# Requirements Document

# Team PB-PI

# February 11, 2018

Table 1: Team

| Name | ID Number |
|---|---|
| Alissa Bellerose | 27377320 |
| Sabrina D'Mello | 27739486 |
| Melanie Damilig | 40032420 |
| Tobi Decary-Larocque | 27407645 |
| Zain Farookhi | 26390684 |
| Giulia Gaudio | 27191766 |
| Jason Kalec | 40009464 |
| Damian Kazior | 40016168 |
| Johnny Mak | 40002140 |
| Philip Michael | 40004861 |
| Ramez Nicolas Nahas | 26718108 |
| Steven Tucci | 40006014 |
| Shunyu Wang | 40043915 |

# 1 System

## 1.1 Purpose

The purpose of this document is to outline the requirements for the myMoney application. It provides direction and background to anyone involved in developing, testing and maintaining the system.

The intended audience of this document is described in table 2.

Table 2: Intended audience

| Reader | Reason |
|---|---|
| Users/Customers | Give feedback |
| System Developers | Understand the functionality and properties the application contains |
| Testers | Test the system |
| User Manual Writers | Source material for manuals |
| Project Team | Keep track of status of project |

## 1.2 Business Goals

Our customers are money-conscious people who would benefit from a system that makes it easier to track how their money moves.

There currently is no efficient way for people to stay on top of their finances. To do it, they must rely on bank statements and receipts, and must learn to navigate through a variety of interfaces to keep track of different accounts.

Our system makes it easier for people to control their finances, by allowing them for example to record cash deposits/withdrawals and track income/expenses across various accounts, all in one place.
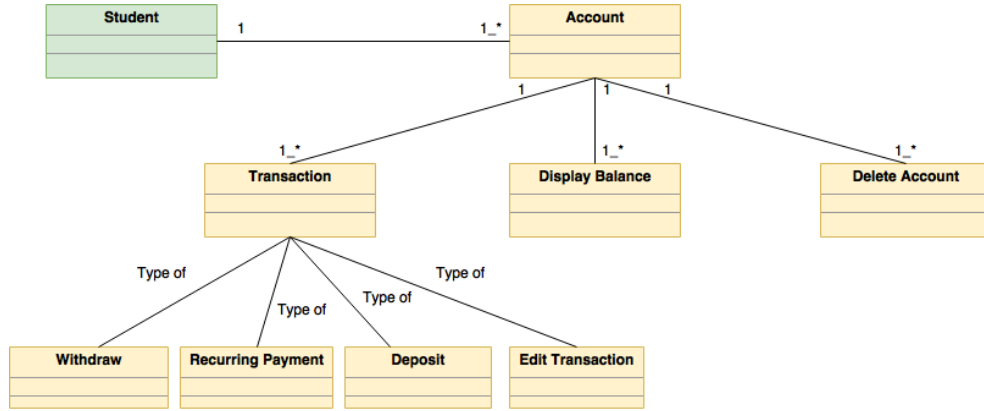
Figure 1: Domain model

# 2 Domain Concepts

Table 3: Main domain concepts

| Concept | Description |
| --- | --- |
| Student | Main user of the system. Student manually keeps the system up to date by entering every transaction performed during his day to day activities. |
| Deposit | An action performed by the student. Whenever they receive money, student indicates that amount in myMoney app as a deposit. |
| Withdrawal | Every time student spends money, withdrawal object is created. It can be a type of credit card purchase or paying a bill. |
| Bill | Cash payment that student performs. Amount of bill paid in cash must be entered manually. |
| Credit card | Indicate credit card purchase. Every credit card payment is manually entered into myMoney app. |

# 3 System overview

The myMoney application is a simple personal funds management system. It tracks a user's deposits and expenses across different banks and accounts within these banks. System is maintained by the user and is not connected to any bank account.

# 4 Actors

Table 4: User groups

| User Group | Description | Number of Users |
|---|---|---|
| Young adults | Student or adult at an early stage of their career | 1 |

# 5 Functional Requirements

Table 5: Functionalities

| Field | Description |
|---|---|
| **ID** | **F1** |
| Version | 1 |
| Feature | Transaction |
| Requirement | The system must be able to make a transaction from an account |
| Source | Team Brainstorm |
| Rationale | For a user to keep track of their budget, they must be able to record changes made within an account by adding transactions to said account |
| Priority | Must |
| Status | Implemented |
| Traces to use cases | Withdrawal & Deposit Amount |

| ID | F2 |
|---|---|
| Version | 1 |
| Feature | Transaction |
| Requirement | The user may only enter a numeric entry as input for the amount to deposit or withdraw |
| Source | User |
| Rationale | Monetary value is numerical, therefore we must ensure that input is valid, for system to accept the input |
| Priority | Must |
| Status | Proposed |
| Traces to use cases | Withdrawal, Deposit Amount & Edit Transaction |

| ID | F3 |
|---|---|
| Version | 1 |
| Feature | Transaction |
| Requirement | Transaction should require confirmation before being processed |
| Source | Team Brainstorm |
| Rationale | To ensure transaction is properly entered, system will prompt the user to confirm the transaction after inputting an amount |
| Priority | Must |
| Status | Proposed |
| Traces to use cases | Withdrawal & Deposit Amount |

# 6 Non-functional Requirements

Table 6: Non-functional requirements

| Field | Description |
|---|---|
| **ID** | **D1** |
| Ver | 1 |
| Requirement | All documentation will be found within the source code in the repository |
| Source | Team Brainstorm |
| Rationale | All documents should be centrally located and accessible by all team members |
| Priority | Want |
| Status | Implemented |
| Traces to use cases | - |
| **ID** | **S1** |
| Ver | 1 |
| Requirement | User's financial information needs to be encrypted in database |
| Source | Organizer |
| Rationale | System will be handling sensitive information about the user's finances. Data such as account numbers will need to be encrypted for protection |
| Priority | Must |
| Status | Proposed |
| Traces to use cases | - |

# 7 Use Cases

## 7.1 Overview
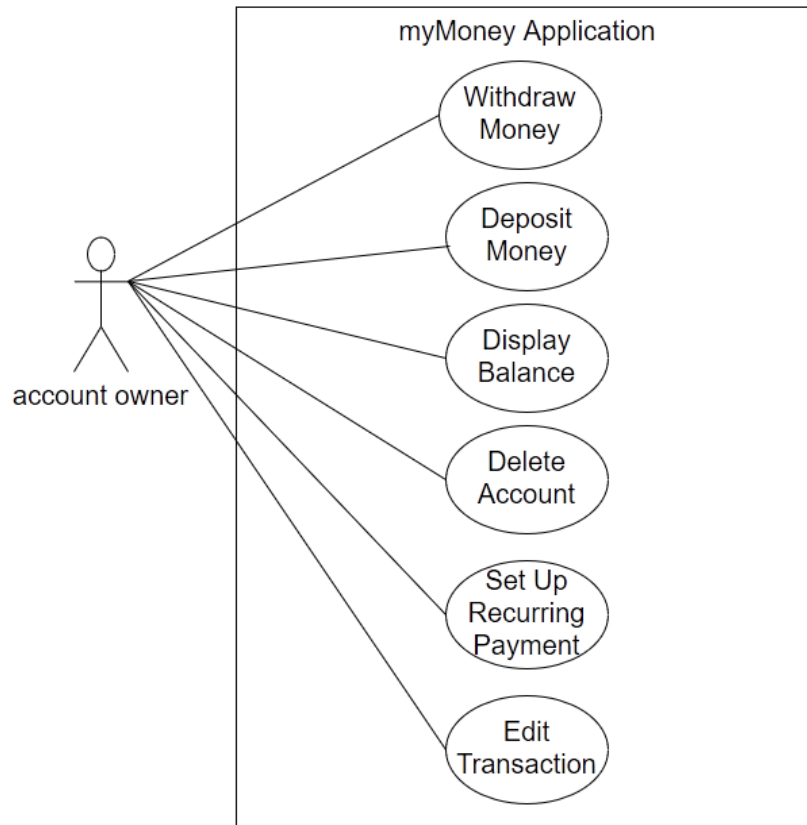


Figure 2: Use Case Diagram

**Use Case 1**

**ID**
UC-MWA-001

**Name**
Withdraw money.

**Goal**
The user withdraws an amount of money from the selected account.

**Actors**
Primary Actor - owner of the account.

**Precondition**
User is the owner of the account.

**Main Scenario**

1. Primary actor indicates to withdraw an amount from a selected account.

2. System verifies the account exists.

3. System prompts primary actor for the amount to withdraw.

4. User enters amount to withdraw.

5. System verifies that the account contains sufficient funds.

6. System prompts user for confirmation.

7. User confirms.

8. System subtracts amount of money requested from selected account, confirms withdrawal.

9. Use case ends succesfully.

**Exceptions**

2a) Account does not exist.

5a) Account has an empty or negative balance.

5b) Account contains insufficient funds.

**Postcondition**
The amount is subtracted from the selected account.

**Priority**
Critical.

**Use Case 2**

**ID**
UC - MDA - 002

**Name**
Deposit Money.

**Goal**
User succesfully deposits an amount of money to the selected account.

**Actors**

Primary Actor - owner of the account.

**Precondition**

User is owner of the account.

**Main Scenario**

1. Primary actor indicates to deposit an amount to a selected account.

2. System verifies that the account exists.

3. System prompts user to enter amount to deposit.

4. System verifies amount is valid.

5. System adds amount of money deposited to selected account and confirms.

6. Use case ends succesfully.

**Exceptions**

2a) Account does not exist.

4a) Amount is zero or negative.

**Postcondition**

The amount is added to the selected account.

**Priority**

Critical.

**Use Case 3**

**ID**

UC - DBA - 003

**Name**

Display Balance.

**Goal**

Display balance of chosen account to user.

**Actors**

Primary Actor - Owner of the account

**Precondition**

Account exists.

**Main Scenario**

1. Primary Actor selects account to be displayed.

2. System retrieves account information from database.

3. System displays balance.

## Exceptions

1a) Account does not exist.

2a) System cannot retrieve data for chosen account.

## Postcondition
Account balance is displayed for user.

## Priority
Critical.

## Use Case 4

### ID
UC - DA - 004

### Name
Delete Account.

### Goal
User deletes selected account.

### Actors
Primary Actor - owner of the account.

### Precondition
User is owner of the account. At least one account exists.

### Main Scenario

1. User indicates to delete the selected account.

2. System verifies that the selected account exists

3. System prompts the user for confirmation.

4. User confirms.

5. System deletes selected account.

6. Use case ends successfully.

## Exceptions

1a) There is no account.

**Postcondition**
Selected account is deleted.
**Priority**
Critical.

**Use Case 5**

**ID**
UC - SRA - 002
**Name**
Set Up Recurring Payment.
**Goal**
User can set up recurring payments.
**Actors**
Primary Actor - owner of the account.
**Precondition**
User is owner of the account. At least one account exists.
**Main Scenario**

1. User indicates to set up a recurring payment.

2. User chooses the appropriate account.

3. System prompts user to enter the amount of payment.

4. User enters amount of payment.

5. System verifies amount is valid.

6. User selects start date.

7. User selects payment frequency.

8. System prompts user for confirmation.

9. User confirms.

10. Use case ends succesfully.

**Exceptions**

1a) There is no account.

5a) Amount is zero or negative.

9a) User cancels or does not confirm.

## Postcondition

System records recurring payment information and initiates <u>Money Withdrawal</u> accordingly.

## Priority

Critical.

## Use Case 6

## ID

UC - ETR - 006

## Name

Edit transaction.

## Goal

User can modify an existing transaction.

## Actors

Primary Actor - owner of the account.

## Precondition

User is owner of the account. At least one account exists. At least one transaction has been saved.

## Main Scenario

1. User indicates to modify a transaction.

2. User chooses the appropriate account.

3. User selects transaction to modify.

4. System prompts user for confirmation.

5. User confirms.

6. User enters new amount and confirms.

7. System prompts user for confirmation with details.

8. User confirms.

9. Use case ends succesfully.

## Exceptions

2a) There is no account.

3a) There are no transactions for selected account.

6a) Amount is not valid.

**Postcondition**
Transaction has been altered to reflect changes.

**Priority**
Critical.

# 8   Constraints

Table 7: System Constraints

| Field | Description |
|---|---|
| ID | C1 |
| Ver | 1 |
| Constraint | The application must run on any well-known desktop operating system including Apple & Windows OSs |
| Source | Team Brainstorm |
| Rationale | Serve all users across different operating systems |
| Priority | Must |
| Status | Proposed |
| Traces to use cases | All use cases with a desktop interface |

# 9  Solution ideas

Table 8: Solution Idea I

| Field | Description |
|---|---|
| ID | SI1 |
| Ver | 1 |
| Solution idea | Total amount of withdrawals & deposits could be displayed using charts |
| Source | Team Brainstorm |
| Rationale | When evaluating interesting budgeting applications, use of charts offers quick and clear overview of the user's financial situation |
| Traces to use cases | Display balance |

# 10  Acronyms and Abbreviations

| | |
|---|---|
| **C x** | Constraint x |
| **ChAc** | Chequing Account |
| **D x** | Documentation x |
| **DBA** | Display Balance |
| **F x** | Functionality x |
| **MDA** | Money Deposit |
| **MWA** | Money Withdrawal |
| **SavAc** | Savings Account |
| **S x** | Security x |
| **SI x** | Solution Idea x |

# 11  References

We obtained an example *User Requirements Document* from the website http://www.soberit.hut.fi/T-76.115/05-06/ohjeet/template/requirements.html.