California State University, Sacramento
College of Engineering and Computer Science
CSC 131-05: Computer Software Engineering

SOFTWARE REQUIREMENTS SPECIFICATION

(Requirements Phase - Waterfall Methodology)

for

PROJECT BRAVO

Prepared By: Sparsh Saraiya

Table Of Contents

Chapter 1: Introduction

1.1 Purpose & Scope:

This Software Requirement Specification (SRS) document was created to define both the requirements and intended functionalities of the **Expense Tracker** software solution that was assigned to Team Bravo. This document will define the functionalities, requirements (both functional & non-functional), scope, objectives, and overall features of said software solution. This document should be used by Team Bravo's development team in accordance with the "Requirements" phase of the Waterfall Methodology. The **Expense Tracker** software solution is a desktop application where end-users will be able to log their daily, weekly, monthly, annual income and expenses; the application will then formulate a summary (written and visual) of said finances for the end-users to interact with. Within the application there will be individual windows that end-users can interact with where they can do the following: add income & expenses for interval tracking, visually display and categorize their expenses into groups, view a monthly review summarizing all their finances, and edit or delete entries whenever the user chooses to. There will also be an administrative window that allows an administrator to view and manage user entries, generate individual and combined financial reports, and oversee all registered users and their roles within the system. This software solution is intended for users who are in need of a reliable, user-friendly, and clean application that helps manage personal finances. **This document matches the Requirements Analysis and Requirements Specification phase of the Waterfall model.**

Chapter 2: System Overview

2.1 Problem Definition:

In order for one to be successful financially one must track their finances. Managing personal finances, such as tracking income and budgeting, manually through paper records is inherently inefficient and time consuming. When using manual approaches the user will often run into problems such as inconsistent record-keeping, calculation errors, and limited insight into spending habits. Without a centralized place to manage finances, users will struggle with managing, budgeting, and planning their individual finances. Because of this problem, there is a need for a centralized, efficient, and user-friendly application that can solve the problems that manual methods present when one manages their personal finances.

2.2 Problem Solution:

The solution to fix the issues that manual financial tracking brings is through the **Expense Tracker** desktop application. This desktop based application allows users to easily and interactively track, categorize, and understand their income and expenses. The application separates the user interface into separate activity windows for adding transactions, investigating monthly summaries, and determining spending habits. The layout of the application makes it simple for end users to understand their financial activity, editing (deleting) transactions at any time to keep their account up to date. In addition to the user features, a separate administrative window will be included in the application for an administrator to manage users, view an individual or combined financial activity, and report information for one user or across multiple users. The administrative user account will serve as an interfacing mechanism to support personal financial management, while also allowing administrator access to oversight. Overall, the **Expense Tracker** will provide a simple, straightforward, and reliable software system to assist users in taking control of their finances with the appropriate data, thus enabling the end user to make spending decisions based on their financial position.

Chapter 3: Requirement Elicitation & Analysis

3.1 Methods Used:

The requirements for the **Expense Tracker** application were identified at a series of meetings and planning sessions held by the team. Several meetings were also held online where each group member was present to read, discuss, and interpret the project requirements provided in the project document. Roles were assigned during the meetings and subgroups were formed to focus on particular elements of the project including, front-end design, back-end development, documentation, and UML diagram design. To increase organization and coordination, the team also used tools such as Trello and other platforms to centralize tracking of tasks, responsibilities, and progress of the project. The use of these platforms also ensured accountability that each task was done by each member, as well as a consistent communication method within the group to communicate what tasks were being completed. The team also searched several existing expense-tracking tools to find what standard and common features were typically used, user interface conventions, and best practices that were commonly found in the expense management website. The features, functions, and limitations of the existing expense tracking applications, along with the requirements outlined in the assignment helped the team in establishing the functional and non-functional requirements for the **Expense Tracker** application.

3.2 Summary Findings:

The team's analysis of the project requirements, connected with our discussions and observations of existing expense tracking applications found online, led to several findings regarding the features and design of the Expense Tracker system. The analysis and discussions provided input on features and priorities for the Expense Tracker system. The highest priority functional requirement was identified as the need for an user-friendly interface that allows end users to input and categorize their income and expenses. Users should see summaries of their financial records that include monthly and yearly summary information, and they should have the ability to modify or delete records as necessary. Important features established in discussion were visual representations of expense records in charts or graphs so users could better interpret and understand their spending trends. Another important feature is the need for some sort of an administrative feature, at least for overseeing and managing data, where an administrator could view and modify user entries, produce individual and combined financial reports, and administer the user roles within the system. From a nonfunctional perspective, the system must accommodate usability and responsiveness and handle data accurately. The application should run reliably on a standard desktop environment that has cross platform compatibility, efficiently store financial data in an organized and secure manner, and be user friendly for those of varying levels of technical knowledge. All of these findings formed the foundation for the functional and nonfunctional requirements section in the requirement specification chapter.

Chapter 4: Requirement Specification

4.1 Functional Requirements:

User Functional Requirements
FR1: The system must allow users to create a new account and securely log in to the application.
FR2: The system must allow users to add income and expense entries for daily, weekly, monthly, or yearly intervals.
FR3: The system must allow users to categorize each expense into predefined or user-created categories (e.g., Food, Rent, Entertainment, Transportation).
FR4: The system must allow users to view all recorded income and expenses in a summarized list or table format.
FR5: The system must generate a monthly financial summary displaying total income, total expenses, and remaining balance.
FR6: The system must visually represent spending data using charts or graphs for easy interpretation.
FR7: The system must allow users to edit or delete existing income or expense entries.
FR8: The system must allow users to filter transactions by date, category, or type (income/expense)
FR9: The system must store all user data in a secure local database or file system.

Administrative Functional Requirements
FR10: The system must allow an administrator to log in with administrative credentials.
FR11: The system must allow the administrator to view all registered users and their respective roles.
FR12: The system must allow the administrator to view, edit, or delete user financial entries as needed.
FR13: The system must allow the administrator to generate individual financial reports for a specific user.
FR14: The system must allow the administrator to generate combined financial reports for all users.
FR15: The system must allow the administrator to manage categories and system configurations.

General Functional Requirements
FR16: The system must support intuitive navigation between all windows (Home, About, Credits, Contact, and Expense Management).
FR17: The system must display a footer section in every window containing clickable icons linking to the project's social media pages (LinkedIn, GitHub, Facebook, Twitter, Instagram).
FR18: The system must provide a Contact window with fields for name, email, phone number, message, and a functional Submit button that sends contact messages to all team members.
FR19: The system must validate all input fields to ensure that data entered by the user is complete and in the correct format.
FR20: The system must allow multiple windows to be open simultaneously and enable users to rearrange them as needed.
FR21: The system must provide accessibility features such as resizable fonts, high-contrast color schemes, and keyboard shortcuts.

4.2 Non-Functional Requirements:

Usability Requirements

NFR1: The system must provide a clean, intuitive, and easy-to-navigate graphical interface suitable for users with minimal technical experience.
NFR2: All key actions (adding, editing, or deleting entries) must be accessible within three or fewer user interactions.
NFR3: The system must use clear labels, icons, and color schemes to ensure consistency and readability across all windows.

Performance Requirements

NFR4: The system must load each primary window (Home, Add Entry, Reports, Admin) within three seconds under normal conditions.
NFR5: The system must support simultaneous operation of multiple windows without significant delay or application crashes.
NFR6: The system must handle a minimum of 5,000 income or expense records without performance degradation.

Reliability and Availability Requirements

NFR7: The system must operate reliably during extended use without unexpected termination or data loss.
NFR8: The system must automatically save new entries to the local database or file system immediately after submission.
NFR9: The system must recover from unexpected shutdowns by loading the most recent saved state.

Security Requirements

NFR10: The system must require secure login credentials for both users and administrators.
NFR11: User passwords must be stored in an encrypted format to prevent unauthorized access.
NFR12: Administrative privileges must be restricted to verified users with appropriate credentials.

Compatibility and Portability Requirements

NFR13: The system must run on major desktop operating systems, including Windows 10/11, macOS, and Linux.
NFR14: The system's interface and core functionalities must display consistently across supported platforms.
NFR15: The application must be developed using technologies that support future scalability and maintenance.

Maintainability Requirements

NFR16: The source code must follow consistent naming conventions, indentation, and documentation for ` readability.
NFR17: The system must be modular to allow easy updates or replacement of components without affecting other features.
NFR18: All changes or updates to the system must be documented in a version log for maintenance tracking.

Accessibility Requirements

NFR19: The system must support keyboard shortcuts for primary actions to enhance accessibility.
NFR20: The system must provide resizable fonts and a high-contrast color option for users with visual impairments.

Chapter 5: Requirement Validation

5.1 Validation Approval:

The requirements for the **Expense Tracker** system were validated through team review meetings and comparison to the official project specifications from the professor. Each functional and non-functional requirement was assessed for feasibility, scope and alignment to the objective of the system. Peer review between team members helped to identify missing/determined requirements and requirements with conflicting nature. Because of this section, we were able to ensure that all requirements were clear, testable and aligned to the project as a whole.

5.2 Validation Checklist:

The team used the following checklist to verify the requirements of the Expense Tracker system:

1. **Completeness**: All requirements for a functional system are present; no requirements that should be included have been omitted.
2. **Consistency**: All requirements are consistent and do not contradict other functional or non-functional requirements.
3. **Feasibility**: All requirements are feasible to implement within the time frame, technical knowledge, and resources allocated.
4. **Clarity**: The requirements are written clearly without ambiguity so that all team members interpret the same requirement consistently.
5. **Traceability**: All requirements can be traced back to the project goals or user needs identified in previous chapters.
6. **Verifiability**: All requirements can be tested or measured during the software validation phase to determine if the requirement has been satisfied.
7. **Relevance**: All requirements are relevant to the solution of the identified problem and are aligned with the overall goals of the project.
8. **Prioritization**: All requirements are prioritized in logical order so that the most critical user and administrative design functions are implemented first.