# Introduction

## Purpose of Document

This is a Requirements Specification document for a banking system for CSC 406 class Team 1. This document describes the scope, objectives, and goal of the new system. In addition to describing non-functional requirements, this document models the functional requirements with use cases, interaction diagrams, and class models. This document is intended to direct the design and implementation of the target system in an object-oriented language.

## Project Summary

Project Name: CSC 406 Banking System

Project Manager: Devin Amos

Project Developers: Corey Morris, Jacob Teasley, Lizzy Rugg, and Mac Koontz

## Background

The objective of this project is to develop a banking system that allows customers to perform various transactions related to their accounts, such as deposits, withdrawals, and transfers. The system should support several types of accounts, including savings, CDs, checking, and loans. In addition, the system should allow bank tellers and management to manage customer accounts and perform necessary operations.

## Project Scope

The scope of this project is a native desktop-based system that supports account creation and facilitation.

## System Purpose

### Users

Those who will use the system include

* Customers
  + Customers will use the system to access and control their bank accounts remotely.
* Tellers
  + Tellers will use the system to manage higher level control of customer bank accounts.
* Managers
  + Managers will use the system to manage the highest level of control for customer bank accounts.

### Location

The system will be available to any potential customer using the internet. Bank tellers and managers may only use the system within the bank network.

### Responsibilities

The primary responsibilities of the system are:

* Provide customers with up to date information that is constantly synced throughout the system
* Allow different levels of access for specific users.

### Need

The system is needed for the bank to keep track of all customer and account data.

## Overview of Document

The rest of this document gives the detailed specifications for the banking system. It is organized as follows:

* Section 2: Functional Objectives

Each objective gives a desired behavior for the system. These objectives are organized by priority.

* Section 3: Non-Functional Objectives

This section is organized by category. Each objective in this section states a technical requirement on the characteristics of the system.

* Section 4: Context Model

This section gives a text description of the goal of the system. This section also provides a model of the scope of the system in the form of a diagram.

* Section 5: Use Case Model

This section provides a use case diagram, as well as a description of select components in the diagram.

* Section 6: Class Model

This section provides a collection of objects within the system as well as their relationships.

# Functional Objectives

## High Priority

### Account Creation

1. The system shall allow customers to create and account by providing their personal information.
2. The system shall allow tellers and managers to create savings and checking accounts for customers.

### Account Types

1. The system shall support the following account types:
   1. Savings
   2. CD
   3. Checking (That’s My Bank)
   4. Checking (Gold/Diamond)
2. The system shall support the following loan types:
   1. Long term mortgage
   2. Short term
   3. Credit card

### Interest Calculation

1. The system shall calculate interest on savings, cd, and gold/diamond accounts daily
2. The system shall calculate interest on loans.
3. The system shall calculate interest on credit card loans.

### Transactions

1. The system shall allow customers to perform the following transactions on their accounts:
   1. Deposit money into account.
   2. Withdraw money from account.
   3. Transfer money between accounts.
   4. Make payments on loan accounts.
   5. Make payments on credit card accounts.
   6. Perform stop payments on checks.
   7. Select savings account as overdraft protection for checking accounts.

### Fees and Penalties

1. The system shall apply fees and penalties as follows:
   1. A $35 fee for stop payments.
   2. A $25 fee for overdrafts.
   3. A $75 late fee for missed mortgage and short-term loan payments.
   4. A $1.25 fee per transfer from TMB accounts.
   5. A $0.75 fee per transaction from TMB accounts.
   6. A $0.75 fee per transaction from Gold/Diamond accounts if balance is below $5000.

### Account Management

1. The system shall allow bank tellers and management to perform the following operations:
   1. View account balances, recent debits, and account status.
   2. Credit or debit any account.
   3. Transfer money between accounts.
   4. Manage loan accounts and initiate monthly processes for credit card and mortgage payments.

# Non-Functional Objectives

## Reliability

1. The system shall perform all operations without crashing or freezing.

## Usability

1. The system shall have an easy-to-use interface for customers, tellers, and managers to perform their respective tasks.

## Performance

1. The system shall be able to handle many windows and transactions without affecting performance.

## Security

1. The system shall ensure the security of customer information and transactions by implementing encryption, access controls, and other security measures.

## Documentation

1. The system shall have readable documentation for the interfaces throughout the system.
2. The documentation shall be accessible to any user of the system.

# The Context Model

## Context Diagram

INSERT CONTEXT DIAGRAM

## System Externals

1. Customer
   1. A customer is any user of the system that is using it with the intention of creating and using bank accounts.
2. Teller
   1. A teller is any user of the system that is employed at the bank below the manager.
3. Manager
   1. A manager is any user with the highest level of access to the system.

# The Use Case Model

## System Use Case Diagram

INSERT USE CASE DIAGRAM

## Use Case Descriptions

Notes:

* For all use cases, the user can navigate back to the previous page
* For all use cases, the user can exit the system and choose to save the contents. This action ends the use case

**Login User**

|  |  |
| --- | --- |
| Use Case Name: | Login User |
| Summary: | To access personal accounts and restricted information, the user must login to their account using their social security number |
| Basic Flow: | 1. The use case starts when a user selects who they are. 2. The system requests a social security number if the user is a customer. 3. The user enters their social security number if they are a customer. 4. The system verifies the social security number is in the database. 5. The system starts a login session for the user. |
| Alternative Flows: | * Step 4: If the social security number is not in the database, the use case goes back to step 2. |
| Extension Points: | none |
| Preconditions: | The user is registered in the database |
| Postconditions: | The user can now perform functions according to their access level |
| Business Rules: | Some data and functions are restricted to certain types of users. |

**Register Customer**

|  |  |
| --- | --- |
| Use Case Name: | Register Customer |
| Summary: | To access the banking system and accounts within the system the customer must be registered |
| Basic Flow: | 1. The use case starts when a customer selects the create account button. 2. The system requests the required information for the account being created. 3. The customer enters the requested information. 4. The system verifies inputs match the desired input format. 5. The system checks to see if the primary key is not a duplicate within the database. 6. The system stores the information in the database. 7. The system redirects the customer to their account page. |
| Alternative Flows: | * Step 4: If the inputs do not match the desired input format the use case goes back to step 2 * Step 5: If the primary key is a duplicate within the database the use case goes back to step 2 |
| Extension Points: | none |
| Preconditions: | none |
| Postconditions: | The customer can now create and access their bank accounts |
| Business Rules: | ? |

# The Class Model

INSERT CLASS MODEL DIAGRAM