Group 3

*Banking System*

*Software Requirements Specification*

**Revision History**

| **Date** | **Revision** | **Description** | **Author** |
| --- | --- | --- | --- |
| 02/22/2024 | 1.0 | Initial Version | Sungmo Koo  Daniel Rodriguez  Breanne Loo |
| 02/28/2024 | 1.1 | Added diagrams and small edits | Sungmo Koo  Daniel Rodriguez  Breanne Loo |
| 02/29/2024 | 1.2 | Created two new classes after seeing what other groups have created. Updated Use Case Specifications. Finalized it. | Sungmo Koo  Daniel Rodriguez  Breanne Loo |
| 03/28/2024 | 1.3 | Update with Design Document | Sungmo Koo  Daniel Rodriguez  Breanne Loo  Dongping Guo |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**ii**

Table of Contents

[**1. Purpose 4**](#_fgv8hlzc2vsd)

[1.1 Scope 4](#_30j0zll)

[1.2 Definitions, Acronyms, Abbreviations 4](#_3znysh7)

[1.3 References 4](#_tyjcwt)

[1.4 Overview 4](#_1t3h5sf)

[**2. Overall Description 5**](#_vq4onbj370qm)

[2.1. Product Perspective 5](#_2s8eyo1)

[2.2. Product Architecture 5](#_3rdcrjn)

[2.3. Product Functionality/Features 5](#_lnxbz9)

[2.4. Constraints 5](#_1ksv4uv)

[2.5. Assumptions and Dependencies 5](#_2jxsxqh)

[**3. Specific Requirements 6**](#_c8qz3vxsnb0)

[3.1. Functional Requirements 6](#_3j2qqm3)

[3.1.1. Common Requirements: 6](#_ophxadl4fspw)

[3.1.2. User Module Requirements: 6](#_5208ql8tqj76)

[3.1.3. Bank User Module Requirements: 6](#)

[3.1.4. Teller user Module Requirements: 6](#)

[3.1.5. ATM Module Requirements: 6](#_hqhvfa96h52)

[3.1.6. Account Module Requirements: 6](#_dxknyelrmubq)

[3.1.7. Server Module Requirements: 6](#_218vpcowddj0)

[3.2. External Interface Requirements 7](#_2bn6wsx)

[3.3. Internal Interface Requirements 7](#_3as4poj)

[**4. Non-functional Requirements 8**](#_dpedc7i333dl)

[4.1. Security and Privacy Requirements 8](#_49x2ik5)

[4.2. Environmental Requirements 8](#_147n2zr)

[4.3. Performance Requirements 8](#_23ckvvd)

**iii**

# Purpose

This document outlines the requirements for the Banking System (B.S.).

## 1.1 Scope

### 

Our banking systems will provide a basic account creation system, contact system, atm system, login system, and fraud protection.

## 1.2 Definitions, Acronyms, Abbreviations

* B.S.: Banking System
* ATM: Automated teller Machine

## 1.3 References

* [Design Document](https://docs.google.com/document/d/1SmrB9MbszO9P1-xuKH4l_imc748U8QrSTDpL7T8aqhE/edit)
* [Project.phase2](https://docs.google.com/presentation/d/1i_j2w5AzO34sKx-MYrIZloXa03jIHaS8_Bs3KpdzPrI/edit#slide=id.g2bde655336b_0_6)
* [GitHub](https://github.com/danny-zebby/Banking_System)

## 1.4 Overview

Our Banking system is a Java application with a GUI that operates over TCP/IP. It allows users to talk to a teller to create, change, and delete checking and savings accounts. Along with this, there is an around the clock ATM service for basic deposits, withdraws, and transfers.

# Overall Description

## Product Perspective

When implemented correctly the B.S.will provide a means to handle transitions through a server. In addition it will also be operational even without working staff thanks to the ATM service.

## Product Architecture

The B.S. will be organized into five major modules: the User module, the Bank User module, Teller module, Owner Module, and the ATM module.

## Product Functionality/Features

The high-level features of the B.S. are as follows (see section 3 of this document for more detailed requirements that address these features): An ATM service that handles Transaction, withdrawals, and deposits. Will have unique account creation. Will have different categories of Users. Users can have multiple accounts, and they can share accounts with other users.

## Constraints

2.4.1 The B.S. will not run without Java installation. (see section 4 of this document for more environmental requirements)

2.4.2. The number of unique users is the same size as the max value an integer could be in java, 2,147,483,647.

## Assumptions and Dependencies

2.5.1 If an account exists we can assume there will be at least one user who owns the account.

# 3. Specific Requirements

## Functional Requirements

### Common Requirements:

* + - 1. There should be a main module called Bank that holds other modules together.
      2. There will be a login page, and way to store this login information.

### Bank User Module Requirements:[[1]](#footnote-0)

3.1.2.1 A user logins in as a bank user or teller with their name, birthday, unique username, and password.

3.1.2.2. The User Module works with the server module for logging in a user in and keeping track of whether or not a user is currently logged in.

3.1.2.3. Bank User will have methods to get their id, username, and password

3.1.2.4 Bank users are presented with two options, ATM service

3.1.2.5 Bank users can own as many accounts as they like.

3.1.2.6 A bank user that is admin to an account can via the teller add other users to the account.

3.1.2.7. Bank Users have cash that they can deposit to their accounts.

### Teller Module Requirements:[[2]](#footnote-1)

3.1.3.1 The Tellers can create and delete saving and checking accounts for bank users.

3.1.3.2 The accounts the teller creates have a pin that is randomly generated.

3.1.3.3 Tellers can change the pin of a user’s account at their request.

3.1.3.4. Tellers can create bank user accounts.

3.1.3.5. Tellers can log in and out of user’s account

3.1.3.6. Once a teller logs in a user’s account, the teller can transfer the account admin status to another user

3.1.3.7. Once a teller logs in a user’s account, the teller can deposit

3.1.3.8. Once a teller logs in a user’s account, the teller can transfer

3.1.3.9. Once a teller logs in a user’s account, the teller can withdrawal

3.1.3.10. Once a teller logs in a user’s account, the teller can help reset an account password when user forgot password

3.1.3.11. Once a teller logs in a user’s account, the teller can create bank accounts

3.1.3.12. Once a teller logs in a user’s account, the teller can delete a bank account

3.1.3.13. Once a teller logs in a user’s account, the teller can create bank accounts

3.1.3.13. Once a teller logs in a user’s account, the teller can add user to a bank account

3.1.3.14. Once a teller logs in a user’s account, remove a user from an account

3.1.3.15. Only one teller is an Admin Teller.

3.1.3.16. An admin teller can create and delete tellers, and view logs.

3.1.3.17. Tellers must login in and out with an id, name, and password

3.1.3.18. Teller refers to an ATM for deposit, withdrawal, and transfer.

### ATM Module Requirements:

3.1.4.1. ATM functions are deposit, withdraw, and transfer.

3.1.4.2. The ATM verifies the account number and pin number from the user before making any transaction to a bank account.

3.1.4.3. ATM transfer funds only work if a Bank User wants to send funds to a different account. A Bank User cannot request funds from another account.

3.1.4.4. Once a transfer is requested, the ATM will verify the name of the account the transfer is sent to.

3.1.4.5. Once a transaction is made, a confirmation message will appear, showing the account and account balance.

3.1.4.6. ATMs require users to log in and out with user id and password.

3.1.4.7. ATMs can show account balances by the press of a button.

### Account Module Requirements:

3.1.7.1. The accounts are linked to at least one or more Bank User.

3.1.7.2. An account has a status type of checking or saving.

3.1.7.3. An account has a balance.

3.1.7.4. To make any changes to an account, the correct pin must be matched to the account.

3.1.5.5 An account has only one admin user.

### Server Module Requirements:

3.1.8.1. The server module works with the bank user and teller modules for logging in user and teller clients, and keeping track of whether or not a user is logged in the B.S..

3.1.8.2. The server is responsible for connecting clients to the server and allowing them to access our B.S.

3.1.8.3. The server can connect multiple clients via multithreading.[[3]](#footnote-2)

3.1.8.4. The server will need keep track of accounts and information in a txt file

3.1.8.5. The server will save logs for all changes in the B.S..

## External Interface Requirements

* + 1. This is a Java application with a GUI that operates over TCP/IP

## Internal Interface Requirements

* + 1. This B.S. requires a server application and client application
    2. There is no web or HTML component. No databases, libraries, frameworks, or other technologies

# 4. Non-functional Requirements

## Security and Privacy Requirements

* + 1. Login into one account from two different attempts should be prevented.
    2. IP addresses of users will not be saved in our B.S.

## Environmental Requirements

* + 1. B.S. must be developed in JDK.
    2. B.S. must at least be ran on a JRE
    3. B.S. must be run on a computer.

## Performance Requirements

* + 1. B.S. must communicate with or between users within an appropriate[[4]](#footnote-3) amount of time.
    2. Server could possibly have some down time.[[5]](#footnote-4)
    3. B.S. will not apply a finite number to data structures, Data structures will be dynamically sized.

1. The Bank user is a user that is not a teller and is not an owner. They can only access ATMs and ask teller’s for help. The bank user is the simplest a user can be. [↑](#footnote-ref-0)
2. Want to make Tellers have the same functionality as ATM, so anything you can do at an ATM you can also do with a teller, like a real bank. Unsure about adding this to the requirements. [↑](#footnote-ref-1)
3. Unsure of how exactly multithreading works as of yet. Not sure if this is the route we will go with for in the long run but this is the idea we have right now. [↑](#footnote-ref-2)
4. How much time can not be promised right now, hoping for at least a second [↑](#footnote-ref-3)
5. How much time can not be promised right now [↑](#footnote-ref-4)