Project - Bank Client-server System
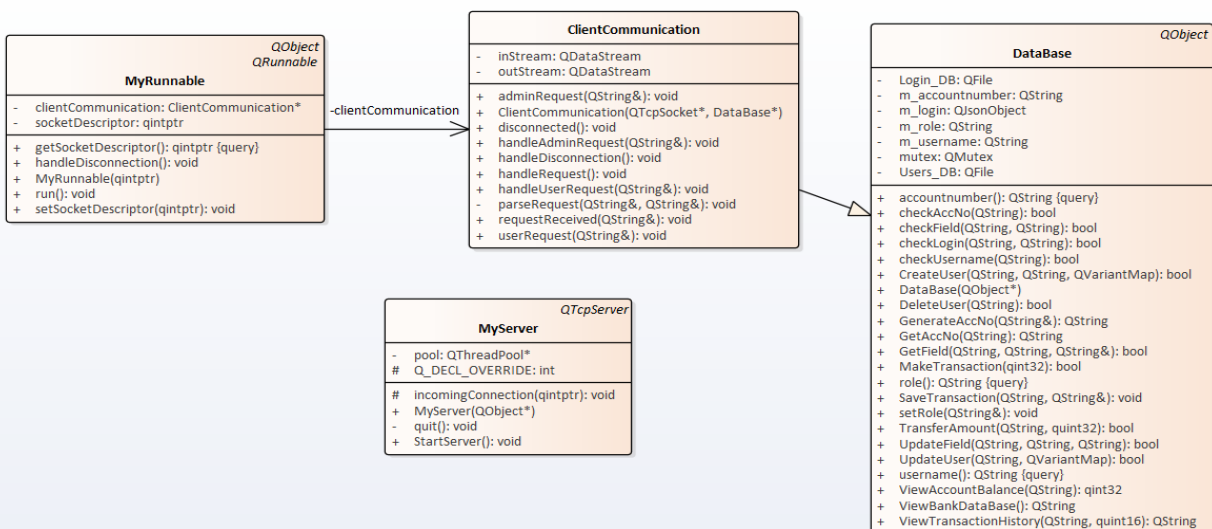
Rana Ali Ibrahim Mohamed

Client Class Diagram:

Admin

- inStream: QDataStream
- m_adminname: QString
- outStream: QDataStream
- socket: QTcpSocket

- + Admin()
- + connected(): void
- + connectToHost(QString, quint16): void
- + CreateNewUser(): void
- + Deleteuser(): void
- + disconnect(): void
- + disconnected(): void
- + error(QAbstractSocket::SocketError): void
- + GetAccNo(): void
- + Login(): bool
- + readyRead(): void
- + sendrequesttoserver(): void
- + Start(bool&): void
- + stateChanged(QAbstractSocket::SocketState): void
- + UpdateUser(): void
- + ViewAccount(): void
- + ViewBankDatabase(): void
- + ViewTransactionHistory(): void

Client *QObject*

- # m_request: QString
- # m_requestflag: QString
- # m_role: QString
- # m_serverrespond: QVariant

- + clearScreen(): void
- + Client(QObject*)
- + *GetAccNo(): void*
- + *Login(): bool*
- + *sendrequesttoserver(): void*
- + *ViewAccount(): void*
- + *ViewTransactionHistory(): void*

User

- inStream: QDataStream
- m_accountNumber: QString
- m_userName: QString
- outStream: QDataStream
- socket: QTcpSocket

- + connected(): void
- + connectToHost(QString, quint16): void
- + disconnect(): void
- + disconnected(): void
- + error(QAbstractSocket::SocketError): void
- + GetAccNo(): void
- + Login(): bool
- + MakeTransaction(): void
- + readyRead(): void
- + sendrequesttoserver(): void
- + Start(bool&): void
- + stateChanged(QAbstractSocket::SocketState): void
- + TransferAccount(): void
- + User()
- + UserOptions(): quint8
- + ViewAccount(): void
- + ViewTransactionHistory(): void

➢ Here, the Client class is the abstraction class for users or admins.

Server Class Diagram:

MyRunnable *QObject QRunnable*

- - clientCommunication: ClientCommunication*
- - socketDescriptor: qintptr

- + getSocketDescriptor(): qintptr {query}
- + handleDisconnection(): void
- + MyRunnable(qintptr)
- + run(): void
- + setSocketDescriptor(qintptr): void

-clientCommunication

ClientCommunication

- - inStream: QDataStream
- - outStream: QDataStream

- + adminRequest(QString&): void
- + ClientCommunication(QTcpSocket*, DataBase*)
- + disconnected(): void
- + handleAdminRequest(QString&): void
- + handleDisconnection(): void
- + handleRequest(): void
- + handleUserRequest(QString&): void
- + parseRequest(QString&, QString&): void
- + requestReceived(QString&): void
- + userRequest(QString&): void

DataBase *QObject*

- - Login_DB: QFile
- - m_accountnumber: QString
- - m_login: QJsonObject
- - m_role: QString
- - m_username: QString
- - mutex: QMutex
- - Users_DB: QFile

- + accountnumber(): QString {query}
- + checkAccNo(QString): bool
- + checkField(QString, QString): bool
- + checkLogin(QString, QString): bool
- + checkUsername(QString): bool
- + CreateUser(QString, QString, QVariantMap): bool
- + DataBase(QObject*)
- + DeleteUser(QString): bool
- + GenerateAccNo(QString&): QString
- + GetAccNo(QString): QString
- + GetField(QString, QString, QString&): bool
- + MakeTransaction(qint32): bool
- + role(): QString {query}
- + SaveTransaction(QString, QString&): void
- + setRole(QString&): void
- + TransferAmount(QString, quint32): bool
- + UpdateField(QString, QString, QString): bool
- + UpdateUser(QString, QVariantMap): bool
- + username(): QString {query}
- + ViewAccountBalance(QString): qint32
- + ViewBankDataBase(): QString
- + ViewTransactionHistory(QString, quint16): QString

MyServer *QTcpServer*

- - pool: QThreadPool*
- # Q_DECL_OVERRIDE: int

- # incomingConnection(qintptr): void
- + MyServer(QObject*)
- + quit(): void
- + StartServer(): void

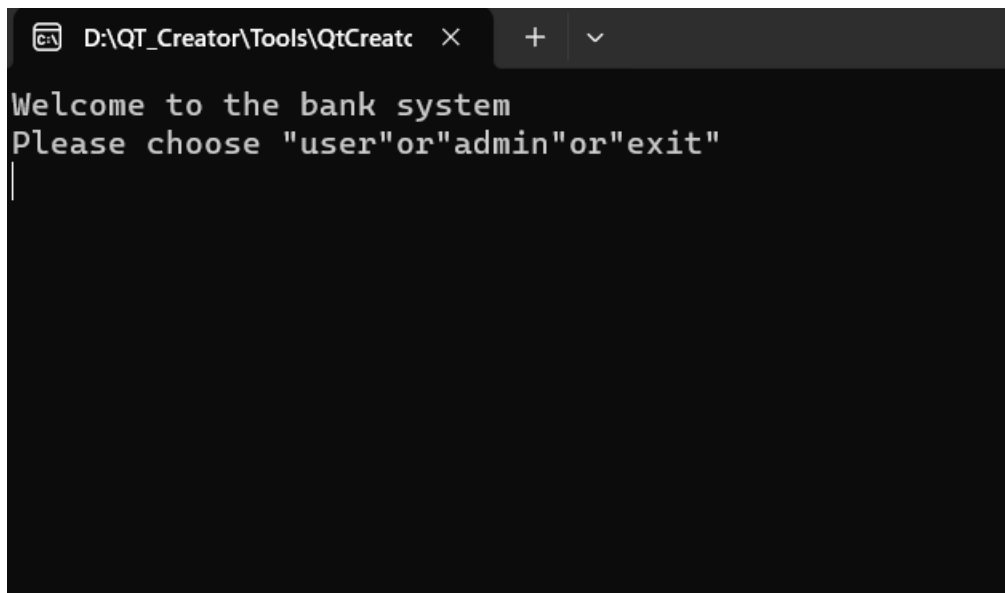**In server console application:**

Database class is the class which includes all the functions dealing with the Json files (DB format).

Client communication class is the class which handles each client (user or admin) connection to the server and processes its requests. This class inherits the Database class to access all the functions dealing with database files to handle the client requests.

MyRunnable class is the piece of code that is executed on the thread in the thread pool. Where QRunnable is not a thread it's the task executed on the thread in the thread pool.

MyServer class is the class inherits the QTcpServer class which catch each new connection using the incomingConnection overridden function from QTcpServer class which assigns each new connection to a runnable task in the pool and the executed code is in run function in MyRunnable class which executed once pool starts executing.
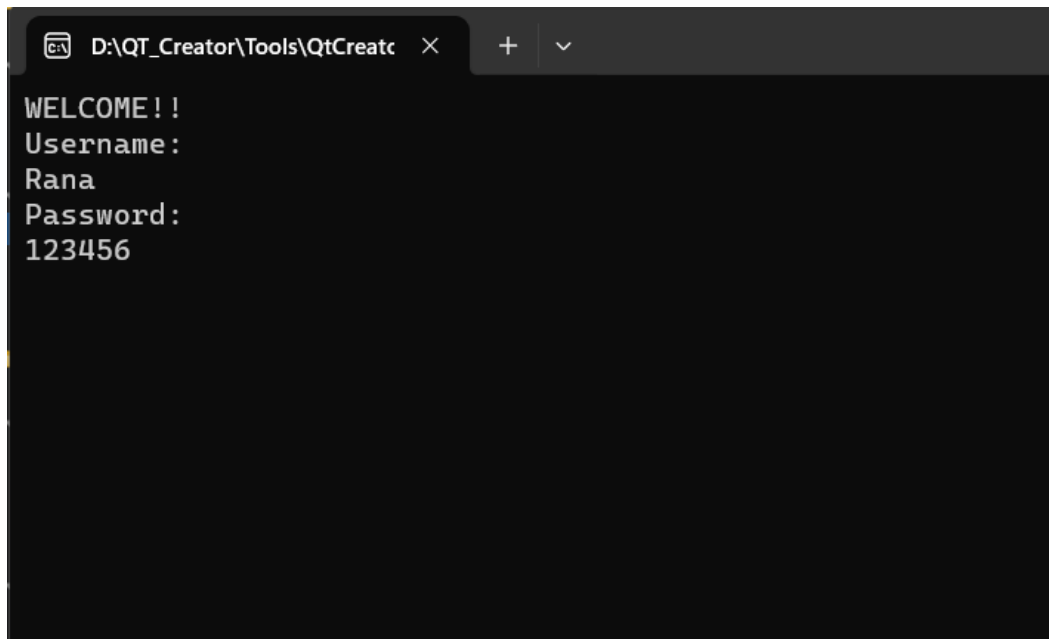
UI (User Interface):
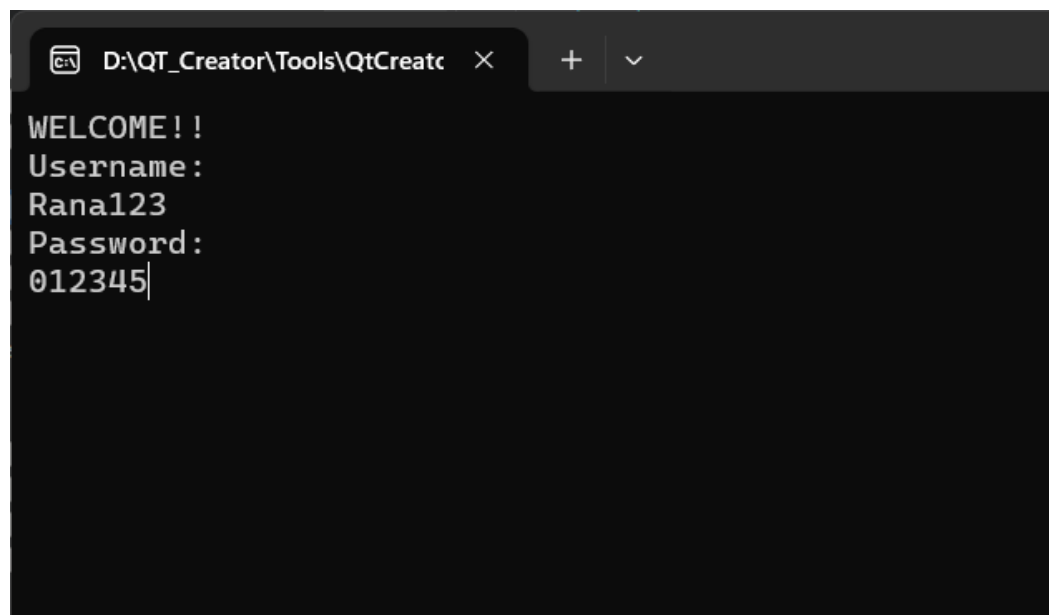
➢ **The client chooses if he is admin or user.**

```
WELCOME!!
Username:
Rana
Password:
123456
```

➢ **If client chooses user: Login function starts to be logged.**


```
WELCOME!!
Username:
Rana123
Password:
012345
```

➢ **If client chooses admin: Login function starts to be logged.**

Inside Login function we check the existence of username and password in our Login file DB then the client can be logged if they existed.

➢ **List for user options.**

➢ **List for admin options.**

Threads: 5
MyServer(0x75ebfffb70) start QThread(0x23b212b66d0)
Listening
pool started
MyRunnable(0x23b212ba170) run on QThreadPoolThread(0x23b212bb2f0, name = "Thread (pooled)")
connected to 880
Thread ID: 0x7348
pool started
connected to 880
MyRunnable(0x23b212ba730) run on QThreadPoolThread(0x23b212bb2c0, name = "Thread (pooled)")
Thread ID: 0x7300
pool started
MyRunnable(0x23b212ba570) run on QThreadPoolThread(0x23b212be880, name = "Thread (pooled)")
connected to 880
Thread ID: 0x3114

➢ **Each connection is on a different thread in the pool.**