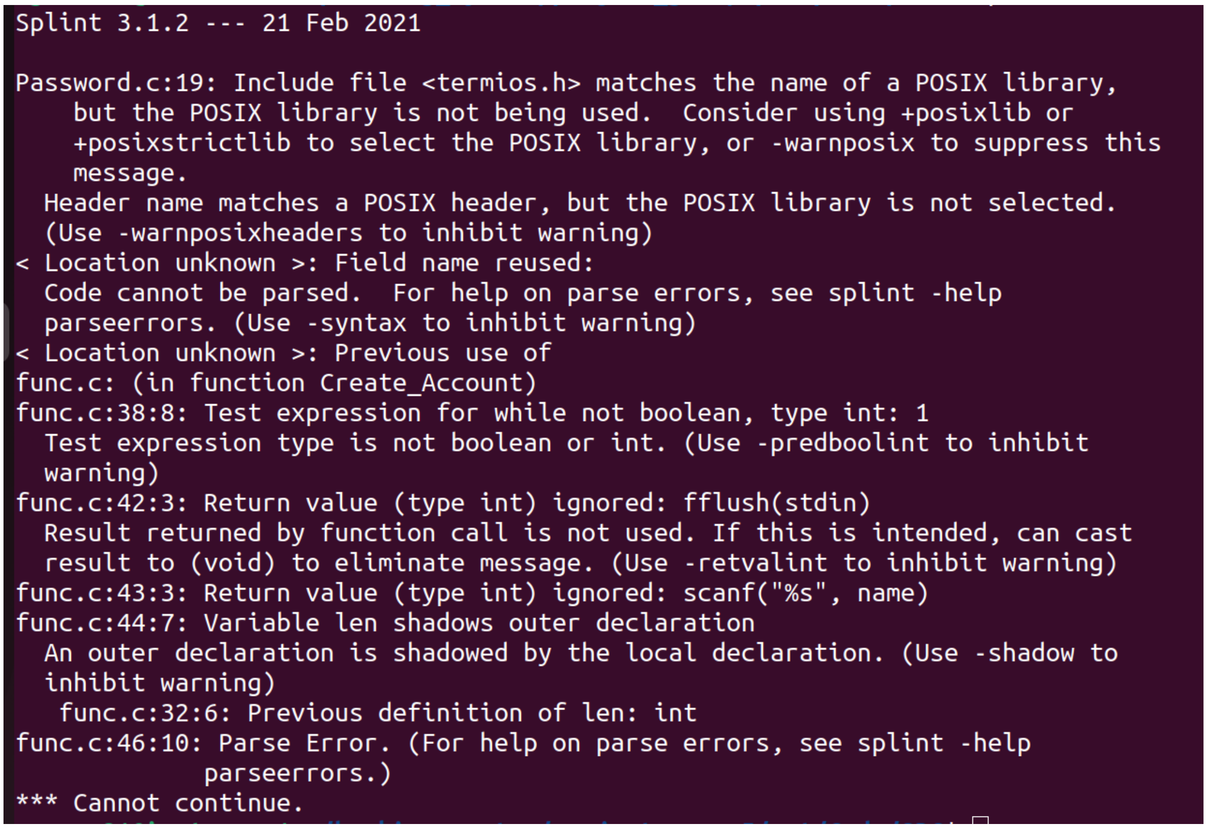


```
        1:   41:                system("read a");
call    0 returned 1
        1:   42:        banker_file_to_list(); //Calling file to list function for banker corner
call    0 returned 1
        1:   43:        start=ptr=prev=new=NULL; //Declares all pointer of customer structure as NULL
        1:   44:        start1=last1=ptr1=NULL; //Declares all pointer of transaction structure as NULL
        1:   45:        int choice=0;
        1:   46:        int Banker_pass=0;
        4:   47:        while(choice!=3)
branch  0 taken 3
branch  1 taken 1 (fallthrough)
        -:   48:        {
        3:   49:                system("clear");
call    0 returned 3
        3:   50:                printf("\n_________________________\n\n");
call    0 returned 3
        3:   51:                printf("\n\n----------WELCOME-----------\n");
call    0 returned 3
        3:   52:                printf("\n------------To--------------\n");
call    0 returned 3
        3:   53:                printf("\n ---The Banking System--- \n\n");
call    0 returned 3
        3:   54:                printf("\n_________________________\n\n");
call    0 returned 3
        3:   55:                printf("\n1. Customer Corner \n2. Banker Corner \n3. Exit \n");
call    0 returned 3
        3:   56:                printf("\nEnter Your Choice\n");
call    0 returned 3
        3:   57:                scanf("%d",&choice);
call    0 returned 3
        3:   58:                switch(choice)
branch  0 taken 1
branch  1 taken 1
branch  2 taken 1
branch  3 taken 0
        -:   59:                {
        1:   60:                        case 1: Customer_Corner();  // Calling Customer_Corner function
call    0 returned 1
        1:   61:                                break;
        1:   62:                        case 2: Banker_pass=checkpassword(); // Check banker's password
call    0 returned 1
        1:   63:                                if(Banker_pass==1)
branch  0 taken 0 (fallthrough)
branch  1 taken 1
        -:   64:                                {
    #####:   65:                                        printf("\nINVALID PASSWORD!!\n");
call    0 never executed
    #####:   66:                                        continue;
        -:   67:                                }
        -:   68:                                else
```



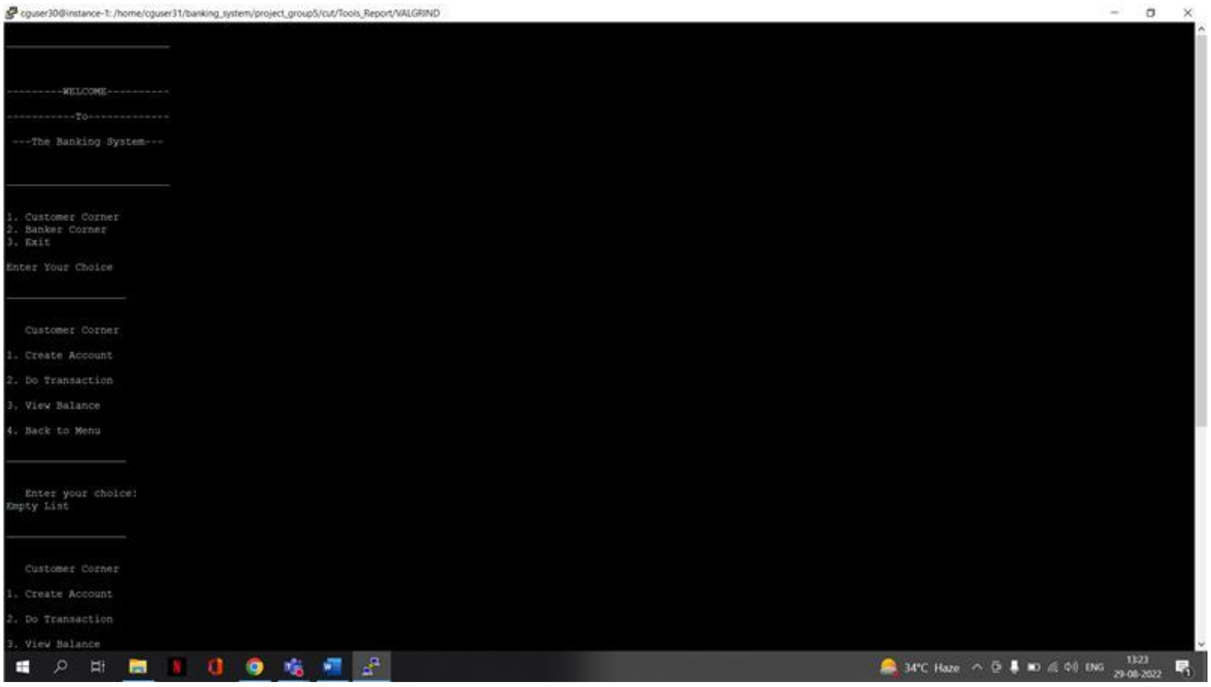
```
call    0 returned 3
        3:   57:                scanf("%d",&choice);
call    0 returned 3
        3:   58:                switch(choice)
branch  0 taken 1
branch  1 taken 1
branch  2 taken 1
branch  3 taken 0
        -:   59:                {
        1:   60:                        case 1: Customer_Corner();  // Calling Customer_Corner function
call    0 returned 1
        1:   61:                                break;
        1:   62:                        case 2: Banker_pass=checkpassword(); // Check banker's password
call    0 returned 1
        1:   63:                                if(Banker_pass==1)
branch  0 taken 0 (fallthrough)
branch  1 taken 1
        -:   64:                                {
    #####:   65:                                        printf("\nINVALID PASSWORD!!\n");
call    0 never executed
    #####:   66:                                        continue;
        -:   67:                                }
        -:   68:                                else
        -:   69:                                {
        1:   70:                                        Banker_Corner();  // Calling Banker_Corner function
call    0 returned 1
        -:   71:                                }
        1:   72:                                break;
        1:   73:                        case 3: break;
    #####:   74:                        default: printf("\nInvalid Choice\n");
call    0 never executed
        -:   75:                }
        -:   76:        }
        1:   77:        if(start)
branch  0 taken 1 (fallthrough)
branch  1 taken 0
        1:   78:                customer_list_to_file(); // Calling  the list to file function of customer corner
call    0 returned 1
        -:   79:
        1:   80:        if(start1)
branch  0 taken 0 (fallthrough)
branch  1 taken 1
    #####:   81:                banker_list_to_file();  //Calling list to file  function for bankers corner
call    0 never executed
        -:   82:
        1:   83:  system("read a");
call    0 returned 1
        -:   84:}
        -:   85:
cguser30@instance-1:/home/cguser31/banking_system/project_group5/cut/Tools_Report/GCOV$
```

5.2 Splint Report:

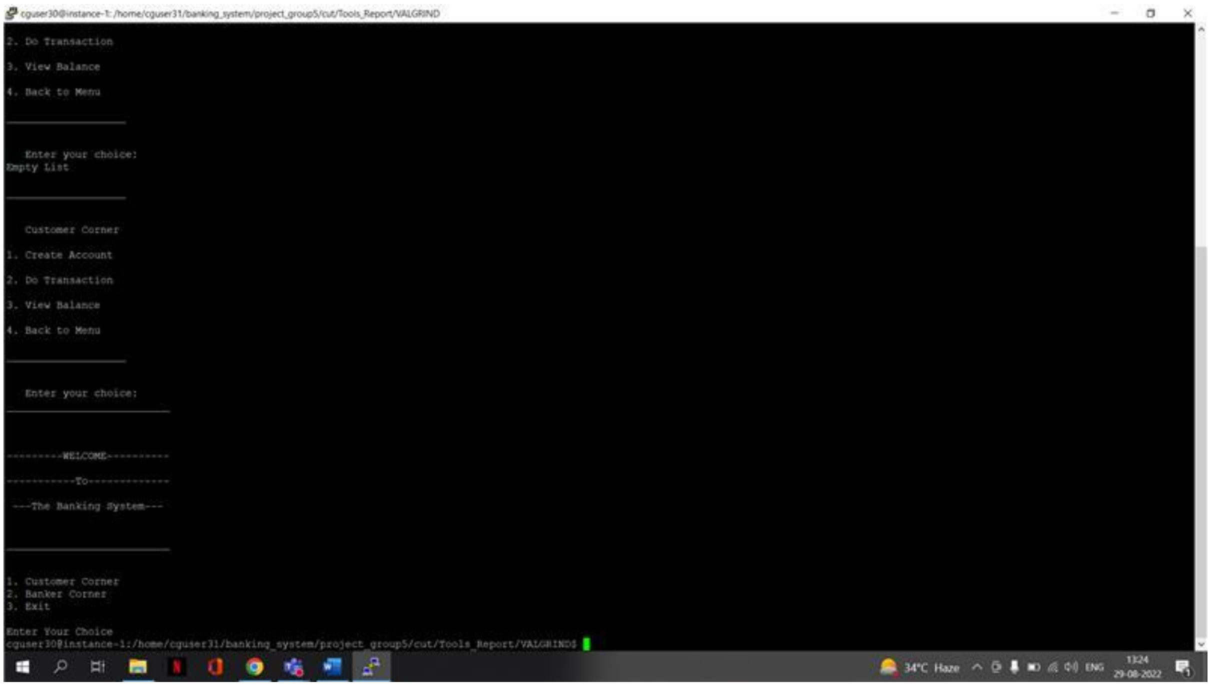


```
Splint 3.1.2 --- 21 Feb 2021

Password.c:19: Include file <termios.h> matches the name of a POSIX library,
    but the POSIX library is not being used.  Consider using +posixlib or
    +posixstrictlib to select the POSIX library, or -warnposix to suppress this
    message.
  Header name matches a POSIX header, but the POSIX library is not selected.
  (Use -warnposixheaders to inhibit warning)
< Location unknown >: Field name reused:
  Code cannot be parsed.  For help on parse errors, see splint -help
  parseerrors. (Use -syntax to inhibit warning)
< Location unknown >: Previous use of
func.c: (in function Create_Account)
func.c:38:8: Test expression for while not boolean, type int: 1
  Test expression type is not boolean or int. (Use -predboolint to inhibit
  warning)
func.c:42:3: Return value (type int) ignored: fflush(stdin)
  Result returned by function call is not used. If this is intended, can cast
  result to (void) to eliminate message. (Use -retvalint to inhibit warning)
func.c:43:3: Return value (type int) ignored: scanf("%s", name)
func.c:44:7: Variable len shadows outer declaration
  An outer declaration is shadowed by the local declaration. (Use -shadow to
  inhibit warning)
   func.c:32:6: Previous definition of len: int
func.c:46:10: Parse Error. (For help on parse errors, see splint -help
               parseerrors.)
*** Cannot continue.
```

5.3 Valgrind Report:

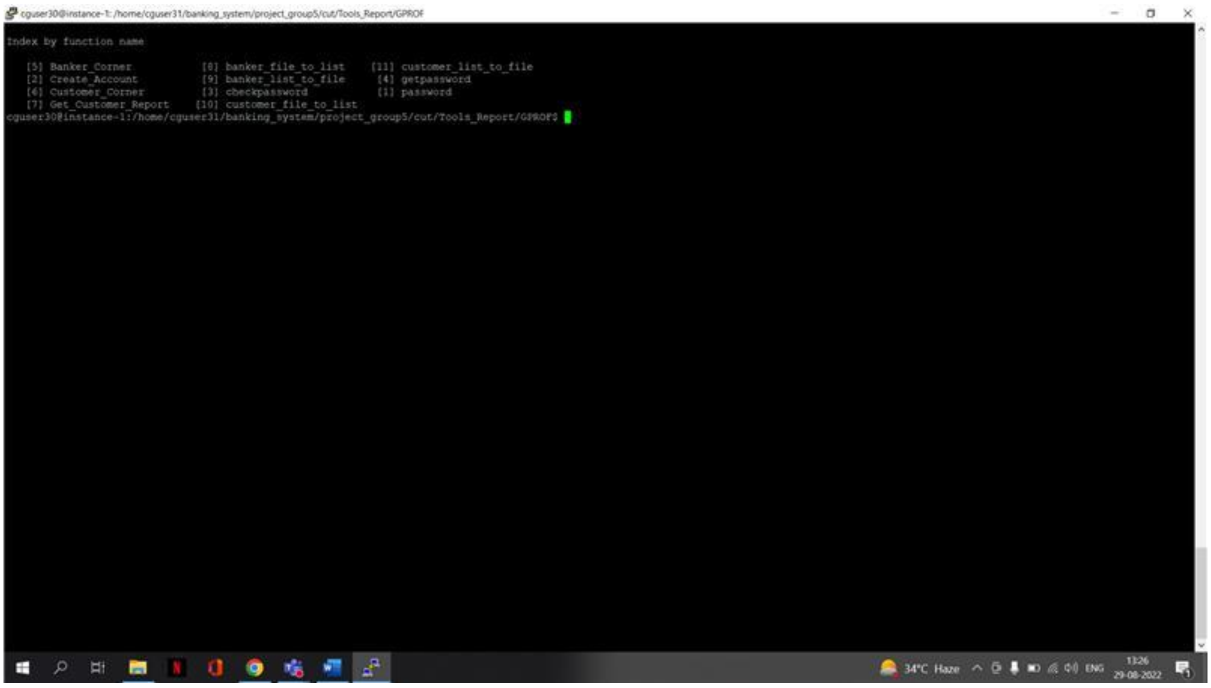


```
cguser30@instance-1: /home/cguser31/banking_system/project_group5/cut/Tools_Report/VALGRIND

_________________________


----------WELCOME----------

------------To-------------

 ---The Banking System---


_________________________


1. Customer Corner
2. Banker Corner
3. Exit

Enter Your Choice


___________________


   Customer Corner

1. Create Account

2. Do Transaction

3. View Balance

4. Back to Menu


___________________


   Enter your choice:
Empty List


___________________


   Customer Corner

1. Create Account

2. Do Transaction

3. View Balance
```



```
cguser30@instance-1: /home/cguser31/banking_system/project_group5/cut/Tools_Report/VALGRIND
2. Do Transaction

3. View Balance

4. Back to Menu

___________________


   Enter your choice:
Empty List

___________________


   Customer Corner

1. Create Account

2. Do Transaction

3. View Balance

4. Back to Menu

___________________


   Enter your choice:
__________________________



----------WELCOME----------

------------To-------------

 ---The Banking System---



__________________________


1. Customer Corner
2. Banker Corner
3. Exit

Enter Your Choice
cguser30@instance-1:/home/cguser31/banking_system/project_group5/cut/Tools_Report/VALGRIND$
```

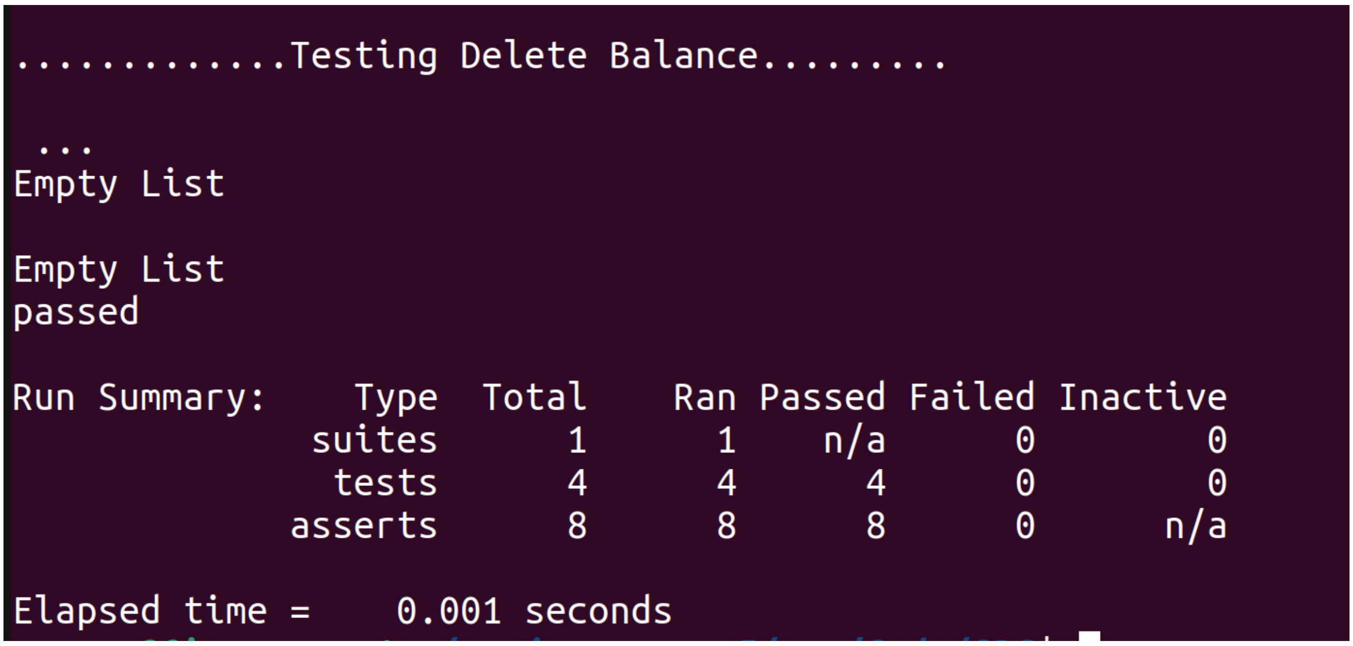
5.4 Gprof



```
cguser30@instance-1: /home/cguser31/banking_system/project_group5/cut/Tools_Report/GPROF
Index by function name

   [5] Banker_Corner           [8] banker_file_to_list    [11] customer_list_to_file
   [2] Create_Account          [9] banker_list_to_file     [4] getpassword
   [6] Customer_Corner         [3] checkpassword           [1] password
   [7] Get_Customer_Report    [10] customer_file_to_list
cguser30@instance-1:/home/cguser31/banking_system/project_group5/cut/Tools_Report/GPROF$
```

6. Testing

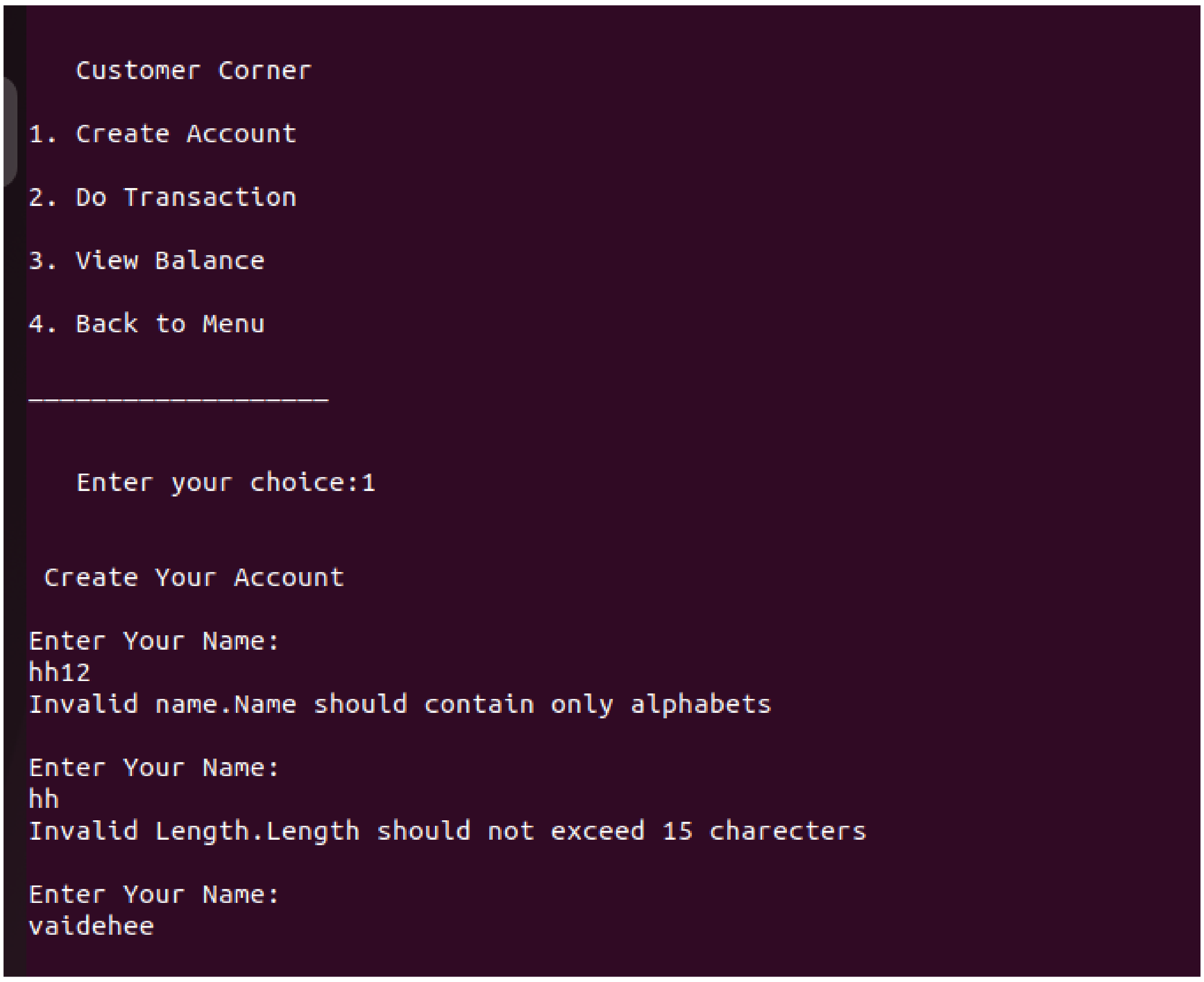
6.1 Unit Testing Report



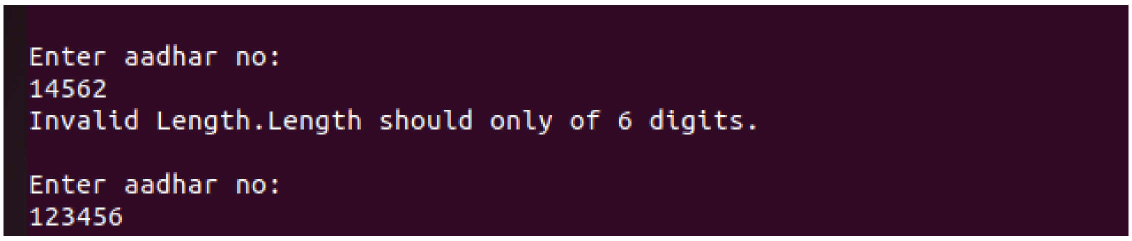
```
.............Testing Delete Balance.........


 ...
Empty List

Empty List
passed

Run Summary:    Type  Total    Ran Passed Failed Inactive
              suites      1      1    n/a      0        0
               tests      4      4      4      0        0
             asserts      8      8      8      0      n/a

Elapsed time =    0.001 seconds
```

6.2 Integration Testing Report

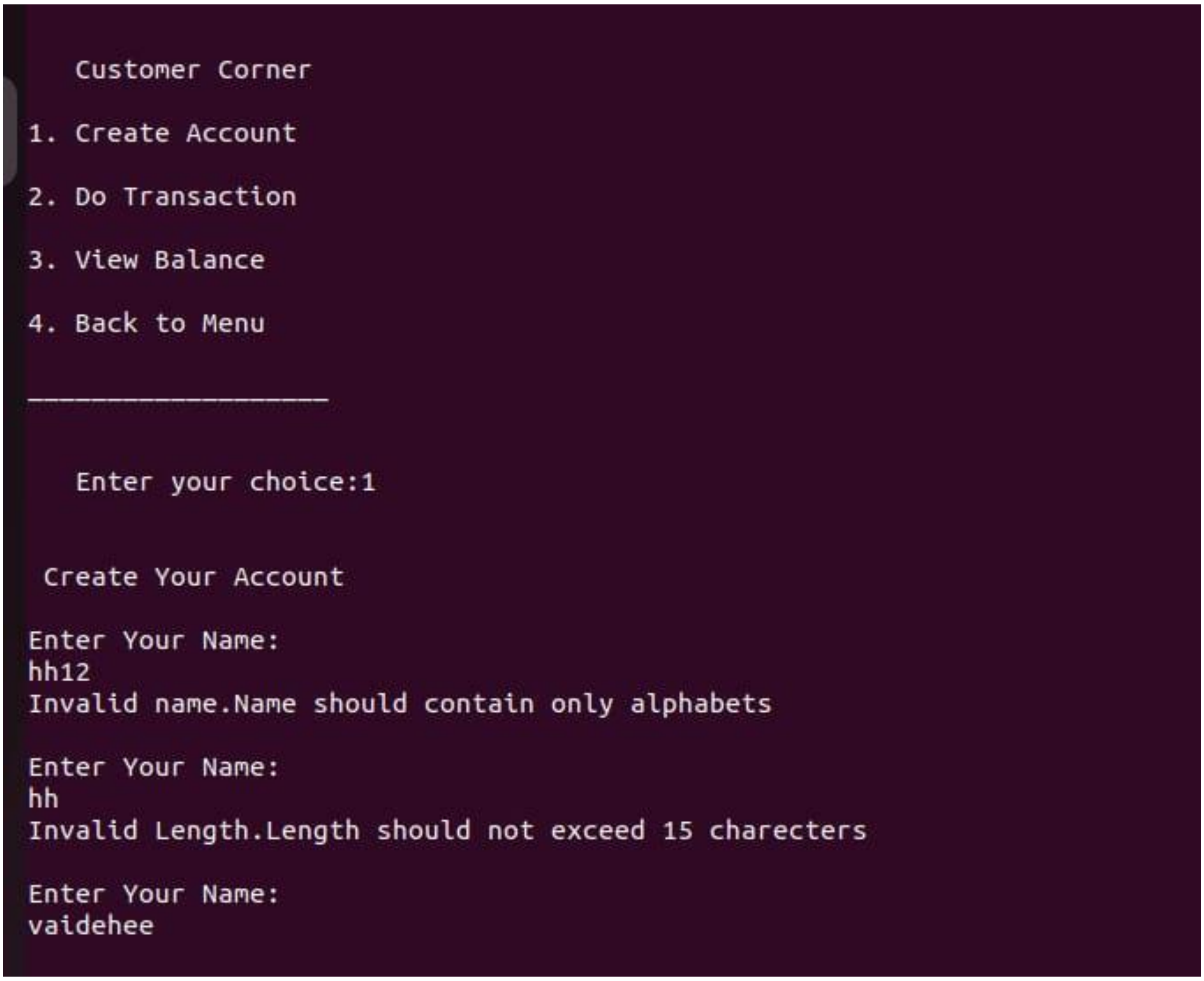
IT_CASE 1: For creating account



```
   Customer Corner

1. Create Account

2. Do Transaction

3. View Balance

4. Back to Menu

___________________


   Enter your choice:1


 Create Your Account

Enter Your Name:
hh12
Invalid name.Name should contain only alphabets

Enter Your Name:
hh
Invalid Length.Length should not exceed 15 charecters

Enter Your Name:
vaidehee
```

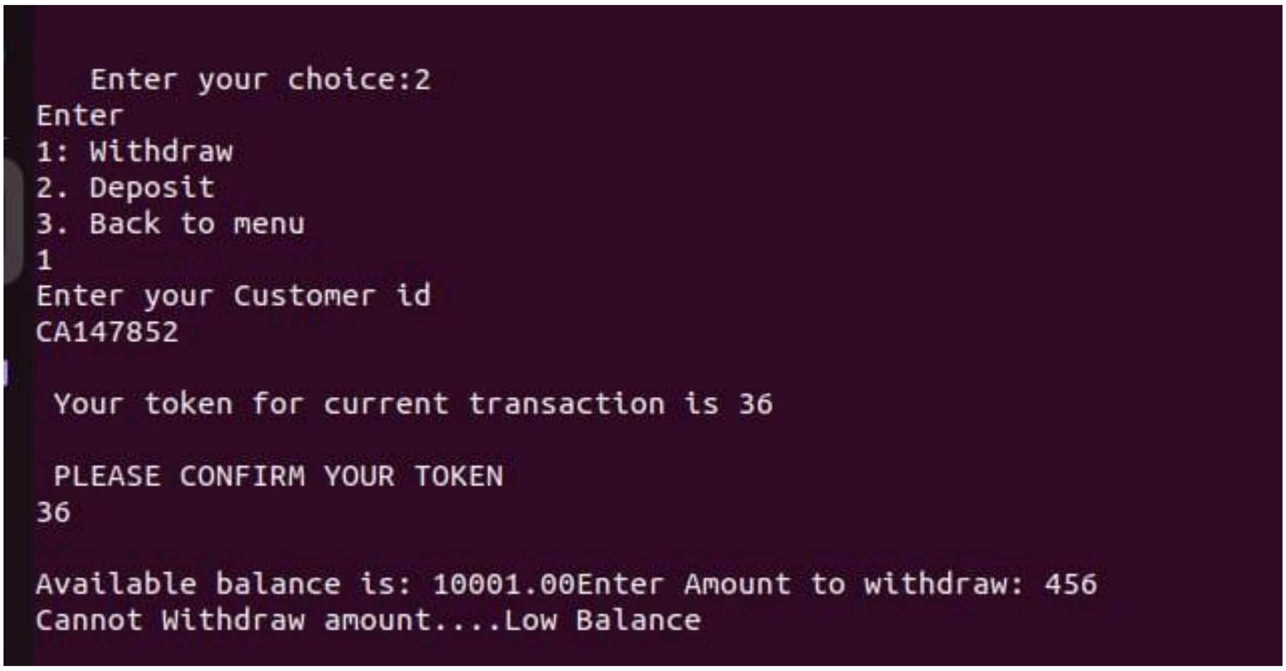


```
Enter aadhar no:
14562
Invalid Length.Length should only of 6 digits.

Enter aadhar no:
123456
```



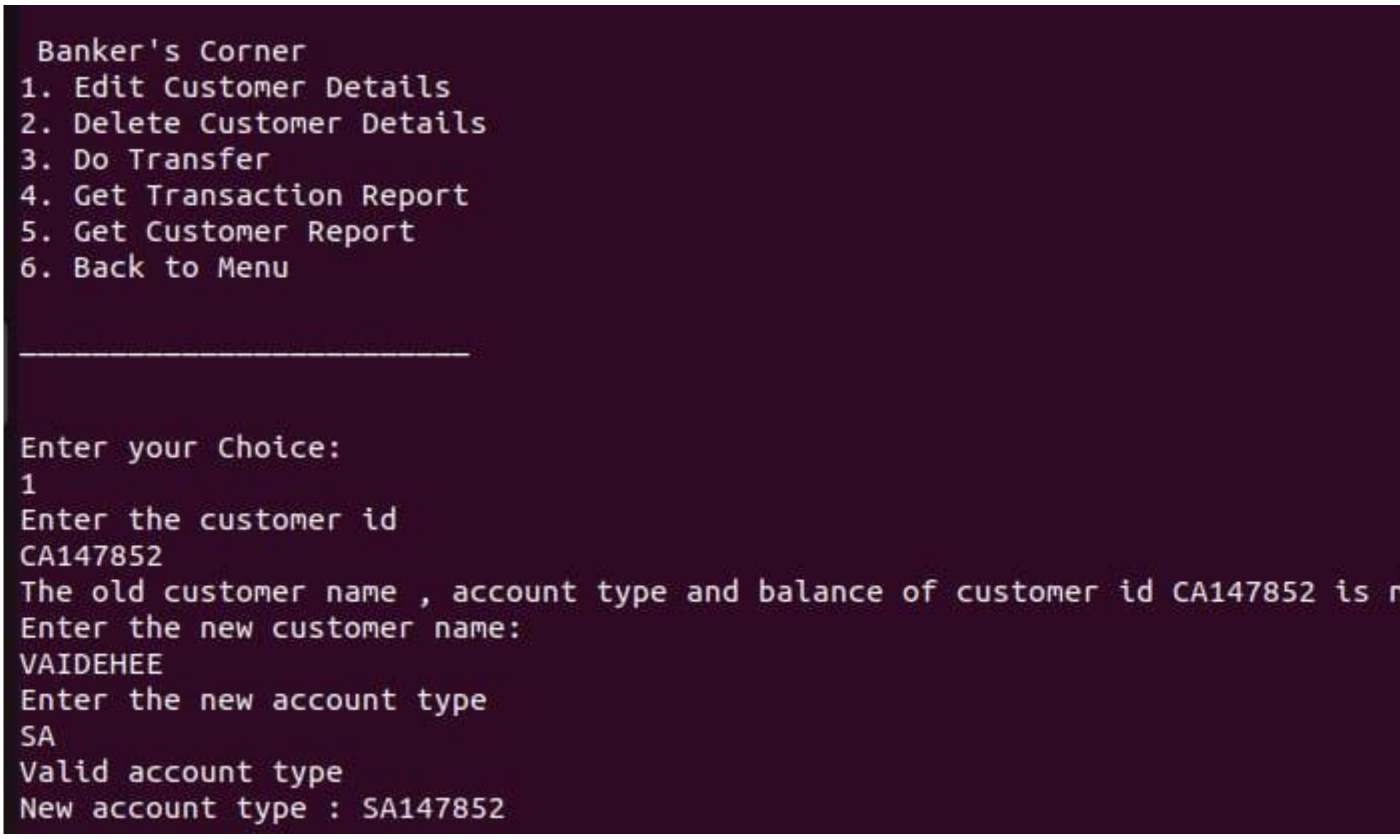
```
   Customer Corner

1. Create Account

2. Do Transaction

3. View Balance

4. Back to Menu

___________________


   Enter your choice:1


 Create Your Account

Enter Your Name:
hh12
Invalid name.Name should contain only alphabets

Enter Your Name:
hh
Invalid Length.Length should not exceed 15 charecters

Enter Your Name:
vaidehee
```

Case2 : to do transaction



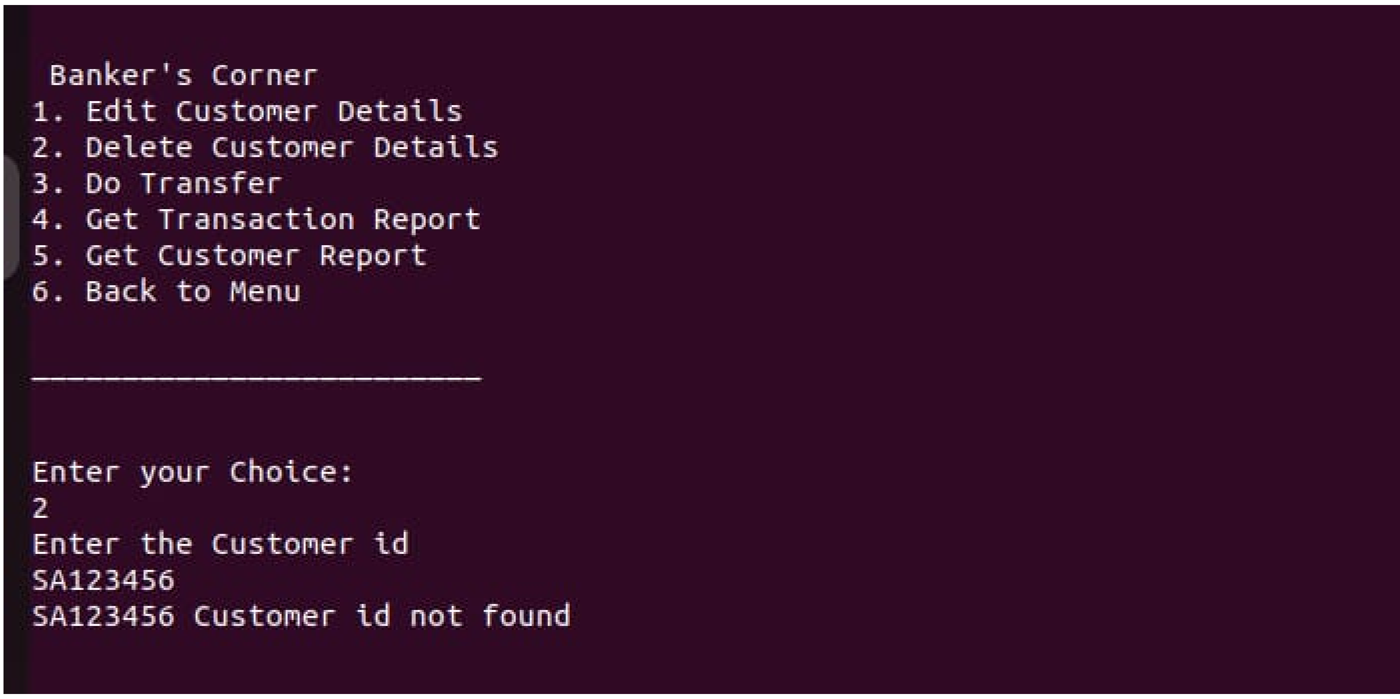
```
   Enter your choice:2
Enter
1: Withdraw
2. Deposit
3. Back to menu
1
Enter your Customer id
CA147852

 Your token for current transaction is 36

 PLEASE CONFIRM YOUR TOKEN
36

Available balance is: 10001.00Enter Amount to withdraw: 456
Cannot Withdraw amount....Low Balance
```

Case3 : Edit details of customer by banker



```
 Banker's Corner
1. Edit Customer Details
2. Delete Customer Details
3. Do Transfer
4. Get Transaction Report
5. Get Customer Report
6. Back to Menu


_________________________


Enter your Choice:
1
Enter the customer id
CA147852
The old customer name , account type and balance of customer id CA147852 is r
Enter the new customer name:
VAIDEHEE
Enter the new account type
SA
Valid account type
New account type : SA147852
```

Case4 : To delete account



```
 Banker's Corner
1. Edit Customer Details
2. Delete Customer Details
3. Do Transfer
4. Get Transaction Report
5. Get Customer Report
6. Back to Menu

_________________________


Enter your Choice:
2
Enter the Customer id
SA123456
SA123456 Customer id not found
```

7.Requirement Traceability Matrix (RTM):

Requirement.	Design Mapping	Code Mapping	UT Mapping	IT Mapping
BS_01	a	customer_corner		
BS_02	b	banker_corner		
BS_03	c	create_account	Test_case_1	IT_01
BS_04	d	do_transaction		IT_02
BS_05	e	view_balance	Test_case_3	
BS_06	f	edit_customer	Test_case_2	IT_03
BS_07	g	delete_customer	Test_case_4	IT_04
BS_08	h	do_transfer		
BS_09	i	get_transaction_report		

8. Reference: -

The references are:

- https://www.programiz.com/dsa/linked-list
- https://www.javatpoint.com/file-handling-in-c
- https://www.educative.io/answers/how-to-create-a-simple-thread-in-c