SOFTWARE REQUIREMENTS SPECIFICATION

For

Expense Tracker

Prepared By: -

Abhishek Dutta
Department of Software Engineering
Arizona State University

Academic Year: 2016 - 2017

Version: 1.0

Preface

This document is designed for people who will be testing or using this application for their own benefits. It describes about the user requirements and functional requirements that was used to design and build the application. Each chapter tells about the functionality of this application and understand the purpose of it. If you are a software engineering student, then it should help you to be skilled in software development requirements documentation and understand on how to document various functional and non-functional requirements from a client user.

Revision History

Date	Revision	Description	Author
11/26/16	1.0	Initial Version	Abhishek Dutta

Table of Contents

1. Introduction

1.1 Purpose

The objective of this document is to illustrate the requirements of the project Expense Tracker. It tells about the functional and non-functional requirements of this system. The project is mainly focused on maintaining user daily expenses like registers and user can refer it back anytime they want to track their expense for maintaining their budget plans. All users have separate record maintained so it can be used by a group of people without impacting each other's data. This document describes about the software requirements for this. This application can be used both on a small scale or large scale as per user need.

1.2 Scope

Expense Tracker basically is maintaining the user expense information with details as such category, date of entry and the amount spent on the category. A user can go back and update an existing entry or delete an existing entry. Users can also add/delete different members or check other user's expenses. The application takes input using command line console and accepts only a valid entry for an existing user.

The project is mainly designed for the use of students, who want to track their monthly expenses or a family person who wants to keep track of the expenses made by the family members. It can be also useful to a small group of people or a small organization/association who wants to track their expenses made from a common fund. The user interface is a simple command line interface and all the options are clear to understand for a person. The programming language used for building this application is Java and data is stored using Json.

1.3 Definitions, Acronyms, and Abbreviations

ANT	Java library.
IDE	Integrated Development Environment
JAVA	Platform Independent Programming Language
JSON	JavaScript Object Notation
JRE	Java Runtime Environment
SRS	Software Requirements Specification

Table 1. Acronyms and Terminology

1.4 References

- Books
  - Software Engineering: Ninth Edition by Ian Sommerville
  - The Personal Software Process(PSPSM), November 2000 by Watts S. Humphrey

2. Overall Description

2.1 Product Perspective

This application allows users to perform basic functions such as storing the data, updating the data, retrieving it and deleting it in an easy to use command line interface. The options are self-explanatory for a user to understand. The application can be used in any environment as it is built in Java which is platform independent. The application can use this application to store their daily expenses and track them in future for various needs. It can be used by any group of people or individual. I tried to keep it as simple as possible with limited functionality to make it more user friendly. There can be lots of enhancements integrated with the application as per the user requirements.

2.2 Product Architecture

The architecture of this application is a two-layer architecture. Command Line interface is used to do the I/O operations to take user input and the application does the read/write operation to the backend on JSON file. If there is some undesired input, then application itself handles it and responds accordingly to the user.

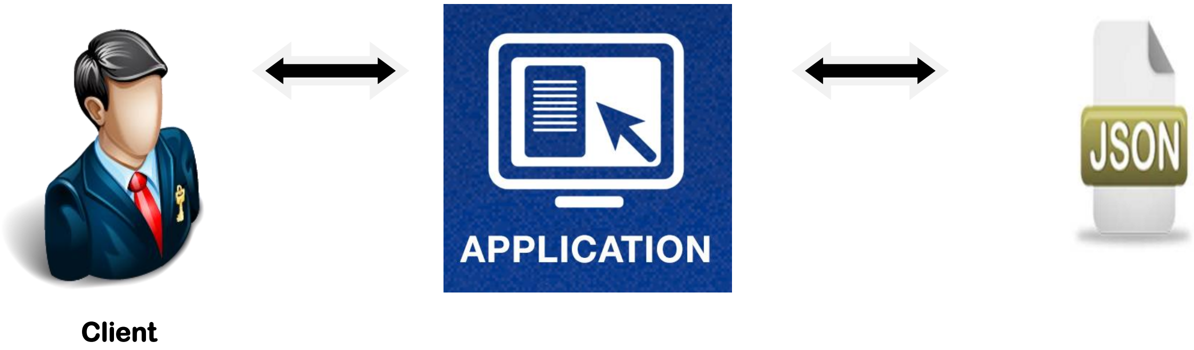

Figure 1. Product Architecture

Client will interact with the command line user interface for giving inputs to the application. Application will process the input given, and if required then will interact with backend JSON to store/retrieve data and respond back to user. If the user input is not a valid input, then application will itself reject user's request with/without interacting with backend JSON files.

2.3 Product Functionality/Features

The expense tracking application offers limited but essential features for different kinds of users. The list of options includes the below options: -

1)User can add/delete members.
2)Users can add/delete expenses for a member.
3)Users can update old entries for a member.
4)Users can view all the entries of expenses for a member.
5)Users can list all members.

3. Specific Requirements

3.1 User Requirements

From a user's perspective, the application should be able to track their daily expenses in the form such that they can refer it in the future and track it for their own purpose. To be specific the application should be able to add or delete any member. It should be able to add/delete or update an existing expense and list all the entries.

3.2 System Requirements

The application can be used in any system. The response should be provided within a fraction of milli seconds. A user should be able to see other members and their expenses. As it is more focused towards homely usage at a small scale, database is not affordable and should not be used. Instead an efficient way to should be used to store data.

3.3 Functional Requirements

As this is a simple application with limited capabilities, there were very few functional requirements associated with it. The main functional requirements from user's perspective was user should be able to add any member, user can add any expense entry for any user in the application, user can modify and delete as well any existing entry. A user can check the expenses of other users. Other functional requirements include that the entry should contain date, category and the amount spend. Date and category field are for user to understand when and where the expense was made in the past. Also, as the application is focused for a common man, it was required to keep the user interface simple and self-explanatory for users to operate easily without any hassle.

3.4 Non-Functional Requirements

Non-functional requirements that are associated with the application are focused more on cost effectiveness keeping in mind about being user friendly. As a non-functional requirement, user can use this application in any environment and that is why we built this application in Java as it is an open source and platform independent. The application should keep on checking for user inputs without being executing it each time for doing any operation. Also, this application deals with data but due to cost effectiveness, it was required to store the data in file system efficiently and retrieve back from it as databases are costlier and would have taken more response time than file based system.

3.5 Software Requirements

The below mentioned software are needed to be installed in the system for running this application:

- Java Development Toolkit JDK 1.8+
- Apache Ant 1.8

3.6 Hardware Requirements

A standard computer system should be able run this application easily. Any system with below minimum configurations are sufficient to host it.

- 2 GB RAM
- X86 or x64 bit Windows/Linux Operating system
- 50 GB Hard drive (Depends on usage to store the data)
- 1.6 GHZ Intel CPU

4. Requirements Traceability

The table below shows the traceability of the application, this table will be constantly updated as the development process progresses. A detailed list of the test cases can be found in the Test Plan document, please refer to that document to see which test cases are being referenced here below. The table below only shows the requirements that have a test plan implemented.

Requirement	T1	T2	T3	T4	T5	T6	T7
Add a Member	X						
Display Members			X				
Add expense for a valid member				X			
Add expense for an invalid username				X			
Add expense for a valid user with an invalid entry				X			
Update expense for a valid user						X	
Update expense for an invalid user						X	
Update expense for a valid user with an invalid entry.						X	
Display the expenses							X
Delete expenses for a valid user					X		
Delete expenses for an invalid user					X		
Delete expenses for a valid user with invalid entry					X		
Delete a member for a valid user		X					
Delete a member for an invalid user		X					

Table 2. Traceability of the requirements

5. Bibliography

1) Software Engineering: Ninth Edition by Ian Sommerville
2) The Personal Software Process(PSPSM), November 2000 by Watts S. Humphrey