The Design and Build of a Simple Personal Finance System, Focused on Budgeting and Expenditure Analysis

Claudius de Moura Brasil
BSc Computing Project Report
Birkbeck College, University of London

May 2018

This report is the result of my own work except where explicitly stated in the text. The report may be freely copied and distributed provided the source is explicitly acknowledged.

1 Abstract

Personal finance systems exist in abundance nowadays, from open source to proprietary ones. They all tend to revolve around a basic common theme: providing accurate information about an individual's income and expenditure. Beyond this, they tend to vary in which features are implemented. The system designed and built for this project focuses on the use of the bookkeeping principle of double entry and the concept of pattern matching to find effective ways to cagetorise a user's expenditure, and provide them with relevant financial information to assist in decision making.

Contents

1 Abstract **i**

2 Introduction **1**

3 Development Method **2**

3.1 The use of Universal Modelling Language (UML) constructs 2

4 Requirements **3**

4.1 Functional Requirements 3

4.1.1 Requirements Model 3

5 Reflections **6**

References **7**

2 Introduction

Vaasen et al. (2009 cited Boczko, 2012, p. 8) suggests that an accounting information system's main purpose is to provide information to internal and external stakeholders. Although this refers to accounting systems for business, it could be argued that the same concept could be applied for personal finance systems – except that, in this case, the main stakeholder would be the individual using the system (that is, the user). In fact, one of the most widely known accounting systems available in the market, Quicken™, was conceived around the idea that there should be more efficient and less tedious ways to organise one's personal financial information than doing it manually (Quicken Inc., 2017). This project has been developed based on these ideas.

This report documents the work for the project. Each chapter delineates a specific aspect of the development lifecycle, which is in line with the development process listed in 3. Chapter 4 identifies the identified requirements which were used as motivation for the system to be developed, and how these requirements were analysed.

Fowler (1997, Chapter 1, Section 1.3) defines a pattern as "an idea that has been useful in one context and will probably be useful in others". This project will therefore attempt to utilise patterns where appropriate in order to prove this concept, and as an attempt to make use of the experience already acquired in the domain (or domains) in question.

3 Development Method

For this project, an approach similar to that adopted by Bennett et al. (2010, p. 77) regarding methodology will be employed, where no specific named methodology is espoused, but concepts of object-oriented analysis and design shall be applied, in an iterative and incremental fashion, using UML. More details about which concepts were used and the methodologies which originated them can be found in the following subsections.

3.1 The use of Universal Modelling Language (UML) constructs

UML is a modelling language created with the intention of providing system architects, software engineers and developers with a common set of modelling tools, with a defined syntax, which would help them better analyse and design software-based systems, and to model business and similar processes (OMG, 2015, p. 43). It defines several constructs which will be employed throughout this report in order to model the specifications of the system, such as:

Use Case diagrams As a useful, high level tool to document users' requirements (Bennett et al., 2010, p. 138), use case diagrams have been used to develop the requirements model of the system.

Activity diagrams

Class diagrams

Sequence diagrams

4 Requirements

Bennett et al. (2010, pp. 140-142) categorises requirements as being of three types:

Functional Requirements The system's functionality – what it is expected to do.

Non-functional Requirements How well the system delivers its functionality. These requirements are related to the performance, scalability, availability, recovery time, security, and others.

Usability Requirements These relate to how effectively, efficiently and satisfactorily users can achieve their goals in the existing system. User interfaces can play a big part in meeting these requirements.

The initial implementation iterations will be focused more on the functional and usability requirements, paying some attention as well to specific non-functional requirements such as performance and security.

4.1 Functional Requirements

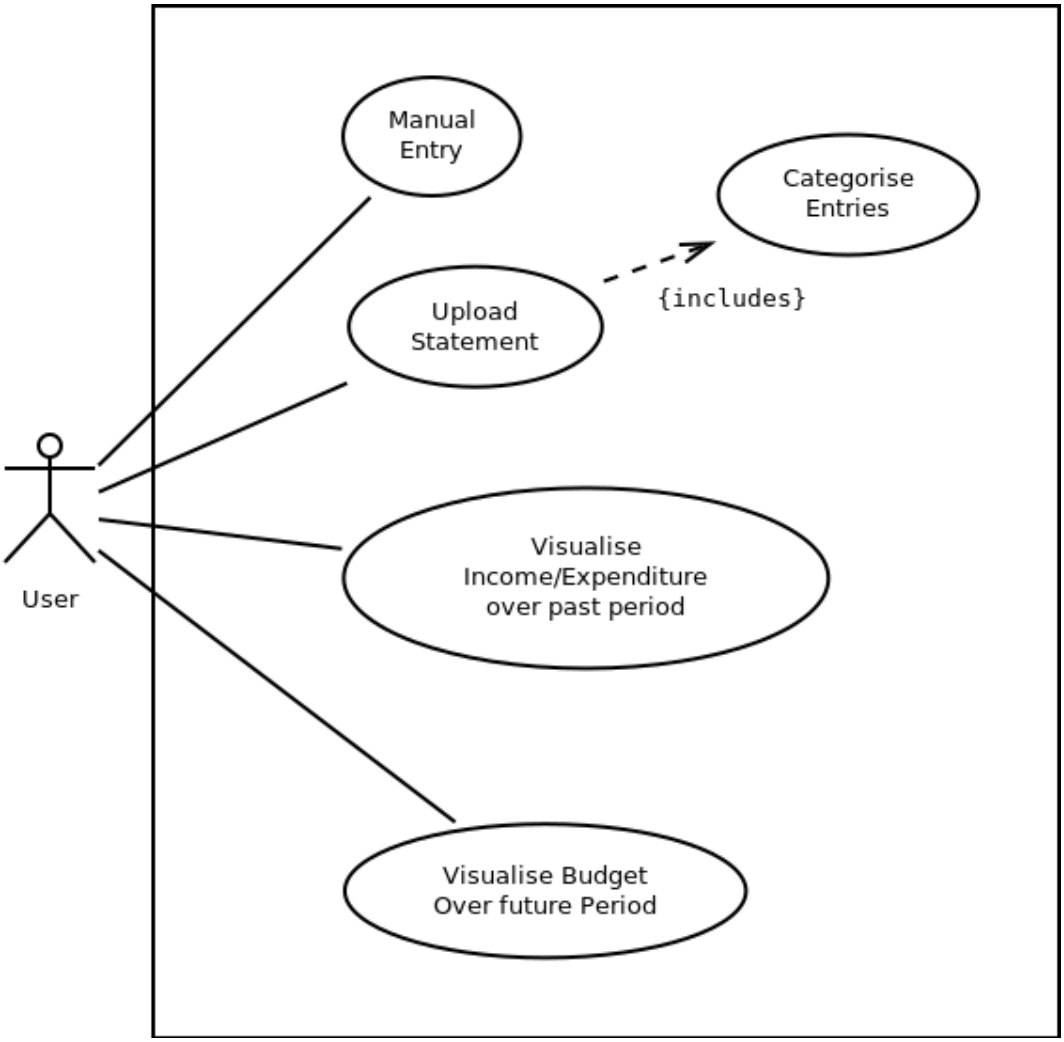
Any personal accounting system should be able to provide accurate and relevant summaries of an individual's financial status. In order to do this, the user needs to be able to supply the system with the relevant data so that it can be analysed and properly converted into the relevant knowledge.

It seems fair to infer that nowadays most of a user's financial transactions currently happen in ways that can be listed electronically – a study by Payments UK (2017), for example, indicates that there has been a rise in debit card payments over the past few years, and that the volumes of this type of transaction is likely to be higher than that of cash payments by the year 2021. Therefore, an assumption has been made that the users will require means of uploading a list of their financial transactions into the system, and this has been identified as a functional requirement.

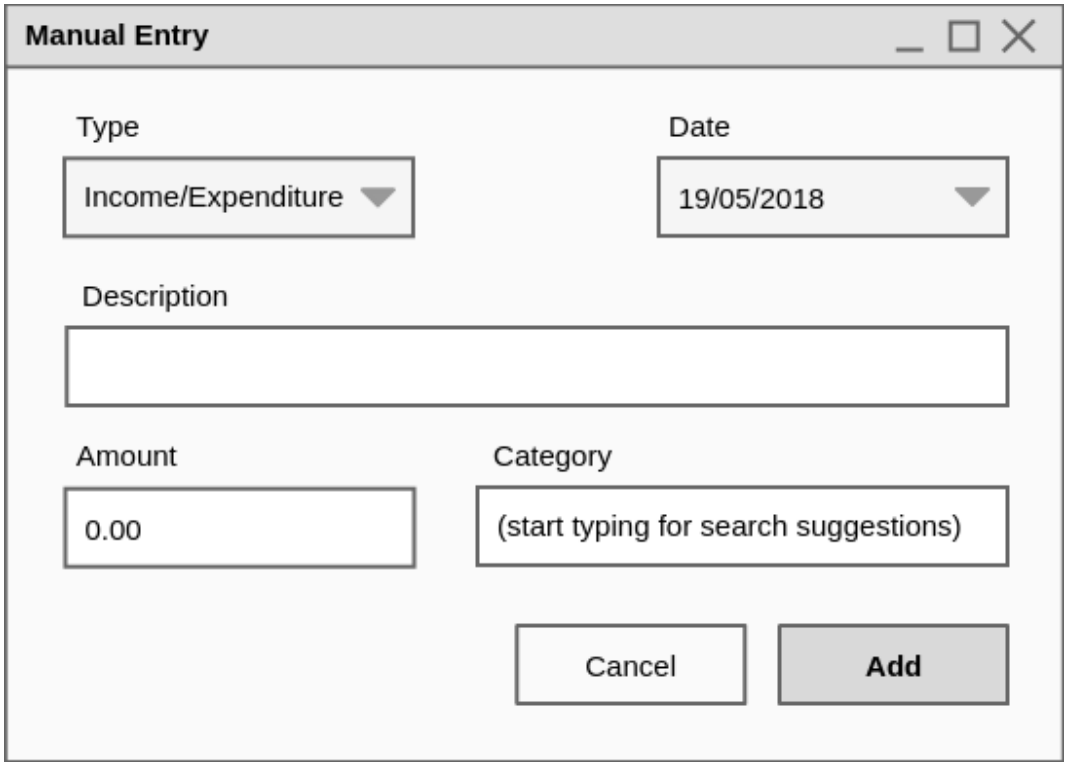
The system created for this project intends to do just this. Its main feature, however, will be a system for the user to categorise expenditure based on patterns in these entries. However, in order not to restrict the system to the uploading of transactions only, it must also be possible for a user to create entries manually.

4.1.1 Requirements Model

Use case diagrams are an UML construct which were developed by Jacobson et al. (1992, cited Bennett et al., 2010, p. 154). The use case diagram below is used to illustrate the functional requirements listed in 4.1:



The wireframe below was created to better illustrate the *Manual Entry* requirement:


Manual Entry
Type
Income/Expenditure
Date
19/05/2018
Description
Amount
0.00
Category
(start typing for search suggestions)
Cancel
Add

5 Reflections

Originally, the author did not know about expert systems when the idea for this project was conceived. However, during the literature search and review for the project the idea for these systems was found, and many of the patterns of what an expert system does and what this system is supposed to do were identified to be similar. This led to the conclusion that the project has the potential to become an expert system, even if just with a budgeting tool. The expert knowledge being provided by it for its first iteration includes:

- Double entry bookkeeping;
- Budgeting by category.

It achieves the above by separating the inference engine, which is the tool responsible for knowing how to apply double entry to transactions, from its knowledge base which is the information input by the user – for example, if the user tells the system a manual entry is income, the system will know to debit the cash book, and credit the category in question.

Brown et al. (1990) declare that the heuristics used by a financial planning system can be interpreted as a "rule of thumb" to be applied to a problem which will normally result in a correct solution for it. In the same article it is also stated that "an expert system is most commonly and most effectively used as an advisor to a human decision maker". I this is considered as the measure by which classify an expert system, then the budgeting tool alone would place this system into it.

References

Bennett, Simon, McRobb, Steve, and Farmer, Ray (2010). *Object-oriented systems analysis and design using UML*. Fourth Edition. McGraw Hill Higher Education. ISBN: 978-0-07-712536-3.

Boczko, Tony (2012). *Introduction to Accounting Information Systems*. Pearson Custom Publishing.

Brown, Carol E, Nielson, Norma I, and Phillips, Mary Ellen (1990). "EXPERT SYSTEMS FOR PERSONAL FINANCIAL PLANNING." In: *Journal of Financial Planning* 3.3.

Fowler, Martin (1997). *Analysis patterns: reusable object models (OBT)*. 1st Edition, Kindle Edition. Addison-Wesley Professional.

OMG (2015). *Unified Modeling Language Specification*. Version 2.5. URL: http://www.omg.org/spec/UML/2.5/PDF (visited on: 16 Dec. 2017).

Payments UK Management Ltd. (2017). *UK Payments Markets Summary*. URL: https://www.paymentsuk.org.uk/files/puk-uk-payment-markets-2017-summarypdf-0 (visited on: 18 Dec. 2017).

Quicken Inc. (2017). *About Us*. URL: https://www.quicken.com/about-us (visited on: 17 Dec. 2017).