SYSTEM REQUIREMENT SPECIFICATION FOR

Expenditure Management Web Application

Ramkumar R G	**16IT132**
Rahul A R	**16IT239**
Swapnil Chavan	**16IT214**

Table Of Contents

A. Table Of Contents

1. Introduction

1.1. Purpose

1.2. Scope

1.3. Definitions, acronyms, and abbreviations

1.4. References

1.5. Overview

2. Overall Description

2.1. Product Perspective

2.2. Product Functions

2.3. User Characteristics

2.4. Constraints

2.5. Assumptions and Dependencies

3. Specific Requirements

3.1. External Interfaces

3.2. System Features

3.3. Performance Requirements

3.4. Logical Database Requirements

3.5. Design Constraints

3.6. Software System Attributes

4. Appendixes

1. Introduction

This Software Requirements Specification specifies the requirements of the Expenditure Management Web Application through which users can enter in their weekly budget, their expenditure details and items they wish to buy using their savings. This Web Application supports all the latest browsers. Expenditure Management is a difficult task and people often forget to keep track of how they have spent their money. This Application will help the users to keep track of their weekly expenditures with details like item name, price, date of purchase, etc. and will give an insight to the users on how they are spending their money and how they can save it by cutting some of the costs. Users can then buy items that they had wish-listed using the money they save which feels rewarding.

1.1 Purpose

The purpose of this document is to give a detailed description of the requirements for the Expenditure Management Web Application. It will explain system constraints, interface and interactions with other external applications. It defines what problems the software development process will face and the solutions. This document is primarily intended to be proposed to a customer for its approval and a reference for developing the first version of the system for the development team and it also serves to ensure that the software is traceable throughout its software development life cycle.

1.2 Scope

The application is free to use as a website on a desktop PC or as a web application on a smartphone. It can be used by people of any profession. Children can use it to manage their pocket money and save up to buy something they like. It can be used by house-makers to manage the house expenses. It can be used by people who have jobs or small businesses to manage their daily expenses. The application requires the user to enter their weekly budget and the details of expenditures and it automatically calculates the savings. All information of the users will be securely stored in a database. An administrator will administer the database and ensure the correctness of the data. The administrator can verify the users and also receive feedback for the application and update the application or fix bugs. The application will be able to provide the users their expenditure history and savings history and also notify users if they have enough savings to buy an item from their wish-list.

1.3 Definitions, acronyms, and abbreviations

User	Enters budget, expenditure details, items in ToDo list, ToBuy list
Admin	Manages database, checks user authenticity, updates app
App	Abbreviated form of Application
Node.Js	Open source server framework
MongoDB	Open-source cross-platform document-oriented database program
GPS	Short for Global Positioning System

1.4 References

[1] IEEE Software Engineering Standards Committee, “IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications”, October 20, 1998.

1.5 Overview

The remainder of this document includes two chapters. The second chapter provides an overview of the system functionality and system interaction with other systems. This chapter also introduces different types of stakeholders that will use the system and what functionality is available for each type.

The third chapter provides the requirements specification in detailed terms and a description of the different system interfaces.

2. Overall Description

This section will give an overview of the whole system. It will provide the perspective of the product, product functions, user characteristics, constraints, assumptions and dependencies.

2.1 Product Perspective

Expenditure Management App enables users to note down their expenditure cost, time and place whenever they spend money so that they do not forget about that later, view their savings and also review their expenditure transactions. It requires the user however to have a device with him/her with web browser installed and connected to the internet. This means that the users of the system do not need to invest in any other software to get the most out of the software system as most of of the devices come with a browser installed.

2.2 Product Functions

The important functions of this software are as follows:

- Authenticate user
- Log down details of expenditure
- Review the transaction history
- Log To Buy details and notify when user can buy
- ToDo list on that day

2.3 User Characteristics

There will be three types of users using the software. Guest user, Signed In user and admin. The Guest user will have access to login / sign up page and homepage about the software. In short they will have to login / sign up.
The signed in user will have access to all the features of the software. It will enable him to add and view transactions, view savings, add things to to buy list, add / delete stuff from ToDo list and also to give feedback.

The admin user will have access to User Database. Thus can remove fake users or reported users. He will also have access to feedback database. They will have access to the server to modify the actual software whenever necessary.

2.4 Constraints

Since the software will be an web application the user must a have web browser installed in the device. The software should work well in all major browsers like Google Chrome, Mozilla Firefox, Microsoft Edge and Internet Explorer, but some functions may not be the same in all as different browsers are coded differently.

Internet Connection is a must since the application fetches data from the database over the internet.

The amount of data our software can hold is constrained by the capacity of the database.

Since there will be no credit / debit card linked to the software the user will have to type the transaction information manually.

2.5 Assumptions and dependencies

- The system will be running on a windows computer with NodeJs and mlab will be used as our database (MongoDB).
- The user must enter all the transactions he makes regularly.
- The user must have his device connected to the internet while updating/viewing transactions, ToDo list, viewing savings and giving feedback.
- The user must type the information about every expenditure correctly to get the full advantage of the software.

3. Specific Requirements

This section specifies the detailed requirements which the system shall meet.

3.1 External Interfaces

3.1.1 User Interfaces

The user interface will be a GUI implemented using html, css and javascript. It will be responsive so that there will be convenience in using it on any device of user's choice.

The guest user of the application will see the log-in page when he/she opens the application. If the user is not registered he will have to sign up by providing Name, username, password and email id or else login by providing username and password in the username and password fields.

Once the guest user is signed in, he will be given the option related to Expenditure, Savings, ToDo List, ToBuy List, Feedback and account settings.

In Expenditure route there will be options for the user to add a new Expenditure which will consist of amount, place, optional description. Next there will be an option to view expenditure history and at last the net amount remaining this month.

In Savings route there will be an option to see savings history and total amount saved.

In ToDo list the logged in user will be able to add or delete ToDo items.

In ToBuy list the logged in user will be able to add or delete ToBuy items. If a item is feasible to buy, the item will have a background colour of light green.

In Feedback route, the user can give number of stars and a form to fill feedback.

In account settings route, the user gets to change budget of the week, add user description and profile picture.

3.1.2 Hardware Interfaces

The web application does not have any direct hardware interfaces.

3.1.3 Software Interfaces

To get the location the application communicates with GPS application to get geographical information. MongoDB database will be used to store the information and the server will be a Node.Js server running on a windows system.

3.1.3 Communication Interfaces

The communication between the different parts of the system is important since they depend on each other. However, in what way the communication is achieved is not important for the system and is therefore handled by the underlying operating systems.

3.2 System Features

3.2.1 User login / Register

3.2.1.1 Introduction/ Purpose of feature

This will allow first users to register into the application and already registered users to login into the application. This feature is used to authenticate the user. This is a high priority feature

3.2.1.2 Stimulus/Response sequence

Stimulus: User requests to login.
Response: System provides a form for the user to enter the username and password. If such a user exists in database then the user will be authenticated.

Stimulus: User requests to register.
Response: System provides a form for the user to fill the essential details. If there does not exist another user with same username the user will be registered and authenticated.

3.2.1.3 Functional Requirements

FR ID	FR Name	Description
FR-A1	Login Authentication	If the user is successfully logged in a token will be issued to the user. The token will expire in 2 days. The user will have to login again afterwards.

FA-A2	Register User	If user enters correct details with username not matching any of the usernames in db, then the user will be successfully registered and authenticated using FR-A1

3.2.2 Expenditure Management

3.2.2.1 Introduction/ Purpose of feature

In this feature the user can view the transaction history, add new expenditure and view the amount left for the week to spend. This is a high priority feature.

3.2.2.2 Stimulus/Response sequence

Stimulus: User requests to add a new expenditure information.
Response: System provides a form for the user to enter the expenditure details with cost, item, item description (optional), default location set to current user location (optional). Remaining budget of the week will be shown. This response will be added on to the database.

Stimulus: User requests to view expenditure history.
Response: The system send all the expenditure information till date to the user from database, which will be rendered on the web page.

Stimulus: User requests to view expenditure history of a particular date.
Response: The system send all the expenditure information of that date to the user from database, which will be rendered on the web page.

Stimulus: User requests to view money left and spent till that day of the week.
Response: System will send the current savings and amount spent to the user.

3.2.2.3 Functional Requirements

FR ID	FR Name	Description
FR-E1	Add Expenditure	The application will allow the to add new expenditure details to his list of expenditures.
FR-E2	Get Location	The application will get the location of the user by locating the device.
FR-E3	View Expenditure History till date	The application will allow user to check expenditure history of the user till that day.
FR-E4	View Expenditure History on a given date	The application will allow user to check expenditure history of the user on the given date..

3.2.3 Savings Management

3.2.3.1 Introduction/ Purpose of feature

In this feature the user can view the savings history week wise and total savings upto that day. It has a medium priority.

3.2.3.2 Stimulus/Response sequence

Stimulus: User requests to view savings history.
Response: The system send all the savings information till date to the user from database, which will be rendered on the web page.

Stimulus: User requests to view savings history of a particular week.
Response: The system send all the expenditure information of that week to the user from database, which will be rendered on the web page.

Stimulus: User requests to view savings till the current week.
Response: System will send the overall savings till the current week.

3.2.3.3 Functional Requirements

FR ID	FR Name	Description
FR-S1	View Savings History till the week	The application will allow user to check savings history of the user till that week.
FR-S2	View Savings History on a given week	The application will allow user to check savings history of the user on the given week.
FR-S3	View Savings till the current week	The application will show the user savings till the current week.

3.2.3 ToDo List

3.2.3.1 Introduction/ Purpose of feature

In this feature the user can add the things he needs to do that day. After that day, the list expires. It has a low priority.

3.2.3.2 Stimulus/Response sequence

Stimulus: User requests to add a new ToDo list item.
Response: The application send a form consisting of what activity and time. 5 mins before the remainder time an email notification will be sent to the user.

Stimulus: User requests to delete a ToDo item.
Response: The system will delete the item from database. No notification about that will be sent.

3.2.3.3 Functional Requirements

FR ID	FR Name	Description
FR-D1	Add item to ToDo list	The application will allow the to add new ToDo item to the user’s ToDo list.
FR-D2	Email Notification	The application will send a email notification to the user 5 mins before the ToDo list item as a reminder.

FR-D3	Delete a ToDo list item	The application will delete the item from the list. Remainder using FR-D2 will not be sent.

3.2.4 ToBuy List

3.2.4.1 Introduction/ Purpose of feature

In this feature the user can add the things he wants to buy but are costlier than INR 5000. After his savings exceed that amount he can buy. It has a low priority.

3.2.4.2 Stimulus/Response sequence

Stimulus: User requests to add a new ToBuy list item.
Response: The application send a form consisting of what the investment is and cost. When user savings exceed the cost of the item a notification will be sent to the user and Buy button will be enabled.

Stimulus: User requests to buy a ToBuy item.
Response: The system will delete the item from database. The item will be added to expenditure list and savings money will be deducted.

Stimulus: User requests to delete a ToBuy item.
Response: The system will delete the item from database. No notification about that will be sent.

3.2.4.3 Functional Requirements

FR ID	**FR Name**	**Description**
FR-B1	Add item to ToBuy list	The application will allow the user to add new ToBuy item to the user's ToBuy list.
FR-B2	Email Notification	The application will send a email notification when the user is eligible to buy the item.
FR-B3	Buy item	The application will delete the item from the Database. The item will be added to expenditure list and savings money will be deducted.

FR-B4	Delete item	The application will delete the item from the list. Remainder using FR-B2 will not be sent.

3.2.5 Feedback System

3.2.5.1 Introduction/ Purpose of feature

In this feature the user will provide feedback to the admin and also rate the application. It has a high priority.

3.2.5.2 Stimulus/Response sequence

Stimulus: User requests to send feedback and ratings.
Response: The application send a form consisting of ratings and feedback. This goes to admin.

3.2.5.3 Functional Requirements

FR ID	**FR Name**	**Description**
FR-F1	Feedback and rating	The application will allow the user to send ratings and feedback to the admin.

3.2.6 Profile Management

3.2.6.1 Introduction/ Purpose of feature

In this feature the user will provide description about the user and a profile picture. Low priority.

3.2.6.2 Stimulus/Response sequence

Stimulus: User requests to update user info and profile picture.
Response: The application send a form so that user can add a new description about the user and also update profile picture.

3.2.6.3 Functional Requirements

FR ID	FR Name	Description
FR-P1	Personal Information update	The application will allow the user to update user information and profile picture.

3.2.6 Admin Features

3.2.6.1 Introduction/ Purpose of feature

In this feature the admin user will be allowed to view the feedbacks and delete fake users. High priority.

3.2.6.2 Stimulus/Response sequence

Stimulus: User requests to view feedbacks.
Response: The application send all the feedbacks received from users and render it on a page.

Stimulus: User delete fake users.
Response: The application delete all the users the admin specifies.

3.2.6.3 Functional Requirements

FR ID	FR Name	Description
FR-A1	View user feedback	The application will show all the feedbacks received from users.
FR-A2	Remove fake users	The admin will provide fake users id and all the information regarding the fake user will be deleted.

3.3 Performance Requirements

Static

- Data in database should be updated in 1s.
- UI must load within 3s.
- At least 10000 users can use the application at the same time.
- Login validation will be done within 2s.

Dynamic

- The software should be able to handle at least 1000 requests / second.
- Average response time for each response should be 0.5s with worst case being 2s which can occur only 5% of the time.

3.4 Logical Database Requirements

Database Format

- As the database will be using MongoDB, so the data will be in JSON format in the database.
- The database will have different collections. User collection, Expenditure, Savings, ToDO, ToBuy and feedback collections. User_id from the User will be the primary key.

Accessibility and Security

- The database can be accessed / read / added by the logged in user through transaction, savings, ToDo and ToBuy routes.
- The database as a whole will be accessible to admin. The admin can read/ modify/ add/ delete any document in the collection of any user.
- To access the database every user will be authenticated to the database through Node js. When request is made to the database for connection, the request will carry authentication details to login to database. Only then database can be used to perform any actions.

3.5 Design Constraints

3.5.1 Technology Constraints

The software which is an webapp will be implemented in javascript both frontend and backend. It will use HTML and CSS for styling. Node js will be used for implementing backend.

3.5.2 Interface Constraints

As the software is a web application the software will work as expected in majority of the popular browsers Google Chrome and Mozilla Firefox. Due to cross browser incompatibility there might be some features which this may not work as expected in lesser used browsers.

3.5.3 Storage Constraints

Since we will be using online database service provider Mlab for our mongoDB database our software will be restricted to 500mb data in DB.

3.6 Software System Attributes

3.6.1 Reliability

The system will not crash on invalid inputs in fields like item price, place of purchase. The user data will be stored on servers and there will always be a backup copy of all user data in case the system crashes. The backup data will be updated frequently.

3.6.2 Availability

The web application will be available to the user throughout the day and will work without any problems if there is a good internet connection. There is no limit to how long the user wants to use the web application.

3.6.3 Security

The software will use token based authentication. The whole database will be accessible to the admin. The user can access only their document present in database to modify the contents like budget, expenditure details, etc. When request is made to the database for connection, the request will carry authentication details to login to database. Only then database can be used to perform any actions. The user authentication token will be passed on to all the routes through authentication header. The user password will be hashed and stored. It will have a 256 bit encryption making it almost hack proof. JSON Web Tokens will be used.

3.6.4 Maintainability

The web application will be built using components/modules that are as independent to each other as possible so that the application is easily modifiable. If one part of the application crashes, the other parts should remain unaffected and should continue to function normally. The crashed part can be fixed and a patch can be provided to the application.

3.6.5 Portability

Since the Expenditure Management application is a web application, it can run on any device irrespective of its platform such as Android, Windows, MacOS, iOS, Linux. It can be accessed through any browser on any device as long as there is an internet connection. This will help the users to enter their details through many devices and sync all their details across all the devices and also access all their information from anywhere.

4.Appendixes

Appendix A: Glossary

- **Guest User:**

Any user who visits the web app.

- **Registered User:**

Users how have registered / subscribed the software and have logged in into the app.

- **Admin User:**

They are generally developers, maintainers and testers of the application, who have special rights.

- **Database:**

A database is a collection of information that is organized so that it can be easily accessed, managed and updated.

- **MongoDB:**

MongoDB is an open source database that uses a document-oriented data model.

- **Collection:**

In MongoDB a set of similar documents is called a collection.

- **Documents:**

Any Object of a specific structure that holds information is called a document. In mongoDB they are BSON objects.

- **BSON:**

BSON is a binary representation of JSON documents, though it contains more data types than JSON.

- **JSON:**

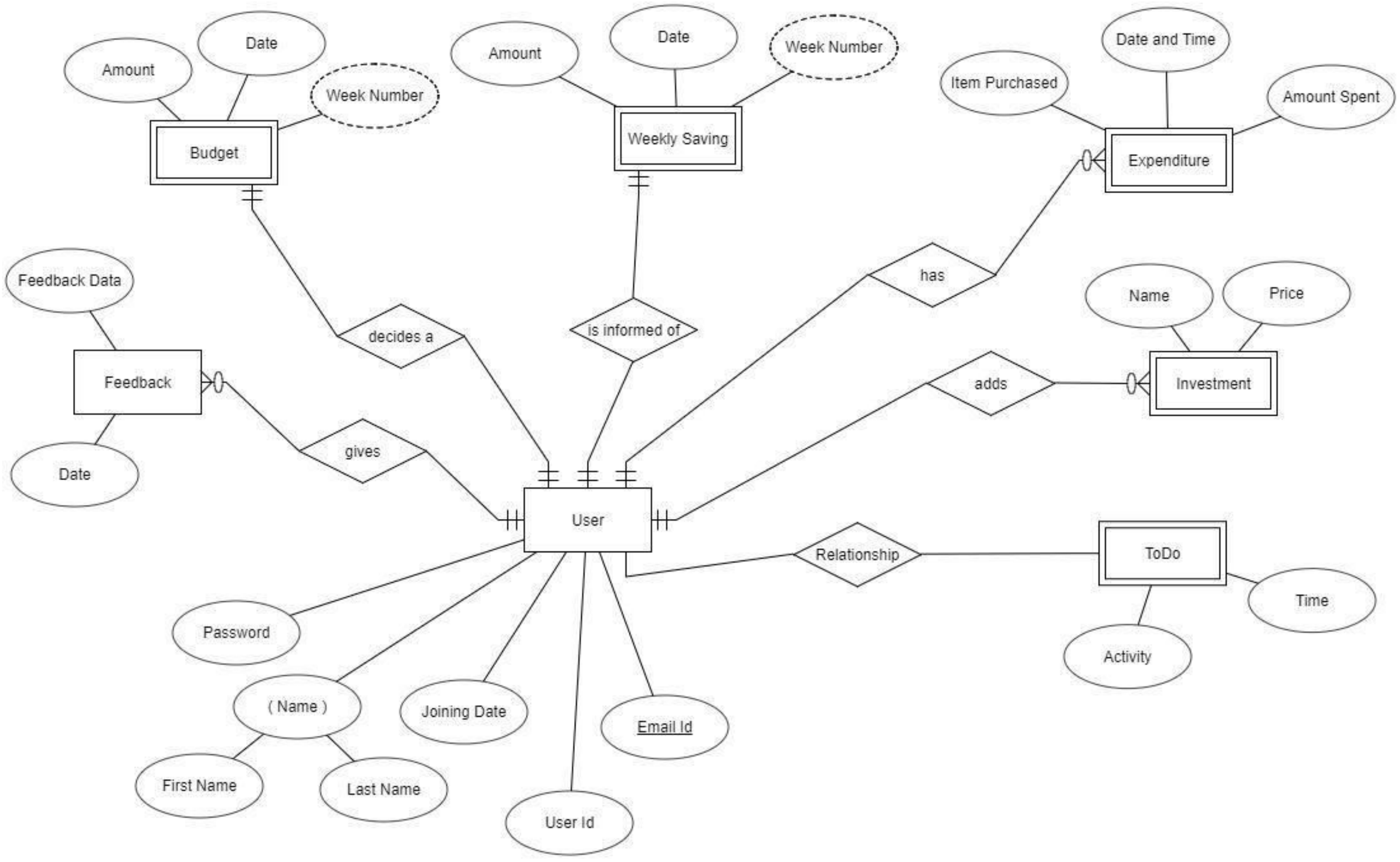
JavaScript Object Notation or JSON is an open-standard file format that uses human-readable text to transmit data objects consisting of attribute–value pairs and array data types.

- **Node Js:**

JavaScript run-time environment for executing JavaScript code server-side.

Appendix B: Analysis Models

- **ER Diagram**



Appendix C: To Be Determined List

Expenditure Management Application intends to ship all the features in its initial release itself. However there is some functionality that could be released at a future date if needed.

- The primary functionality that could be held back for future release is the Email Notification feature. The ability to send out emails to users when a transaction is made or notifications of to buy. It is however a feature that the system can be used without and still function in its original scope and requirement.

- If it is required that the Email Notification feature is postponed, it is highly recommended that this feature be developed and added on to the system as quickly as physically possible. As it is highly important to the ease of use by the customer and a big selling point which will help keep the system ranked high.

- Debit card information gathering of users and automatically adding card usage on expenditure directly to database.

- Using OAuth and OAuth2.0 also along with token based authentication system allowing users to login through their Google / Facebook accounts.