

# Software Requirements Specification

Vincent Boivin, ID 000000000      Kisife Giles, ID 40001926  
Marc-Antoine Jette-Leger, ID 27895038  
Jean-Loup Johnston, ID 000000000  
Fabian Vergara, ID 000000000      Michael Xu, ID 27206356

February 9, 2018

# 1 Revision History

**Tuesday February 6th, 2018 Version 1.0**

**Tuesday February 7th, 2018 Version 1.1** Reworked context, added Business rules, Non-functional requirements

**Tuesday February 9th, 2018 Version 1.2** Described Use Case 3, added Domain Model, added the domain model entries to the glossary, formatted tables and lists

# Contents

<b>1</b>	<b>Revision History</b>	<b>2</b>
<b>2</b>	<b>Project Description</b>	<b>4</b>
2.1	Introduction . . . . .	4
2.2	Context . . . . .	4
2.3	Business goals . . . . .	4
2.4	Scope . . . . .	4
2.5	Domain model . . . . .	5
2.6	Actors . . . . .	5
<b>3</b>	<b>Functional requirements</b>	<b>6</b>
3.1	Overview . . . . .	6
3.2	Use cases . . . . .	6
3.2.1	Create user accounts . . . . .	6
3.2.2	Access user accounts . . . . .	7
3.2.3	Load and display transactions data . . . . .	7
3.2.4	Create and visualize budget . . . . .	8
3.3	Business Rules . . . . .	8
<b>4</b>	<b>Non-functional requirements</b>	<b>8</b>
<b>5</b>	<b>Design Constraints</b>	<b>8</b>
<b>6</b>	<b>Glossary</b>	<b>9</b>
<b>7</b>	<b>References</b>	<b>9</b>

## **2 Project Description**

### **2.1 Introduction**

The primary goal of this project is to develop a program to manage one's own finances to satisfy a targeted savings amount. Using this program, the users can obtain critical information about their spending and make better, educated decisions on how to manage their finances.

### **2.2 Context**

what is this document for

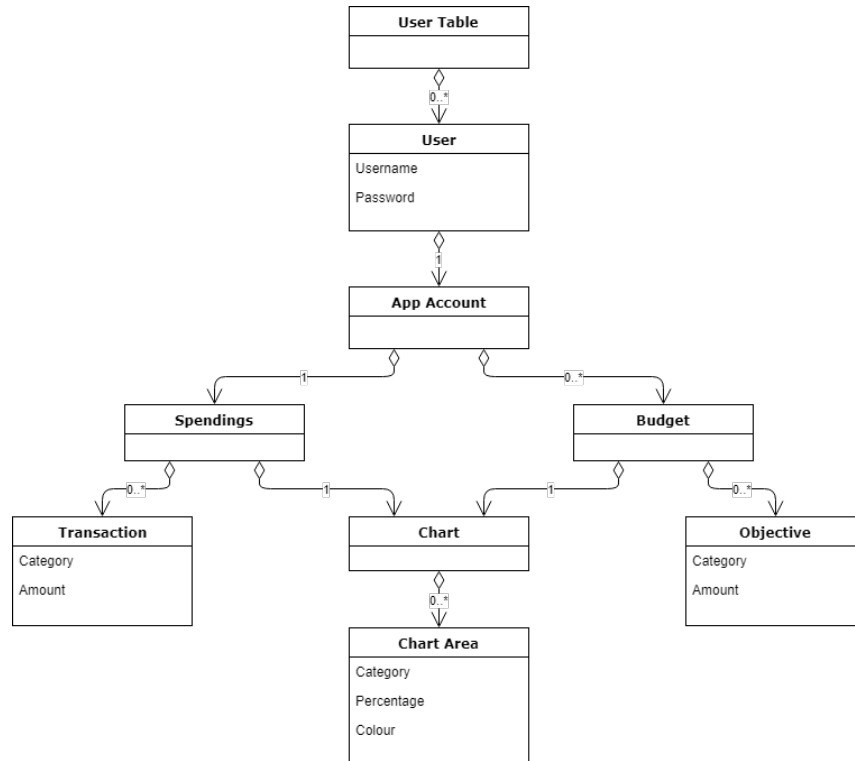
### **2.3 Business goals**

The primary business goal is to make our customers save more and to make them adopt our software applications, which will allow us to provide a better service as a more well rounded financial company. This in turn allows us to increase efficiency (needs less employees to serve customers on the same topic), increase our market share, as we did not have this service beforehand which will in turn increase our profit margin because we have access to more customers than before.

### **2.4 Scope**

The MyMoneyApp software is a financial management application, developed in Java as a standalone desktop application targeting young consumers. This system's aim is to help users to make wise and accurate decisions when they have a target amount of savings they want to have for a month. It performs this operation by first getting the user's bank statement, then displaying them on-screen to differentiate what amount of money has went into what type of service, and to then be able to change those amounts in the coming months. The main qualities of MyMoneyApp, is that it is easy-of-use, user-friendly and efficient, which will allow the purchasing company to satisfy its customers' needs. All in all, this software will help the company to hold its market leader position in the financial domain.

## 2.5 Domain model



## 2.6 Actors

This document is intended to be read by:

- Users of the software: this document allows them to have a more complete idea about the system and its functionalities.
- Team developers: they can use this document as a primary resource for all subsequent project development phases (design, coding, testing and maintenance phases).
- Testers: To be able to test the system in accordance to the specified requirements.
- CEO of the company that has hired us: this document will allow them to deeper understand and have a more comprehensive idea about the requirements of the project.

## 3 Functional requirements

### 3.1 Overview

This section includes all the details regarding the use cases and features afforded to the user of the MyMoney application. Those features include creating and logging into an account, accessing various information about the transactions made with a bank account, and creating, inspecting and deleting budgetary goals to be stored in a list.

### 3.2 Use cases

#### 3.2.1 Create user accounts

<b>Action</b>	Account management
<b>Case ID</b>	1.1
<b>Summary</b>	User provides the necessary information for the creation of an account.
<b>Scope</b>	Budget management application
<b>Trigger</b>	Registration button
<b>Precondition</b>	None
<b>Postcondition</b>	Account is created
<b>Primary Actor</b>	User
<b>Secondary Actor</b>	Filesystem

Main Scenario-Step	Action
1	User Clicks on Register.
2	User enters a username, password, and password confirmation, and click “Register Account” button.
3	System verifies if the username is already taken.
4	System gets login credentials from File System.
5	System verifies in login file if the password and password confirmation are the same.
6	System creates account object.
7	System saves the username and password combination to the database/textpad.
8	System displays login menu, with account creation confirmation, and asks user to login.
9	System goes idle.

### 3.2.2 Access user accounts

<b>Action</b>	Account management
<b>Case ID</b>	1.2
<b>Summary</b>	User provides the necessary information to login.
<b>Scope</b>	Budget management application
<b>Trigger</b>	Login button
<b>Precondition</b>	Have a registered account
<b>Postcondition</b>	Account is accessed
<b>Primary Actor</b>	User
<b>Secondary Actor</b>	Filesystem

Main Scenario-Step	Action
1	User inputs username and password.
2	User clicks on the “login” button.
3	System verifies if the username, password pair exists.
4	System displays the user’s transaction logs.
5	System goes idle.

### 3.2.3 Load and display transactions data

<b>Action</b>	Load and display transaction data
<b>Case ID</b>	2
<b>Summary</b>	User provides the the number of his credit card number for the reviewing of his spending.
<b>Scope</b>	Budget management application
<b>Trigger</b>	Get my info button
<b>Precondition</b>	To be logged in as a user
<b>Post condition</b>	Transaction data is displayed
<b>Primary Actor</b>	User
<b>Secondary Actor</b>	Filesystem

Main Scenario-Step	Action
1	User Enters his credit card number.
2	User clicks on the get transaction data button.
3	System pulls the information from the textpad/database.
4	Transaction data is displayed on the screen.
5	System goes idle.

### 3.2.4 Create and visualize budget

<b>Action</b>	Account customization
<b>Case ID</b>	3
<b>Summary</b>	The user is able to create a budget and visualize it with a chart
<b>Scope</b>	Budget management application
<b>Trigger</b>	Budget button
<b>Precondition</b>	To be logged in as a user
<b>Postcondition</b>	Budget information displayed and editable, chart is accessible
<b>Primary Actor</b>	User

<b>Main Scenario-Step</b>	<b>Action</b>
1	User clicks the Budget button
2	User chooses a category
3	User enters an amount
4	User clicks the Add to budget button
5	User's budget is updated in real time
6	User clicks Create chart button
7	User's budget is instantly computed and illustrated by a chart in a new window
8	User clicks the Reset budget button
9	User's budget values are all set to 0
10	User's budget is updated in real time

### 3.3 Business Rules

- The customer must not be able to alter his balance that he entered (read only)
- The customer must have a credit card with a balance on it (textpad in this case)

## 4 Non-functional requirements

- The login information (password/username) is encrypted
- The application is intuitive to use requires no computer knowledge

## 5 Design Constraints

The programming language used in this software is Java. The main feature is giving a clear representation of the user's spending over the course of a bank statement. The representation will take the form of graphs and charts defining where their money went. Users can only view their past data and cannot alter



it. The maintenance and feature upgrades are handled by us, the developers of MyMoneyApp.

## 6 Glossary

**User table** The user table is the collection of registered users, along with their login information

**User** A user is the virtual representation of a person using the application

**App account** An account is the model used by the application to store and use a user's information and interact with all the different features present in the application

**Spendings** A user's spending is the agglomeration of every transaction done and entered by the user

**Budget** A user's budget is the collection of the user's projected spendings

**Chart** A chart is the graphic illustration of the user's spending or budget information

**Chart area** A chart area represents a category of the user's projected or actual spendings, represented by a percentage of the total spendings or budget and unique color in the chart, indicating which category the area relates to

**Transaction** A transaction summarizes a positive or negative money transaction made by the user in real life and is composed of the amount of money exchanged during the transaction and the category which the transaction relates to (e.g. food, home, transportation, salary, ...)

**Objective** An objective is an entry reflecting a projected amount spent in a given category. It is composed of a category and an amount. These values are used by the system to produce a budget for the user, illustrated by a chart if the user so chooses)

## 7 References

...