# SOFTWARE REQUIREMENT SPECIFICATION

*INFO 2413: System Development Project (A10)*

**FAM BUDGETLY**

An Online Budget Management System – Web Application

*Submitted By*
*Ali Manghat – 100356935*
*Akshit Thakur – 100394694*
*Ramandeep Singh – 100394782*
*Gurpreet Singh – 100400870*
*Sunpreet Kaur – 100410018*

OCTOBER 3, 2022

## Table of Contents

# 1. Introduction

## 1.1 Purpose

The purpose of this document is to develop a basic framework for an online budget system that will be represented through a web application. This document helps establish a baseline for things like functional requirements, non-functional requirements, databases, etc. The focus here is to make a user-friendly Online Budget Management System for a family's needs. A short description has been written for each functional requirement and non-functional requirement. Database logic has also been explained at a fundamental level. The intended audience for this SRS is Dr. Jendy Wu.

## 1.2 Scope

The software product to be produced is a Simple Online Budget Management Web Application named "Fam Budgetly". The main purpose of this system will be to have a simple easy-to-use website that will allow its users to manage their budgets more wisely. The system will be designed for a single family. Each member of the family will have their own personal user account. Through this system keeping track of spending on a month-by-month basis will be made easy and more visual. This will result in having wiser spending and more control over what the money is being spent on.

## 1.3 Definitions Acronyms and Abbreviations

- HTML - Hypertext Markup Language
- Cloud - Virtual spaces exist on the internet, called the cloud. Files, applications, software, and servers can be stored in this storage area.
- Node.js - An open-source, cross-platform JavaScript runtime environment, node.js is designed to build network applications that run outside of web browsers and run JavaScript code.
- HTML5 - It is a markup language to structure and present content on the World Wide Web.
- PostgreSQL - Postgres is a free, open-source, and extensible SQL-compliant relational database management system.
- CSV file - Comma-separated values files are text files with values separated by commas. In this file, each line represents a data record. Comma-separated fields make up each record.

- Hash code - In hashing, keys or strings of characters are transformed into new values. As a result, shorter, fixed-length values or keys are usually used to represent the original string and make it easier to locate or use

## 1.4 References

- Andrew Burak – "Your 2022 Guide to Writing a Software Requirements Specification (SRS) Document"
  https://relevant.software/blog/software-requirements-specification-srs-document/

- "Node.js Introduction"
  https://www.w3schools.com/nodejs/nodejs_intro.asp

- Nishadha, 28 September 2022 – "Use Case Diagram Tutorial (Guide with Examples )"
  https://creately.com/blog/diagrams/use-case-diagram-tutorial/

## 1.5 Overview

The SRS contains functional requirements, non-functional requirements, and logical database requirements. Functional requirements are requirements that will enable the users to complete all the required tasks. Non-functional requirements are measurable traits that will be things such as usability, security, reliability, etc. Database logic being the entities of the database such as budget/spending and more.

# 2. Overall Description

## 2.1 Product Perspective

The "Fam Budgetly" is a platform which is meant to serve a family to manage their monthly budget. Our goal is to make a web application which will make it very easy to manage household expenses for each member of the family.

### System Interface

Windows Server 2016, a cloud-ready Operating System will be used to host the Website. HTML will be used to create web pages for the user interface and node.js will be used to connect the frontend with the backend database. PostgreSQL will be used to create the database.

### User Interface

The user interface for the web application will be a simple and natural interface. The navigation between the pages will be user-friendly and easy to manage. Different tasks such as Starting a budget for a month, updating spending, and generating reports will be easier to handle and work with.

### Software Interface

On the user side, a web browser should be capable of supporting JavaScript and HTML5 like Firefox or Chrome. On the server side, a database will be hosted through Windows Server 2016 using PostgreSQL.

## 2.2 Product Functions

The web application has various functions to make it user friendly and easy to use. Some major functions are as follows:

a) Registration: Regular users can register themselves on the web application.

b) Login: All users can log into their account.

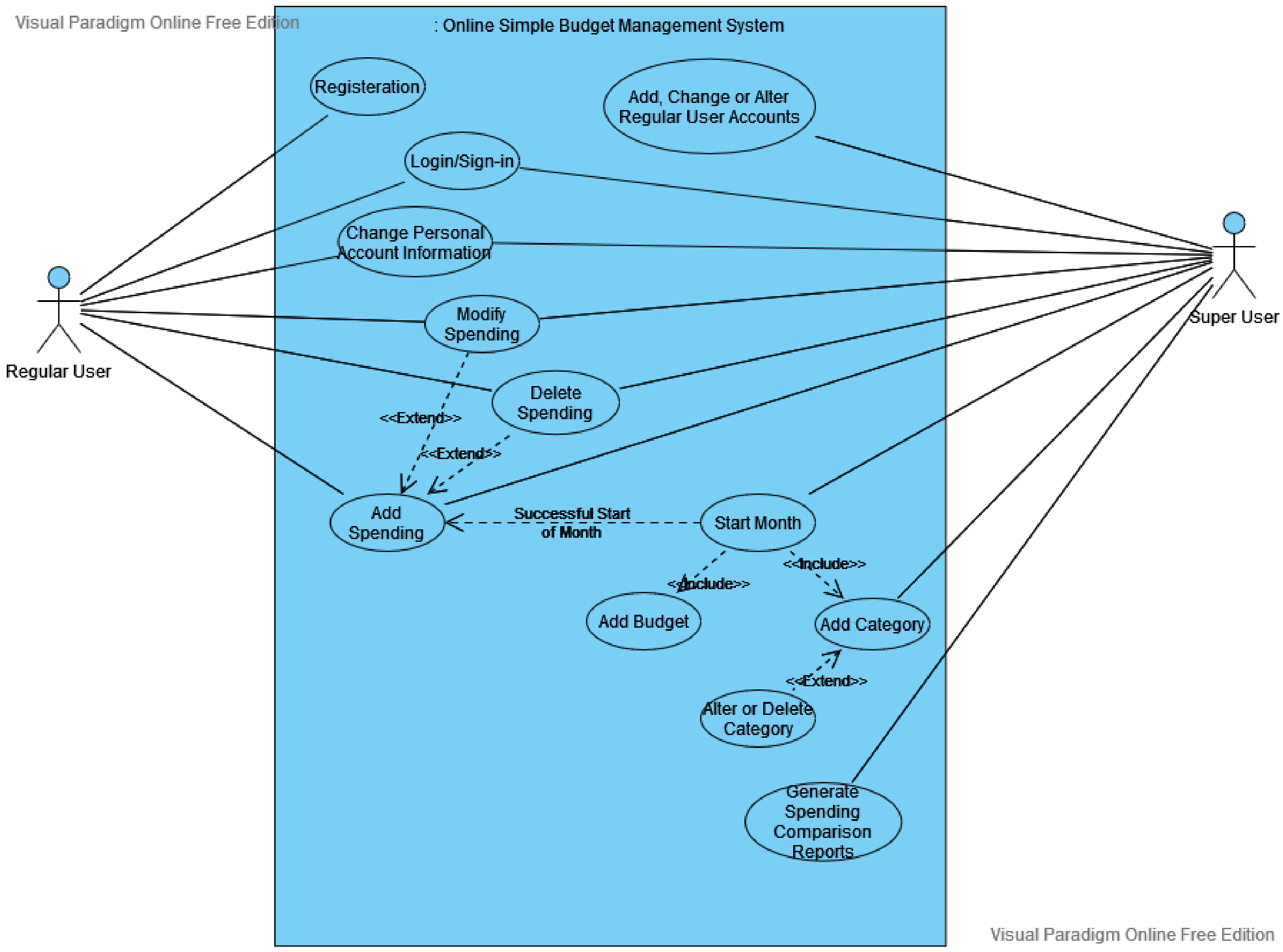
c) Change personal Information: All users can change their personal information like name, email, nickname etc.

d) Add, change or Alter Accounts: Super users have privileges to add, change or alter regular user's account data.

e) Start Month: Super Users can start the month by adding categories and specifying budget.

f) Add Spendings: Once the month starts, all users can start adding their spending.

g) Generate Reports: Super Users can generate spending reports after the month ends.


Visual Paradigm Online Free Edition
: Online Simple Budget Management System
Registeration
Add, Change or Alter Regular User Accounts
Login/Sign-in
Change Personal Account Information
Modify Spending
Delete Spending
<<Extend>>
<<Extend>>
Add Spending
Successful Start of Month
Start Month
<<Include>>
<<Include>>
Add Budget
Add Category
<<Extend>>
Alter or Delete Category
Generate Spending Comparison Reports
Regular User
Super User
Visual Paradigm Online Free Edition

# 3. Specific Requirements

## 3.1 Functional requirements

### 3.1.1 Registration

The website will have many functionalities for its visitors. The regular users will be able to sign up themselves if they provide their name, email, phone number, username, and password. This data will be used to create a regular user and added to the database. After the registration, the user can access the web application using his/her user credential.

### 3.1.2 Sign-In/login

Users must be able to log in either as regular or super users based on their account type, using their username and password. The user credentials will be used to search the account in the database, and if and only if the account information is present in the database, the user will be able to log in. During the login process, the system should pick up the account type and grant the user those permissions based on whether they are a regular user or a super user.

### 3.1.3 Ability to change personal information

All users should have the privilege to alter their account name, email, and phone number. After logging into their account, the users can update their personal information which will update their information in the database with new data.

### 3.1.4 Ability to add, alter or remove Regular Users

Super users should have the privilege to alter their account name, email, and phone number. Along with the ability to add, alter, and delete regular user accounts. Any change will replace the old information with the new data in the database.

### 3.1.5 Start of Month

On every first of the month, the website should enable super users to add the budget of the new month for each category that they have created such as food, clothing, rent, transportation cost, etc. The categories can be added, removed or altered during the process of starting a month. Once the month has been started, all the users will be able to add spending.

### 3.1.6 Alter Categories

The website will also enable functionality for super users to add, delete, and alter each category if there is a change in their budget plan. If there is any change done, all the users will be able to see the changed categories once they log into their account because the overall information will also be updated in the backend (database). This functionality is important because if a super user forgets to add a category during the start of the month, they can add it later.

### 3.1.7 Notification (Start of Month)

If a super-user forgets to add their budget plan for the month, the system should send out an email on the 2nd day of the month, reminding the super-user to do so. An email alert should continue to get sent out once daily until the super user adds their budget plan for that month.

### 3.1.8 Spending

Both super users and regular users should be able to input, alter and delete their spending for each category during the current active month. The system should make sure that the super users have started the budgeting for that month before allowing input/changes to the spending. Any changes in the spending during an active month will update the data in the backend (database) as this will ensure the integrity and availability of most recent updated data.

### 3.1.9 Altering/Deleting Spending Data (After Month's Completion)

When the month is passed, the system will not allow regular users to input, alter, or delete any spending data. Super users should be able to add or alter spending data, but they will not be able to delete data.

### 3.1.10 Notification Email (Exceeding Budget Warning)

In a situation where spending reaches 80% of the budget for a specific category, the system should send an alert email to all users regardless of their account type.

### 3.1.11 Generating Reports

The super user account type should be able to generate a monthly report for the following:
1) Comparing the monthly budget and the actual spending in total.
2) Comparing the actual spending with every category's budget.
3) Analysing the current month in comparison with the previous month
4) The top 3 categories for which the spending has exceeded 90% or more of the budget in the last 3 months.
They can export/download these reports in the form of csv files.

## 3.2 Non-Functional Requirements

### 3.2.1 User-Friendly

Our system will have the following specifications under a user-friendly interface. After clicking on any link, our system will open it in a new tab rather than in the same tab, so this will help the user to easily navigate around the website. Our Web Application will use a simple and readable font which can be read by users without difficulty for long hours along with good colour combinations. Our system will allow the user to search the application easily like profile settings (name, email address, phone number), login and logout links. To measure if it is User-Friendly or not, we can do a focus group of let's say 10 people and if 8 or more than 8 people find our web application User-Friendly, then we can consider that it is user-friendly.

### 3.2.2 Security

Nowadays, Developers consider Security as the utmost requirement while developing any application, so our application will also not compromise with any security concerns and vulnerabilities. We will provide maintenance and updates on a regular basis to check for any concerns. Our system will prevent any unauthorised access to read, write, edit, or delete any information. The Password set by various users will be stored as a hash code in the Database. To measure the security, we can use some third-party applications over the internet to run various tests to check for vulnerabilities in our web application. We can consider our web application to be secure if 3 out of 5 applications have more than an 80% success secure rate while testing our web application.

### 3.2.3 Reliability

In today's world, only those systems are successful that are quite reliable. Our web application will provide 70% error tolerance. For example, if any update made by the user does not go to next stage, then it will come back to the original state. To test it out, we can put in some incorrect values or incorrect functions and see how the application performs. For it to pass, it should bring the page back to its original state for 7 out of 10 errors at least. The system will be easy to update. It will not be hard coded so it can be easily modified when needed. With time, the application may demand more features to be added, and it will be easier to add them.

### 3.2.4 Performance

The website will not crash frequently because it will be able to handle 10 - 15 users at a time. The website will process 80% of the transactions in less than five seconds. To test this, we can try to do multiple functions from different accounts and let's say for 10 functions 8 should be completed within 5 seconds. Our system will also make sure that any work done by the user will be saved. This will ensure that no data is lost in case of their system shutdown.

## 3.3 Database Logic

The Database will store all the information about the regular users and super users, like their names and email address, phone numbers, usernames, and passwords (as hash-code). Any data that will be requested to be updated from the web application will be updated in the database. The database will also contain information about budget categories and spending. The node.js will be used to connect the front end with the back-end database which will help in viewing, altering, and updating data in the database by the user.