DAYANANDA SAGAR COLLEGE OF ENGINEERING

(An Autonomous Institute affiliated to Visvesvaraya Technological University (VTU), Belagavi,
Approved by AICTE and UGC, Accredited by NAAC with 'A' grade & ISO 9001-2015 Certified Institution)

Department of Information Science and Engineering

(Accredited by NBA Tier 1: 2022-2025)

SOFTWARE ENGINEERING AND SOFTWARE TESTING (IPCC22IS63)

Academic Year 2024-25

ALTERNATE ASSESSMENT TECHNIQUE

DESIGN DOCUMENT

I. **Group No: C03**

II. **Team Members:**

Name	USN
Shreenivas Nayakawadi	1DS22IS143
Shreesha A	1DS22IS144
Siddeshwar M	1DS22IS155
Prashant S N	1DS23IS415

III. **Project Title:** Budget Management System

SIGNATURE OF FACULTY

1. INTRODUCTION

The Personal Budget Management System is designed to provide a comprehensive, user-friendly financial management solution that streamlines expense tracking, budget planning, and financial goal achievement in today's dynamic economic environment. With the increasing complexity of personal finances, traditional budgeting methods often prove inadequate due to manual tracking, limited visibility into spending patterns, and inefficient expense categorization. This system addresses these challenges by leveraging modern technologies such as real-time transaction tracking, automated categorization, and intelligent budget analytics. At the core of the platform is an intuitive dashboard that enables users to instantly monitor their financial health, track expenses across multiple categories, and receive intelligent alerts for budget overruns. The platform includes a web-based interface for users to manage their budgets, set financial goals, and analyze spending patterns, while also offering detailed transaction management for precise expense tracking and categorization. This design document outlines the architecture, system components, user interfaces, data models, and other essential elements that form the foundation of the platform's functionality. It serves as a comprehensive guide for developers, testers, project stakeholders, and future maintainers, ensuring that the system is implemented with precision, scalability, and reliability to achieve its primary goal of empowering users to make informed financial decisions and achieve their monetary objectives.

2. SYSTEM OVERVIEW AND ARCHITECTURAL DESIGN

The architecture of the Personal Budget Management System follows a modern, scalable, and maintainable design that supports real-time financial tracking, secure user interactions, and efficient data management. The system implements a three-tier architecture consisting of the presentation layer, application logic layer, and data layer. The presentation layer features a responsive React-based web application, providing intuitive interfaces for users to manage budgets, track transactions, and visualize financial analytics. The application layer houses core services including user authentication, budget management, transaction processing, and financial analytics. These services are built using Express.js and are deployed on a secure server infrastructure to ensure reliable performance and data protection. The application's core functionality is structured through RESTful APIs, maintaining a clear separation of concerns and enabling seamless integration of future features. The data layer utilizes MySQL through Prisma ORM for storing user profiles, budget configurations, transaction records, and financial analytics data. The architecture incorporates real-time updates for transaction tracking and budget monitoring, ensuring users always have access to their current financial status.

System Architecture diagram

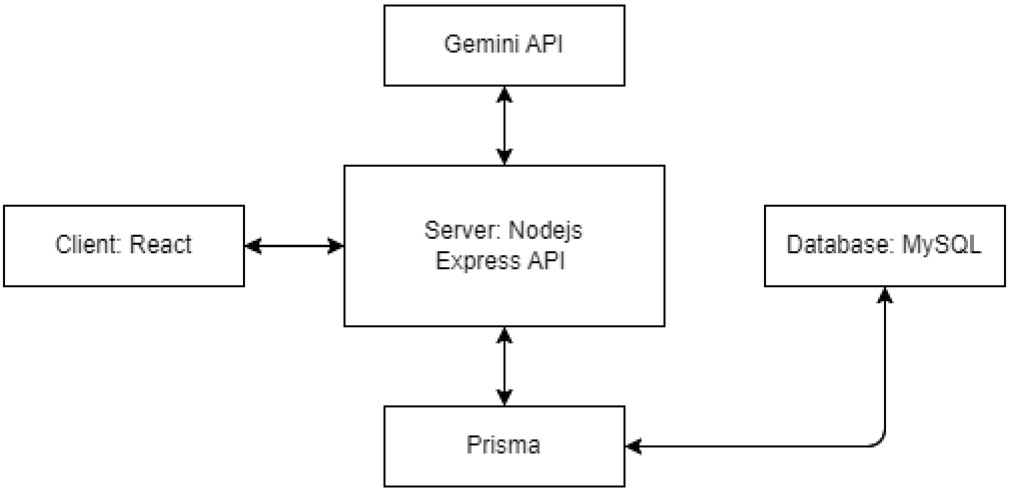


Figure 1: System Architecture of Budget Management System

3. SYSTEM MODELS

3.1 Data Flow Diagram (DFD)

The Budget Management System is an integrated platform designed to streamline personal financial management by offering the following key functionalities:

1. User Authentication – Secures system access through a login system, user authentication, and token generation.
2. Budget Management – Allows users to create and categorize budgets using a structured budget form, with the data stored for future reference.
3. Transaction Processing – Supports both automated receipt scanning (via image processing) and manual transaction entry to collect accurate financial data.

4. AI Analysis Engine – Processes combined budget and transaction data to generate insightful financial analysis.

5. Visualization System – Presents processed insights through intuitive visual displays, enabling users to understand and manage their financial activities effectively.

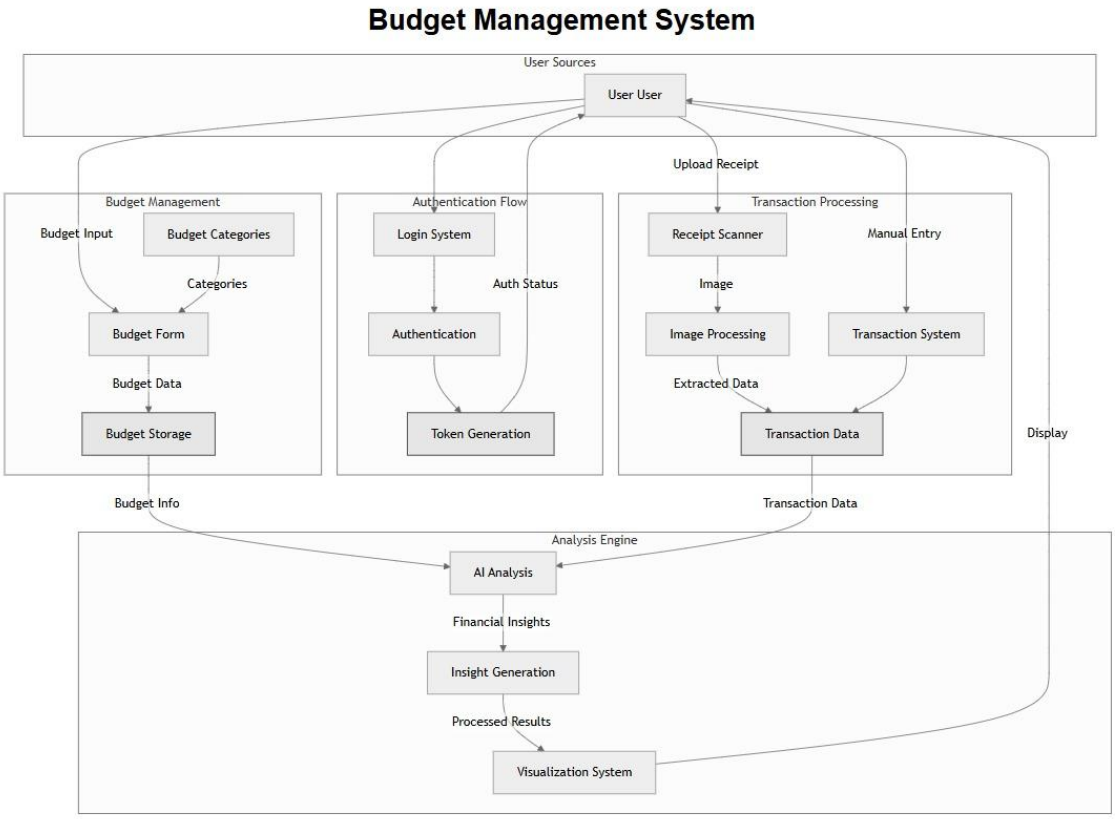


Figure 2: Data flow diagram of Budget Management System

3.2 Use Case Diagram

The User is the primary actor in the system and can interact with the Budget Management System through the following use cases:

- Register – Allows new users to create an account in the system.
- Login – Authenticates a registered user to access system features.
- Logout – Terminates the current user session securely.
- Create Budget – Enables users to define a new budget plan.
- Update Budget – Allows users to make changes to an existing budget.

- Delete Budget – Permits users to remove an unwanted or outdated budget entry.
- Fetch All Budgets – Retrieves a list of all budgets created by the user.
- Create Transaction – Lets users add a new transaction manually.
- Update Transaction – Enables editing of existing transaction records.
- Delete Transaction – Removes a specific transaction from the system.
- Fetch All Transactions – Retrieves all transaction data for review.
- Scan Receipts – Allows users to upload receipts for automated transaction data extraction using image processing.
- Get AI Insights – Provides users with financial insights generated through AI analysis of budget and transaction data.

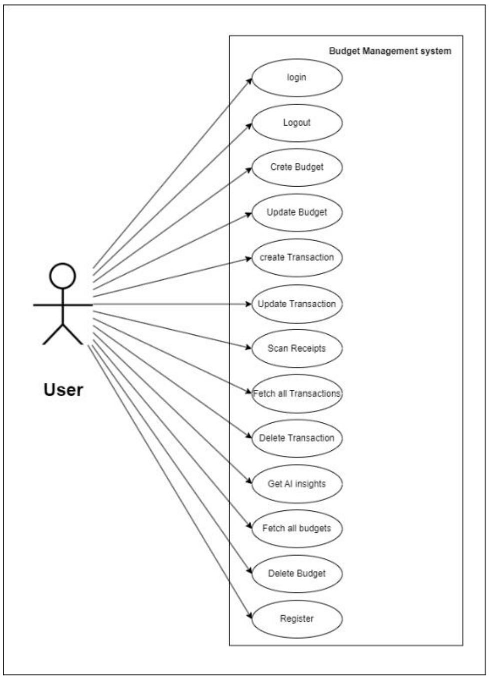


Figure 3: Use case diagram of the Budget Management System

3.3 ER Diagram

The ER diagram for a Budget Management System has three main entities: users, budgets, and transactions. Here's a concise breakdown:

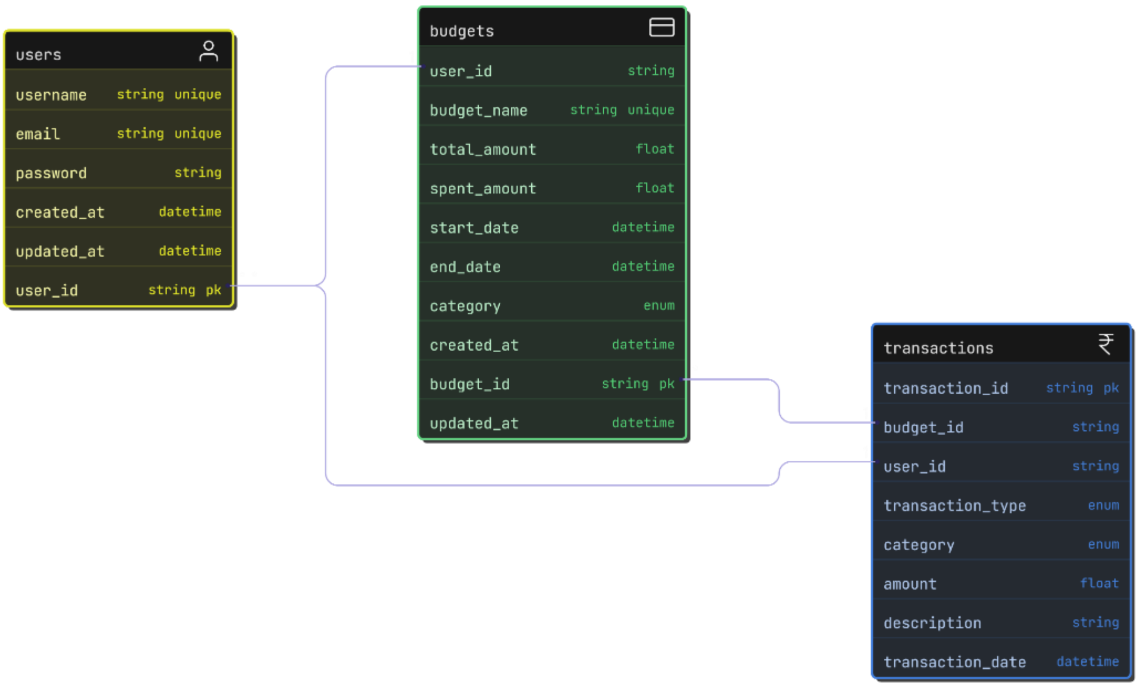
1. **Users**
   - Attributes: user_id (PK), username, email, password, created_at, updated_at.
   - Relationships: One user can have multiple budgets and transactions.

2. Budgets

- Attributes: budget_id (PK), user_id (FK), budget_name, total_amount, spent_amount, start_date, end_date, category, created_at, updated_at.
- Relationships: One budget can have multiple transactions, linked to one user.

3. Transactions

- Attributes: transaction_id (PK), budget_id (FK), user_id (FK), transaction_type, category, amount, description, transaction_date.
- Relationships: Each transaction is linked to one user and one budget.



users	
username	string unique
email	string unique
password	string
created_at	datetime
updated_at	datetime
user_id	string pk

budgets	
user_id	string
budget_name	string unique
total_amount	float
spent_amount	float
start_date	datetime
end_date	datetime
category	enum
created_at	datetime
budget_id	string pk
updated_at	datetime

transactions	
transaction_id	string pk
budget_id	string
user_id	string
transaction_type	enum
category	enum
amount	float
description	string
transaction_date	datetime

Figure 4: ER diagram of Budget Management System

3.4 Sequence Diagram

Authentication sequence diagram:

This authentication sequence diagram illustrates the user registration, login, and logout processes in a system:

- **Registration**: The user submits their details via /api/auth/signup. The system checks if the user already exists. If not, it hashes the password, creates the user, and returns a JWT with user data. If the user exists, an error is returned.
- **Login**: The user provides credentials via /api/auth/login. The system verifies them. If valid, it generates a JWT and returns user data. If invalid, an error is returned.
- **Logout**: When the user logs out via /api/auth/logout, the system clears the JWT cookie and redirects to the login page.

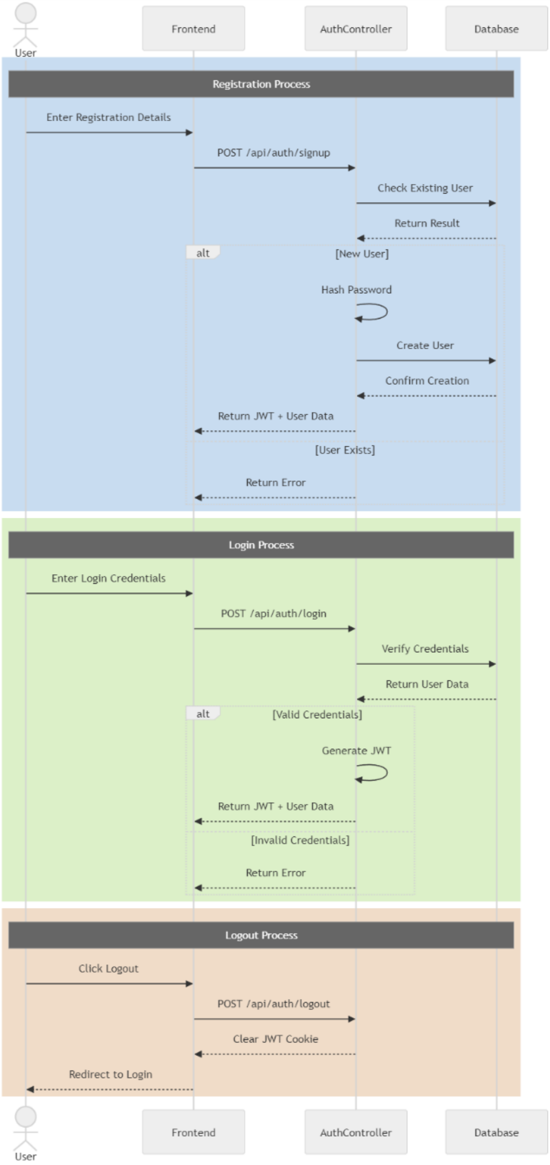


Figure 5: Sequence diagram for authentication in Budget Management System

Budgets sequence diagram:

This sequence diagram shows the **budget operations** in a system:

- **Create Budget**: User submits details via /api/budget/create. The system checks for duplicates, creates the budget if none exists, and returns success or error.
- **Update Budget**: User sends updated info to /api/budget/update. The system updates the budget and responds with the updated data.
- **Delete Budget**: User requests deletion via /api/budget/:id. The system deletes the budget and confirms the action.

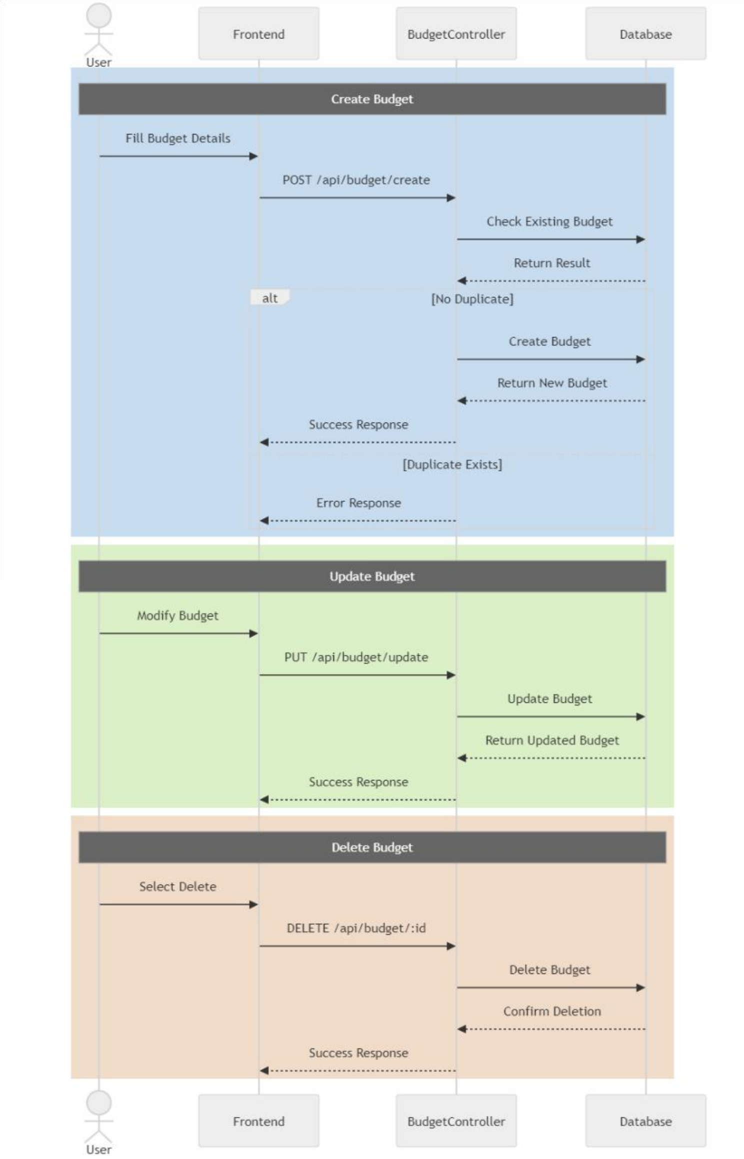


Figure 6: Sequence diagram for Budgets in Budget Management System

Transactions sequence diagram:

This sequence diagram shows the transaction operations in a system:

- Create Transaction: User submits data via /api/transaction/create. The system creates the transaction, updates the related budget, and sends a success response.
- View Transactions: User sends a GET request to /api/transactions. The system retrieves and returns the list of transactions.
- Update Transaction: User updates a transaction using /api/transaction/:id. The system modifies the data, recalculates the budget, and confirms the update.

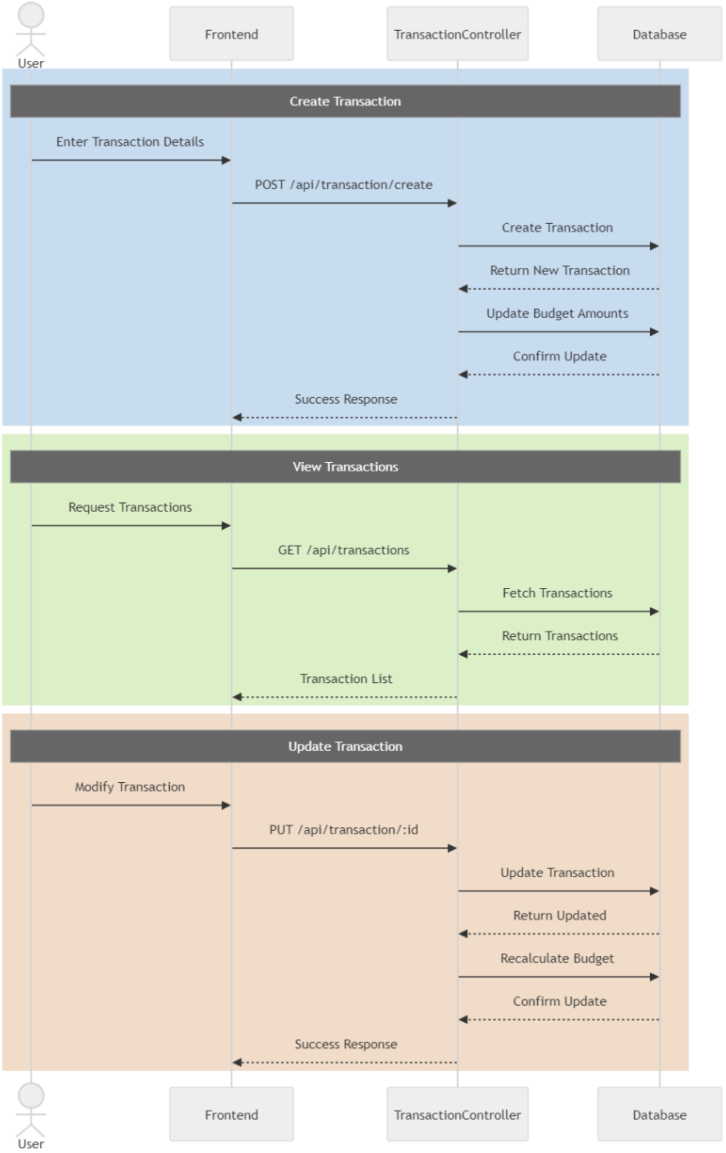


Figure 7: Sequence diagram for Transactions for Budget Management System

AI sequence diagram:

This sequence diagram shows the **AI-powered financial features** of the system:

- **Receipt Scanning**: User uploads a receipt via /api/scan-receipt. The image is processed by AI, data is extracted, and a transaction is created and saved.
- **Generate Insights**: User sends a GET request to /api/insights. Financial data is fetched, analyzed by AI, and insights are returned.
- **Spending Analysis**: User requests spending analysis via /api/analysis. Transaction data is analyzed for patterns by AI, and results are displayed to the user.

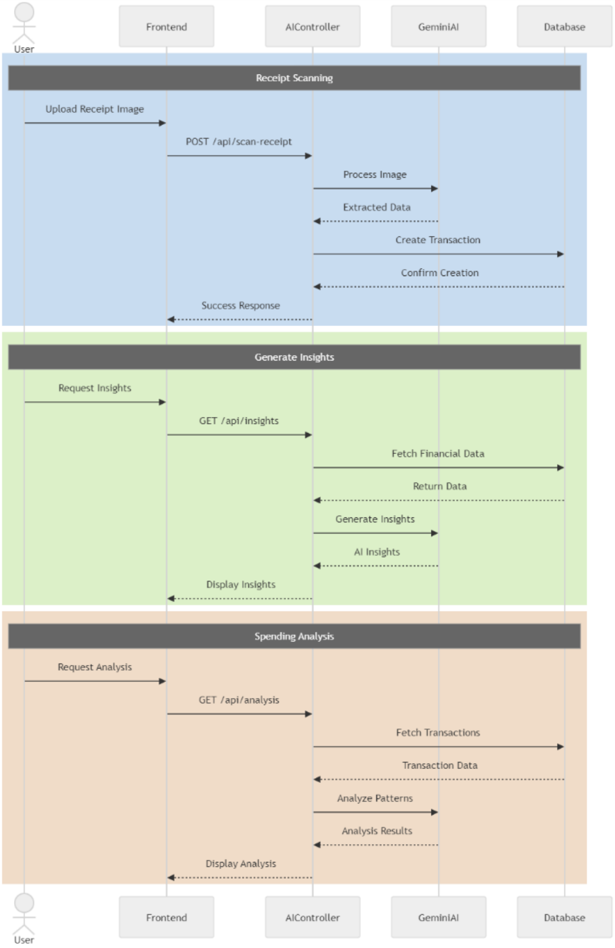


Figure 8: Sequence diagram for AI in Budget Management System