Software Requirements Specification

Coin Calendar

Team 5, Section 5 : Joey Mule, Bodie Nies, Matthew Smith

2024-04-18

University of Maryland, Baltimore County

CMSC 447 : Software Engineering I

Contents

1 Introduction **2**
- 1.1 Purpose . . . 2
- 1.2 Intended Audience . . . 2
- 1.3 Scope . . . 2
- 1.4 Definition . . . 2
- 1.5 References . . . 3

2 Overall Description **3**
- 2.1 User Interface . . . 3
- 2.2 System Interface . . . 4
- 2.3 Software and Hardware Requirements . . . 4
- 2.4 Constraints . . . 5

3 System Features and Requirements **5**
- 3.1 User Interface . . . 5
- 3.2 System Interface . . . 6
- 3.3 External Interface Requirements . . . 6
- 3.4 Database Requirements . . . 6
- 3.5 Non-Functional Requirements . . . 6

4 Deliver for Approval **7**

1 Introduction

1.1 Purpose

This SRS (Software Requirements Specification) outlines the functional and non-functional requirements for the deployment and release 1.0 of the Coin Calendar project. This document should be utilized by both development members as well as the associated UMBC staff to verify the correct functionality of this software. All requirements declared and explained in this document are going to be in this initial release.

1.2 Intended Audience

The intended audience for this web application includes people of all ages who are budget-conscious and want to manage their personal finances and schedules more effectively. For youth, it provides a user-friendly platform to learn budgeting skills. For the more experienced, it can streamline finances and plans for budgeting while also managing appointments or notable events. Its versatility promotes financial literacy and efficient time management across all life stages.

1.3 Scope

The Coin Calendar project entails the development of an innovative event planner and calendar concept integrated with a budget management tool. Users will have the capability to seamlessly track their expenses, deposits, and recurring bills within both a calendar interface and a comprehensive dashboard setting. This multifaceted system aims to provide users with enhanced financial visibility and organization while effectively coordinating their schedules and events.

1.4 Definition

The Coin Calendar application is a financial planning and event management tool designed to help users manage their finances and schedules effectively. It allows users to add, edit, and remove events, expenses, and recurring bills on a calendar, as well as track their balance and expected balance based on the expenses they add.

1.5 References

- **React Documentation**: React JavaScript library
- **React Big Calendar Documentation**: React Big Calendar library used to render the calendar
- **Date-Fns Documentation**: date-fns library used for date formatting and manipulation
- **Axios Documentation**: Axios library for making HTTP requests to the backend API

2 Overall Description

2.1 User Interface

The Coin Calendar application provides an intuitive and visually appealing user interface that consists of several key components:

- **Calendar View**: A monthly or weekly view that displays events, expenses, and recurring bills. Users can easily navigate between dates and manage their schedules.
- **Dashboard**: An overview of the user's financial status, including balances, recent transactions, and upcoming expenses.
- **Event and Expense Management**: Forms and interfaces allow users to add, edit, and remove events and expenses. They can categorize and set reminders for each entry.
- **Budget Tracker**: Users can monitor their spending against a set budget, providing insights into their financial habits.

The interface is designed to be responsive and user-friendly, adapting to various screen sizes and providing a consistent experience across devices.

2.2 System Interface

The Coin Calendar application interacts with other systems and services in the following ways:

- **Backend API**: The application communicates with a backend API to retrieve and update data related to events, expenses, and user information.
- **Authentication**: Integration with an authentication service for secure user login, password reset, overall and account management.
- **Notification Service**: Sends alerts and reminders for upcoming events and due dates through email or in-app notifications.

These interfaces ensure smooth operation and enable the application to leverage external resources efficiently.

2.3 Software and Hardware Requirements

The Coin Calendar application requires the following software and hardware to function effectively:
Software:

- Web browser (latest versions of Chrome, Firefox, Safari, or Edge)
- Backend server with the necessary API and database support
- Frontend frameworks/libraries such as React and associated dependencies

Hardware:

- Desktop or mobile device with internet connectivity
- Server infrastructure for hosting the backend and database

These requirements may vary depending on the specific technology stack chosen for development.

2.4 Constraints

The Coin Calendar application faces certain constraints during development and operation:

- **Regulatory Compliance**: Adhering to privacy and data protection laws (e.g., GDPR) to safeguard user data.
- **Performance**: Ensuring fast and efficient performance, even with a large volume of events and expenses.
- **Compatibility**: Maintaining compatibility with different devices, browsers, and operating systems.
- **Scalability**: The application should scale efficiently to accommodate growing user numbers and data volumes.

3 System Features and Requirements

3.1 User Interface

Key features and requirements for the user interface include:

- **Event Management**: Users can create, edit, and delete events and expenses on the calendar.
- **Categorization**: Users can categorize events and expenses for easier tracking.
- **Recurring Bills**: Users can set up and manage recurring bills with reminders.
- **Budget Tracking**: Users can set budgets and monitor their expenses against them.

The interface should be intuitive and accessible for a broad audience.

3.2 System Interface

- **API Integration**: Ensure seamless communication with the backend API for data retrieval and updates.
- **Notification Handling**: Integration with notification services for sending alerts and reminders.
- **User Authentication**: Interface with authentication services for user login and account management.

3.3 External Interface Requirements

- **Analytics Tools**: Optionally, integration with analytics tools for monitoring user interactions and improving the application.

3.4 Database Requirements

- **Data Storage**: The database should handle user data, events, expenses, and balance settings securely.
- **Data Security**: Implement strong security measures to protect user data from unauthorized access.

3.5 Non-Functional Requirements

- **Performance**: Fast load times and efficient handling of user interactions.
- **Reliability**: The application should operate consistently and reliably, with minimal downtime.
- **Scalability**: The application should scale smoothly as the user base and data volume grow.
- **Usability**: The application should be easy to navigate and use, even for those unfamiliar with similar tools.
- **Security**: Protect user data and transactions with encryption and secure authentication methods.

4 Deliver for Approval

- Ensure the document is complete, accurate, and meets the criteria set by the **CMSC 447** Professor and TA.
- Obtain formal approval from the TA and professor of **CMSC 447**.