**Centralized Multi-User Concurrent Bank Account Manager**

Name : Haravind Rajula

ID : JC68015

How to compile and run the code :

1. Firstly to compile the server and client code, make use of makefile
2. Open the terminal in the src folder
3. Run the commands :   
   a. Make clean  
   b. make compile
4. Open another terminal to execute server code and run the ./server command
5. Similar to server code, open an terminal and execute the ./client command

Program implementation:

Client side :

1. Firstly it reads the transaction file line by line.
2. A separate block is created for the socket execution, connect() and sending the content to the server.
3. Implemented the if else blocks in each stage to check the correct execution
4. Loggers are implemented to track each level of execution.
5. When data is sent to the server, the content/data is logged and displayed on terminal.

Server side:

1. Created models for the customers/account holders and the incoming data from the client and those data will be stored on the objects of these models
2. A separate functional block is designed for the reading of records.txt and get stored in the customer model.
3. Initialised the pthread\_mutex thread array with zero so that it doesn’t cause the issues when threads are executing if in case with the previous vaues.
4. Socket(), binding() and listening() are used. And listening queue is implemented till the SOMAXCONN.
5. Threads are created for 20000 arraysize. To accommodate the incoming requests. And each thread is executed respectively.
6. While accept() the transaction block is designed to implement the core logic
7. In transaction block, recv() is used for each incoming data catching
8. The incoming data is partioned in such a way to content model
9. Used for loop to find the model containing the same ID and perform the withdraw and deposit action.
10. Before and after executing the block, pthread mutex lock and unlock is called to secure the segment.
11. At the end closing the sockets and files opened.