MyMoney App

Comp354

PB-PJ

Matthew Ferderber, Matthew Dugal, Mylene Haurie,
Viktoriya Malinova, Eric Morgan, Artem Khomich,
Kai Nicoll-Griffith, Max Malderle

February 2018

1 Revision History

any big changes done in how the project is set up

Contents

2 Project Description

2.1 Introduction

MyMoney is a desktop application that targets young adults organize and helps them analyse their financial data. The app relies on a user input of data, and can considers multiple accounts at the same time in its analysis. The main end goal of MyMoney is to help visually and statistically represent whether a person is making enough money for its expenses and show which fields of expenses (e.g. eating out, beverages, clothes shopping, house necessities, etc.) is most costly and what percentage of the budget it currently uses up. The app analyses these statistics based on a monthly cycle of expenditures and can alert when a person is spending more money they can afford in a given cycle.

2.2 Context

For iteration 1, we look mainly at transactions. Our app is centered around the idea that each customer class will have one or many bank accounts which can be of any type (ex. Savings, Checking, etc.). The customer class can add transactions for each of its accounts which hold a certain value (if positive, we have a deposit; if negative, we have a withdrawer). The customer class holds a name and a unique ID.

2.3 Business Goals

Max's text

2.4 Scope

MyMoney is a three-month long project that aims to deliver a finance-analysing application. By end of iteration 1, the app should be able to support separate accounts linked to an individual, to have functional and user interactive sections where the client can submit data and Iteration 2 will take 3 weeks and will support statistical analysis. Iteration 3 will take 2 weeks and will put in place money generating advertisement.

2.5 Domain Model

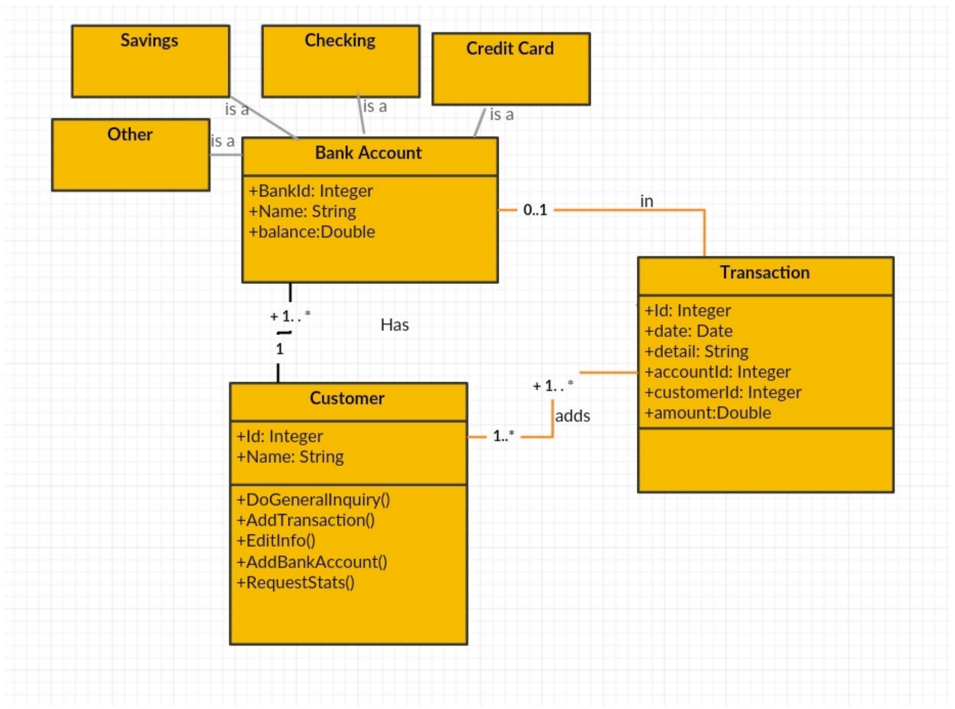

Figure 1: MyMoney Domain Model

2.6 Actors

There are several important actors that will interact with our application perpetually. The most important ones worth considering are the client, the stakeholders, the business manager(s), coders and the testers.

2.6.1 Client

The client (or user) uses the system for the service it offers. He or she may use all or some of the features of the program, namely data entry, analysis of finances, budget reminders, etc. Clients will also view and click the ads which will generate revenue. Their feedback may help fix missed bugs during the testing stage or give direction to features the app is missing.

2.6.2 Stakeholder

Similarly to clients, stakeholder may wish to have an account app to view what their investment looks like after an update or at any other given moment. Based on their observations and interactions with the app, they may inform the business manager the changes they want or the direction they wish the app will take in a future iteration.

2.6.3 Project Manager

He or she will interact with the app and will deal with feedback of the clients that he or she will investigate with his team and will report to coders. He or she may also train new employees and help them get familiar with using the app. As the company grows, we may get more than one business manager, and the quantity of interaction a given manager will have with the app will depend on which sector of the company he or she is managing.

2.6.4 Coder

At the very least, the coders and maintenance staff will have to be familiar with the app to be able to do changes, updates and work on future iterations. They may do testing as they code develops to fix bugs or other problems that may emerge along the way.

2.6.5 Tester

They will put the app through a variety of tests at various points of development due to the test-driven nature of development for this project. After each iteration, they will interact with the final product looking for bugs or inaccuracies in the interface of the program and give feedback based on their observations to the coders.

3 Functional Requirements

Technical functionality, what the program does/will do

3.1 Overview

3.2 Use cases

3.3 Business Rules

4 Non-Functional Requirements

HOW the program functions/will function (performance, reliability, etc)

4.1 Usability

4.2 Reliability

4.3 Responsiveness

4.4 Flexibility

5 Design Constraints

5.1 Time

As a student project, one of the primary restraints is time. Due to the majority of team members being full time students, the scope of the project has to be limited to one that is comfortably feasible within the time frame of the semester in accordance with responsibilities the team has to other courses.

5.2 Team Size

Another constraint is team size. Due to the variety of positions that must be filled, we will only have about 3 programmers at a time. In combination with the aforementioned time constraints, we must limit the functionality of our program to the what we determine are the essential features, and then expand on them as time permits.

5.3 Software

In accordance with the requirements laid out in the project description, the program must be a desktop application developed in Eclipse using Java, and make use of one of Java's GUI packages. Additionally, we are required to use GitHub to maintain and iterate upon our design and LaTeX for documentation. The documentation will also require we utilize UML modeling software to input diagrams when appropriate. Because we must have a way to store and track all of the user's financial information, we also must utilize an SQL database.

5.4 Knowledge

Many of the softwares and some of the coding languages we currently work with are unfamiliar to members of the team. Attending crash tutorials and reviewing online resources in an attempt to fill in the knowledge gaps is both time consuming and cognitively tiring when it is in such concentration. Slack, GitHub and LaTeX all presented us with a learning curve and we still have not discovered all the features that these indispensable tools offer us.

5.5 Budget

The budget is also a constraint, or more accurately, the lack of one is. As a class project, we are not developing the software under

contract or financial limitations, which means it all is being developed on the hardware and software already available to us.

6 Glossary

Eclipse: An integrated development environment used for programming mainly in Java. It is one of the most popular of its kind.

GitHub: Version control system that keeps all changes in a project up-to-date while multiple people work on it simultaneously

GUI: Short for graphical user interface. This interface represents what the end user sees and allows the user to interact with the system without needing to code commands.

Iteration: A cycle consisting of design, development and testing of a new feature that is ultimately added in a overall application

LaTeX: Document preparation software used mainly for scientific and academic documents.

Slack: Online collaboration app based heavily on compartmentalized instant messaging.

SQL database: SQL, short of structured query language, is a language used to manipulate databases. A SQL database is a database which is accessed through this specific programming language.

Test-driven project:A project that is driven by different tests put in place before coding even starts. The goal is that the end product passed these tests that were set from the start.

UML modeling: A visual representation of the main classes, attributes and methods composing an application, website or other code-based product.

References

[1] Computer Hope. "GUI," *Computer Hope*, Oct. 30 2017. [Online]. Available: https://www.computerhope.com/jargon/g/gui.htm. [Accessed: Jan. 31,2018].

[2] JCG. "JavaFX ListView Example," *Java Code Geeks*, March 15, 2o16. [Online]. Available: https://examples.javacodegeeks.com /desktop-java/javafx/listview-javafx/javafx-listview-example/. [Accessed: Jan. 28,2018].

[3] K. Brown. "What Is GitHub, and What Is It Used For?," *How-To Geek*, Sept. 21 2016. [Online]. Available: https://www.howtogeek.com/180167/htg- explains-what-is-github-and-what-do-geeks-use-it-for/. [Accessed: Jan. 28,2018].

[4] M. Mansfield. "What is Slack and How Do I Use It for My Team?," *Small Business Trends*, June 7, 2016. [Online]. Available: https://smallbiztrends.com/2015/12/slack-use-team.html. [Accessed: Jan. 31,2018].

[5] Oracle. "Introduction to FXML," *Oracle*, June 21, 2008. [Online]. Available: https://docs.oracle.com/javafx/2/api/javafx/fxml/doc-files/introduction_to_fxml.html. [Accessed: Jan. 28,2018].

[6] Oracle. "Packages," *Oracle*, 2014. [Online]. Available: https://docs.oracle.com/javafx/2/api/overview-summary.html. [Accessed: Jan. 28,2018].

[7] Oracle. "Skinning JavaFX Applications with CSS," *Oracle*, June 2013. [Online]. Available: https://docs.oracle.com/javafx/2/css_tutorial/jfxpub-css_tutorial.htm. [Accessed: Jan. 28,2018].

[8] Refsnes Data. "Introduction to SQL," *w3schools.com*, 2018. [Online]. Available:https://www.w3schools.com/sql/sql_intro.asp. [Accessed: Jan. 31,2018].

[9] The LaTeX Project Public. "LaTeX A document preparation system," *The LATEX Project*. [Online]. Available: https://www.latex-project.org/. [Accessed: Jan. 31,2018].