

High Level Design & Low Level Design

**Document Control :**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Revision History** | | | | | | | | |
|  |  |  | |  |  |  |  |  |
| **Date** | **Version** | **Author** | **Brief Description of Changes** | | | | **Approver Signature** | |
| 07.10.2022 | 1.0 | Group 7 |  | | | |  | |
|  |  |  |  | | | |  | |
|  |  |  |  | | | |  | |

**Index**

1. Introduction ------------------------------------------------ 4

1.1 Intended audience ------------------------------------------------ 4

1.2 Project purpose ------------------------------------------------ 4

1.3 Key project objective ------------------------------------------------ 4

1.4 Project scope and limitation ------------------------------------------------ 4

1.5 Functional overview ------------------------------------------------ 4

1.5.1Header files ------------------------------------------------ 4

1.5.2 Functions ------------------------------------------------ 5

2. Design overview ------------------------------------------------ 6

2.1 Design objective ------------------------------------------------ 7

2.2 Design alternative ------------------------------------------------ 7

2.3 User interface paradigms ------------------------------------------------ 7

2.4 Error detection/ Exceptional Handling ------------------------------------------------ 8

2.5 Performance ------------------------------------------------ 8

2.6 Maintenance ------------------------------------------------ 8

3. System architecture ------------------------------------------------ 8

3.1 Structure ------------------------------------------------ 8

4. Detailed system design ------------------------------------------------ 9

5. Environment description ------------------------------------------------10

5.1 Time zone support ------------------------------------------------10

5.2 Language support ------------------------------------------------10

5.3 User desktop requirement ------------------------------------------------10

5.4 Server-side requirement ------------------------------------------------11

5.4.1Deployment consideration ------------------------------------------------9

5.4.2 Application server disk space ------------------------------------------------10

5.4.3 Database server disk space ------------------------------------------------11

5.4.4 Integration requirements ------------------------------------------------11

5.4.5 Network ------------------------------------------------11

5.5 Configuration ------------------------------------------------11

5.5.1 Operating system ------------------------------------------------11

6. Reference ------------------------------------------------11

1. **Introduction: -**
   1. **Intended Audience: -**

The target audience for this project comprises of both the individuals working for the banking organisations and their clients, who are the people who use their services.

* 1. **Project Purpose: -**

The banking system has features for account opening, deposits, and withdrawals and was created exclusively for banking applications. Second, the programme only enables authorised bank employees to transfer funds across accounts held by the same bank. Additionally, this system enables banking staff to create client reports and transaction reports as well as edit and delete customer information. Customers can use this system to open an account, make deposits or withdrawals, and view their balance.

* 1. **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. **Project scope and limitation: -**

The scope of the banking application's functionality, in particular, to guarantee seamless banking operations The application's primary goal is to automate system records. It offers the fundamental services needed by the bank to maintain 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. Through this application, one may also monitor their ongoing transactions and keep a check on their accounts. It’s not only useful for the customers but also for the admin.

* 1. **Functional Overview: -**

1.5.1 Following header files are included in the program:

* #include <stdio.h>
* #include <string.h>
* #include <stdlib.h>
* #include <time.h>
* #include <pthread.h>
  + 1. Login
* Customer : Customer logins by entering customer’s account number & a login password.
* Banker: Banker logins by entering banker’s id & a login password.
  + 1. Customer’s Corner
       1. Create Account

The customer can create account by entering aadhar number and account type (either SA((Savings Account) or CA(Current Account)).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
    - 1. Do\_Transaction

1. Deposit: This function will add the deposited amount to the current balance.
2. 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.

* + - 1. View\_Balance

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

* + 1. Banker’s Corner
       1. Edit\_Customer\_Details

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

* + - 1. Delete\_Customer\_Details

The entire customer record is deleted from database.

* + - 1. Do\_Transfer

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

* + - 1. Get\_Transaction\_Report

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

* + - 1. Get\_Customer\_Report

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

# Design Overview: -

Banking Application comprises of the following modules:

|  |  |
| --- | --- |
| Name of the Module | Create account and Do transaction |
| Handled by | Anisha Surana |
| Description | It will create account and do transfer from one account to another. |

|  |  |
| --- | --- |
| Name of the Module | Edit and Do Transfer |
| Handled by | Anshul Kumar |
| 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 | Abhinav Kumar Singh |
| 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 | Sukanya Sahoo |
| 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 | Yashwant Tadepalli |
| Description | It will get transaction report and and file\_to\_list and list\_to\_file will read and write from file. |

## 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
  1. **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.

### 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.

### 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 12 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”

### 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

### Maintenance: -

Very little maintenance should be required for this setup. An initial configuration will be the only system required interaction after system is put together. The only other user maintenance would be any changes to settings after setup, and any specified special cases where user settings or history need to be changed. Physical maintenance on the system’s parts may be required, and would result in temporary loss of data or Internet. Upgrades of hardware and software should have little effect on this project but may result in downtime.

1. **SYSTEM ARCHITECTURE: -**

**3.1 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.

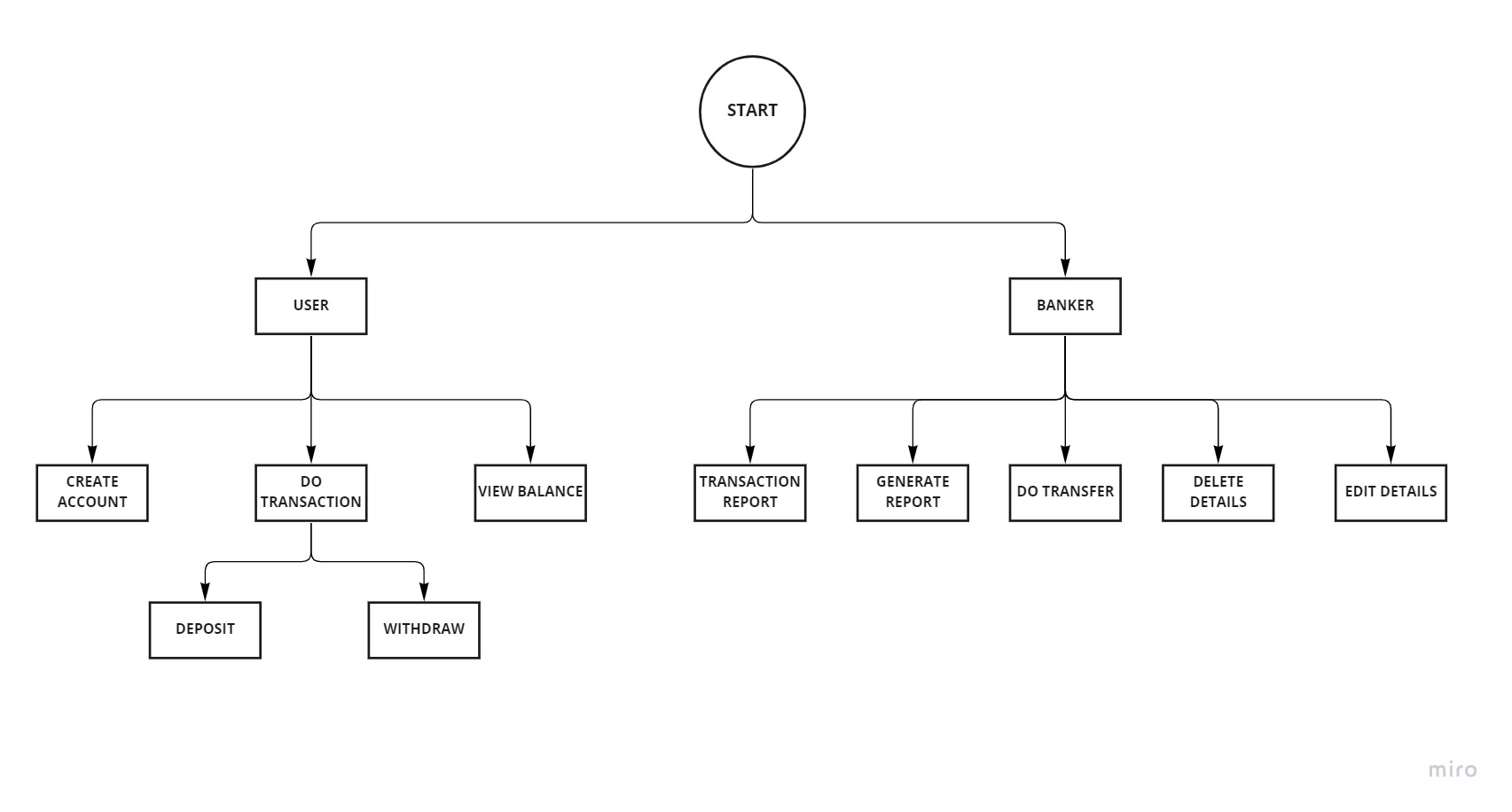
* Transaction

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.

1. **Detailed System Design**

**Activity Diagram:**



1. **Environment Description: -**
   1. **Time Zone Support: -** IST- Kolkata
   2. **Language Support: -** English
   3. **User Desktop Requirements: -**

* 64-bit processor, 1 GHz or faster
* At least 10 GB free hard drive space
* At least 1 GB RAM **Server**
  1. **-Side Requirements: -**
* 64-bit processor, 1 GHz or faster
* At least 2 GB free hard drive space
* At least 1GB RAM
  1. Deployment Considerations: -
  + Local storage is used
  + No network latency to consider
  + To scale buy a bigger CPU, more memory, larger hard drive, or additional hardware

**5.4.2**. Application Server Disk Space: -

No such disk space is required as the program is fully functional on online

IDE(s) as well. Local Operating System is required and two txt file to store the

records of processes.

**5.4.3**. Database Server Disk Space: -

No such disk space is required as the program is fully functional on online IDE(s) as

well. Local Operating System is required and two txt file to store the records of

processes.

**5.4.4**. Integration Requirements: -

* Language: - C
* Tools: - Valgrind, Makefile ,Cunit,gprof,splint,gcov
* Complier: - gcc
* Linux Environment

**5.4.5**. Network: - End to End

**5.5 Configuration: -**

**5.5.1**. Operating System: - Linux environment

1. **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