# **Project Design Document**

## **Overview**

The project is about creating a multithreading TCP client-server system that runs to manage user accounts, money transfers, and authentication procedures.

**Components**

1. **Client**
   * **Responsibility**: Engages in money operations, account administration, user registration, and login interactions with the server.
   * **Features**: Registration, Login, Lodgement, User Listing, Password Update, Transaction Viewing, Logout.
2. **Server**
   * **Responsibility**: Responds to incoming client requests, maintains transaction records, and oversees user accounts.
   * **Features**: User Registration, Login Authentication, Lodgement, Password Update, User Listing, Transaction Processing.
3. **Account**
   * **Purpose**: This represents a user account, storing the users’ details and balance.
   * **Attributes**: The attributes are User ID, User Details (ppsn, name, email, password, address), Current Balance.
   * **Features**: Getter and Setter methods for user details and balance management.
4. **User**
   * **Purpose**: Stores user data on its own.
   * **Attributes**: Name, ID, Email, Password, Address, Balance.
   * **Features**: Getter methods for user details.

**Client Implementation**

* **User Interface**: Offers a GUI interface for user interactions.
* **Login & Registration**: Once a user registers and makes an account then granted Login functionalities.
* **Account Operations**: Allows users to lodge money, view registered users, transfer funds, view transactions, and update passwords.

**Server Implementation**

* **Socket Handling**: Waits for new connections and starts a brand-new thread for every client.
* **Account Management**: Handles financial transactions, keeps track of transactions, manages user accounts, and confirms logins.
* **Data Persistence**: Stores user account information in a text-based database file.

**Communication Protocol**

* **Client-Server:** Preset protocol messages are used along with object streams for communication below are the commands.
* **Protocol Commands**: REGISTER, LOGIN, LODGE, TRANSFER, UPDATE\_PASSWORD, RETRIEVE\_USERS, VIEW\_TRANSACTIONS, LOGOUT.

**Error Handling**

* **Client Side**: Handles invalid inputs, login failures, and unsuccessful transactions in case of making a mistake.
* **Server Side**: Handles cases of recipient not found, insufficient balances, mismatched passwords, and duplicate IDs or emails.

**Enhancements**

* **Encryption**: Implements encryption as to allow for secure data transmission.
* **Improved User Experience**: Provide a graphical user interface for the application running on the client side.
* **Optimization**: Improve server performance to manage multiple connections running at once.

**Conclusion**

In summary, the strong TCP-based client-server application described in this Project Design Document allows the creation of a safe connection between clients and the server, guaranteeing consistent account management and smooth processing of banking transactions.Top of Form

A screenshot of a computer

Description automatically generated