In Partial Fulfillment of the Requirements for the Course

Software Engineering 2

**Budge-!T: Personal Expense and Savings Optimizer**  
**Team Members:**

**Submitted By:**

Pagulayan, Kamira Allison F.

Rico, Ronaldo Jr. D.

Asuncion, Beatriz Uy

Galvez, Aldrin

Esquejo, Sherdon Rappah

**Date of Submission:**

March 3, 2025

Table of Contents

[**Executive Summary** 4](#_Toc190721443)

[**Introduction** 4](#_Toc190721444)

[**Background of the Problem** 4](#_Toc190721445)

[**Project Objectives** 4](#_Toc190721446)

[**Scope and Limitations** 4](#_Toc190721447)

[**Requirements Analysis** 4](#_Toc190721448)

[**Stakeholder Identification** 4](#_Toc190721449)

[**Use-Case Diagrams and Descriptions** 5](#_Toc190721450)

[**System Design** 5](#_Toc190721451)

[**Vision Statement** 5](#_Toc190721452)

[**Architectural Design** 5](#_Toc190721453)

[**Design Patterns Applied** 5](#_Toc190721454)

[**Entity-Relationship Diagrams (ERD)** 5](#_Toc190721455)

[**User Interface Mockups** 5](#_Toc190721456)

[**Project Management** 6](#_Toc190721457)

[**Agile Practices and Sprint Planning** 6](#_Toc190721458)

[**Gantt Chart or Project Timeline** 6](#_Toc190721459)

[**Development Process** 6](#_Toc190721460)

[**Coding Standards and Best Practices** 6](#_Toc190721461)

[**Tools and Technologies Used** 6](#_Toc190721462)

[**Implementation Details** 6](#_Toc190721463)

[**Testing** 6](#_Toc190721464)

[**Test Case Documentation** 6](#_Toc190721465)

[**Results of Test Execution** 6](#_Toc190721466)

[**Bug Tracking and Resolution** 6](#_Toc190721467)

[**User Guide for the System** 6](#_Toc190721468)

[**Conclusion** 7](#_Toc190721469)

[**Summary of Project Outcomes** 7](#_Toc190721470)

[**Challenges Faced and Lessons Learned** 7](#_Toc190721471)

[**Recommendations for Future Work** 7](#_Toc190721472)

[**References** 7](#_Toc190721473)

[**Appendices** 7](#_Toc190721474)

[**Appendix A: Meeting Minutes** 7](#_Toc190721475)

[**Appendix B: Screenshots** 7](#_Toc190721476)

[**Appendix C: Other Supporting Documents** 7](#_Toc190721477)

# **Executive Summary**

Budge-IT is a personal budgeting tool designed to help users manage their finances easily and visually appealing. It has a simple interface that lets you track your spending, set savings goals, and review your finances over time. Budge-IT utilizes Agile principles, particularly the Dynamic Systems Development Method (DSDM), to ensure it meets user's needs. This strategy focuses on developing tiny, incremental changes based on frequent input from consumers. Some of the app’s most beneficial features include automated cost categorization, interactive financial dashboards with customized charts, and real-time notifications that let users know when they’re close to overspending. By combining these aspects, Budge-IT delivers a practical, easy-to-use tool for personal financial planning, while also keeping up with recent trends in software development and design.

# **Introduction**

## **Background of the Problem**

Many individuals struggle with personal finance, leading to overspending, poor savings habits, and financial stress. Existing solutions either lack user engagement or are overly complex.

## **Project Objectives**

1. Provide an intuitive budgeting tool for tracking expenses and savings.
2. Offer data-driven insights to improve financial literacy.
3. Implement real-time alerts and goal-setting features.

## **Scope and Limitations**

1. **Scope:** Includes expense tracking, savings goal management, and financial analytics.
2. **Limitations:** Does not provide direct banking transactions. Lack of scalability to handle the amount of user accounts.

# **Requirements Analysis**

## **Stakeholder Identification**

1. **Primary Users:** Gen Z.
2. **Secondary Users:** Students, young adults, financial advisors, educators, and professionals looking for financial planning tools.
3. **Functional and Non-functional Requirements**
   * **Functional Requirements:**
     + Expense tracking and categorization.
     + Savings goal setting.
     + Real-time financial insights.
     + Notifications for overspending.
   * **Non-functional Requirements:**
     + User-friendly interface.
     + Secure data storage.
     + Responsive design for mobile and web use.

## A diagram of a person with a triangle and a triangle AI-generated content may be incorrect.**Use-Case Diagrams and Descriptions**

The use-case diagram illustrates the interactions between the user (Gen Z) and the key functionalities of the Budge-IT application. It highlights the system’s primary operations categorized under Create, Read, Update, and Delete (CRUD) actions.

1. **Actor:**
   * **User (Gen Z):** Represents the target audience who will interact with the app to manage their personal finances effectively.
2. **Use Cases:**
   * **Login:** Users authenticate their identity to access their personal budget dashboard.
   * **Add Expenses/Income (Create):** Users can input their income and expenses to track their cash flow.
   * **Set Savings Goals (Create):** Users define their financial goals, such as saving for a vacation or emergency fund.
   * **Track Spending (Create):** Users monitor their spending habits to ensure they are staying within their budget.
   * **Receive Notifications (Read):** Users get alerts and reminders related to their budget and savings goals.
   * **View Financial Reports (Read):** Users can access detailed reports, including spending summaries and savings progress.
   * **Update Transaction (Update):** Users can modify previously entered transactions (e.g., correct amounts or categories).
   * **Delete Transaction (Delete):** Users can remove incorrect or outdated transactions from their records.
3. **CRUD Operations:**
   * **Create:** Login, Add Expenses/Income, Set Savings Goals, Track Spending
   * **Read:** Receive Notifications, View Financial Reports
   * **Update:** Update Transaction
   * **Delete:** Delete Transaction

# **System Design**

## **Vision Statement**

We want Budge-IT to become the preferred budgeting tool for those who wish to take charge of their finances, establish objectives, and maintain financial stability. Our software will be entertaining and easy to use while offering clear insights, practical tools, and incentives to promote conservative spending and saving.

## **Architectural Design**

1. **Presentation Layer**
   1. **Purpose:** The presentation layer is responsible for delivering an intuitive and interactive user experience for the budgeting application. It handles user interactions, displays data, and communicates with the application logic layer to perform actions. This layer ensures the application is visually appealing, easy to use, and accessible across multiple devices.
   2. **Components:**
      * **User Interface (UI):**
        + **Dashboard:** Provides a summary of expenses, income, and budget utilization.
        + **Expense Tracking Page:** Allows users to input, edit, and categorize expenses.
        + **Budget Management Page:** Enables users to set and adjust budgets for specific categories. Tracks budget usage with real-time updates.
        + **Savings and Goals Page:** Tracks savings progress and displays goals visually. Provides articles to achieve savings targets.
        + **Report Page:** Generates detailed financial reports (monthly, yearly, or custom range) and by categories such as food, transportation, and others, based on the user input.
      * **User Interaction Elements:**
        + Buttons, dropdowns, sliders, and input fields for seamless data entry.
        + Interactive elements like drag-and-drop for reorganizing budgets or goals.
   3. **Technologies**
      * **Figma:** For designing and prototyping user interfaces.
2. **Application Logic Layer**
   1. **Purpose:** The application logic layer serves as the core of the budgeting application, where the business rules and functionality are implemented. It handles calculations, manages data flow between the user interface and the database, enforces application rules, and ensures that the application meets user requirements, such as expense tracking, budget management, and financial goal setting.
   2. **Components:**
      * **Expense Management Module:** Calculates total spending and compares it with allocated budgets. Supports features like adding, editing, and deleting expenses.
      * **Goal Tracking Module:** Monitors progress and offers insights on goal fulfillment.
      * **Notification Service:** Generates alerts for overspending, savings milestones, and debt repayment reminders.
      * **Data Aggregation Service:** Compiles data for charts, graphs, and financial summaries.
   3. **Technologies:**
      * MySQL for structured data.
      * Libraries for machine learning and financial calculations.
      * Firebase for real-time data sync.
3. **Data Management Layer**
   1. **Purpose:** Manages all data-related operations.
   2. **Components:**
      * **Database Management System (DBMS):** Stores user data, transaction records, and objectives.
      * **Data Processing Module:** Processes raw data into user-friendly representations (e.g., charts, insights).
      * **Backup and Recovery System:** Ensures data security and recovery in case of failure.
   3. **Technologies:**
      * SQL Databases.
      * Data storages for analytics.
4. **Infrastructure Layer**
   1. **Purpose:** The infrastructure layer for the budgeting application ensures that the application operates efficiently, securely, and reliably. It provides the necessary foundation to handle user traffic, store financial data, and maintain high performance and availability. This layer supports the backend logic, user interface, and data management while enabling future scalability.
   2. **Components:**
      * **Cloud Storage:** Hosts user data and application files.
      * **Server Configuration:** Manages APIs and backend services.
      * **Authentication Services:** Handles user authentication (e.g., OAuth).
      * **Scalability Solutions:** Ensures smooth app performance as the user base grows.
   3. **Technologies:**
      * Cloud providers (e.g., AWS, Azure)
5. **Security Layer**
   1. **Purpose:** Ensures data protection and application integrity.
   2. **Components:**
      * **Encryption:** Protects sensitive user data like passwords and transactions.
      * **User Authentication:** Implements multi-factor authentication.
      * **Audit Logs:** Tracks user actions for accountability.
      * **Regulatory Compliance:** Aligns with Philippine Data Privacy Act and other data protection laws.
   3. **Technologies:**
      * SSL/TLS protocols.
      * Security tools for penetration testing.

## **Design Patterns Applied**

MVC (Model-View-Controller) for software modularity.

Singleton for financial calculations.

## **Entity-Relationship Diagrams (ERD)**

Depicts relationships between users, transactions, budgets, and goals.

## **User Interface Mockups**

Includes wireframes for onboarding, dashboards, and settings.

# **Project Management**

## **Agile Practices and Sprint Planning**

1. Follows DSDM with iterative releases.
2. User feedback incorporated in each sprint.

## **Gantt Chart or Project Timeline**

Project phases: Feasibility, Foundations, Exploration, Engineering, Deployment.

# **Development Process**

## **Coding Standards and Best Practices**

1. Adheres to clean coding principles.
2. Uses version control (GitHub).

## **Tools and Technologies Used**

1. **Frontend:** React.js
2. **Backend:** Node.js with Express.js
3. **Database:** Firebase

## **Implementation Details**

1. Secure authentication with Firebase Authentication.
2. Real-time database updates using Firebase.

# **Testing**

## **Test Case Documentation**

Covers unit testing for expense tracking and notification features.

## **Results of Test Execution**

High pass rate for core functionalities.

## **Bug Tracking and Resolution**

Managed through Jira.

## **User Guide for the System**

Step-by-step instructions for setting budgets and tracking expenses.

# **Conclusion**

## **Summary of Project Outcomes**

Budge-IT successfully provides an interactive budgeting experience with automation and insights to promote financial literacy.

## **Challenges Faced and Lessons Learned**

Balancing simplicity with feature richness.

Addressing user alert fatigue.

## **Recommendations for Future Work**

Integration with banking APIs for real-time tracking.

AI-based financial recommendations.

# **References**

Myers, G. J., Sandler, C., & Badgett, T. (2012). *The Art of Software Testing*. John Wiley & Sons. <https://malenezi.github.io/malenezi/SE401/Books/114-the-art-of-software-testing-3-edition.pdf>.

Zafar, I., Nazir, A., & Abbas, M. (2017). *The Impact of Agile Methodology (DSDM) on Software Project Management*. <https://www.researchgate.net/publication/323572478_The_Impact_of_Agile_Methodology_DSDM_on_Software_Project_Management>.

Lienert, J. (2020). *SSWM – Stakeholder Identification*. <https://sswm.info/es/humanitarian-crises/prolonged-encampments/planning-process-tools/exploring-tools/stakeholder-identification>.

# **Appendices**

## **Appendix A: Meeting Minutes**

Summaries of key discussions during development.

## **Appendix B: Screenshots**

UI mockups and test results.

## **Appendix C: Other Supporting Documents**

Survey results and additional diagrams.