Digital Wallet System (Console-Based C++ Application)

---

1. System Overview

The system is a console-based C++ digital wallet application (similar to InstaPay).
It supports:

- **Secure login system**
- **Client account management (Admin)**
- **Money transfers**
- **Admin operations for system monitoring and account control**

The system uses modular C++ OOP design and stores all data in memory during runtime.

2. Actors

2.1 Client (User)

A registered user who can:

- Register
- Log in
- View profile
- Check balance
- Transfer money
- Log out

2.2 Admin

A privileged user who can:

- View all clients
- Search clients
- Enable/disable accounts
- View system statistics

3. Functional Requirements (FR)

3.1 Authentication & Login

FR1: The system shall prompt the user to log in using a predefined username and password before accessing operations.
FR2: The system shall prevent disabled client accounts from logging in.
FR3: The system shall display clear error messages (“Invalid credentials”).
FR4: Passwords must never be displayed after registration.

3.2 Client Registration and Account Management

FR5: The system shall allow new clients to register by entering:

- Username
- Password
- Phone number
- Initial Balance

FR6: The system shall ensure **unique username and phone number**.
FR7: The system shall automatically generate a **unique account ID/number**.
FR8: The system shall allow clients to view their full profile:

- Phone number
- Username
- Balance
- Account status

3.3 Transactions & Balance Management

FR9: The system shall allow clients to transfer money to another account using Account ID
FR10: The system shall reject:

- Negative amount
- Zero amounts
- Invalid numeric inputs

FR11: The system shall prevent transactions that exceed the balance and show an error message.
FR12: The system shall allow users to check their current wallet balance.

3.4 Validation

FR13: The system shall validate all numeric inputs and reject non-numeric or negative values.

3.5 Money Transfer Between Clients

FR14: The system shall allow clients to transfer money using:

- Receiver Account Number

FR15: The system shall validate that the receiver exists and is active.
FR16: The system shall reject transfers that are zero or negative.
FR17: The system shall deduct the amount from the sender and add it to the receiver atomically.

3.6 Admin Panel

FR18: The admin shall be able to view all registered clients.
FR19: The admin shall be able to search by account number
FR20: The admin shall be able to enable or disable a client account.
FR21: The admin shall view system statistics:

- Total number of clients
- Number of active clients
- Number of disabled clients
- Total balance across all accounts

3.7 System Interaction

FR22: The system shall display a clear menu with available actions for clients and admins.
FR23: The system shall return to the main menu after each operation.
FR24: The system shall allow the user to log out.
FR25: The system shall allow users to exit the application safely.

4. Non-Functional Requirements (NFR)

4.1 Performance

NFR1: All operations (login, registration, search, transfer) must complete in **under 2 seconds**.

4.2 Usability

NFR3: The system shall provide a clear, simple, console-based UI with descriptive menus.
NFR4: Error messages shall be clear (e.g., “Invalid amount”, “Account not found”).

4.3 Security

NFR6: Admin access must require valid admin credentials.
NFR7: Disabled accounts must be prevented from logging in or transferring money.

4.4 Reliability

NFR9: Money transfers must be **atomic (all-or-nothing)** to avoid inconsistent balances.
NFR10: Account data must remain consistent during the session.