Group 3 Banking Application

Software Requirements Specification

Revision History

Date	Revision	Description	Author
9/22/2023	1.0	Initial Version	All
9/22/2023	1.1	Brainstorm terms/abbreviations	Daniel Leal
9/25/2023	1.2	Started on system requirements	Daniel Leal
10/2/2023	1.3	Added requirements to modules	All
10/3/2023	1.4	Edited assumptions and constraints	All
10/3/2023	1.5	Transferred Document from Google docs to MS Word	David Kromka
10/5/2023	1.6	Requirements overhaul, added references	Sean Kongsle et. al.

Table of Contents

1. Purpose

This document outlines the requirements for our server-based banking system.

1.1. Scope

This document will catalog the requirements for the server-based banking system. It will not, however, document how these requirements will be implemented.

1.2. Definitions, Acronyms, Abbreviations

Acronyms/Abbreviations:

1.2.1. **ATM:** System that lets users do transactions with their bank account such as deposit, withdraw and see balance in their accounts.

1.2.2. **GUI:** Graphical User Interface

1.2.3. **Teller**: Bank worker who is able to open/freeze and modify user accounts based on their needs.

1.2.4. **(End) User**: Bank customer who has an account or multiple accounts and is able to do transactions.

1.2.5. **Deposit**: When a user puts money in a bank account.

1.2.6. **Withdraw**: When a user takes out money from their account.

1.2.7. **Freeze Accoun**t: when a bank account is put on “hold” in order to prevent any unwanted transaction from taking place.

1.2.8. **Deactivate Account:** Make account unusable aside from reactivation.

1.3. References

Sequence Diagram - Sequence Diagram.png

UML Case Diagram - UML Case Diagrams.png

UML Class Diagram - UML Class Diagram.png

Use Case Specification Document - Use Case Specifications.pdf

1.4. Overview

The banking software provides tellers with the ability to access a member’s account and provide banking services for them. This will include ATM software that allows members to access banking services without a teller. The software will have many essential features to handle modern-day banking needs. We will provide both members and tellers with a streamlined system for managing funds and accounts.

2. Overall Description

2.1. Product Perspective

The system will be insular, and not interface with anything outside of its server and GUIs in this case.

2.2. Product Architecture

The system will be organized into five core modules: the server module, the ATM GUI module, the teller GUI module, the end user account module, and the bank account module.

2.3. Product Functionality/Features

The high-level features of the system are as follows (see section 3 of this document for more detailed requirements that address these features):

ATMs allow users to log in and deposit and withdraw money from the bank accounts they're attached to. For other account operations, they can contact a teller over the server, who can approve or deny their action using the dedicated teller UI.

2.4. Constraints

2.4.1. Since the GUIs must be able to run on any commonly used hardware, it must be implemented in pure Java.

2.5. Assumptions and Dependencies

2.5.1. Assumptions:

2.5.1.1. It is assumed that the server's Internet connection will never be faulty or terminate.

2.5.1.2. It is assumed that the system will only operate within the United States and therefore only needs to support US dollars/cents as a unit of currency.

2.5.2. Dependencies:

2.5.2.1. The system will be dependent on the TCP/IP protocol for networking communications between the server and its various clients.

3. Specific Requirements

3.1. Functional Requirements

3.1.1. Common Requirements:

3.1.1.1. System must be able to support storing the information of virtually unlimited tellers and end users.

3.1.1.2. System must use the TCP/IP communication protocol for its networking.

3.1.2. Server Module Requirements:

3.1.2.1. The server must be able to support a large number of users and tellers using the system interface at once.

3.1.2.2. The system must use a multi-threaded approach to handle multiple simultaneous clients.

3.1.2.3. The server must store the information for all users and accounts.

3.1.2.4. The server must store all user, bank account, and log info into files, which are saved when the server is shut down and loaded when the server is run.

3.1.2.5. The server must store logs of all operations on accounts that occur.

3.1.2.6. All “math” to do with account balances and editing accounts must occur within the server, and not within the client, in order to avoid abuse.

3.1.2.7. The server must manage the process of receiving user-to-teller requests and serving them to an available teller.

3.1.2.8. The server must track which connected tellers are currently occupied and unoccupied in order to serve requests appropriately.

3.1.3. ATM GUI Module Requirements:

3.1.3.1. The ATM GUI must initially prompt the end user for username and password.

3.1.3.2. If the username/password combination is invalid, the GUI must return to the login screen.

3.1.3.3. If the username/password combination is valid, the GUI must display the user’s associated bank accounts, and allow them to select one to interact with.

3.1.3.4. Once an account is selected, the GUI must display options allowing the user to deposit, withdraw, check balance, and contact a teller.
3.1.3.5. The GUI's deposit and withdraw operations must allow the user to input a value in US dollars and cents.
3.1.3.6. Contacting the teller must allow the user to ask for the teller to add/remove users from the account, freeze/unfreeze account, create a new account, and deactivate the account.
3.1.3.7. The GUI must provide a waiting screen while the user waits for a teller to approve or deny their action.
3.1.3.8. Once an action is either completed or rejected, the connection must be terminated and the GUI must close.

3.1.4. End User Account Module Requirements:

3.1.4.1. An end user must be able to be associated with any number of bank accounts, including none.

3.1.4.2. Each end user must have a unique user ID (username), and a password associated with this username.

3.1.4.3. There must be a system for comparing a given username and password against an account's to verify correctness.

3.1.5. Bank Account Module Requirements:

3.1.5.1. Each bank account must store its current balance in USD.

3.1.5.2. Bank accounts must be able to be deposited to (balance added) and withdrawn from (balance subtracted).

3.1.5.3. Each bank account must have a minimum balance. Attempting a withdrawal that would leave the account with a balance lower than this minimum must be denied.

3.1.5.4. Each bank account must be able to be associated with and accessed by many end users.

3.1.5.5. Each bank account must have a status which is either normal, frozen, or deactivated.

3.1.5.6. Frozen accounts must not be able to be deposited to or withdrawn from.

3.1.5.7. Deactivated accounts must allow no account operations other than being reactivated by a teller.

3.1.6. Teller GUI Module Requirements:

3.1.6.1. The Teller GUI must initially prompt the end user for username and password.

3.1.6.2. If the username/password combination is invalid, the GUI must return to the login screen.

3.1.6.3. If the username/password combination is valid, the GUI must display a waiting screen until a user request is served to the associated teller.

3.1.6.4. Once a request is received, the GUI must provide options to confirm or deny the transaction, as well as to view all logs associated with the relevant account.

3.1.6.5. Once the request is approved or denied, the GUI must return to the waiting screen.

3.2. External Interface Requirements

N/A

3.3. Internal Interface Requirements

3.3.1.1. The server must interface with a connected drive to store the info of its users, accounts, and logs.

4. Non-Functional Requirements

4.1. Security and Privacy Requirements

4.1.1. Highly sensitive data like usernames and passwords should not be able to be accessed externally from the system.

4.2. Environmental Requirements

4.2.1. The bank system should be able to run at any time of the day.

4.3. Performance Requirements

4.3.1. System should have enough memory to store an unlimited amount of users accounts, tellers accounts, and user information

4.3.2. The system should be able to verify the user's ID and password in a timely manner in order for the user to experience as little lag as possible while using the bank system.

4.3.3. The system should be able to perform actions on any account in a timely manner to ensure that they experience minimal delay and risk.