Software Requirements Specification

For

Personal Finance Tracker

Prepared by:
Anshuman Singh
Krishna Kumar
Mohit Kumar Dubey
Shivam Singh Barman

10th October 2024

Table of Contents

1. Introduction

1.1 Purpose
This Software Requirements Specification (SRS) provides a comprehensive overview of the Personal Finance Tracker application, detailing both functional and nonfunctional requirements. It serves as a foundation for development, ensuring all stakeholders align on project goals and deliverables.

1.2 Document Conventions
Requirement: A feature or function that the system must support.
User: An individual utilizing the Personal Finance Tracker.
Admin: A user with management privileges overseeing application functionality.

1.3 Intended Audience and Reading Suggestions
This SRS targets project stakeholders, including project managers, developers, testers, and end-users. It is recommended that readers review the purpose and scope sections to grasp the overall vision before examining specific requirements.

1.4 Product Scope
The Personal Finance Tracker enables users to effectively manage their financial transactions through CRUD operations. It provides a secure interface for user authentication, expense tracking, and reporting capabilities, ensuring that users can monitor their financial health efficiently.

1.5 References
React.js Documentation
MongoDB Documentation
JWT Authentication Standards
Flask Documentation

2. Overall Description

2.1 Product Perspective
The Personal Finance Tracker will be a web-based application developed using HTML, CSS, React.js, and JavaScript for the frontend, and Flask with MongoDB for the backend. The system will employ JWT for secure user authentication, facilitating a seamless user experience.

2.2 Product Functions
User Signup and Login: Secure user registration and authentication processes.
CRUD Operations for Expenses: Create, read, update, and delete expense entries.
Expense Reports: Generate and view summaries of expenses over selected time frames.
User Authentication and Session Management: Ensure secure user access and manage user sessions.

2.3 User Classes and Characteristics
Regular Users: Individuals managing personal finances, requiring ease of use and quick access to expense data.
Admin Users: Optional users who manage the application and oversee user accounts, requiring access to administrative functionalities.

2.4 Operating Environment
The application will operate on modern web browsers (e.g., Chrome, Firefox, Safari) and will be hosted on a cloud-based server to ensure high availability and scalability.

2.5 Design and Implementation Constraints
The application must be developed using React.js for the frontend.
MongoDB must be the chosen database for data storage.
JWT must be implemented for secure user authentication.

2.6 User Documentation
Comprehensive user manuals and online help documentation will be available, providing guidance on system navigation and troubleshooting common issues.

2.7 Assumptions and Dependencies
Users will have internet access to use the application.
Users are expected to possess basic web navigation skills.

3. External Interface Requirements

3.1 User Interfaces
Login/Signup Pages: Simple and secure forms for user authentication.
Dashboard: A centralized interface for users to manage expenses, view summaries, and access reports.
Expense Management Forms: User-friendly forms for creating and editing expense entries.
Reports Page: A dedicated section for generating and viewing expense reports.

3.2 Hardware Interfaces
The application does not require specific hardware interfaces but must be compatible with devices capable of running modern web browsers.

3.3 Software Interfaces
MongoDB: For database management and storage of user and expense data.
JWT: For secure authentication and session management.

3.4 Communications Interfaces
REST API: For communication between the frontend and backend, enabling data exchange and functionality implementation.

4. System Requirements

4.1 User Signup/Login
4.1.1 Description and Priority
This feature allows users to create an account and log in securely, providing essential access to the application. Priority: High.

4.1.2 Stimulus/Response Sequences
Stimulus: User submits the signup/login form.
Response: System validates credentials and provides appropriate feedback (success or error message).

4.1.3 Functional Requirements
Users must be able to sign up using a valid email address and password.
Users must be able to log in using their registered credentials.

4.2 Create Expense
4.2.1 Description and Priority
Allows users to record new expenses, enabling effective financial tracking. Priority: High.

4.2.2 Stimulus/Response Sequences
Stimulus: User submits the new expense form.
Response: The expense is saved in the database, and confirmation is displayed.

4.2.3 Functional Requirements

Users must be able to enter expense details, including amount, category, date, and description.

4.3 Read Expenses

4.3.1 Description and Priority

Enables users to view their recorded expenses, providing insight into their financial activities. Priority: High.

4.3.2 Stimulus/Response Sequences

Stimulus: User requests to view expense reports.
Response: The system displays a list of expenses based on user-defined filters.

4.3.3 Functional Requirements

Users must be able to filter expenses by date range and category.

4.4 Update Expense

4.4.1 Description and Priority

Allows users to modify existing expenses, ensuring accuracy in financial records. Priority: Medium.

4.4.2 Stimulus/Response Sequences

Stimulus: User submits an updated expense form.
Response: The system updates the expense in the database, and confirmation is displayed.

4.4.3 Functional Requirements

Users must be able to edit any detail of a recorded expense.

4.5 Delete Expense

4.5.1 Description and Priority

Enables users to remove expenses, maintaining accurate financial tracking. Priority: Medium.

4.5.2 Stimulus/Response Sequences

Stimulus: User requests to delete an expense.
Response: The system confirms deletion and updates the records accordingly.

4.5.3 Functional Requirements

Users must be able to delete an expense from their records.

5. Nonfunctional Requirements

5.1 Performance Requirements

The application should support at least 100 concurrent users without noticeable degradation in performance, ensuring smooth operation during peak usage.

5.2 Safety Requirements

User data must be protected against unauthorized access through secure storage and management practices.

5.3 Security Requirements

All user data should be encrypted in transit (using HTTPS) and at rest (in the database).
JWT should be securely implemented to protect user sessions, ensuring that tokens are signed and validated properly.

5.4 Software Quality Attributes

Usability: The interface should be intuitive, allowing users to navigate easily and efficiently.
Reliability: The application should maintain 99% uptime, ensuring users can access their financial data when needed.

5.5 Business Rules

Users must provide valid email addresses during signup.
Each expense must belong to a specific category, which should be predefined in the system.

Appendices

Appendix A: Glossary
CRUD: Create, Read, Update, Delete.
JWT: JSON Web Token, used for secure user authentication.

Appendix B: Analysis Models
Use Case Diagram (to be created).
Data Flow Diagram (to be created).

Appendix C: To Be Determined List
Further details on performance metrics.
Additional user interface specifications, including accessibility considerations.