In Partial Fulfillment of the Requirements for the Course

Software Engineering 2

Budge-!T: Personal Expense and Savings Optimizer

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

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

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

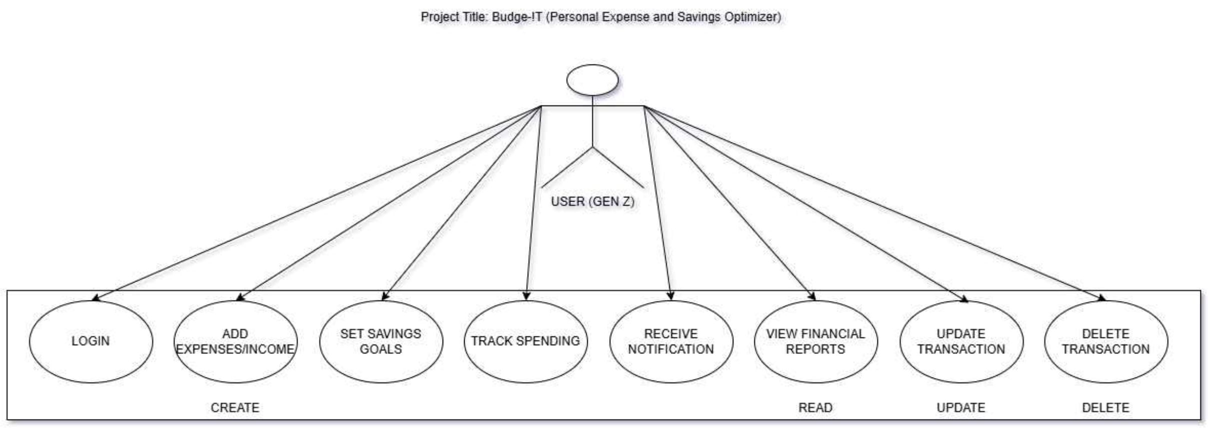
Requirements Analysis

Stakeholder Identification

A. **Primary Users:** Gen Z.
B. **Secondary Users:** Students, young adults, financial advisors, educators, and professionals looking for financial planning tools.
C. **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.

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.

A. **Actor:**
   - **User (Gen Z):** Represents the target audience who will interact with the app to manage their personal finances effectively.

B. **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.

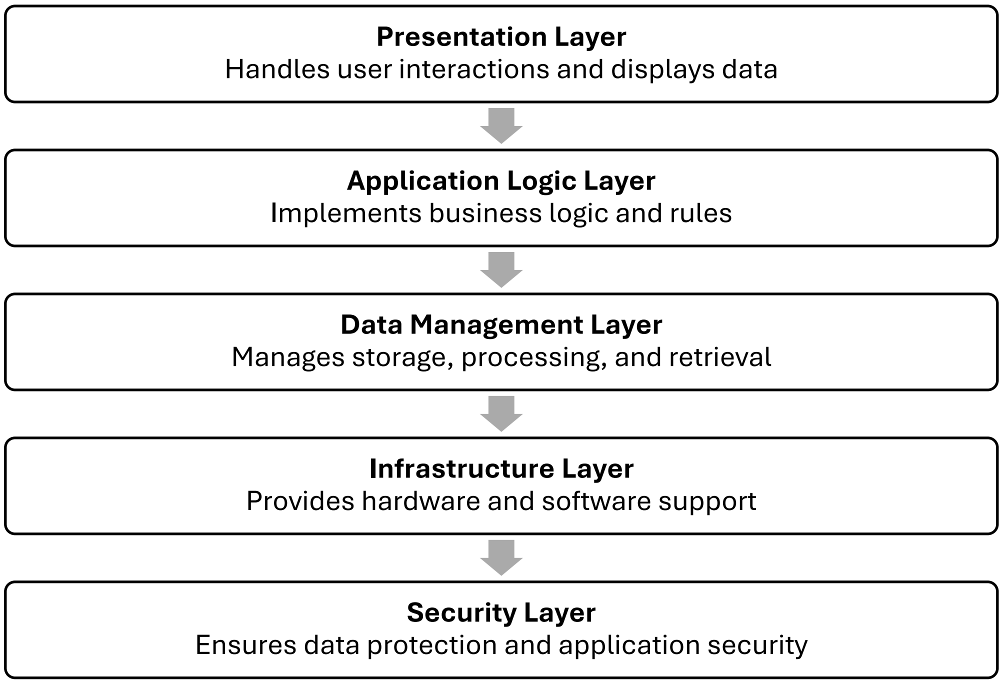
C. **CRUD Operations:**
- **Create:** Login, Add Expenses/Income, Set Savings Goals, Track Spending
- **Read:** Receive Notifications, View Financial Reports
- **Update:** Update Transaction
- **Delete:** Delete Transaction

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


Presentation Layer
Handles user interactions and displays data
Application Logic Layer
Implements business logic and rules
Data Management Layer
Manages storage, processing, and retrieval
Infrastructure Layer
Provides hardware and software support
Security Layer
Ensures data protection and application security

1. **Presentation Layer**
   A. **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.
   B. **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.
   C. **Technologies**
      - **Figma:** For designing and prototyping user interfaces.
2. **Application Logic Layer**
   A. **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.
   B. **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.

C. **Technologies:**

- MySQL for structured data.
- Libraries for machine learning and financial calculations.
- Firebase for real-time data sync.

3. **Data Management Layer**

A. **Purpose:** Manages all data-related operations.

B. **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.

C. **Technologies:**

- SQL Databases.
- Data storages for analytics.

4. **Infrastructure Layer**

A. **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.

B. **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.

C. **Technologies:**

- Cloud providers (e.g., AWS, Azure)

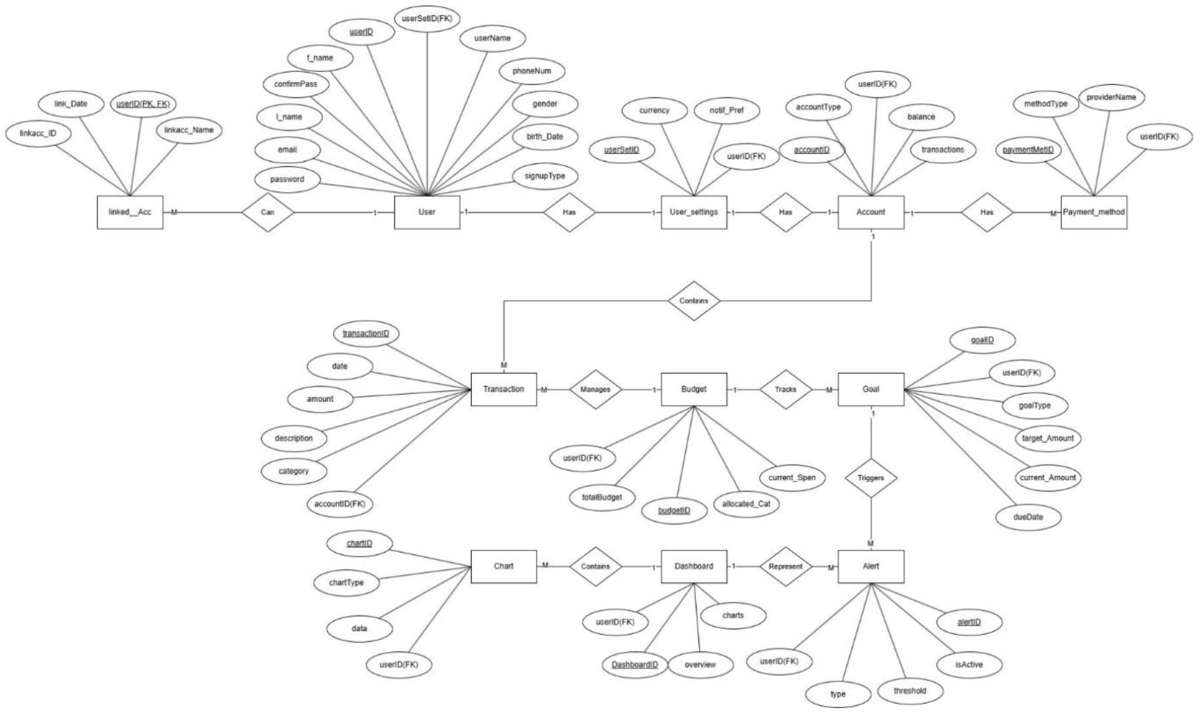
5. **Security Layer**
   A. **Purpose:** Ensures data protection and application integrity.
   B. **Components:**
      - **Encryption:** Protects sensitive user data like passwords and transactions.
      - **User Authentication:** Implements multi-factor authentication.
      - **Audit Logs:** Tracks user actions for accountability.
      - **Regulatory Compliance:** Aligns with Philippine Data Privacy Act and other data protection laws.
   C. **Technologies:**
      - SSL/TLS protocols.
      - Security tools for penetration testing.

Design Patterns Applied

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

Singleton for financial calculations.

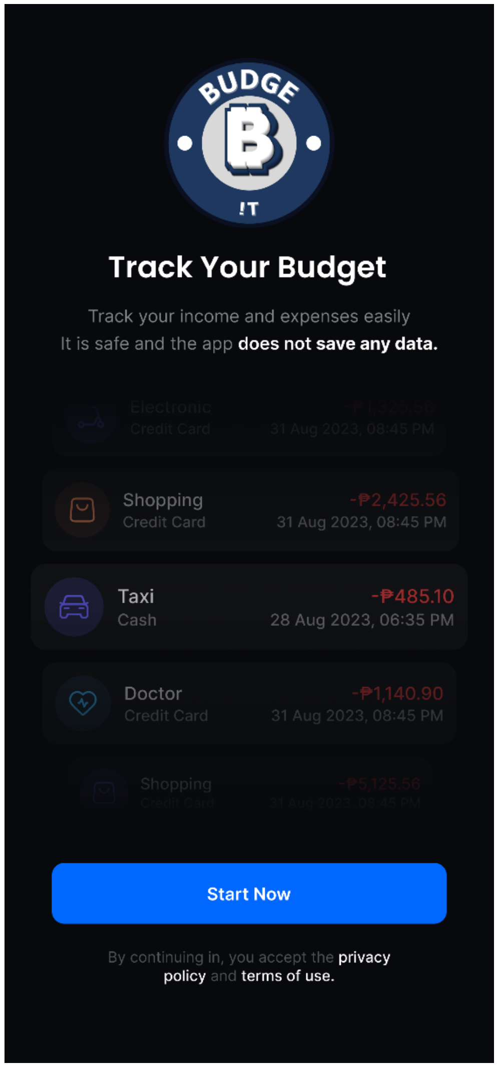
Entity-Relationship Diagrams (ERD)



Depicts relationships between users, user settings, linked accounts, transactions, payment methods, budgets, and goals.

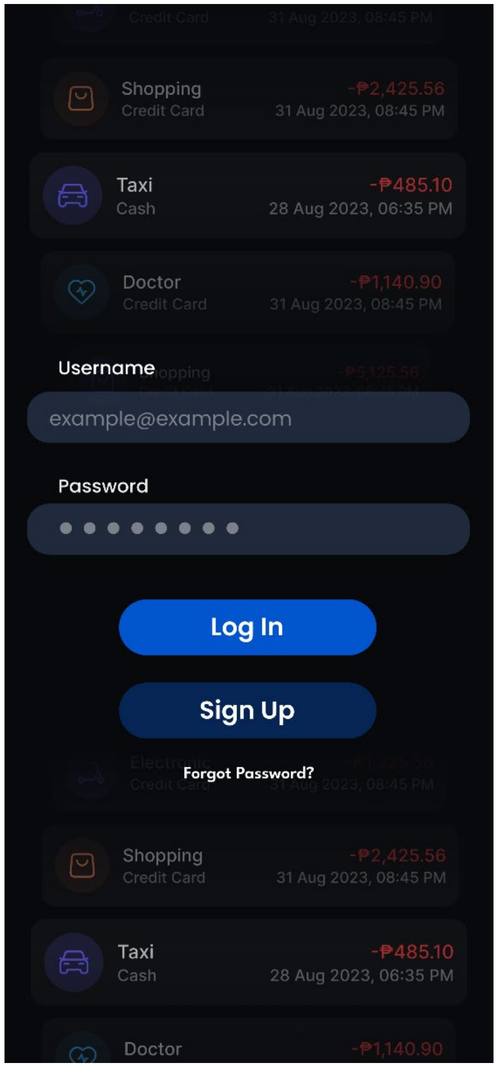
User Interface Mockups

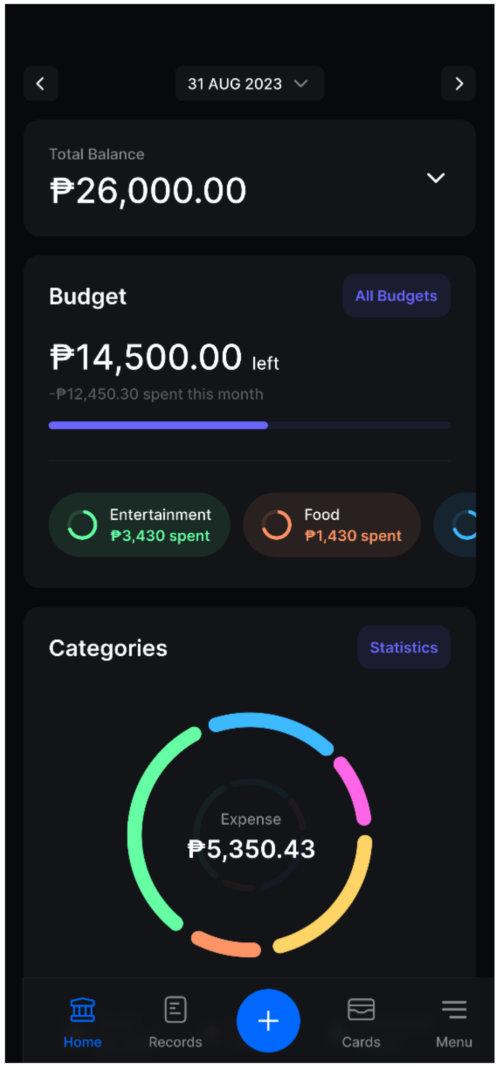
Onboarding, Login, and Dashboard Page


BUDGE IT
Track Your Budget
Shopping Credit Card -₱2,425.56 31 Aug 2023, 08:45 PM
Taxi Cash -₱485.10 28 Aug 2023, 06:35 PM
Doctor Credit Card -₱1,140.90 31 Aug 2023, 08:45 PM
Start Now

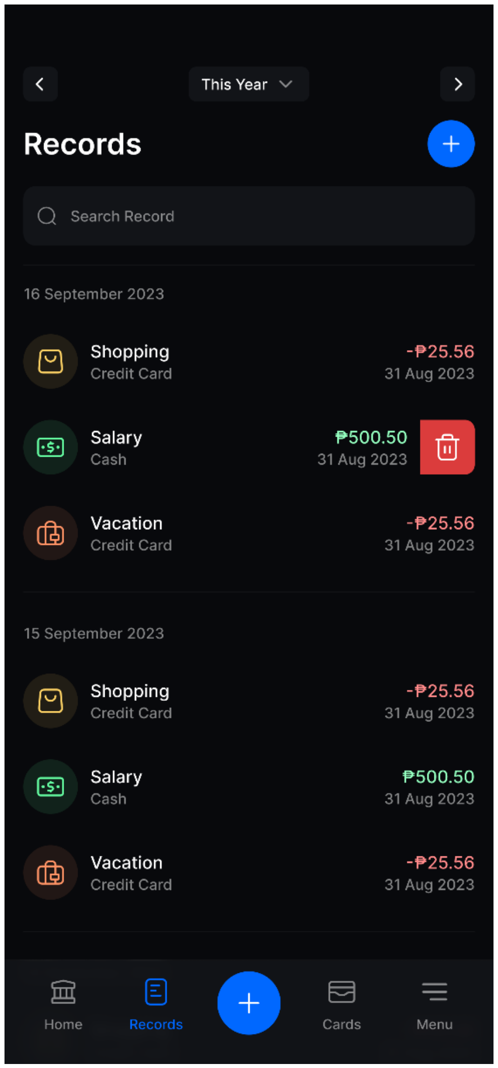
Track your income and expenses easily
It is safe and the app **does not save any data.**

By continuing in, you accept the **privacy policy** and **terms of use.**


Shopping Credit Card -₱2,425.56 31 Aug 2023, 08:45 PM
Taxi Cash -₱485.10 28 Aug 2023, 06:35 PM
Doctor Credit Card -₱1,140.90 31 Aug 2023, 08:45 PM
Username
example@example.com
Password
Log In
Sign Up
Forgot Password?
Shopping Credit Card -₱2,425.56 31 Aug 2023, 08:45 PM
Taxi Cash -₱485.10 28 Aug 2023, 06:35 PM
Doctor -₱1,140.90



Records, Cards, and Menu Page


This Year

Records

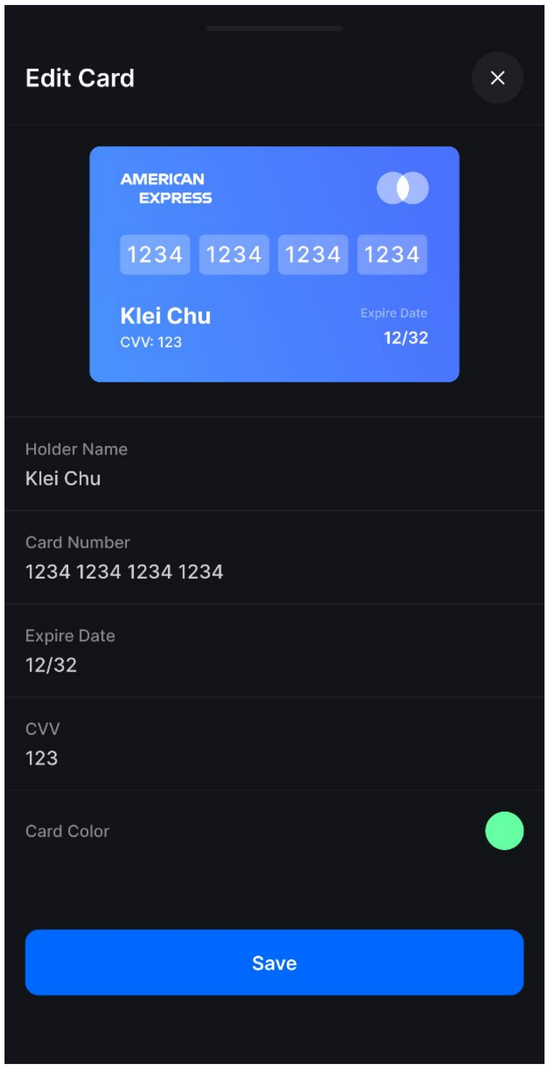
Search Record

16 September 2023

Shopping	Credit Card	-₱25.56 31 Aug 2023
Salary	Cash	₱500.50 31 Aug 2023
Vacation	Credit Card	-₱25.56 31 Aug 2023

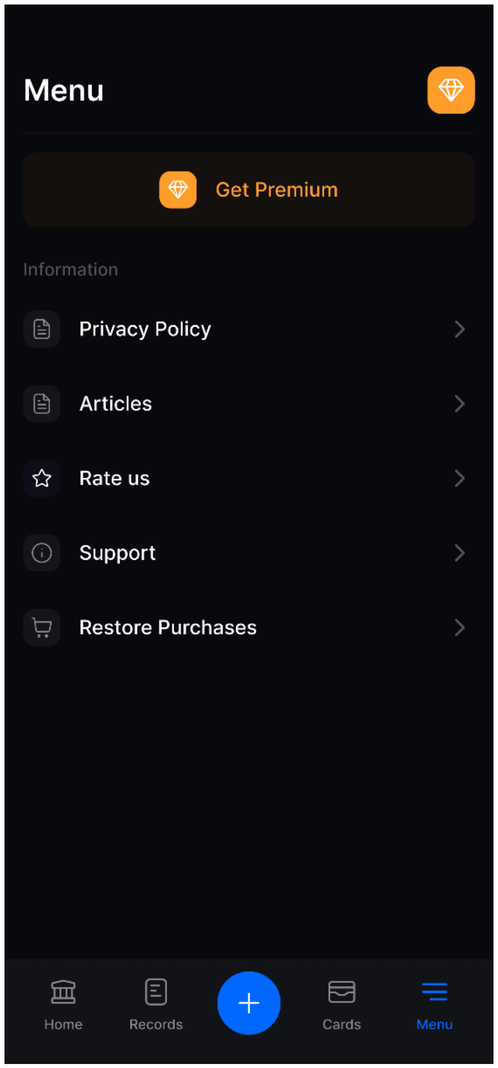
15 September 2023

Shopping	Credit Card	-₱25.56 31 Aug 2023
Salary	Cash	₱500.50 31 Aug 2023
Vacation	Credit Card	-₱25.56 31 Aug 2023


Edit Card

AMERICAN EXPRESS
1234 1234 1234 1234
Klei Chu
CVV: 123
Expire Date 12/32

Holder Name: Klei Chu
Card Number: 1234 1234 1234 1234
Expire Date: 12/32
CVV: 123
Card Color


Menu

Project Management

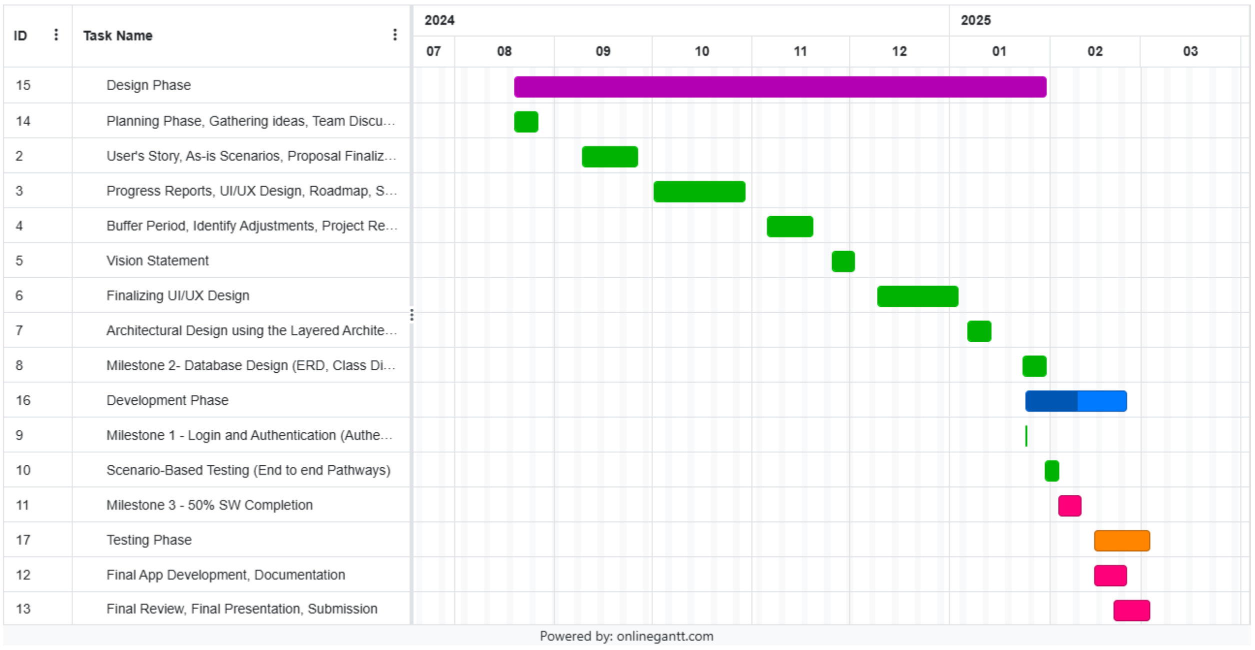
Agile Practices and Sprint Planning

1. **Dynamic Systems Development Method (DSDM)**
    - DSDM, or Dynamic Systems Development Method, was developed in the early 1990s by a consortium of companies in the UK, known as the DSDM Consortium.
    - Dynamic Systems Development Method (DSDM) is an agile project management framework that delivers business value through iterative development. It emphasizes user involvement, collaboration, and flexibility, making it essential for organizations seeking to enhance project delivery and adapt to change.
2. **Key DSDM Principles**
    - **Focus on Business Needs:** The project addresses the need for better personal finance tools, especially for helping young people keep track of their buying and saving habits.
    - **Deliver on Time:** Budge-IT's features, such as tracking expenses and setting goals, are being built according to a plan to make sure they are delivered on time.
    - **Collaborate:** Working with users on a regular basis will help us improve standards and make sure our solutions are still useful and easy to use.
    - **Never Compromise Quality:** Important tools like transaction labeling, real-time alerts, and interactive charts are built with high quality standards in mind.
    - **Build Incrementally from Firm Foundations:** Budge-IT is being built gradually, beginning with basic features that will be added to and made better based on comments from users.
    - **Develop Iteratively:** We're dedicated to making the app better all the time by updating it regularly based on what users say and how they like it.
    - **Communicate Continuously and Clearly:** We keep lines of communication open within the team and with partners to make sure we stay on the same page throughout the project.
3. **Project Phases**
    - **Feasibility:** We reviewed Budge-IT's sustainability and proved it fits well with the goal of promoting financial literacy.
    - **Foundations:** Defined basic objectives, gave team roles, and set project timelines.
    - **Exploration:** Using continuous testing, we've gathered user feedback on core features like cost categories and overspending alerts.

- **Engineering:** The focus here is on creating and merging features such as financial reports, transaction tracking, and goal-setting tools.
- **Deployment:** Budge-IT will be launched with user support to make training easy, gather comments, and watch performance.

4. **Roles in DSDM for Budge-IT**
    - **Project Manager:** Coordinates the project schedule, handles resources, and watches progress to ensure that Budge-IT stays on track with its goals and targets.
    - **Researcher:** Collects and analyzes data on user needs, spending habits, and current financial tools to guide the development of Budge-IT's features and functions.
    - **Programmer:** Develops the app's core functions, including transaction tracking, planning tools, and goal-setting features, ensuring smooth merging and usage.
    - **Documenter:** Maintains clear and organized records of project objectives, user feedback, and progress updates, building thorough paperwork for future reference and team alignment.
    - **Designer:** Creates user-friendly layouts and entertaining visual elements for Budge-IT, focused on layout, color schemes, and dynamic components that improve financial management for users.
5. **DSDM Techniques and Tools Applied**
    - **Timeboxing:** We set fixed times for the development of key features like cost tracking, planning, and debt management.
    - **Incremental Delivery:** Prioritizing key features for earlier versions, starting with tracking and progress reporting.
    - **Prototyping:** Creating wireframes and mockups to test UI ideas and gather feedback early.
    - **User Involvement:** Actively engaged end-users, especially during Exploration and Engineering, to ensure usefulness.
6. **Advantages and Considerations**
    - **Advantages:** Our user-centered method promises faster release of useful features, flexibility, and improved customer happiness.
    - **Considerations:** Balancing user interaction with a tight timeline, handling alert tiredness, and ensuring freedom without losing quality.

Gantt Chart or Project Timeline



ID	Task Name
15	Design Phase
14	Planning Phase, Gathering ideas, Team Discu...
2	User's Story, As-is Scenarios, Proposal Finaliz...
3	Progress Reports, UI/UX Design, Roadmap, S...
4	Buffer Period, Identify Adjustments, Project Re...
5	Vision Statement
6	Finalizing UI/UX Design
7	Architectural Design using the Layered Archite...
8	Milestone 2- Database Design (ERD, Class Di...
16	Development Phase
9	Milestone 1 - Login and Authentication (Authe...
10	Scenario-Based Testing (End to end Pathways)
11	Milestone 3 - 50% SW Completion
17	Testing Phase
12	Final App Development, Documentation
13	Final Review, Final Presentation, Submission

The Budge-IT (Personal Expense and Savings Optimizer) project's timeline and progress are shown in the Gantt chart, which highlights important stages, deadlines, and activities from July 2024 to March 2025. The project proceeds through planning, design, development, testing, and final deployment using an organized development methodology.

Project Phases & Timeline

1. **Design Phase (July 2024 – January 2025)**
    - Encompasses planning, UI/UX design, architectural design, and vision statement formulation.
    - Involves gathering user stories, defining system architecture, and database design.
2. **Development Phase (January 2025 – February 2025)**
    - Includes key milestones such as login authentication, partial software completion, and scenario-based testing.
3. **Testing Phase (February 2025)**
    - Ensures the system is functional through end-to-end testing.
4. **Final Development & Documentation (February - March 2025)**
    - Finalizes app development, documentation, and prepares for the final presentation.
5. **Final Review & Submission (March 2025)**
    - Concludes with project submission and presentation.

Key Milestones

A. **Milestone 1:** Implementation of login & authentication.
B. **Milestone 2:** Database design (ERD, Class Diagram).
C. **Milestone 3:** 50% software completion.
D. **Final Milestone:** Application deployment and final review.

Development Process

Coding Standards and Best Practices

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

Tools and Technologies Used

A. **Frontend:** Flutter
B. **Backend:** Java
C. **Database:** MySQL
D. **Figma:** For designing and prototyping user interfaces
E. **OWASP (ZAP):** For vulnerabilities testing

Implementation Details

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

Testing

Test Case Documentation

1. **Test Case ID:** #BITC001
   - **Test Scenario:** To register a user on Budge-IT
   - **Test Steps:**
     - ➢ The user navigates to Budge-IT.
     - ➢ The user clicks ‘Start Now’ button.
     - ➢ The user clicks ‘Sign Up’.
     - ➢ The user fills out the information.
     - ➢ The user clicks the ‘Next’ button.
     - ➢ The user enters their desired username.
     - ➢ The user enters their desired password and re-enter to confirm it.
     - ➢ The user clicks the ‘Sign Up’ button.

- **Prerequisites:** The user must be connected to the internet.
- **Browser:** Google Chrome v.133.0.6943.142 (Official Build) (64-bit) Device: Lenovo Laptop (ideapad) – Type 80TU
- **Test Data:** Legitimate name, age, gender, phone, birthdate, email, username, and password.
- **Expected/Intended Results:** Once name, age, gender, phone, birthdate, email, username, and password are entered, the web page redirects to the Home Screen, displaying the dashboard that contains the visual graphs of the user's current budget data.
- **Actual Results:** As Expected Results
- **Test Status – Pass/Fail:** Pass

2. **Test Case ID:** #BITC002
   - **Test Scenario:** To login a user on Budge-IT
   - **Test Steps:**
     - ➢ The user navigates to Budge-IT.
     - ➢ The user clicks 'Start Now' button.
     - ➢ The user enters a registered username and password in the respective fields.
     - ➢ The user clicks 'Log In' button.
   - **Prerequisites:** A registered email with a unique username and password.
   - **Browser:** Google Chrome v.133.0.6943.142 (Official Build) (64-bit) Device: Lenovo Laptop (ideapad) – Type 80TU
   - **Test Data:** Legitimate username and password.
   - **Expected/Intended Results:** Once username and password are entered, the web page redirects to the Home Screen, displaying the dashboard that contains the visual graphs of the user's current budget data.
   - **Actual Results:** As Expected Results
   - **Test Status – Pass/Fail:** Pass
3. **Test Case ID:** #BITC003
   - **Test Scenario:** To change the password of a user on Budge-IT
   - **Test Steps:**
     - ➢ The user navigates to Budge-IT.
     - ➢ The user clicks 'Start Now' button.
     - ➢ The user clicks 'Forgot Password?'.
     - ➢ The user enters the username of the forgotten password.
     - ➢ The user clicks 'Next'.
     - ➢ After system verification of an existing account, the user enters a new password and re-enters it to confirm.
     - ➢ The user clicks 'Confirm'.
   - **Prerequisites:** A registered user.

- **Browser:** Google Chrome v.133.0.6943.142 (Official Build) (64-bit) Device: Lenovo Laptop (ideapad) – Type 80TU
- **Test Data:** Existing account.
- **Expected/Intended Results:** Once the password has been changed, the web page redirects to the Login page to enter the username and new password.
- **Actual Results:** As Expected Results
- **Test Status – Pass/Fail:** Pass

4. **Test Case ID:** #BITC004
    - **Test Scenario:** To add cash amount to the Total Balance on Budge-IT home screen.
    - **Test Steps:**
        - ➢ The user navigates to Budge-IT.
        - ➢ The user clicks 'Start Now' button.
        - ➢ The user enters a registered username and password in the respective fields.
        - ➢ The user clicks 'Log In' button.
        - ➢ The user clicks the drop-down button on the right of the Total Balance.
        - ➢ The user selects 'Cash' and enters the amount of cash.
        - ➢ The user clicks 'Confirm' button to add the entered value.
    - **Prerequisites:** A registered user must be at the Home page of the application.
    - **Browser:** Google Chrome v.133.0.6943.142 (Official Build) (64-bit) Device: Lenovo Laptop (ideapad) – Type 80TU
    - **Test Data:** Cash value.
    - **Expected/Intended Results:** Once the value has been confirmed, the entered value will be added to the current Total Balance and display the latest Total Balance.
    - **Actual Results:** As Expected Results
    - **Test Status – Pass/Fail:** Pass
5. **Test Case ID:** #BITC005
    - **Test Scenario:** To view, add, or delete budget categories on All Budgets section of the home screen.
    - **Test Steps:**
        - ➢ The user navigates to Budge-IT.
        - ➢ The user clicks 'Start Now' button.
        - ➢ The user enters a registered username and password in the respective fields.
        - ➢ The user clicks 'Log In' button.
        - ➢ The user navigates to Budget section.
        - ➢ The user clicks 'All Budgets' button.
        - ➢ The user may view all of the present budget; or

    - ➢ The user may add a budget that can be categorized in various expense contexts, each having their own unique icons for quick identification by clicking on the plus sign button; or
    - ➢ The user may delete a budget by clicking the delete button.
  - **Prerequisites:** A registered user must be at the section of 'All Budgets' of the application. The user must have an existing budget to delete a budget category.
  - **Browser:** Google Chrome v.133.0.6943.142 (Official Build) (64-bit) Device: Lenovo Laptop (ideapad) – Type 80TU
  - **Test Data:** Information and values of the budget.
  - **Expected/Intended Results:** The user will be able to view, add, or delete the budget.
  - **Actual Results:** Result may vary depending on the present budget categories. The user may be unable to delete a budget category if none are present.
  - **Test Status – Pass/Fail:** Pass
6. **Test Case ID:** #BITC006
  - **Test Scenario:** To view Statistics on Budge-IT home screen categories.
  - **Test Steps:**
    - ➢ The user navigates to Budge-IT.
    - ➢ The user clicks 'Start Now' button.
    - ➢ The user enters a registered username and password in the respective fields.
    - ➢ The user clicks 'Log In' button.
    - ➢ The user navigates to Categories section by scrolling down.
    - ➢ The user clicks 'Statistics'.
  - **Prerequisites:** A registered user must be at the bottom of the Home page on Categories section of the application.
  - **Browser:** Google Chrome v.133.0.6943.142 (Official Build) (64-bit) Device: Lenovo Laptop (ideapad) – Type 80TU
  - **Test Data:** Statistical values of the budget.
  - **Expected/Intended Results:** The user will be able to view the statistics represented by the data of their budget.
  - **Actual Results:** Results may vary according to the user's current budget.
  - **Test Status – Pass/Fail:** Pass
7. **Test Case ID:** #BITC007
  - **Test Scenario:** To view, add, or delete income records on the Record tab under the tabs bar of the application.
  - **Test Steps:**
    - ➢ The user navigates to Budge-IT.
    - ➢ The user clicks 'Start Now' button.
    - ➢ The user enters a registered username and password in the respective fields.

- ➢ The user clicks ‘Log In’ button.
  - ➢ The user clicks the Records tab below.
  - ➢ The user may view all of the income records present; or
  - ➢ The user may add an income record and choose a category that suits their income’s origin; or
  - ➢ The user may delete an income record by clicking the delete button.
- **Prerequisites:** A registered user must be at the ‘Records’ tab of the application under the tabs bar. The user must have an existing income record to delete an income record.
- **Browser:** Google Chrome v.133.0.6943.142 (Official Build) (64-bit) Device: Lenovo Laptop (ideapad) – Type 80TU
- **Test Data:** Information and values of the income.
- **Expected/Intended Results:** The user will be able to view, add, or delete an income record.
- **Actual Results:** Result may vary depending on the present income record. The user may be unable to delete an income record if none are present.
- **Test Status – Pass/Fail:** Pass

8. **Test Case ID:** #BITC008
   - **Test Scenario:** To view, add, or delete wallet cards on the Cards tab under the tabs bar of the application.
   - **Test Steps:**
     - ➢ The user navigates to Budge-IT.
     - ➢ The user clicks ‘Start Now’ button.
     - ➢ The user enters a registered username and password in the respective fields.
     - ➢ The user clicks ‘Log In’ button.
     - ➢ The user clicks the Cards tab below.
     - ➢ The user may view all of the wallet cards available and hide or unhide their card information; or
     - ➢ The user may add a wallet card and choose which card they want to add; or
     - ➢ The user may delete a wallet card by clicking the delete button.
   - **Prerequisites:** A registered user must be at the ‘Cards’ tab of the application under the tabs bar. The user must have a legitimate valid card to add a wallet card. The user must have an existing wallet card to delete a wallet card.
   - **Browser:** Google Chrome v.133.0.6943.142 (Official Build) (64-bit) Device: Lenovo Laptop (ideapad) – Type 80TU
   - **Test Data:** Information of a legitimate valid card.
   - **Expected/Intended Results:** The user will be able to view, add, or delete a wallet card.

- **Actual Results:** Result may vary depending on the wallet card available. The user may be unable to add a card if it is illegitimate or invalid. The user may be unable to delete a wallet card if none are present.
- **Test Status – Pass/Fail:** Pass

9. **Test Case:** #BITC009
   - **Test Scenario:** To access the Premium subscription page on Budge-IT.
   - **Test Steps:**
     - ➢ The user navigates to Budge-IT.
     - ➢ The user clicks ‘Start Now’ button.
     - ➢ The user enters a registered username and password in the respective fields.
     - ➢ The user clicks ‘Log In’ button.
     - ➢ The user clicks on the ‘Menu’ tab on the bottom navigation bar.
     - ➢ The user clicks the ‘Get Premium’ button.
   - **Prerequisites:** The user must be logged in.
   - **Browser:** Google Chrome v.133.0.6943.142 (Official Build) (64-bit) Device: Lenovo Laptop (ideapad) – Type 80TU
   - **Test Data:** A registered account (optional: payment details if purchasing).
   - **Expected/Intended Results:** The user is redirected to a Premium subscription page displaying the available plans and benefits.
   - **Actual Results:** As Expected Results
   - **Test Status – Pass/Fail:** Pass
10. **Test Case:** #BITC010
    - **Test Scenario:** To access the Privacy Policy page on Budge-IT.
    - **Test Steps:**
      - ➢ The user navigates to Budge-IT.
      - ➢ The user clicks ‘Start Now’ button.
      - ➢ The user enters a registered username and password in the respective fields.
      - ➢ The user clicks ‘Log In’ button.
      - ➢ The user clicks on the ‘Menu’ tab on the bottom navigation bar.
      - ➢ The user clicks the ‘Privacy Policy’ button.
    - **Prerequisites:** The user must be logged in.
    - **Browser:** Google Chrome v.133.0.6943.142 (Official Build) (64-bit) Device: Lenovo Laptop (ideapad) – Type 80TU
    - **Test Data:** None.
    - **Expected/Intended Results:** The user is redirected to a page displaying the Privacy Policy.
    - **Actual Results:** As Expected Results
    - **Test Status – Pass/Fail:** Pass

11. **Test Case:** #BITC011
    - **Test Scenario:** To access the Articles section on Budge-IT.
    - **Test Steps:**
        - ➢ The user navigates to Budge-IT.
        - ➢ The user clicks ‘Start Now’ button.
        - ➢ The user enters a registered username and password in the respective fields.
        - ➢ The user clicks ‘Log In’ button.
        - ➢ The user clicks on the ‘Menu’ tab on the bottom navigation bar.
        - ➢ The user clicks the ‘Articles’ button.
    - **Prerequisites:** The user must be logged in.
    - **Browser:** Google Chrome v.133.0.6943.142 (Official Build) (64-bit) Device: Lenovo Laptop (ideapad) – Type 80TU
    - **Test Data:** None.
    - **Expected/Intended Results:** The user is redirected to an Articles page containing tips and tricks on budget management.
    - **Actual Results:** As Expected Results
    - **Test Status – Pass/Fail:** Pass
12. **Test Case:** #BITC012
    - **Test Scenario:** To access the ‘Rate Us’ feature on Budge-IT.
    - **Test Steps:**
        - ➢ The user navigates to Budge-IT.
        - ➢ The user clicks ‘Start Now’ button.
        - ➢ The user enters a registered username and password in the respective fields.
        - ➢ The user clicks ‘Log In’ button.
        - ➢ The user clicks on the ‘Menu’ tab on the bottom navigation bar.
        - ➢ The user clicks the ‘Rate Us’ button.
    - **Prerequisites:** The user must be logged in.
    - **Browser:** Google Chrome v.133.0.6943.142 (Official Build) (64-bit) Device: Lenovo Laptop (ideapad) – Type 80TU
    - **Test Data:** None.
    - **Expected/Intended Results:** The user is redirected to the app store or a feedback form to rate and review the app.
    - **Actual Results:** As Expected Results
    - **Test Status – Pass/Fail:** Pass

13. **Test Case:** #BITC013
    - **Test Scenario:** To access the Support section on Budge-IT.
    - **Test Steps:**
        - ➢ The user navigates to Budge-IT.
        - ➢ The user clicks 'Start Now' button.
        - ➢ The user enters a registered username and password in the respective fields.
        - ➢ The user clicks 'Log In' button.
        - ➢ The user clicks on the 'Menu' tab on the bottom navigation bar.
        - ➢ The user clicks the 'Support' button.
    - **Prerequisites:** The user must be logged in.
    - **Browser:** Google Chrome v.133.0.6943.142 (Official Build) (64-bit) Device: Lenovo Laptop (ideapad) – Type 80TU
    - **Test Data:** None.
    - **Expected/Intended Results:** The user is redirected to the Support page, which may contain FAQs or a contact form for assistance.
    - **Actual Results:** As Expected Results
    - **Test Status – Pass/Fail:** Pass
14. **Test Case:** #BITC014
    - **Test Scenario:** To restore purchases on Budge-IT
    - **Test Steps:**
        - ➢ The user navigates to Budge-IT.
        - ➢ The user clicks 'Start Now' button.
        - ➢ The user enters a registered username and password in the respective fields.
        - ➢ The user clicks 'Log In' button.
        - ➢ The user clicks on the 'Menu' tab on the bottom navigation bar.
        - ➢ The user clicks the 'Restore Purchases' button.
    - **Prerequisites:** The user must have made a previous purchase on Budge-IT.
    - **Browser:** Google Chrome v.133.0.6943.142 (Official Build) (64-bit) Device: Lenovo Laptop (ideapad) – Type 80TU
    - **Test Data:** Previous transaction data associated with the user's account.
    - **Expected/Intended Results:** The app verifies the user's purchase history and restores any premium features previously bought.
    - **Actual Results:** As Expected Results
    - **Test Status – Pass/Fail:** Pass

Bug Tracking and Resolution

Overview

Budge-IT's functionality and performance was hindered by a number of problems that arose during development. To guarantee a reliable and completely working application, efficient bug tracking and resolution procedures were put in place. The main problems encountered, their origins, and the associated fixes are described in depth in the sections that follow.

Issues Encountered and Resolutions

1. **Could Not Connect to Database**
   Issue: The application failed to establish a connection with the database, preventing data retrieval and storage.
   A. **Possible Causes:**
      - Incorrect database credentials (username, password, host, port, or database name)
      - Database server not running or misconfigured
      - Network connectivity issues

   B. **Resolution:**
      - Verified and corrected the database credentials in the configuration file.
      - Ensured the database server was running and accepting connections.
      - Allowed necessary ports through the firewall.
      - Implemented error handling and retry mechanisms for better stability.
2. **Could Not Append Values to the Database**
   Issues: Data could not be inserted into the database tables.
   A. **Possible Causes:**
      - Incorrect SQL queries or syntax errors
      - Missing or incorrect database schema constraints
      - Insufficient database user permissions

   B. **Resolution:**
      - Reviewed and corrected SQL queries for proper syntax and structure.
      - Modified table schemas to accommodate the expected data.
      - Granted necessary permissions to the database user to allow inserts.

3. **Could Not Fetch Data from the Database**
   Issues: The application was unable to retrieve data from the database.
   A. **Possible Causes:**
      - Incorrect SQL SELECT statements
      - Connection timeout or disconnection issues
      - Database schema mismatches

   B. **Resolution:**
      - Checked and optimized SQL queries for accurate data retrieval.
      - Implemented query debugging and logging mechanisms.
      - Ensured the database schema matched the application's expected structure.
4. **Missing Dependencies and Packages in Flutter**
   Issues: Several dependencies required for Flutter were missing, causing build failures and runtime errors.
   A. **Possible Causes:**
      - Dependencies not installed due to incorrect pubspec.yaml configurations
      - Flutter SDK not updated

   B. **Resolutions:**
      - Ran flutter pub get to fetch all dependencies.
      - Ensured pubspec.yaml contained correct package versions.
      - Updated the Flutter SDK using flutter upgrade.
      - Cleared and reinstalled dependencies using flutter clean && flutter pub get.
5. **Could Not Connect to Firebase**
   Issues: Firebase services, such as authentication and database, were not accessible from the application.
   A. **Possible Causes:**
      - Incorrect Firebase project configuration
      - Missing or incorrect google-services.json or GoogleService-Info.plist
      - Firebase API keys not correctly set up

   B. **Resolution:**
      - Verified Firebase project setup in the Firebase console.
      - Downloaded and added the correct google-services.json (Android) and GoogleService-Info.plist (iOS) files.
      - Ensured API keys were properly configured in the Firebase dashboard.
      - Implemented proper authentication and security rules for accessing Firebase services.

6. **Some Dependencies Were Not Installed (Node.js, Java SDK)**
   Issues: The application required certain dependencies like Node.js and Java SDK, which were not installed, leading to errors in development and builds.
   A. **Possible Causes:**
      - Missing installations or incorrect versions
      - Environment variables not configured correctly
   B. **Resolution:**
      - Installed the necessary dependencies using official sources.
      - Configured system PATH variables for Node.js and Java SDK.
      - Verified installations using node -v and java -version.
7. **Some Packages Were Installed Incorrectly**
   Issues: Some required packages were installed incorrectly, leading to unexpected behavior in the application.
   A. **Possible Causes:**
      - Incorrect package versions
      - Conflicting dependencies
      - Corrupt installations
   B. **Resolution:**
      - Removed and reinstalled problematic packages.
      - Used correct version constraints in pubspec.yaml.
      - Ran flutter clean && flutter pub get to refresh dependencies.
      - Checked for dependency conflicts and resolved version mismatches.

User Guide for the System

1. **Getting Started**
   1.1 **Installation and Setup**
      - Download the Budge-IT app from the official store (Google Play / App Store).
      - Open the app and sign up using an email address or social media login.
      - Verify your email address and log in.

   1.2 **Linking Your Bank Account**
      - Navigate to the "Accounts" section in the main menu.
      - Select "Add Bank Account" and choose from the list of supported banks.
      - Enter your banking credentials to securely link your account.
      - Once connected, Budge-IT will automatically fetch transaction data.

2. **Setting Up Your Budget**

   2.1 **Defining Your Income and Expenses**
   - Go to "Budget Settings" in the dashboard.
   - Enter your monthly income from all sources (e.g., salary, allowances, side jobs).
   - List down recurring expenses such as rent, utilities, groceries, and subscriptions.

   2.2 **Creating Budget Categories**
   - Navigate to "Budget Categories" in the settings menu.
   - Add expense categories such as Entertainment, Dining, Transport, and Savings.
   - Set spending limits for each category based on your financial goals.
   - Save the settings to enable real-time expense tracking.

3. **Tracking Expenses and Insights**

   3.1 **Logging Expenses Manually**
   - Click on "Add Expense" from the dashboard.
   - Enter the expense amount, select a category, and add a short description.
   - Save the entry to update your budget progress.

   3.2 **Automated Expense Tracking**
   - If a bank account is linked, Budge-IT will fetch transactions automatically.
   - Categorized transactions will be displayed in the dashboard for review.
   - Manually adjust categories if needed for better accuracy.

   3.3 **Visual Spending Insights**
   - Open "Spending Trends" to see graphical representations of your expenses.
   - Identify top spending areas and compare them with your budget limits.
   - Adjust spending habits based on insights provided.

4. **Managing Overspending and Alerts**

   4.1 **Setting Up Spending Alerts**
   - Navigate to "Notifications & Alerts" in settings.
   - Enable real-time alerts for when spending reaches 80% of a category budget.
   - Customize notifications for daily, weekly, or monthly spending summaries.

   4.2 **Responding to Overspending Warnings**
   - If you receive an Overspending Alert, check the spending breakdown.
   - Identify frequent purchases that contribute to excess spending.
   - Reduce unnecessary expenses and revise your budget accordingly.

5. **Achieving Financial Goals**
   5.1 **Setting Savings Goals**
   - Navigate to "Savings Goals" in the main menu.
   - Enter the amount you want to save and set a deadline.
   - Budge-IT will track progress and suggest cost-cutting strategies.

   5.2 **Optimizing Spending**
   - Reduce impulse spending by tracking non-essential purchases.
   - Use discounts, student offers, and meal prepping to save money.
   - Regularly review financial insights to make informed decisions.
6. **Reviewing Financial Summary and Next Steps**
   6.1 **Generating Monthly Reports**
   - Open "Financial Summary" at the end of each month.
   - View categorized expenses, savings progress, and budget deviations.
   - Export reports for personal review or tax purposes.

   6.2 **Transferring Extra Savings**
   - If you have surplus funds, allocate them to your savings goal.
   - Adjust budget categories based on new financial priorities.
   - Plan future expenses to maintain financial stability.

Conclusion

Summary of Project Outcomes

The Budge-IT project was successfully created as a personal savings and expense optimizer designed to assist users in efficiently managing their money, especially Gen Z students. Through an easy-to-use user interface, the system integrated real-time expenditure data, savings goal monitoring, and essential budgeting features. The project's main results are as follows:

- **User-Centric Budgeting Features:** The application provided an easy-to-use budgeting system that allowed users to track income, categorize expenses, and set financial goals.
- **Automated Expense Tracking:** Users could link their bank accounts to retrieve transactions automatically, ensuring accuracy in financial data.
- **Real-Time Notifications:** Overspending alerts helped users avoid financial pitfalls by providing timely reminders about their budgets.
- **Interactive Dashboards:** Graphical visualizations of spending habits enabled users to make informed financial decisions.

- **Successful Agile Implementation:** The project followed the Dynamic Systems Development Method (DSDM) to ensure iterative improvements and user feedback integration.
- **Effective Testing and Bug Resolution:** The application underwent rigorous scenario-based and end-to-end testing, ensuring stability and reliability.
- **Scalability Considerations:** While the current system supports core budgeting functionalities, the foundational design allows for future expansions, such as AI-based financial recommendations and multi-user support.

Challenges Faced and Lessons Learned

Challenges Faced

- **Database Connectivity Issues:** Establishing a stable connection between the app and the database caused delays due to incorrect configurations.
- **Data Handling Errors:** Initial difficulties in appending and retrieving data led to inconsistencies in expense tracking.
- **Missing Dependencies in Flutter:** Several required packages were missing or incorrectly installed, causing build failures.
- **Firebase Integration Issues:** Authentication and real-time database syncing required multiple troubleshooting iterations.
- **User Interface Optimization:** Balancing a feature-rich interface while maintaining simplicity was a key design challenge.
- **Overspending Alert Sensitivity:** Users initially found frequent notifications overwhelming, requiring fine-tuning to avoid alert fatigue.
- **Limited Real-Time Banking Integration:** Due to API restrictions, direct real-time banking transactions were not feasible in the current version.

Lessons Learned

- **Thorough Dependency Management is Crucial:** Ensuring all necessary dependencies (Flutter, Firebase, Node.js, Java SDK) are properly installed before development minimizes integration issues.
- **Comprehensive Testing Enhances Stability:** Scenario-based testing and iterative debugging significantly improved system reliability.
- **User Feedback is Invaluable:** Early usability testing helped refine UI elements and adjust the frequency of notifications.
- **Documentation Facilitates Collaboration:** Maintaining clear technical documentation streamlined bug tracking and resolution processes.
- **Incremental Development Yields Better Results:** The DSDM approach ensured that features were built progressively, reducing last-minute development pressure.

Recommendations for Future Work

To further enhance Budge-IT, several improvements and additional features are recommended for future development:

1. **Bank API Integration:** Implementing direct integration with banking APIs would enable real-time expense tracking without requiring manual entry.
2. **AI-Based Financial Recommendations:** Machine learning models could analyze user spending patterns and provide personalized budgeting suggestions.
3. **Multi-User and Shared Budgeting Features:** Allowing multiple users to collaborate on shared budgets (e.g., roommates, couples) would expand the app's use cases.
4. **Advanced Security Measures:** Strengthening security with multi-factor authentication and end-to-end encryption for financial data.
5. **Gamification Elements:** Adding reward-based incentives for meeting savings goals could enhance user engagement and motivation.
6. **Cross-Platform Availability:** Expanding beyond mobile to include a dedicated web application for increased accessibility.
7. **Localization and Currency Support:** Implementing multi-currency support and localized financial guidelines for international users.

References

Myers, G. J., Sandler, C., & Badgett, T. (2012). *The Art of Software Testing*. John Wiley & Sons. Retrieved from 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*. Retrieved from https://www.researchgate.net/publication/323572478_The_Impact_of_Agile_Methodology_DSDM_on_Software_Project_Management.

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

Browser Stack. (2025). *How to write Test Cases in Software Testing? (with Format & Example)*. BrowserStack. Retrieved from https://www.browserstack.com/guide/how-to-write-test-cases

Appendices

Appendix A: Meeting Minutes

Meeting #1: Project Coding Session

- **Date:** February 27, 2025
- **Attendees:** Aldrin Galvez, Ronaldo D. Rico Jr.
- **Agenda:** Code session continuation and debugging.
- **Assigned Team Roles:**
  - **Backend Developer:** Aldrin Galvez
  - **Documentation:** Ronaldo D. Rico Jr.

Meeting #2: Project Visuals Development Presentation

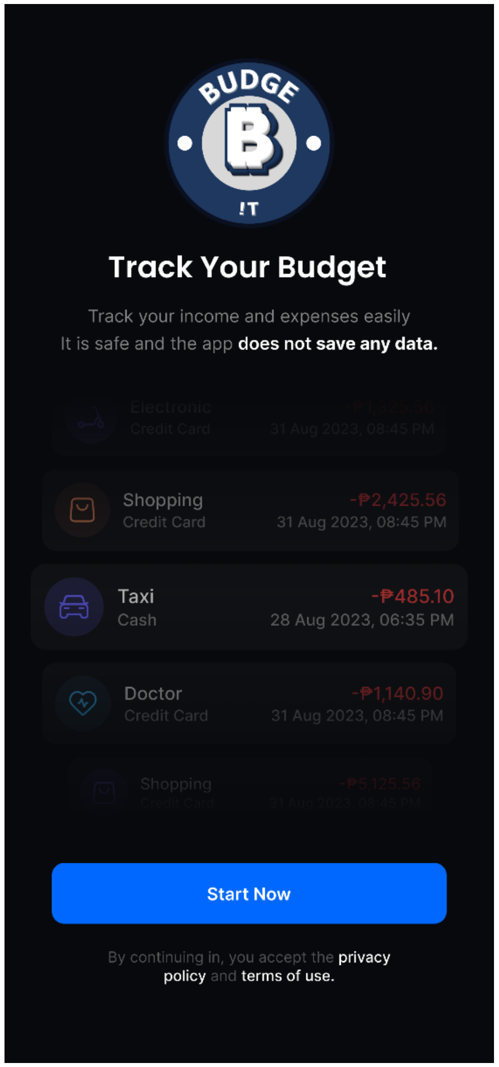
- **Date:** March 1, 2025
- **Attendees:** Kamira Allison F. Pagulayan, Beatriz U. Asuncion, Sherdon Rappah Esquejo
- **Agenda:** Create project presentation.
- **Assigned Team Roles:**
  - **Leader:** Kamira Allison F. Pagulayan
  - **Designer:** Beatriz U. Asuncion
  - **Researcher:** Sherdon Rappah Esquejo

Meeting #3: Project Coding Session

- **Date:** March 2, 2025
- **Attendees:** Beatriz U. Asuncion, Aldrin Galvez, Ronaldo D. Rico Jr.
- **Agenda:** Finalization of backend
- **Assigned Team Roles:**
  - **Designer:** Beatriz U. Asuncion
  - **Backend Developer:** Aldrin Galvez
  - **Documentation:** Ronaldo D. Rico Jr.

Appendix B: Screenshots

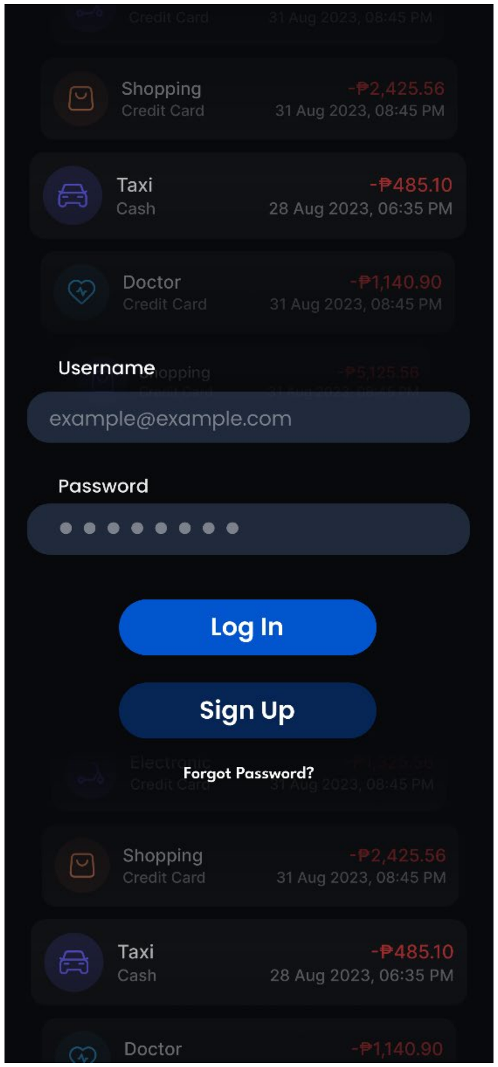
Onboarding, Login, and Dashboard Page

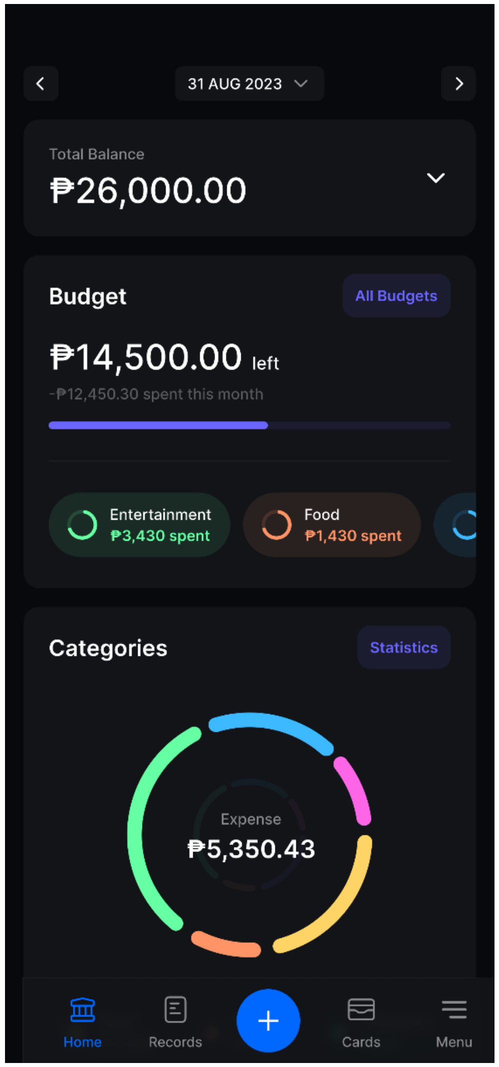


Track Your Budget

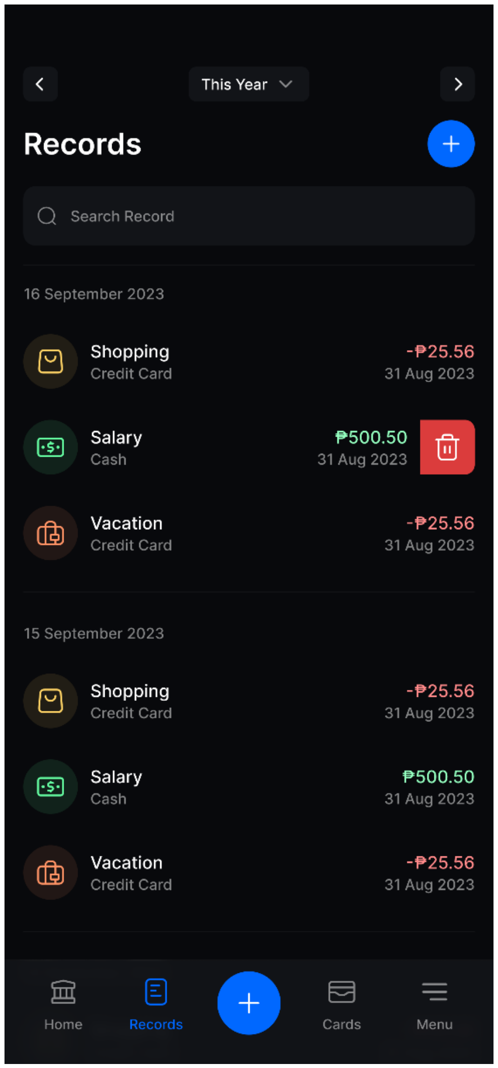
Track your income and expenses easily
It is safe and the app **does not save any data.**

By continuing in, you accept the **privacy policy** and **terms of use.**





Records, Cards, and Menu Page


This Year

Records

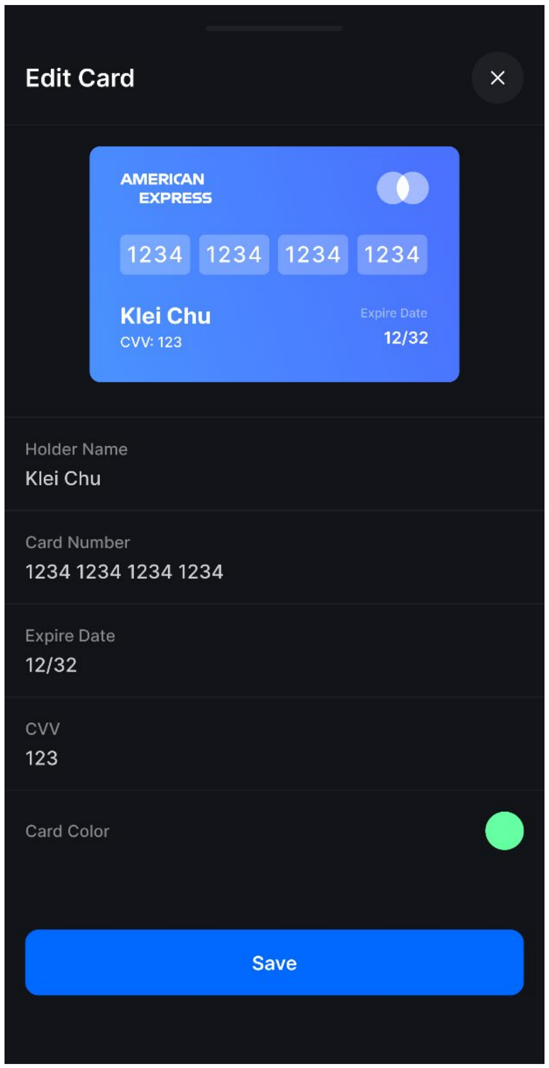
Search Record

16 September 2023

Shopping	Credit Card	-₱25.56 31 Aug 2023
Salary	Cash	₱500.50 31 Aug 2023
Vacation	Credit Card	-₱25.56 31 Aug 2023

15 September 2023

Shopping	Credit Card	-₱25.56 31 Aug 2023
Salary	Cash	₱500.50 31 Aug 2023
Vacation	Credit Card	-₱25.56 31 Aug 2023


Edit Card

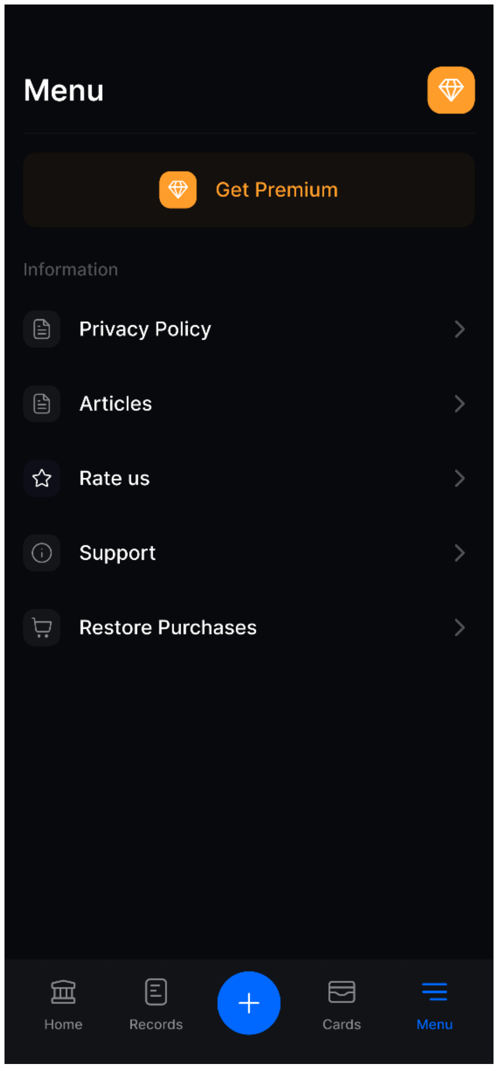
Holder Name: Klei Chu

Card Number: 1234 1234 1234 1234

Expire Date: 12/32

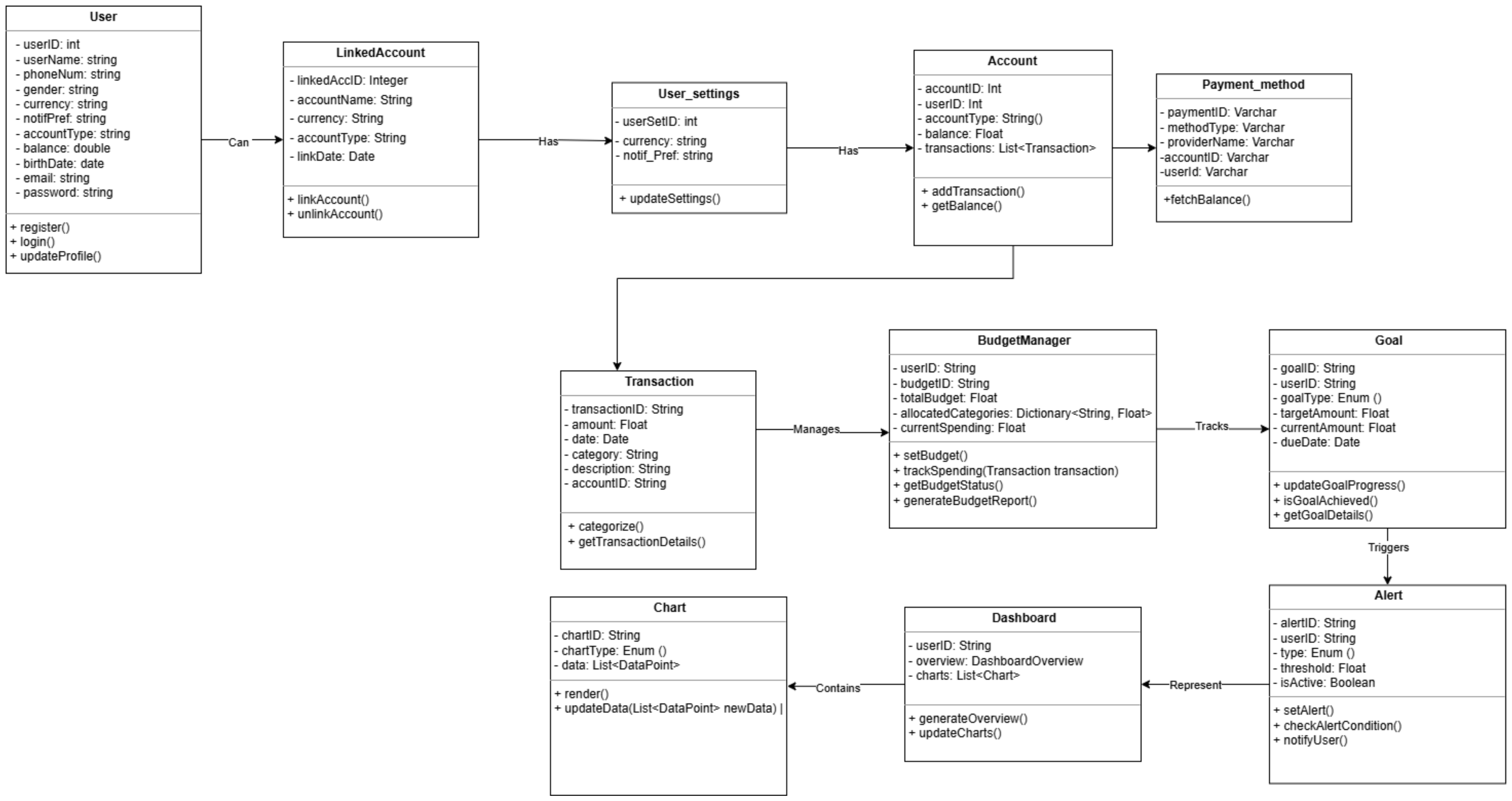
CVV: 123

Card Color

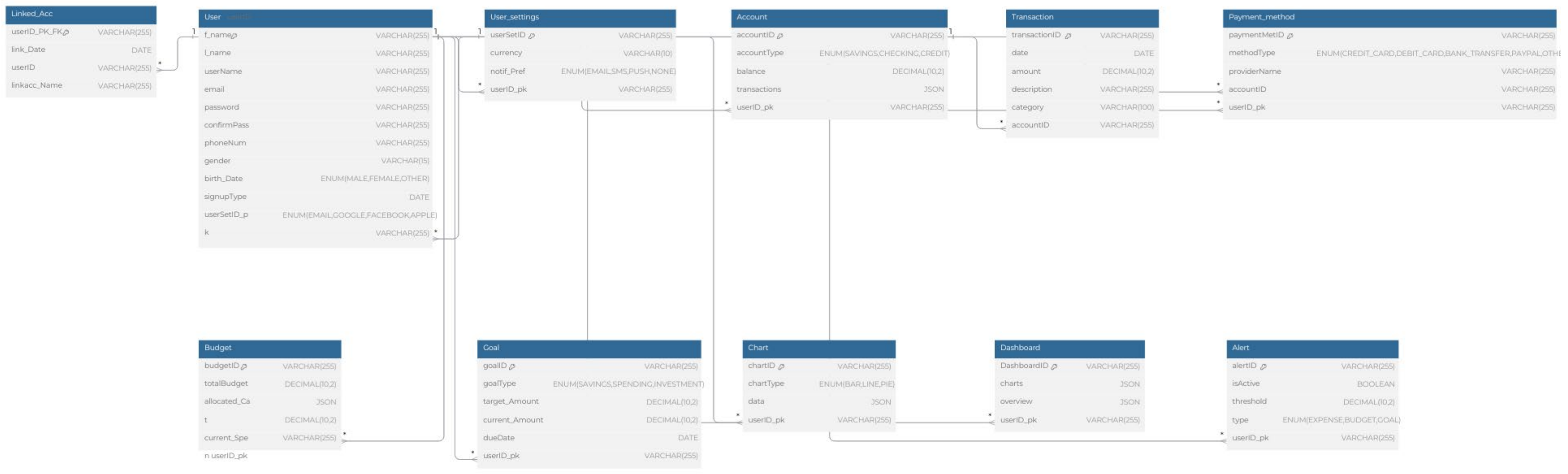

Menu

Appendix C: Other Supporting Documents

Class Diagram



Database Schema


Linked_Acc
userID_PK_FK VARCHAR(255)
link_Date DATE
userID VARCHAR(255)
linkacc_Name VARCHAR(255)
User
f_name VARCHAR(255)
l_name VARCHAR(255)
userName VARCHAR(255)
email VARCHAR(255)
password VARCHAR(255)
confirmPass VARCHAR(255)
phoneNum VARCHAR(255)
gender VARCHAR(15)
birth_Date ENUM(MALE,FEMALE,OTHER)
signupType DATE
userSetID_p ENUM(EMAIL,GOOGLE,FACEBOOK,APPLE)
k VARCHAR(255)
User_settings
userSetID VARCHAR(255)
currency VARCHAR(10)
notif_Pref ENUM(EMAIL,SMS,PUSH,NONE)
userID_pk VARCHAR(255)
Account
accountID VARCHAR(255)
accountType ENUM(SAVINGS,CHECKING,CREDIT)
balance DECIMAL(10,2)
transactions JSON
userID_pk VARCHAR(255)
Transaction
transactionID VARCHAR(255)
date DATE
amount DECIMAL(10,2)
description VARCHAR(255)
category VARCHAR(100)
accountID VARCHAR(255)
Payment_method
paymentMetID VARCHAR(255)
methodType ENUM(CREDIT_CARD,DEBIT_CARD,BANK_TRANSFER,PAYPAL,OTHE
providerName VARCHAR(255)
accountID VARCHAR(255)
userID_pk VARCHAR(255)
Budget
budgetID VARCHAR(255)
totalBudget DECIMAL(10,2)
allocated_Ca JSON
t DECIMAL(10,2)
current_Spe VARCHAR(255)
n userID_pk
Goal
goalID VARCHAR(255)
goalType ENUM(SAVINGS,SPENDING,INVESTMENT)
target_Amount DECIMAL(10,2)
current_Amount DECIMAL(10,2)
dueDate DATE
userID_pk VARCHAR(255)
Chart
chartID VARCHAR(255)
chartType ENUM(BAR,LINE,PIE)
data JSON
userID_pk VARCHAR(255)
Dashboard
DashboardID VARCHAR(255)
charts JSON
overview JSON
userID_pk VARCHAR(255)
Alert
alertID VARCHAR(255)
isActive BOOLEAN
threshold DECIMAL(10,2)
type ENUM(EXPENSE,BUDGET,GOAL)
userID_pk VARCHAR(255)
1
*