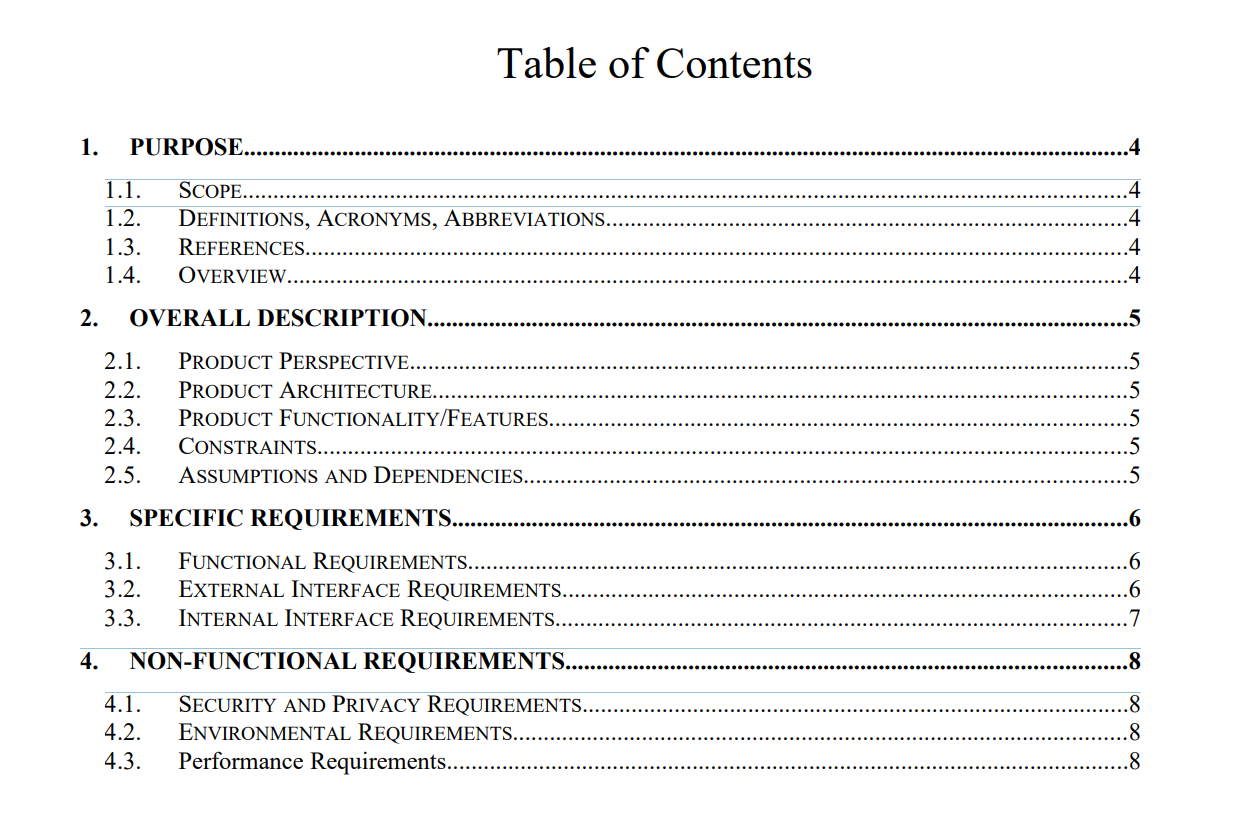
CS-401\_Group Project Requirements

# Software Requirements Specification

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Revision | Description | Author |
| mm/dd/yyyy | 1.0 | Initial Version | Your Name |
| 9/21/2024 | 1.1 | Saving and checkings Account Module | Rohan Kumar |
| 9/24/24 | 1.2 | SuperUser (Admin) Module | Alexis Rojas |
| 9/17/24 | 1.2.1 | 3.1.1.1-2 common requirement module (accounts) | Phuong Nguyen |
| 9/24/2024 | 1.3 | External Requirements | Rohan Kumar |
| 9/25/2024 | 1.4 | Revise modules to new modules type per client discussion: Operator, Account, Client, Host, Message | Phuong Nguyen |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |



1. Purpose

This document outlines the requirements for a Large Banking System.

**1.1. Scope**

the large banking

**1.2. Definitions, Acronyms, Abbreviations**

*List any acronyms, terms etc. that need to be defined.*

SU - SuperUser, refers to the class of user with more permissions and functionalities.

**1.3. References**

// not doing the diagrams this is for the design phase.

//todo: we do

**1.4. Overview**

The large banking system will provide users with an interface for deposits, and withdrawals from their checking or savings accounts. The SU can do the same for the use, additionally, the SU can also add or remove accounts, add people to an account, or take off people from an account.

2. Overall Description

**2.1. Product Perspective**

**2.2. Product Architecture**

*The system will be organized into 5 major modules: the Operator module, the Account module, the Host module, the Message, and the Client module. Note: System architecture should follow standard OO design practices.*

**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):

**2.4. Constraints**

*List appropriate constraints.*

*Constraint example: Since users may use any web browser to access the system, no browser-specific code is to be used in the system.*

**2.5. Assumptions and Dependencies**

*List appropriate assumptions*

*Assumption Example: It is assumed that the maximum number of users at a given time is 15,000.*

3. Specific Requirements

**3.1. Functional Requirements**

**3.1.1. Common Requirements:**

3.1.1.1 Each account will have a unique serial ID used to identify the account.

3.1.1.1.1 Serial ID is generated upon account initialization.

3.1.1.1.2 Serial ID has a pattern for identification.

3.1.1.1.3 Serial ID will be used to authenticate an Account, along with a user generated passcode, which is initiated during account creation. (currently disallowing changing passcode)

3.1.1.2 User may be registered to an account.

3.1.1.2.1 Each registered user gains access to the account through authentication with the account’s reference list.

3.1.1.2.2 Adding and Removing users to account requires the SU.

3.1.1.2.3 Multiple Users may be registered to the same account.

3.1.1.2.4 The removal of the last User in an account will result in its closure.

3.1.1.3 The ability to transfer funds.

**3.1.2. User Module Requirements:**

**3.1.2. SuperUser (Admin) Module Requirements:**

3.1.2.1 The superuser will have the ability to create new accounts for bankers

3.1.2.2 If a checking account has not been active in the last 6 months, the superuser has the ability to deactivate the said account.

3.1.2.3 If an account has a negative balance or a history of overdrafts, the superuser will have the ability to declare the account closed.

3.1.2.4 The superuser has the ability to assign roles that have appropriate permissions

**3.1.3 Account Module Requirements:**

**3.1.3. Checking Account Module Requirements:**

3.1.3.1 The checking account has no limit on the number of withdrawals.

3.1.3.2 The checking account has a $5 maintenance fee per month.

**3.1.3. Saving Account Module Requirements:**

3.1.3.1 Savings account has a withdrawal limit of 6 monthly withdrawals without a fee.

3.1.3.2 If the user withdraws more than 6 times, they will be charged a $5 fee for every withdrawal.

3.1.5.2 The savings account will have a 0.10% annual increase rate.

**3.1.4 Message Module Requirements:**

3.1.4.1 Message between client and host through the internet will be encapsulated.

3.1.4.1.1 Encapsulated data will be referred to a package.

3.1.4.1.2 Package will contain data, current address, receiving address, and type of data.

3.1.4.2 The encapsulated method will accept different type of information.

3.1.4.3 The encapsulated data is able to be decapsulated by the recipient.

3.1.4.4 There will be a processor, a sender, and a receiver on both side of the connection.

3.1.4.4.1 Processor can encapsulate data into package or decapsulate package into data.

3.1.4.4.2 Sender will let processor create package from passed data and send package.

3.1.4.4.3 Receiver will listen on a designated port for packages and send them to processor to get data.

3.1.4.4.4 Processor, Sender, and Receiver can handle multiple packages synchronously.

**3.1.5 Client Module Requirements:**

**3.1.6 Host Module Requirements:**

**3.2. External Interface Requirements**

**3.3. Internal Interface Requirements**

4. Non-Functional Requirements

**4.1. Security and Privacy Requirements**

The data will be stored in a text file.

Allow access if the user and password are correct.

**4.2. Environmental Requirements**

The system must be deployable on any OS, for Example: Linux, Mac, or Windows.

The system has JVM as it’s required to run Java code.

**4.3. Performance Requirements**

The system should be fast enough for the user to not experience a delay greater than 3 seconds.