Software Requirements Specification

for

<Budget Buddy >

**Version 1.0 approved**

**Prepared by Ergita Curri , Adelajda Torollari**

**<Epoka University >**

**<16.05.2024 >**

**Table of Contents**

[1. Introduction 1](#_Toc166951211)

[1.1 Purpose 1](#_Toc166951212)

[1.2 Document Conventions 1](#_Toc166951213)

[1.3 Intended Audience and Reading Suggestions 1](#_Toc166951214)

[1.4 Product Scope 1](#_Toc166951215)

[1.5 References 1](#_Toc166951216)

[2. Overall Description 2](#_Toc166951217)

[2.1 Product Perspective 2](#_Toc166951218)

[2.2 Product Functions 2](#_Toc166951219)

[2.3 User Classes and Characteristics 2](#_Toc166951220)

[2.4 Operating Environment 2](#_Toc166951221)

[2.5 Design and Implementation Constraints 2](#_Toc166951222)

[2.6 User Documentation 2](#_Toc166951223)

[2.7 Assumptions and Dependencies 3](#_Toc166951224)

[3. External Interface Requirements 3](#_Toc166951225)

[3.1 User Interfaces 3](#_Toc166951226)

[3.2 Hardware Interfaces 3](#_Toc166951227)

[3.3 Software Interfaces 3](#_Toc166951228)

[3.4 Communications Interfaces 3](#_Toc166951229)

[4. System Features 4](#_Toc166951230)

[4.1 System Feature 1 4](#_Toc166951231)

[4.2 System Feature 2 (and so on) 4](#_Toc166951232)

[5. Other Nonfunctional Requirements 4](#_Toc166951233)

[5.1 Performance Requirements 4](#_Toc166951234)

[5.2 Safety Requirements 5](#_Toc166951235)

[5.3 Security Requirements 5](#_Toc166951236)

[5.4 Software Quality Attributes 5](#_Toc166951237)

[5.5 Business Rules 5](#_Toc166951238)

[6. Other Requirements 5](#_Toc166951239)

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose

The following document serves as the Software Requirements Specification (SRS) for the initial release of the Budget Buddy web application, version 1. 0 This document will detail the necessary requirements for the development and functionality of the application. Budget Buddy is a multifaceted instrument developed for individuals to efficiently manage and oversee their personal finances. The following Software Requirements Specification (SRS) document offers an in-depth delineation of the functionalities, interfaces, and constraints pertaining to the Budget Buddy application. Its aim is to establish a common comprehension among all stakeholders engaged in the software's development and implementation process. This document comprehensively addresses the entirety of the system, incorporating user requirements, functional and non-functional specifications, and external interface requirements. The purpose of this Software Requirements Specification (SRS) is to act as a comprehensive reference for developers, project managers, testers, and end-users at all stages of the software development process. Its goal is to ensure that the final product aligns with the specified requirements and fulfills the expectations of the stakeholders.

## Document Conventions

The Software Requirements Specification (SRS) adheres to various industry standards and conventions to uphold the principles of clarity and consistency.

**Requirements Identification**:involves assigning a unique sequence number to each requirement for the purpose of facilitating easy reference.

**Priority Levels**:is determined by assigning them a priority level (High, Medium, Low) as a means of indicating their relative importance.

**Formatting**:utilizes high-level requirements in bold for emphasis, while presenting detailed requirements in plain text.

**Terminology**: specific terminology, the definitions of which can be found in the glossary (Appendix A).

**Placeholder Notation**: "TBD" (To Be Determined) is employed to signify sections within a document or presentation where information is currently unavailable and will be provided at a later time.

**Cross-Referencing**:is utilized within the Software Requirements Specification (SRS) to provide easily identifiable references to other sections or documents, thereby aiding in efficient navigation.

## Intended Audience and Reading Suggestions

This document is designed to cater to a wide range of stakeholders, all of whom have unique requirements and areas of interest.

**Developers:** In order to comprehend the specific prerequisites essential for the implementation of the software. It is recommended that developers prioritize their attention towards Sections 3 and 4 in order to address both the functional and non-functional requirements.

**Project Managers:** In order to effectively manage the development process, it is essential to meticulously plan and supervise the progression of the project to ensure that critical milestones and deliverables are achieved. Project managers are encouraged to thoroughly review the entire document, commencing with the Introduction and Overall Description, and allocate particular attention to Section 5 in order to identify any project constraints and dependencies.

**Testers:** The task at hand involves the creation and implementation of test cases in accordance with the prescribed requirements. Testers are advised to direct their attention towards the functional requirements outlined in Section 4, as well as the non-functional requirements documented in Section 5.

**End Users:** In order to comprehend the functionality and potential of the software, it is essential to conduct an analysis. Initially, it is recommended that the end users commence with the Introduction and Overall Description in order to obtain a comprehensive understanding of the system. Following this, they should direct their attention to the specific system features outlined in Section 4.

**Documentation Writers:** In order to produce user manuals, assistance guides, and various forms of documentation materials, adequate preparation is necessary. It is imperative for them to thoroughly reassess the entirety of the Software Requirements Specification (SRS), placing specific emphasis on both the user interface and functional requirements sections.

It is recommended that readers commence with the Introduction in order to obtain a comprehensive understanding of the document's intended purpose and extent. Subsequently, one should advance to the Overall Description in order to gain a comprehensive grasp of the product. The following sections provide an in-depth description of the specific requirements and system features, while the non-functional requirements and constraints are addressed in the latter part of this document.

## Product Scope

Budget Buddy is a web-based application that has been developed to aid individuals in effectively managing their personal finances. The principal aims of Budget Buddy are to streamline the process of budgeting, offer comprehensive expense monitoring, and aid in the establishment and oversight of financial objectives. The software has been designed with the goal of enabling individuals to make well-informed financial decisions, enhance their financial well-being, and attain their financial goals.

Key benefits of Budget Buddy include:

**Simplified Budgeting:** It is possible for individuals to effortlessly establish and oversee financial budgets, distribute funds to different categories, and monitor their expenditures in relation to these budgets in real-time.

**Detailed Expense Tracking:** Users have the option to either manually log their expenses or import them from their bank statements. Furthermore, they have the ability to categorize their expenses and generate comprehensive reports in order to analyze their spending patterns.

**Financial Goal Setting:** Individuals have the capability to establish financial objectives, such as earmarking funds for a holiday or settling outstanding debt, and track their advancement towards these objectives through consistent updates and suggestions.

**User-Friendly Interface:** The design of the application prioritizes usability, aiming to ensure that individuals with diverse levels of technical proficiency can effectively navigate and utilize the software.

**Integration with Financial Institutions:** Budget Buddy facilitates integration with a variety of banks and financial institutions, enabling users to import transaction data with ease and efficiency.

The Budget Buddy product is consistent with corporate objectives and business tactics as it offers a comprehensive financial management solution suitable for broad marketing, ultimately contributing to increased customer interaction and contentment. The objective of the application is to serve as a fundamental tool for the management of personal finances, thereby enhancing the financial well-being of users and facilitating long-term financial planning.

## References

Each reference includes sufficient information for the reader to access a copy.

Title: IEEE Std 830-1998

Author: IEEE

Version Number: 1998

Date: 1998

Source: IEEE Standards Association

Budget Buddy User Interface Style Guide

Title: Budget Buddy UI Style Guide

Author: Budget Buddy Design Team

Version Number: 1.0

Date: March 2024

Source: Internal Document

Budget Buddy Vision and Scope Document

Title: Budget Buddy Vision and Scope

Author: Ergita Curri, Adelajda Torollari

Version Number: 1.2

Date: February 2024

Source: Internal Document

General Data Protection Regulation (GDPR)

Title: Regulation (EU) 2016/679

Author: European Union

Version Number: N/A

Date: April 27, 2016

Source: GDPR Official Text

Web Content Accessibility Guidelines (WCAG) 2.1

Title: Web Content Accessibility Guidelines (WCAG) 2.1

Author: World Wide Web Consortium (W3C)

Version Number: 2.1

Date: June 5, 2018

Source: WCAG 2.1

California Consumer Privacy Act (CCPA)

Title: California Consumer Privacy Act (CCPA)

Author: State of California

Version Number: N/A

Date: June 28, 2018

Source: CCPA Text

PostgreSQL Documentation

Title: PostgreSQL 13 Documentation

Author: PostgreSQL Global Development Group

Version Number: 13

Date: October 2020

Source: PostgreSQL Documentation

MySQL Documentation

Title: MySQL 8.0 Reference Manual

Author: Oracle Corporation

Version Number: 8.0

Date: January 2021

Source: MySQL Documentation

These references provide the necessary background and guidelines for the development and maintenance of the "Budget Buddy" web application.

# Overall Description

## Product Perspective

The "Budget Buddy" application represents a recently developed, self-contained web application that has emerged as a result of a software engineering class project at Epoka University. This application does not serve as a subsequent iteration within a product line or as a substitute for current systems. Instead, its purpose is to fulfill the distinct and specific requirement for a user-accessible personal finance management tool. The principal objective is to streamline the process of budget management, tracking of expenses, and establishment of financial goals for individuals.

Major Components and Subsystem Interconnections

The system architecture of Budget Buddy can be divided into several major components:

***User Interface (UI)***

Description: The frontend component that interacts with users.

Technologies: HTML, CSS, JavaScript, React.js.

Responsibilities: Displaying information, capturing user input, and providing an intuitive and responsive user experience.

***Backend Server***

Description: The server-side component that processes user requests.

Technologies: Node.js, Express.js.

Responsibilities: Handling business logic, managing user sessions, and interfacing with the database.

***Database***

Description: The data storage component.

Technologies: MongoDB.

Responsibilities: Storing and retrieving user data, budgets, expenses, and financial goals.

***API Gateway***

Description: The interface through which the frontend communicates with the backend.

Technologies: RESTful API.

Responsibilities: Facilitating communication between the frontend and backend, ensuring data security and integrity.

***Interfaces and External Systems***

Budget Buddy will interface with external systems primarily through third-party APIs to enhance its functionality. Some potential external interfaces include:

***Banking APIs***

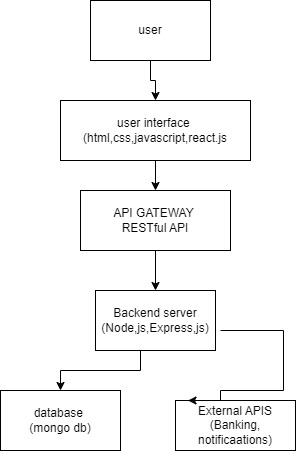
Purpose: To allow users to import their transaction data directly from their bank accounts.

Integration: Secure API calls with proper authentication and authorization mechanisms.

***Notification Services***

Purpose: To send users notifications about their budget status, upcoming bills, or financial goals.

Integration: Services like Twilio or Firebase Cloud Messaging (FCM) can be used.



## Product Functions

Budget Buddy will provide a comprehensive set of functions designed to help users manage their personal finances efficiently. These functions include:

***User Account Management:***

Account Creation: Users can register by providing necessary details and creating a profile.

Profile Management: Users can update personal information, change passwords, and manage security settings.

Account Deletion: Users can permanently delete their accounts and all associated data.

***Budget Creation and Management:***

Budget Setup: Users can create budgets by specifying income, expenses, and savings goals.

Budget Categories: Users can categorize their budgets (e.g., housing, food, transportation) for detailed tracking.

Budget Adjustments: Users can modify budgets as their financial situation changes.

***Expense Tracking:***

Expense Logging: Users can log daily expenses manually or import transactions from linked bank accounts.

Expense Categorization: Users can categorize expenses to monitor spending patterns.

Expense Analysis: Users can view summaries and detailed reports of their expenses over various periods.

***Financial Goal Setting:***

Goal Creation: Users can set financial goals (e.g., saving for a vacation, paying off debt) with target amounts and deadlines.

Progress Tracking: Users can monitor their progress towards achieving their goals.

Goal Adjustment: Users can update goals based on changes in their financial situation.

***Reports and Analytics:***

Spending Reports: Users can generate reports that detail their spending habits and identify areas for improvement.

Budget Reports: Users can view reports on how well they are adhering to their budgets.

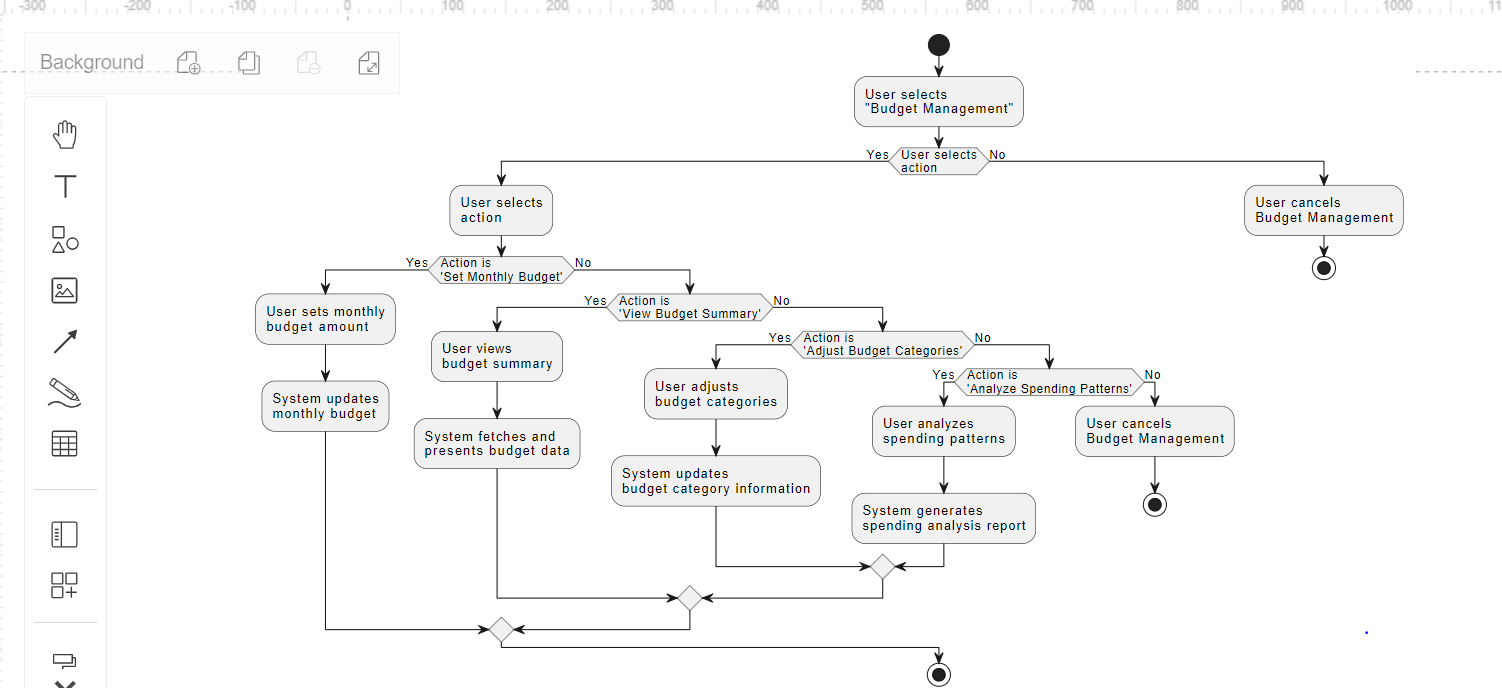
Financial Trends: Users can analyze financial trends over time with graphical representations and charts.

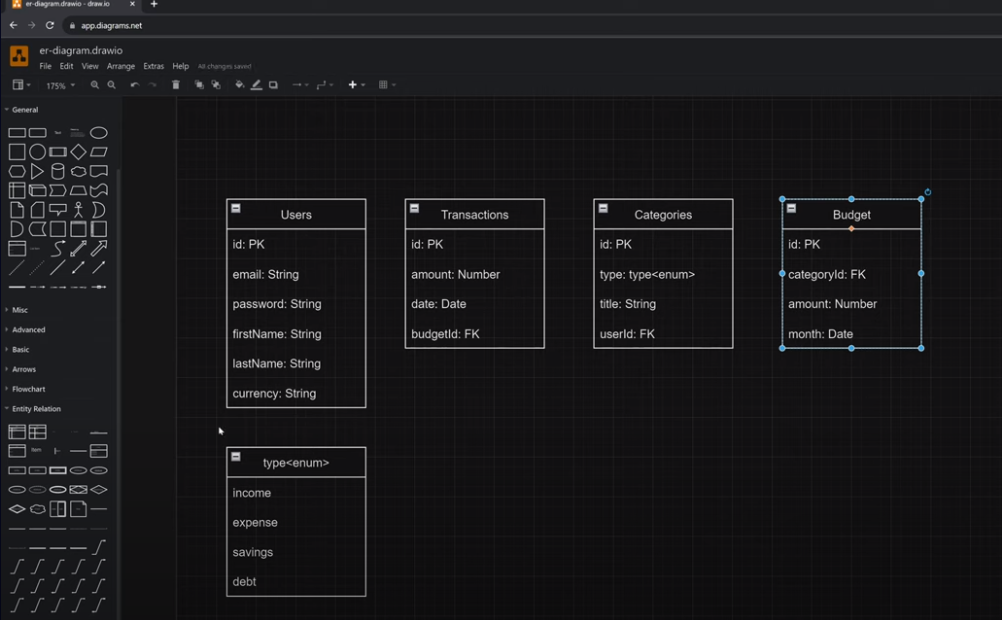
***Notifications and Alerts:***

Budget Alerts: Users receive alerts when they approach or exceed their budget limits.

Bill Reminders: Users receive notifications for upcoming bills and due dates.

Goal Progress Alerts: Users receive updates on their progress towards financial goals.





## User Classes and Characteristics

***1. General Users***

Characteristics: Individuals managing personal finances.

Expertise: Basic computer skills.

Frequency of Use: Daily or weekly.

Functions Used: Budget creation, expense tracking, goal setting, and viewing reports.

***2. Power Users***

Characteristics: Advanced financial management skills.

Expertise: High.

Frequency of Use: Daily.

Functions Used: All features, including advanced reports and banking integrations.

***3. Administrators***

Characteristics: System management and maintenance.

Expertise: High technical skills.

Frequency of Use: As needed.

Functions Used: User account management, system monitoring, and data integrity checks.

***4. Financial Advisors***

Characteristics: Professionals assisting clients with finances.

Expertise: Moderate to high.

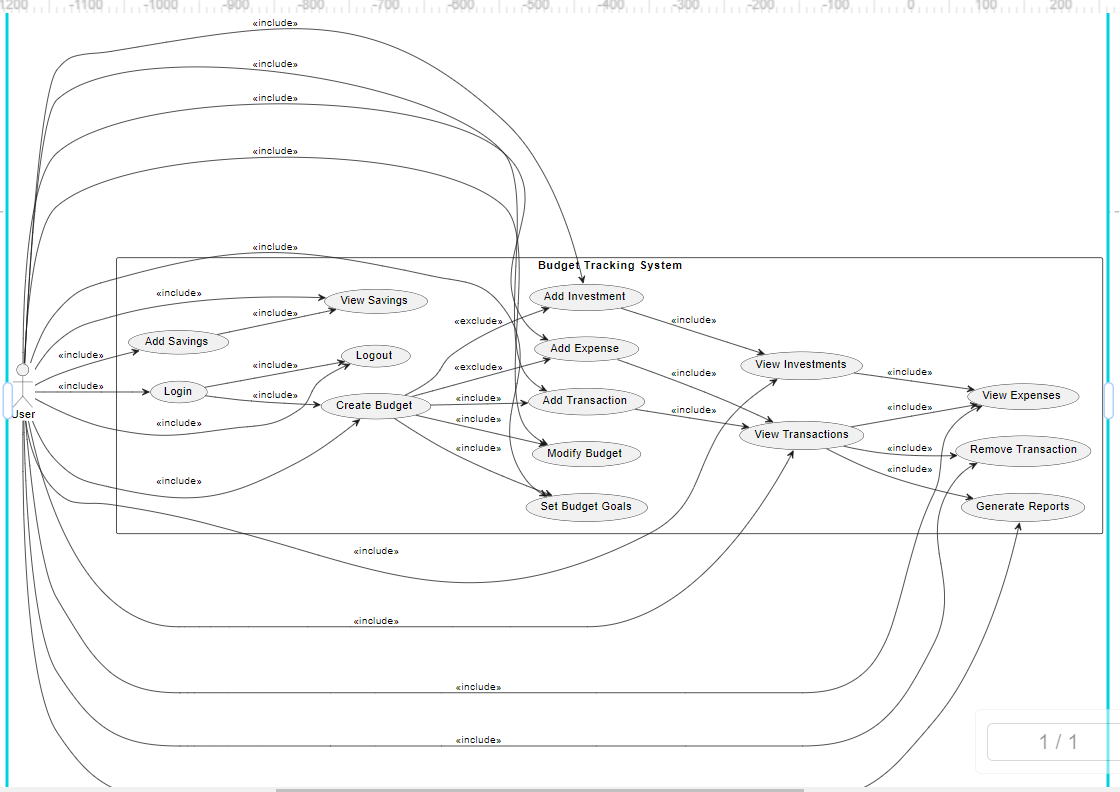
Frequency of Use: Weekly or as needed.

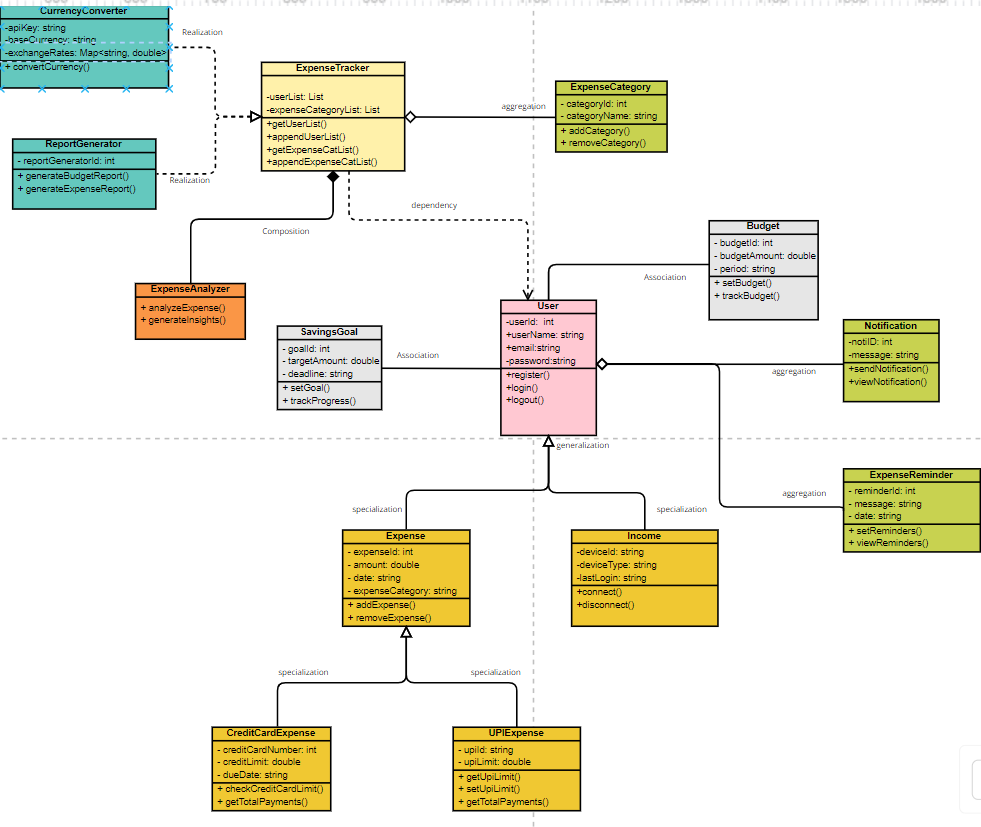
Functions Used: Viewing client reports, tracking financial goals, and providing advice.

***Importance:***

Most Important: General Users, Power Users.

Less Important: Administrators, Financial Advisors.





## Operating Environment

***Hardware Platform:***

The application will primarily run on standard web server hardware, including processors capable of handling concurrent user requests, sufficient memory to store application data and handle processing requirements, and reliable storage to store user data and application files. Client devices accessing the application will require basic hardware components, such as a web browser and an internet connection, to interact with the web application

Client Side: Any device with internet access (PCs, laptops, tablets, smartphones).

Server Side: Cloud-based or dedicated server infrastructure.

***Operating System:***

The web server hosting the application will typically run a popular operating system like Linux, Windows, or macOS. The specific version of the operating system will depend on the server hardware and the compatibility of the web application with the operating system.

Client Side: Any OS with a modern web browser (Windows, macOS, Linux, iOS, Android).

Server Side: Linux-based server OS (e.g., Ubuntu Server, CentOS).

***Web Browsers:***

The "Budget Buddy" web application will be accessible to users through popular web browsers such as Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari. The application will be developed using web standards (HTML, CSS, JavaScript) that are compatible with these browsers, ensuring a consistent user experience across different platforms.

Latest versions of Chrome, Firefox, Safari, and Edge.

***Software Components:***

The "Budget Buddy" application will be built using web technologies such as HTML, CSS, and JavaScript for the frontend, and possibly a backend framework like Django, Flask, or Node.js for the server-side logic. The application will also interact with a database management system (DBMS) to store and retrieve user data. The choice of DBMS will depend on factors such as scalability, data integrity, and compatibility with the chosen backend framework.

Frontend: HTML, CSS, JavaScript (React.js).

Backend: Node.js, Express.js.

Database: MongoDB.

Other: RESTful APIs for external integrations (e.g., banking APIs).

Coexisting Applications:

Compatible with common web applications and services (e.g., banking APIs, notification services).

## Design and Implementation Constraints

***Technological Constraints:*** The project is limited to web-based technologies, as it is a web application. This means the developers must use languages and frameworks suitable for web development, such as HTML, CSS, JavaScript, and possibly a backend framework like Django, Flask, or Node.js.

***Database Constraints:*** The application must use a database system for storing user data, transaction records, and other relevant information. The choice of database system should align with the project's requirements and the technologies being used.

***Security Considerations:*** Due to the nature of the application handling financial data, security is a critical constraint. The application must implement proper authentication, authorization, and data encryption techniques to ensure user data is protected.

***Compliance and Regulatory Policies:*** The application must comply with relevant regulations and standards for financial applications, such as GDPR, PCI DSS, or other regional regulations depending on the target audience.

***User Interface Constraints:*** The application must have a user-friendly interface that is easy to navigate and understand, considering the diverse range of users who may have varying levels of technical expertise.

***Performance Constraints:*** The application should be designed to handle a potentially large number of users and transactions, requiring efficient coding practices and possibly scalability features.

***Testing and Maintenance Constraints:*** The application should be developed with testing and maintenance in mind, ensuring that it can be easily maintained and updated in the future.

***Integration Constraints:*** The application may need to integrate with external services or APIs for features such as financial data aggregation, which could impose constraints on the technology stack and implementation approach.

## User Documentation

The user documentation for the "Budget Buddy" web application will include the following components:

***User Manual:*** A comprehensive guide that provides step-by-step instructions on how to use the application. The manual will cover basic features, such as account setup, expense tracking, budget creation, and goal setting, as well as advanced features, such as reporting and data analysis.

***On-line Help:*** Users are able to access context-sensitive help within the application. The assistance system is designed to offer guidance on particular tasks or features in accordance with the user's ongoing location or activities within the application.

***Tutorials:*** The interactive tutorials within the application serve to instruct users through the execution of commonplace tasks or workflows. The tutorials are intended to facilitate the swift orientation of new users and to offer experienced users techniques and strategies for optimizing their usage of the application.

***FAQs:*** A list of frequently asked questions and their answers, covering common issues or queries that users may encounter while using the application.

***Release Notes:*** The documentation outlines the new features, improvements, and bug fixes incorporated in every software release. This data will facilitate user comprehension of the alterations and enhancements made to the application.

***Known Issues:*** An inventory of recognized challenges or constraints within the application, in addition to any alternative strategies or resolutions. This data will assist individuals in identifying and resolving any issues that may arise during the utilization of the application.

***Delivery Formats:*** The delivery of user documentation will encompass digital formats including PDF files for the user manual, integration of online help within the application, and access to web-based tutorials through the application or a designated website. The documentation will conform to standard formats and conventions in order to ensure clarity and consistency.

The user documentation for the "Budget Buddy" application is intended to be user-friendly, easily accessible, and informative, with the goal of providing users with the necessary information to proficiently utilize the application for the management of their personal finances.

## Assumptions and Dependencies

***Assumptions:***

***Third-Party Libraries:*** The project operates under the assumption that third-party libraries and frameworks, including React for the frontend and Django for the backend, are readily available and compatible for integration in the development process. Modifications to these libraries or their accessibility may have implications for the project's timeline and implementation approach.

***Internet Connectivity:*** It is assumed by the application that users will possess a reliable and sustained internet connection in order to access and utilize the application. Inadequate or unstable internet connectivity can have a detrimental impact on both the user experience and the functionality of the application.

***Data Security:*** The project is predicated upon the implementation of robust security measures, including encryption and secure storage, to safeguard user data. The inability to adhere to such practices may result in security breaches and jeopardize user confidentiality.

***User Adoption:*** The success of the project depends on the belief that people will adopt and make use of the app to manage their personal finances. In order to promote widespread adoption, it may be necessary to implement user engagement strategies.

***Dependencies:***

***External APIs:*** The project depends on external APIs for features such as financial data aggregation and payment processing. Any changes to these APIs or their availability may affect the functionality of the application.

***Database Management System: The*** project relies on a database management system (DBMS) for storing and retrieving user data. Changes to the DBMS or its configuration may impact the application's performance and functionality.

***Web Hosting Services:*** The application depends on web hosting services to be accessible to users. The availability and reliability of these services may affect the application's uptime and performance.

***Browser Compatibility:*** The project depends on the compatibility of the application with popular web browsers. Changes to browser versions or standards may require updates to the application to ensure compatibility.

***Regulatory Compliance:*** The project depends on compliance with regulatory standards, such as GDPR, PCI DSS, and others, for handling user data. Failure to comply with these standards may result in legal issues and fines.

It is important to monitor these assumptions and dependencies throughout the project lifecycle and make adjustments as necessary to ensure the successful completion of the project.

# External Interface Requirements

## User Interfaces

The "Budget Buddy" web application will have the following user interface components:

***Dashboard:*** The main screen where users can view an overview of their financial status, including current balance, spending trends, and progress towards budget goals. The dashboard will also provide quick access to other features of the application.

***Expense Tracking:*** A feature that allows users to input and categorize their expenses. Users will be able to add new expenses, view a list of past expenses, and edit or delete existing expenses.

***Budget Creation:*** A feature that allows users to create budgets for different categories (e.g., groceries, rent, utilities). Users will be able to set budget limits, track their spending against these limits, and receive alerts when they are nearing or exceeding their budget.

***Goal Setting:*** A feature that allows users to set financial goals, such as saving for a vacation or paying off debt. Users will be able to track their progress towards these goals and receive notifications when they reach milestones.

***Reports:*** A feature that allows users to generate reports and visualizations of their financial data, such as spending trends over time or comparisons between budgeted and actual expenses.

***Settings:*** A section where users can customize their experience, such as setting up recurring expenses, changing their budget categories, or updating their profile information.

***User Interface Design:***

The user interface will follow modern web design principles, with a clean and intuitive layout that prioritizes ease of use. It will feature a responsive design to ensure optimal viewing and interaction across a variety of devices and screen sizes.

***Standard Buttons and Functions:***

***Help:*** A help button or link that provides users with access to a help center or FAQ section.

***Logout:*** A button that allows users to securely log out of their account.

***Notifications:*** A notification icon or area where users can see alerts and updates.

***Settings:*** A button or link that takes users to the settings section of the application.

***Error Message Display Standards:***

Error messages will be displayed prominently and in a clear, concise manner to inform users of any issues or errors. They will include suggestions for resolving the issue whenever possible.

***Keyboard Shortcuts:***

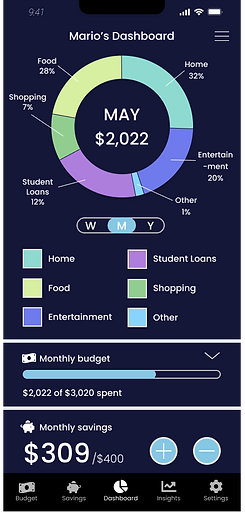
Keyboard shortcuts may be implemented for common actions to improve user efficiency, such as using the Tab key to navigate between fields or using Ctrl+S to save changes.

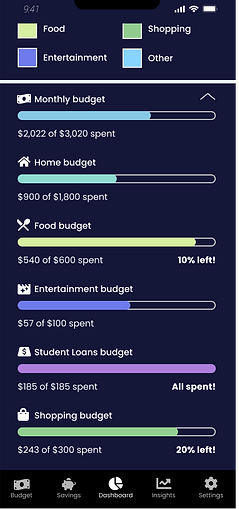
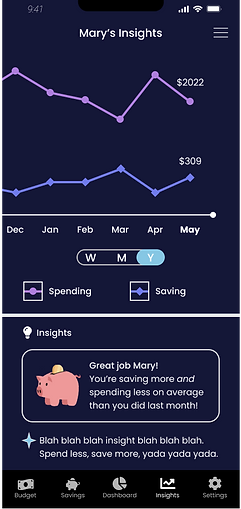
***GUI Standards:***

The user interface will adhere to standard web design guidelines and best practices to ensure a consistent and familiar experience for users. This includes using common UI elements, such as buttons, forms, and menus, in a way that is consistent with other web applications.

***Screen Layout Constraints:***

The screen layout will be designed to maximize usability and readability, with a focus on organizing information in a logical and intuitive manner. Constraints may include fixed or flexible layouts depending on the device and screen size.

## Hardware Interfaces

The "Budget Buddy" web application will interact with standard hardware components, including client devices like desktops, laptops, tablets, and smartphones. It will communicate over the internet using standard web protocols (HTTP/HTTPS) and may use WebSocket for real-time updates. The application's design will ensure compatibility and responsiveness across different devices and network environments.

## Software Interfaces

The "Budget Buddy" web application will interface with several software components, including databases, operating systems, libraries, and possibly integrated commercial components. Here is a summary of the software interfaces:

***Database:*** The application will interact with a database management system (DBMS) to store and retrieve user data. The specific DBMS and version will be determined during the development phase. Data items coming into the system include user registration information, expense records, budget details, and financial goals. Data going out includes reports, notifications, and summaries of financial data.

***Operating System:*** The application will run on a web server that supports the required operating system (e.g., Linux, Windows, macOS). The operating system version will depend on the hosting environment and compatibility requirements.

***Web Frameworks/Libraries***: The application may use web frameworks and libraries such as React for the frontend and Django or Flask for the backend. Specific versions will be determined based on compatibility and project requirements.

***External APIs:*** The application may integrate with external APIs for features such as financial data aggregation, payment processing, or notifications. The APIs and communication protocols will be specified during the development phase.

***Communication:*** The application will use standard web protocols (HTTP/HTTPS) for communication between client devices and the server. WebSocket may be used for real-time updates and notifications.

***Data Sharing:*** Data sharing across software components will be implemented using RESTful APIs for client-server communication and possibly message queues for asynchronous processing. Specific implementation details will be documented in the API specifications.

The detailed application programming interface (API) protocols and data sharing mechanisms will be documented in separate documents during the development phase. Implementation constraints, if any, will be specified based on the chosen technologies and requirements of the application.

## Communications Interfaces

The "Budget Buddy" web application will employ various communication interfaces to ensure smooth operation and user engagement:

***Web Browser Interaction:*** The application will be accessed by users through web browsers such as Google Chrome, Mozilla Firefox, or Safari. The application will employ HTTP/HTTPS protocols to facilitate the transmission of data from the client (browser) to the server, thereby ensuring a secure and efficient mode of communication.

***Email Notifications:*** The application has the capability to transmit notifications and alerts to individuals through the use of electronic mail. The system will establish communication with email servers through the usage of protocols, including SMTP for outbound email transmission and IMAP or POP3 for inbound email retrieval, thereby ensuring that users are consistently apprised of their financial transactions.

***Data Formatting:*** In order to enable smooth and efficient data interchange, the application will employ JSON (JavaScript Object Notation) or XML (Extensible Markup Language) for data formatting. These formats enjoy widespread support and facilitate straightforward parsing and interpretation of data.

***Communication Security:*** In order to safeguard sensitive information, the application will employ HTTPS (HTTP Secure) to encrypt data exchanged between the client and the server. The encryption employed guarantees the confidentiality and security of user data, encompassing financial and personal information.

***Data Transfer Rates:*** The velocity of data transfer will be contingent on elements including network celerity and server efficacy. The software will be tailored to optimize data transfer times with the goal of providing users with a highly responsive and efficient experience.

***Synchronization Mechanisms***: The proposed application will employ both synchronous and asynchronous communication mechanisms in order to ensure the integrity of data and enhance responsiveness. Synchronous requests are designed to facilitate immediate user interactions, whereas asynchronous requests are intended for handling background tasks and real-time updates in order to deliver a smooth and uninterrupted user experience.

Overall, these communication interfaces will ensure that the "Budget Buddy" application operates smoothly and efficiently, providing users with a secure and user-friendly platform for managing their finances.

# System Features

The system features of the "Budget Buddy" web application can be organized based on the major services provided by the product. These features include:

***User Registration and Authentication:***

Allow users to create accounts and log in securely.

Verify user credentials and manage authentication processes.

***Dashboard:***

Display a summary of the user's financial status.

Provide visualizations of spending trends, budget progress, and goal achievements.

***Expense Tracking:***

Allow users to add, edit, and delete expenses.

Categorize expenses and track them over time.

***Budget Management:***

Enable users to create, edit, and delete budgets for different categories.

Set budget limits and receive alerts for exceeding them.

***Goal Setting:***

Allow users to set financial goals (e.g., saving for a vacation, paying off debt).

Track progress towards goals and receive notifications for milestones.

***Reporting and Analysis:***

Generate reports and visualizations of financial data.

Provide insights into spending patterns and financial habits.

***Settings and Customization:***

Allow users to customize their profile settings.

Set up recurring expenses and other preferences.

***Notifications and Alerts:***

Send notifications for upcoming bills, budget alerts, and goal achievements.

Allow users to manage notification preferences.

***Data Security and Privacy:***

Ensure the security and privacy of user data.

Implement encryption and secure authentication mechanisms.

***Accessibility and Usability:***

Ensure the application is accessible to users with disabilities.

Provide a user-friendly interface and intuitive navigation.

***Integration with External Services:***

Integrate with external APIs for features such as financial data aggregation and payment processing.

Ensure seamless communication with external services.

These system features collectively provide users with a comprehensive platform for managing their personal finances effectively and efficiently. Each feature is designed to meet specific user needs and enhance the overall user experience of the application.

## User registration and authentication

* + 1. Description and Priority

User Registration and Authentication allows users to create accounts and log in securely. It includes verifying user credentials and managing authentication processes.

Priority: High\_PRIORITY

Component Ratings:

Benefit: 8

Penalty: 4

Cost: 6

Risk: 7

This feature is crucial for allowing users to access and use the application securely. It provides the foundation for user engagement and data security.

* + 1. Stimulus/Response Sequences

***User Registration:***

User navigates to the registration page.

System displays a registration form.

User enters their email address, username, and password.

System validates the entered information (e.g., checks for duplicate usernames, validates email format).

If the information is valid, the system creates a new user account and redirects the user to the login page.

If the information is invalid, the system displays error messages and prompts the user to correct the information.

***User Login:***

User navigates to the login page.

System displays a login form.

User enters their username and password.

System validates the credentials.

If the credentials are correct, the system logs the user in and redirects them to the dashboard.

If the credentials are incorrect, the system displays an error message and prompts the user to try again.

***User Logout:***

User clicks on the logout button.

System logs the user out and redirects them to the login page.

***Password Reset:***

User clicks on the "forgot password" link on the login page.

System prompts the user to enter their email address.

User enters their email address.

System sends a password reset link to the user's email address.

User clicks on the password reset link and enters a new password.

System validates the new password and updates the user's account.

These sequences of actions and system responses simulate the behavior of the User Registration and Authentication feature, allowing users to securely create accounts, log in, and manage their passwords.

***4.1.3 Functional Requirements***

REQ-1: The system shall allow users to create a new account by providing a valid email address, a unique username, and a password.

If the email address is not in a valid format, the system shall display an error message.

If the username is already in use, the system shall display an error message.

If the password does not meet the minimum complexity requirements, the system shall display an error message.

REQ-2: The system shall allow users to log in to their account using their username and password.

If the username or password is incorrect, the system shall display an error message.

If the account is not verified, the system shall prompt the user to verify their email address.

REQ-3: The system shall allow users to reset their password if they forget it.

Users shall be able to request a password reset link be sent to their email address.

The password reset link shall expire after a specified time period.

Users shall be able to set a new password after clicking the password reset link.

REQ-4: The system shall encrypt user passwords using a secure hashing algorithm before storing them in the database.

REQ-5: The system shall send a verification email to the user's email address upon successful account creation.

The verification email shall contain a link that the user must click to verify their email address.

The account shall not be usable until the email address is verified.

REQ-6: The system shall provide a secure session management mechanism to ensure that users remain logged in securely.

Sessions shall expire after a period of inactivity.

Users shall be automatically logged out after a certain period of time to prevent unauthorized access.

REQ-7: The system shall provide mechanisms to prevent brute force attacks on user accounts, such as limiting the number of login attempts and implementing CAPTCHA verification.

REQ-8: The system shall provide a "Remember Me" option on the login page to allow users to stay logged in across browser sessions.

## Expense Tracking

* + 1. Description and Priority

***Description:*** Expense Tracking enables users to manage their expenses by adding, editing, and deleting entries. They can categorize expenses and track them over time to monitor their financial activities effectively.

***Priority:*** High\_PRIORITY

* + 1. Stimulus/Response Sequences

***User adds a new expense:***

User navigates to the "Add Expense" section of the application.

System displays a form with fields for entering the expense details (e.g., amount, category, date).

User enters the expense details and submits the form.

System validates the input and adds the expense to the user's list of expenses.

System displays a confirmation message indicating that the expense has been added successfully.

***User edits an existing expense:***

User navigates to the "Edit Expense" section of the application.

System displays a form pre-populated with the details of the selected expense.

User modifies the expense details and submits the form.

System validates the input and updates the expense with the new details.

System displays a confirmation message indicating that the expense has been updated successfully.

***User deletes an existing expense:***

User selects the expense to be deleted.

User confirms the deletion action.

System deletes the selected expense from the user's list of expenses.

System displays a confirmation message indicating that the expense has been deleted successfully.

***User categorizes an expense:***

User selects an expense to categorize.

User chooses a category from a predefined list.

System updates the expense's category.

System displays a confirmation message indicating that the expense has been categorized successfully.

* + 1. ***Functional Requirements***

REQ-1: The system shall provide a form for users to add a new expense.

The form shall include fields for entering the expense amount, category, date, and any additional notes.

The system shall validate the expense amount to ensure it is a valid numerical value.

The system shall validate the date to ensure it is a valid date format.

REQ-2: The system shall allow users to edit an existing expense.

Users shall be able to select an expense from a list of their expenses to edit.

The system shall display a form pre-populated with the selected expense details.

Users shall be able to modify the expense amount, category, date, and notes.

The system shall validate the edited expense amount and date.

REQ-3: The system shall allow users to delete an existing expense.

Users shall be able to select an expense from a list of their expenses to delete.

The system shall prompt users to confirm the deletion action.

The system shall remove the selected expense from the list of expenses.

REQ-4: The system shall allow users to categorize expenses.

Users shall be able to select an expense from a list of their expenses to categorize.

Users shall be able to choose a category from a predefined list of categories.

The system shall update the expense's category based on the user's selection.

# Other Nonfunctional Requirements

The "Budget Buddy" web application should be fast, user-friendly, and secure. It should handle at least 1000 users, load in 3 seconds, and respond in 1 second. The interface should be accessible, data encrypted, and user data backed up regularly. The application should comply with regulations, be easy to maintain, and undergo performance testing.

## Performance Requirements

Performance requirements for the "Budget Buddy" web application are crucial for providing a responsive and efficient user experience. These requirements are as follows:

***Response Time:*** The application should respond to user actions (such as clicking a button or submitting a form) within 1 second to provide a smooth and interactive experience. This ensures that users can navigate the application quickly and efficiently.

***Loading Time:*** The application should load within 3 seconds on average, including all necessary resources such as HTML, CSS, JavaScript, and images. This ensures that users can access the application without experiencing significant delays.

**Concurrency:** The application should be able to handle at least 1000 concurrent users without a significant decrease in performance. This ensures that the application can scale to meet the needs of a large user base.

***Data Processing:*** The application should be able to process and display financial data (such as expense tracking, budget calculations, and goal progress) efficiently, even with large datasets. This ensures that users can access and manage their financial information without delays.

***Reliability:*** The application should have an uptime of at least 99.9% to ensure that users can access the application whenever they need to. This includes scheduled maintenance and updates, which should be performed during off-peak hours to minimize disruption to users.

***Scalability:*** The application should be designed to scale horizontally to handle increased traffic and user load. This includes using load balancers and caching mechanisms to distribute traffic and optimize performance.

**Security:** The application should adhere to security best practices to protect user data and prevent unauthorized access. This includes using encryption for data transmission, implementing secure authentication mechanisms, and regularly updating security patches.

These performance requirements are essential for ensuring that the "Budget Buddy" web application meets user expectations for speed, reliability, and efficiency. They will guide the development process and help developers make suitable design choices to achieve optimal performance.

## Safety Requirements

Safety requirements for the "Budget Buddy" web application primarily focus on protecting user data and ensuring the application's secure operation. These requirements include:

***Data Protection:*** The application must encrypt sensitive user data (such as passwords and financial information) to protect it from unauthorized access.

***User Authentication:*** The application must use secure authentication mechanisms to verify the identity of users and prevent unauthorized access to accounts.

***Secure Communication:*** The application must use HTTPS to encrypt data transmitted between the client and the server, preventing interception by malicious actors.

***Access Controls:*** The application must implement access controls to ensure that users can only access data and features that are appropriate for their role.

***Backup and Recovery:*** The application must regularly backup user data and have a plan in place for recovering data in the event of a system failure or data loss.

**Compliance:** The application must comply with relevant data protection regulations (such as GDPR, CCPA) and financial regulations (such as PCI DSS) to ensure the security and privacy of user data.

***Security Audits:*** The application must undergo regular security audits to identify and mitigate potential vulnerabilities.

***User Education:*** The application must provide users with guidance on how to protect their data and use the application securely.

These safety requirements aim to protect users' data and ensure the secure operation of the "Budget Buddy" web application, reducing the risk of loss, damage, or harm resulting from its use.

## Security Requirements

Security requirements for the "Budget Buddy" web application are essential to protect user data and ensure the application's secure operation. These requirements include:

***Data Encryption:*** User data, including passwords and financial information, must be encrypted both in transit and at rest to prevent unauthorized access.

***User Authentication:*** The application must use strong authentication mechanisms, such as multi-factor authentication, to verify the identity of users and prevent unauthorized access.

***Access Control:*** The application must implement role-based access control (RBAC) to ensure that users can only access data and features appropriate for their role.

***Secure Communication:*** The application must use HTTPS to encrypt data transmitted between the client and the server, preventing interception by malicious actors.

***Security Audits:*** The application must undergo regular security audits to identify and mitigate potential vulnerabilities.

***Data Protection Regulations:*** The application must comply with relevant data protection regulations, such as GDPR or CCPA, to ensure the security and privacy of user data.

***Secure Development Practices:*** The application must be developed using secure coding practices to prevent common vulnerabilities, such as SQL injection and cross-site scripting (XSS).

***User Privacy:*** The application must respect user privacy preferences and provide mechanisms for users to control their data.

***Incident Response:*** The application must have an incident response plan in place to quickly address security breaches or data breaches.

***Security Certifications:*** The application must satisfy any security certifications required by external policies or regulations.

These security requirements aim to protect user data and ensure the secure operation of the "Budget Buddy" web application, reducing the risk of security breaches or data breaches.

## Software Quality Attributes

Software quality attributes for the "Budget Buddy" web application are crucial for ensuring its effectiveness and user satisfaction. These attributes include:

***Usability:*** The application should be easy to use, with an intuitive interface and clear navigation. Users should be able to accomplish tasks quickly and efficiently.

***Reliability:*** The application should be reliable, with minimal downtime and errors. It should perform consistently under normal and peak loads.

***Security:*** The application should be secure, protecting user data and preventing unauthorized access. It should comply with relevant security standards and best practices.

***Performance:*** The application should perform well, with fast response times and efficient use of resources. It should be able to handle a large number of users and transactions without slowing down.

***Maintainability:*** The application should be easy to maintain, with clear and well-structured code. It should be easy to update and modify as needed.

***Scalability:*** The application should be scalable, able to handle growth in users and data volume. It should be able to scale both vertically and horizontally.

***Interoperability:*** The application should be interoperable, able to work with other systems and platforms. It should support standard protocols and data formats.

***Portability:*** The application should be portable, able to run on different platforms and devices. It should be able to adapt to different environments without modification.

***Testability:*** The application should be testable, with well-defined test cases and procedures. It should be easy to identify and fix bugs.

***Adaptability:*** The application should be adaptable, able to evolve and meet changing requirements. It should be able to integrate new features and technologies.

These software quality attributes are important for ensuring that the "Budget Buddy" web application meets the needs of its users and developers, providing a reliable, secure, and efficient platform for managing personal finances.

## Business Rules

Business rules for the "Budget Buddy" web application govern how individuals or roles can perform functions under specific circumstances. These rules help guide the behavior and usage of the application. Some examples of business rules for "Budget Buddy" include:

1. Only individuals with a valid email address can create an account. Each email address can be associated with only one account.
2. Users can categorize their expenses into predefined categories such as "Food," "Transportation," and "Utilities." Expenses must be assigned to a category.
3. Users can create budgets for different categories (e.g., setting a monthly budget for groceries). Budgets can be updated or deleted by the user.
4. Users can choose to receive notifications for upcoming bills, budget limits exceeded, or other financial events. Users can also choose the frequency and method of notifications.
5. User financial data is private and should not be shared with third parties without explicit user consent. Data should be stored securely and in compliance with relevant data protection regulations.
6. Different roles (e.g., regular user, administrator) have different levels of access to the application. Administrators have the ability to view and manage all user accounts and data.
7. Users cannot modify or delete financial data (e.g., expenses, budgets) that has already been confirmed or processed. Data can only be modified or deleted if it is in a draft or pending state.
8. The application should support multiple currency formats and date formats based on user preferences or location.

# Other Requirements

***Database Requirements***

Storage Solution: Implement a reliable and scalable DBMS such as PostgreSQL or MySQL.

Backup Strategy: Schedule regular backups and establish a data recovery plan.

Consistency Measures: Maintain data integrity with appropriate constraints and transaction management.

***Internationalization***

Language Options: Support multiple languages.

Regional Settings: Adapt to local date, time, number formats, and currency symbols.

*Legal and Compliance*

Data Privacy: Ensure compliance with data protection laws like GDPR and CCPA.

Financial Standards: Follow relevant financial regulations and standards.

***Reuse and Modularity***

Component Reusability: Develop reusable modules and components to ease future updates.

Integration: Provide APIs for integrating with other financial services and tools.

***Accessibility***

Inclusive Design: Adhere to WCAG 2.1 standards for accessibility.

Keyboard Accessibility: Ensure all features are accessible via keyboard.

***Environmental Considerations***

Energy Usage: Optimize for low energy consumption, particularly for mobile devices.

Resource Efficiency: Manage memory and CPU usage efficiently to ensure smooth performance.

***Performance Monitoring***

Real-time Insights: Use tools for continuous performance monitoring to quickly address issues.

User Reports: Implement feedback mechanisms for users to report bugs or performance problems.

**Appendix A: Glossary**

This glossary defines terms, acronyms, and abbreviations used in the Software Requirements Specification (SRS) for the "Budget Buddy" web application.

***Terms***

Budget: A financial plan that outlines expected income and expenses over a specified period.

Expense Tracking: The process of recording and monitoring spending to manage finances effectively.

User Authentication: The process of verifying the identity of a user attempting to access the application.

Role-Based Access Control (RBAC): A system of regulating access to parts of the application based on the user's role.

User Interface (UI): The visual elements through which a user interacts with the application.

Backup: A copy of data stored separately to ensure data recovery in case of loss.

Localization: Adapting the application to meet the language, cultural, and other specific needs of a particular region.

Encryption: The process of converting information or data into a code to prevent unauthorized access.

***Acronyms and Abbreviations***

API: Application Programming Interface. A set of functions and procedures that allow for the creation of applications accessing the features or data of an operating system, application, or other service.

CCPA: California Consumer Privacy Act. A state statute intended to enhance privacy rights and consumer protection for residents of California, USA.

DBMS: Database Management System. Software that uses a standard method of cataloging, retrieving, and running queries on data.

GDPR: General Data Protection Regulation. A regulation in EU law on data protection and privacy for individuals within the European Union and the European Economic Area.

GUI: Graphical User Interface. A type of user interface that allows users to interact with electronic devices through graphical icons and visual indicators.

HTTPS: Hypertext Transfer Protocol Secure. An extension of HTTP for secure communication over a computer network.

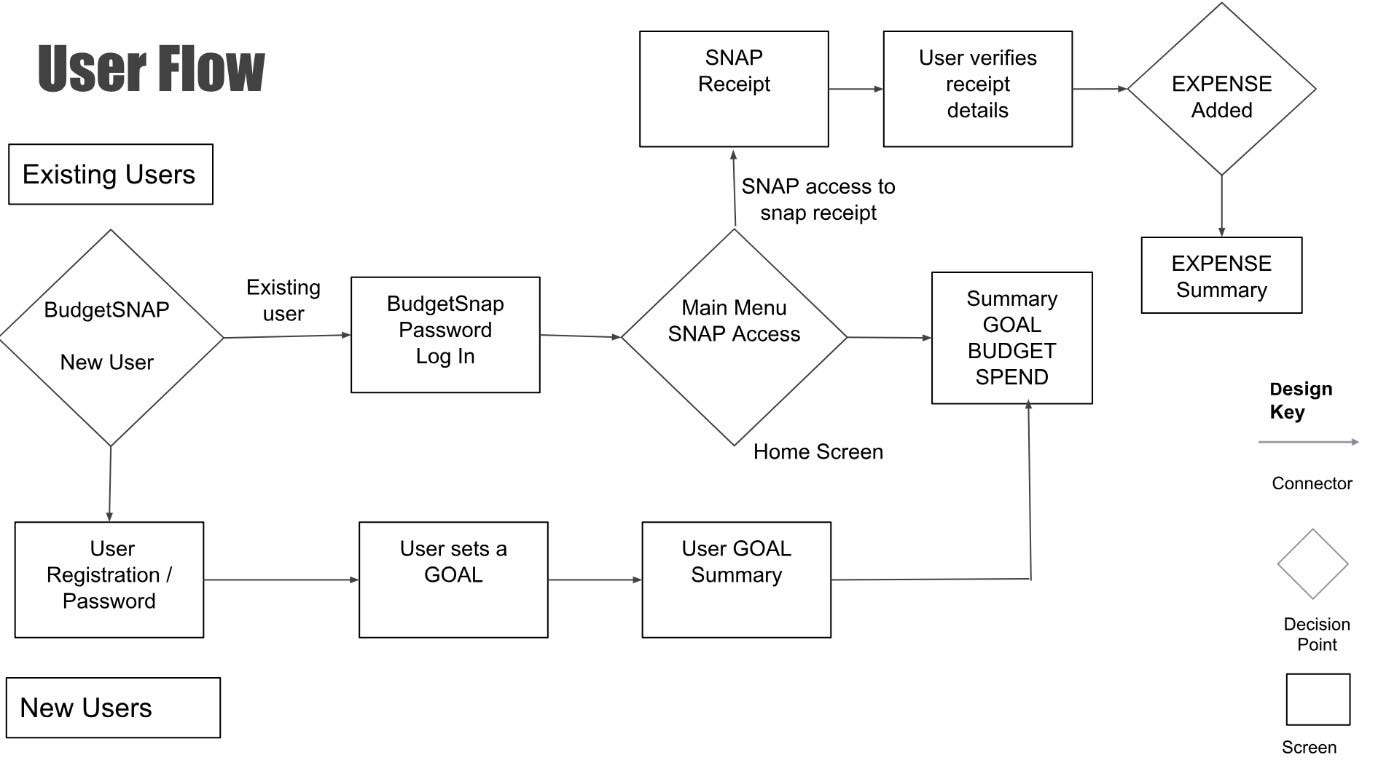
RBAC: Role-Based Access Control. A method of restricting access to authorized users based on their role within an organization.

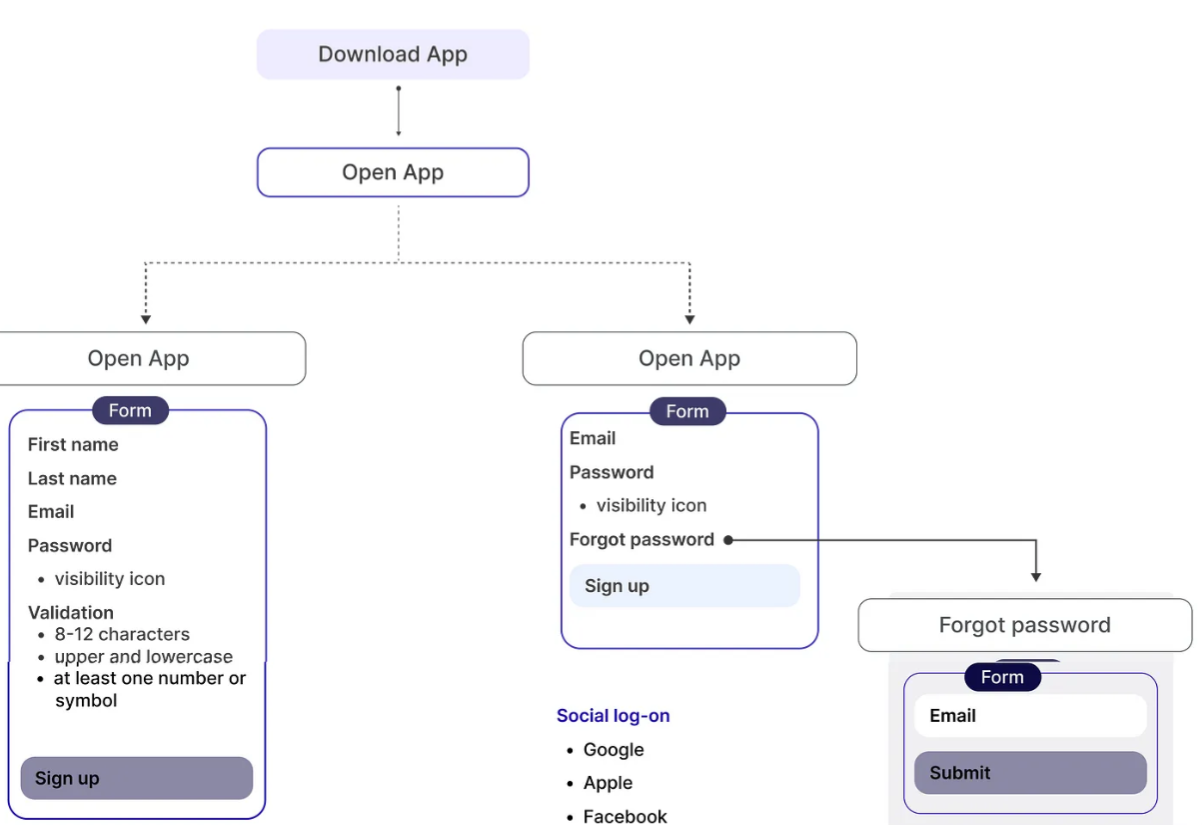
SRS: Software Requirements Specification. A document that describes the functionalities and constraints of a software system.

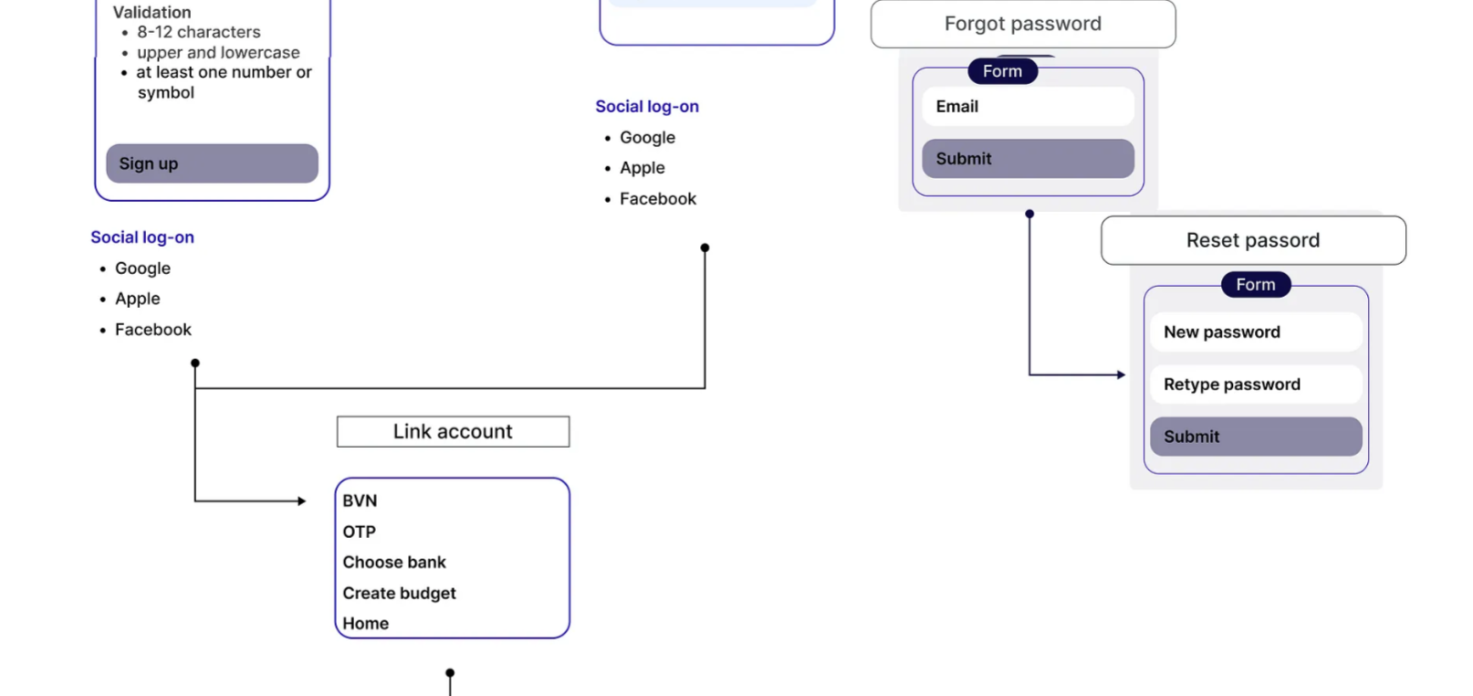
WCAG: Web Content Accessibility Guidelines. A set of guidelines for making web content accessible to people with disabilities.

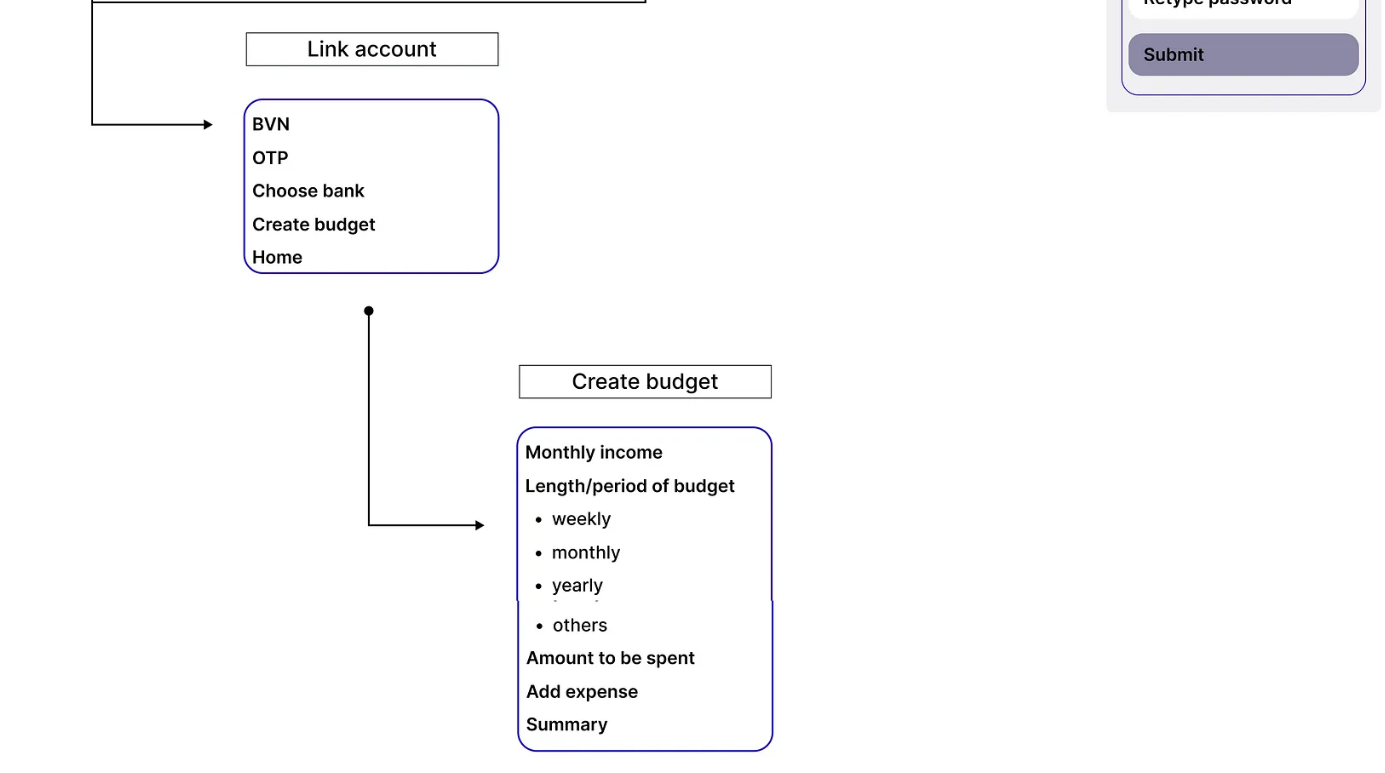
This glossary is intended to help readers understand the specific terms and abbreviations used in the context of the "Budget Buddy" project.

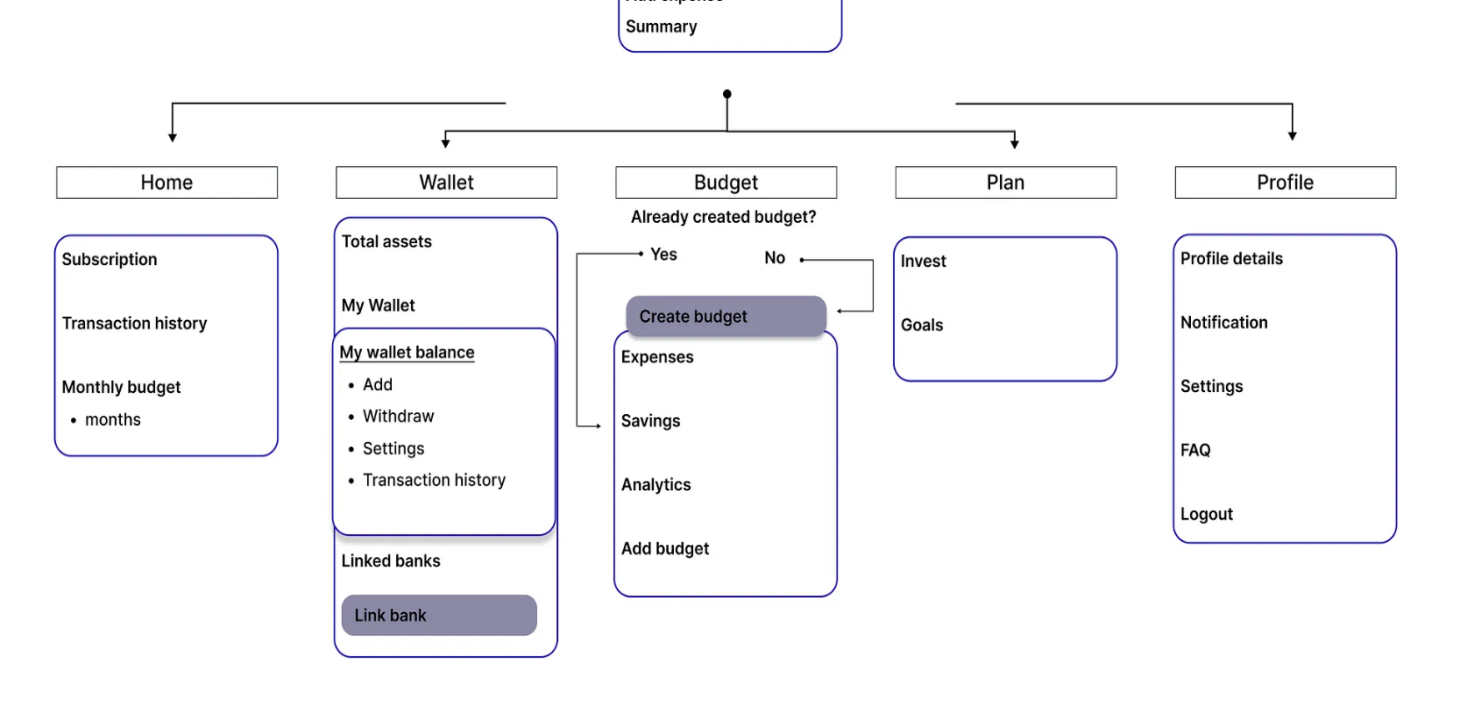
**Appendix B: Analysis Models**

****

****

****

****

****

**Appendix C: To Be Determined List**

This appendix contains a numbered list of all "To Be Determined" (TBD) items referenced in the Software Requirements Specification (SRS) for the "Budget Buddy" web application. These items need further clarification or information and should be tracked until they are resolved.

***User Role Definitions***

TBD-1: Specific roles and their permissions within the application.

**Notification Preferences**

TBD-2: Detailed options for user notification settings.

***Backup Schedule***

TBD-3: Frequency and timing of regular data backups.

***Localization Options***

TBD-4: Specific languages and regional settings to be supported at launch.

***API Integration Details***

TBD-5: List of third-party financial tools and services for API integration.

***Accessibility Features***

TBD-6: Specific accessibility features to be implemented.

***Security Measures***

TBD-7: Detailed security protocols and encryption methods to be used.

***Performance Benchmarks***

TBD-8: Specific performance metrics and benchmarks the application must meet.

***User Interface Guidelines***

TBD-9: Detailed GUI standards or style guides to be followed.

***Data Retention Policy***

TBD-10: Policy for how long user data will be retained and the conditions for data deletion.

***Testing Requirements***

TBD-11: Specific testing procedures and environments required for validation.

***Third-Party Components***

TBD-12: List of third-party or commercial components to be used in the development process.

***Legal and Compliance Standards***

TBD-13: Detailed legal and regulatory standards the application must comply with.

***Financial Calculations***

TBD-14: Specific algorithms or methods for financial calculations used in budgeting and expense tracking.

***Data Migration Plan***

TBD-15: Plan for migrating existing user data to the new system, if applicable.

Each TBD item should be addressed and resolved as the project progresses to ensure the completeness and accuracy of the SRS.