**Banking System**

Software Requirements Specification

Revision History

| **Date** | **Revision** | **Description** | **Author** |
| --- | --- | --- | --- |
| 06/21/2022 | 1.0 | Initial Version | Group 3 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

**1.** **Purpose 4**

1.1. Scope 4

1.2. Definitions, Acronyms, Abbreviations 4

1.3. References 4

1.4. Overview 4

**2.** **Overall Description 5**

2.1. Product Perspective 5

2.2. Product Architecture 5

2.3. Product Functionality/Features 5

2.4. Constraints 5

2.5. Assumptions and Dependencies 5

**3.** **Specific Requirements 6**

3.1. Functional Requirements 6

3.2. External Interface Requirements 6

3.3. Internal Interface Requirements 7

**4.** **Non-Functional Requirements 8**

4.1. Security and Privacy Requirements 8

4.2. Environmental Requirements 8

4.3. Performance Requirements 8

# Purpose

This document outlines the requirements for the Banking System. It will explain the purpose and features of the system, what the system will do, and the interface of the system. This document is intended for the client of the system.

## Scope

This document will catalog the user, system, and hardware requirements for the Banking system. It will not, however, document how these requirements will be implemented.

## Definitions, Acronyms, Abbreviations

Account Teller - Bank Staff that provides information to the user. Also able to open and

close accounts for the user

User - The person who needs the system to do its task. An account holder or bank visitor.

Database - All the information monitored and collected by the banking system

## References

Use Case Specification Document

UML Use Case Diagrams Document

## Overview

The Banking system will provide an interface for authorized banking employees in order to assist customers. There will also be an ATM system designed to perform most standard ATM functions.

# Overall Description

## Product Perspective

The Banking System is a Java Application with a GUI that operates with a server and client side application. The system interfaced with two other systems: a bank teller system and a user system. The system provides a secure environment for all financial transactions and for storing and retrieving user data

## Product Architecture

The system will be organized into two major modules: the bank employee module, the ATM module.

## 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):

Account holders will be able to view transaction history, deposit and withdraw.

Employees, bank tellers, will be able to do what account holders can do on their behalf as well as administrative actions such as open/close accounts and add account holders to joint accounts.

Bank information such as account credentials and account information (balance, transaction history, etc.) will be stored on a text file that will act as the “database” for the bank.

## Constraints

2.4.1 Accounts will only be opened and closed by employees.

## Assumptions and Dependencies

2.5.1 There should be no limit on the amount of account holders logged in at a given time.

# Specific Requirements

## Functional Requirements

### Common Requirements:

3.1.1.1 Ids will be unique, no two accounts can have the same id, case sensitive alphanumeric strings 8 characters in length.

3.1.1.2 Passwords will be case sensitive alphanumeric strings between 6 and 20 characters in length.

### Employee Module Requirements:

3.1.2.1 Employees will log into their bank teller application using their own personal account id and password.

3.1.2.2 Employees should be able to use the customer’s id and pin to view their account.

3.1.2.3 Employees should be able to deposit and withdraw, without over withdrawing, from the user’s account on behalf of the account holder.

3.1.2.4 Employees will be able to add other bank members to a joint account.

3.1.2.5 Employees should be able to open, close, and freeze customer accounts.

3.1.3.6 Employees should be able to transfer money between customer accounts

### ATM Module Requirements:

3.1.3.1 Users should be allowed to log in using their issued id and pin.

3.1.3.2 Users should be able to deposit money into their account.

3.1.3.3 Users should be able to withdraw money from their account while not being able to over withdraw from their account.

3.1.3.4 Users should be able to transfer money between their accounts.

## External Interface Requirements

3.2.1 The system must provide an interface for employees to check the history of actions of an account such as checking balance, open/freeze of account, addition or removal of people from a joint account, deposits, and withdraws. These actions will be stored in a file tied to the given account.

3.2.2 The system will provide an interface for account holders that allows them to view the history of actions on their account such as checking balance, deposits, withdraws, and money transfers. These actions will be stored in a file tied to the given account.

## Internal Interface Requirements

3.3.1 The system should process a data-feed from the banking system such that bank customers' information are stored alongside their accounts and account balance. The data will be in the form of a comma separated text file. It will be exported frequently.

3.3.2 The system should process a data-feed from the banking system and ATM or the bank employee system in which transactions are stored in a transactions history document. Actions such as Deposits, withdrawals, and account balance views as well as the dates they have occured will be stored with comma separations. The file is to be exported upon customer request or administrative review.

# Non-Functional Requirements

## Security and Privacy Requirements

4.1.1 Account holder information such as id, pin, and account transactions should be

encrypted.

## Environmental Requirements

4.2.1 System must communicate with the database and update it live.

## Performance Requirements

4.3 The program should be able to run on low-end hardware.

4.4 Transactions should be verified and completed within a small period of time.

UML Diagram

