**1. Introduction**

**1.1 Purpose**

This document specifies the software requirements for *PursePro: Personal Finance Manager and Digital Wallet*. It covers the system's features, objectives, and implementation details. The focus is to provide an all-in-one desktop application that combines personal finance management tools with digital wallet functionality. This Software Requirements Specification (SRS) outlines the system’s capabilities to ensure seamless financial management and ease of use for its target audience.

**1.2 Product Scope**

*PursePro* aims to revolutionize how individuals manage their finances by integrating personal finance management and digital wallet capabilities into a single platform. The application will enable users to:

* Log and categorize expenses and income.
* Create and track budgets with reminders and alerts.
* Set and achieve savings goals.
* Perform digital transactions, including sending and receiving money and paying bills.

By addressing gaps in existing financial tools, *PursePro* delivers a cohesive solution for students, working professionals, and everyday users. Unlike existing tools such as Mint, YNAB, and SadaPay, *PursePro* offers a unique blend of features to provide comprehensive financial control.

**1.3 Title**

**PursePro: Personal Finance Manager and Digital Wallet**

The title reflects the project’s aim to provide a modern, accessible solution for managing personal finances and performing digital transactions, offering users greater control over their financial health.

**1.4 Objectives**

The primary objectives of *PursePro* are:

1. Simplify personal financial management for everyday users.
2. Empower users to take charge of their finances by tracking income, expenses, and budgets.
3. Provide tools for setting and achieving financial goals, including savings and debt repayment.
4. Facilitate secure and seamless digital transactions within the application.
5. Encourage better financial habits through personalized tips and insights.

By meeting these objectives, the application aspires to reduce financial stress and improve the financial literacy and health of its users.

**1.5 Problem Statement**

**The Challenge:**  
Managing personal finances is often overwhelming and fragmented, with users resorting to multiple tools to track expenses, set budgets, manage debts, and handle digital transactions. Current solutions like Mint and YNAB focus on budgeting but lack digital wallet capabilities, while apps like SadaPay and NayaPay excel at transactions but fail to offer robust financial planning tools. This leads to inefficiencies, poor financial decisions, and financial stress.

**The Solution:**  
*PursePro* addresses this gap by providing an integrated desktop solution that combines personal finance management with digital wallet functionalities. It empowers users to track spending, manage savings, handle debts, and make secure transactions—all within a single platform. The application is designed to be intuitive, scalable, and adaptable, ensuring it meets evolving user needs and offers long-term value.

**Feasibility:**  
This project leverages proven technologies such as MySQL for the database, Java for backend development, and modern design principles to ensure a reliable and scalable system. With a growing demand for digital financial tools, *PursePro* offers a timely and practical solution to an increasingly digital-first audience.

Great, I see the structure of your use cases. Let’s proceed with drafting the **Overall Description** section along with its subheadings. Here's how it looks:

**2. Overall Description**

**2.1 Product Perspective**

*PursePro: Personal Finance Manager and Digital Wallet* is a new, self-contained product that integrates personal finance management tools with digital wallet functionalities. Unlike existing standalone tools such as Mint, YNAB, and SadaPay, *PursePro* merges features like expense tracking, budget planning, savings goal-setting, debt management, and secure digital transactions into one seamless desktop application.

**Key Components of the Product:**

1. **Personal Finance Management:** Includes tracking expenses and income, creating budgets, and managing savings and debts.
2. **Digital Wallet Features:** Facilitates secure transactions, including sending and receiving money, bill payments, and virtual card management.
3. **Financial Insights and Education:** Offers tailored financial health scores, tax calculators, and personalized insights for better decision-making.

A simple architecture of *PursePro* shows its major modules interacting with the database and GUI.

**2.2 Product Functions**

The key functions of *PursePro* include:

* **Expense Tracking:** Users can log and categorize their daily expenses.
* **Income Tracking:** Helps users monitor all income sources.
* **Budget Planning:** Allows users to set and track budgets with alerts to prevent overspending.
* **Transaction Management:** Users can send and receive money securely and make bill payments.
* **Financial Analytics:** Offers features like financial health score analysis and tax calculators.
* **Virtual Card Management:** Facilitates issuing and managing virtual payment cards.
* **Currency Conversion:** Provides currency conversion into multiple currency.

**2.3 List of Use Cases**

The use cases in *PursePro* are grouped under three contributors:

1. **Mehboob Ali:**
   * Track Expense
   * Plan Budget
   * Transfer Money
   * Analyze Financial Health Score
   * Make Donations
2. **Rana Bilal Akbar:**
   * View Transaction History
   * Use Tax Calculator
   * Share Finance Account
   * Request Payment
   * Issue Virtual Card
3. **Muhammad Bilal:**
   * Track Income
   * Make Bill Payments
   * Reward Loyalty Points
   * Manage Subscriptions
   * Convert Currency

**2.4 Extended Use Cases**

**Fully Dressed Use Cases**

**1. Track Expense**

**a. Use Case Name:**

Track Expense

**b. Scope:**

Wallet Application

**c. Level:**

User Goal

**d. Primary Actor:**

User (Everyday users such as working professionals, students)

**e. Stakeholders and Interests:**

* **User:** Wants to log and track expenses efficiently to monitor financial health.
* **PursePro:** Provides transaction data for digital payments and ensures hassle-free expense tracking.

**f. Preconditions:**

* User is logged into the wallet application.
* Expense categories (e.g., groceries, rent, entertainment) are defined or can be created.
* User has paid or sent money using PursePro at least once.

**g. Postconditions:**

* The expense is recorded and categorized.
* The updated balance and expense breakdown are displayed to the user.
* The expense contributes to overall budget calculations.

**h. Main Success Scenario:**

| **User Action** | **System Response** |
| --- | --- |
| 1. User selects the 'Track Expense' option from the dashboard. | 2. The system shows all expenses under each category and provides an option to add an expense manually. |
| 3. User selects the 'Add Expense' option. | 4. The system displays a form to enter expense details (e.g., amount, category, date). |
| 5. User enters the required information and confirms the expense. | 6. The system records the expense in the relevant category. |
|  | 7. The system updates the relevant budget category. |
|  | 8. The system shows a confirmation message that the expense was successfully recorded. |
|  | 9. The system updates the financial overview, reflecting the new expense. |

**i. Extensions:**

* **3a. Do Not Add Expense:**  
  If the user does not want to enter a new expense manually, they can review past expenses and exit “Track Expense.”
* **5a. Invalid Input:**  
  If the user enters invalid or incomplete data (e.g., amount is not a number), the system displays an error message and prompts the user to correct the input.

**2. Plan Budget**

**a. Use Case Name:**

Plan Budget

**b. Scope:**

Wallet Application

**c. Level:**

User Goal

**d. Primary Actor:**

User (Everyday users such as working professionals, students)

**e. Stakeholders and Interests:**

* **User:** Wants to manage monthly or annual budgets to track spending and achieve financial goals.
* **PursePro:** Ensures users manage their budgets effectively to maintain app engagement.

**f. Preconditions:**

* User is logged into the wallet application.
* Expense categories (e.g., groceries, rent, entertainment) are defined or can be created.

**g. Postconditions:**

* The budget is created or updated with specified limits.
* The system starts monitoring the user's expenses against the set budget.
* The user is alerted when approaching or exceeding budget limits.
* The budget is reflected in the financial summary.

**h. Main Success Scenario:**

| **User Action** | **System Response** |
| --- | --- |
| 1. User selects the 'Budget Planning' option from the main dashboard. | 2. The system prompts the user to choose if they want to add a new budget or track current budgets. |
| 3. User selects the 'Add New Budget' option. | 4. The system prompts the user to choose between a monthly or annual budget. |
| 5. User selects a time frame and enters desired budget limits for different categories (e.g., groceries, entertainment, utilities). | 6. The system confirms the budget details with the user. |
|  | 7. The system saves the budget and begins tracking user expenses against the set limits. |
|  | 8. As the user makes transactions, the system tracks the remaining budget in each category. |
|  | 9. If the user approaches the budget limit for any category (e.g., 90% spent), the system sends an automated alert. |
|  | 10. If the user exceeds the budget in any category, the system sends another alert. |
|  | 11. The system displays a dashboard overview showing the current budget status for all categories. |

**i. Extensions:**

* **3a. Track Current Budget:**
  1. If the user chooses to track their current budget, the system displays a dashboard overview showing the current budget status for all categories.
  2. If the user wants to modify an existing budget, they can adjust the limits for individual categories, and the system recalculates accordingly.
* **4a. Invalid Input:**  
  If the user enters invalid data (e.g., budget amount not numeric), the system displays an error message and prompts the user to correct the input.

**3. Transfer Money**

**a. Use Case Name:**

Transfer Money

**b. Scope:**

Wallet Application

**c. Level:**

User Goal

**d. Primary Actor:**

User (Individuals sending money to friends, family, or businesses)

**e. Stakeholders and Interests:**

* **User:** Wants a quick, secure, and easy way to send money to other individuals or businesses.
* **Recipient:** Wants to receive the payment promptly and securely.
* **PursePro:** Handles the transaction securely and efficiently.

**f. Preconditions:**

* User is logged into the PursePro application.
* The recipient’s PursePro Wallet information is available (e.g., account number).

**g. Postconditions:**

* The money is successfully transferred from the user to the recipient.
* The system updates the user’s balance to reflect the transaction.
* The recipient is notified and receives the money.
* A transaction record is stored and available for both the sender and the recipient.

**h. Main Success Scenario:**

| **User Action** | **System Response** |
| --- | --- |
| 1. User selects the 'Send Money' option from the dashboard. | 2. The system prompts the user to enter the recipient’s PursePro account details (e.g., account number). |
| 3. User enters the recipient's information and the amount to be sent. | 4. User also selects a category or purpose for the transaction. |
|  | 5. The system shows the total amount (including any fees, if applicable) and requests confirmation. |
| 6. User confirms the transaction. | 7. The system processes the payment and deducts the amount from the sender’s balance. |
|  | 8. The system notifies the recipient that the money has been sent and updates the recipient's balance. |
|  | 9. Both the sender and recipient can view the transaction details in their respective transaction histories. |
|  | 10. The expense is added to the selected category. |

**i. Extensions:**

* **3a. Invalid Recipient Information:**  
  If the user enters invalid recipient details, the system displays an error and prompts the user to correct the information.
* **4a. Insufficient Funds:**  
  If the user has insufficient funds, the system informs the user.
* **6a. Network Failure:**  
  If the payment fails due to network or technical issues, the system informs the user and offers to retry the transaction.
* **7a. Delayed Transfer:**  
  If the transfer takes longer than expected, the system notifies the user and provides an estimated completion time.

**4. Analyze Financial Health Score**

**a. Use Case Name:**

Analyze Financial Health Score

**b. Scope:**

PursePro Financial Management Application

**c. Level:**

User Goal

**d. Primary Actor:**

User (Individual managing finances)

**e. Stakeholders and Interests:**

* **User:** Wants to understand their financial health and receive actionable insights to improve their financial stability.
* **PursePro:** Provides personalized financial insights and recommendations based on the user's financial data.

**f. Preconditions:**

* The user has logged into the PursePro app.
* The system has data for the user's income and expenses.

**g. Postconditions:**

* The user receives a financial health score based on their data.

**h. Main Success Scenario:**

| **User Action** | **System Response** |
| --- | --- |
| 1. User navigates to the Financial Health Score section and requests an analysis. | 2. The system retrieves the user's income and expense information. |
|  | 3. Based on the user's financial data, the system calculates a health score by evaluating: |
|  | - Income-to-expense ratio |
|  | - Savings Potential Percentage |
|  | 4. The system displays the calculated financial health score to the user in an easy-to-understand format (e.g., a percentage or grade). |
|  | 5. The system generates suggestions based on the user’s score, including: |
|  | - Reducing expenses |
|  | - Increasing monthly savings |
|  | - Investing saved money |
| 6. User reviews the score and improvement suggestions. | 7. User can save the suggestions, set new financial goals, or update current budgets. |

**i. Extensions:**

* **4a. Insufficient Data for Analysis:**  
  The user’s data is insufficient (e.g., does not have sufficient expense data) to calculate the financial score. The system notifies the user and returns to the main dashboard.

**5. Make Donations**

**a. Use Case Name:**

Make Donations

**b. Scope:**

Wallet Application

**c. Level:**

User Goal

**d. Primary Actor:**

User (Individuals sending money to friends, family, or businesses)

**e. Stakeholders and Interests:**

* **User:** Wants a quick, secure, and easy way to make charity donations.
* **Recipient Organization:** Wants to receive the payment promptly and securely.
* **PursePro:** Handles the transaction securely and efficiently.

**f. Preconditions:**

* User is logged into the PursePro application.
* The recipient organization is a registered charity and is in PursePro’s organizations list.

**g. Postconditions:**

* The money is successfully transferred from the user to the organization.
* The system updates the user’s balance to reflect the transaction.
* A transaction record is stored and available for both the sender and the recipient organization.

**h. Main Success Scenario:**

| **User Action** | **System Response** |
| --- | --- |
| 1. User selects the 'Make Donations' option from the dashboard. | 2. The system displays all current recurring donations and prompts the user to either add a new donation or update current donations. |
| 3. User selects 'Add New Donation.' | 4. The system displays a list of all the registered charities. |
| 5. User selects the registered charity they want to donate to. | 6. The system prompts the user to enter the amount and select if the donation is one-time or recurring. |
| 7. User enters the required information. | 8. The system shows the total amount and requests confirmation. |
| 9. User confirms the transaction. | 10. The system processes the payment and deducts the amount from the sender’s balance. |
|  | 11. The system notifies the recipient that the money has been sent and updates the recipient's balance. |
|  | 12. Both the sender and recipient can view the transaction details in their respective transaction histories. |
|  | 13. The expense is added to the 'Donations' category. |

**i. Extensions:**

* **3a. Update Current Donation:**
  1. If the user wants to update existing donations, the system will prompt the user to select the donation they want to update.
  2. The user selects and can either update the amount of any donation or remove the donation.
* **10a. Insufficient Funds:**  
  If the user has insufficient funds, the system informs the user.
* **10b. Network Failure:**  
  If the payment fails due to network or technical issues, the system informs the user and offers to retry the transaction.

**6. View Transaction History**

**a. Use Case Name:**

View Transaction History

**b. Scope:**

PursePro - Personal Finance Manager and Digital Wallet App

**c. Level:**

User-Level Goal

**d. Primary Actor:**

User

**e. Stakeholders and Interests:**

* **User:** Wants to view and track past financial transactions, ensuring accuracy and clarity.
* **System Administrator:** Ensures system performance is not affected by high transaction data retrievals.
* **Government/Regulatory Agencies:** Interested in proper financial record-keeping and compliance with financial regulations (e.g., for audits or tax purposes).

**f. Preconditions:**

* The user is logged into their account.

**g. Postconditions:**

* Transaction history is successfully displayed.
* The user can export or review detailed transaction data for any selected period.

**h. Main Success Scenario:**

| **User Action** | **System Response** |
| --- | --- |
| 1. User successfully logs into PursePro and navigates to the "Transaction History" section. | 2. The system retrieves and displays a chronological list of all transactions made by the user. |
| 3. User scans through the list and selects a particular transaction to view more details. | 4. The system provides a detailed view of the selected transaction, showing essential information such as the transaction date, amount, and involved parties (e.g., sender/recipient). |
| 5. User requests a summary of their financial activity for a specific period (e.g., the past month). | 6. The system generates the summary of the requested period. |
| 7. User requests to export the transaction history. | 8. The system prepares the transaction data, formats it into a downloadable file, and stores it in the designated format for export (e.g., CSV). |
| 9. User confirms the export request. | 10. The system ensures the file is correctly saved and stored for future use. |
|  | 11. System sends a confirmation that the export was successful. |

**i. Extensions:**

* **2a. No Transactions Exist:**  
  The system displays a message stating, "No transactions found for this period."
* **3a. User Selects a Transaction That Is Incomplete or Failed:**  
  The system provides an error message and suggests contacting customer support or retrying the transaction.
* **8a. CSV Export Fails:**  
  The system displays an error message, and the user retries the export.

**7. Use Tax Calculator**

**a. Use Case Name:**

Use Tax Calculator

**b. Scope:**

PursePro - Personal Finance Manager and Digital Wallet App

**c. Level:**

User-Level Goal

**d. Primary Actor:**

User

**e. Stakeholders and Interests:**

* **User:** Wants to estimate tax liabilities, track deductible expenses, and generate accurate tax reports for filing.
* **System Administrator:** Ensures smooth functioning of the tax calculation feature with accurate data syncing and report generation.
* **Tax Preparation Services:** Interested in integrations or reports for professional tax filing.

**f. Preconditions:**

* The user is logged into their account.
* The user has entered or synced income data (e.g., salary, investments, freelance earnings).
* The user has categorized deductible expenses (e.g., donations, business expenses).

**g. Postconditions:**

* The user receives an accurate tax estimate.
* The user can generate a detailed tax report.

**h. Main Success Scenario:**

| **User Action** | **System Response** |
| --- | --- |
| 1. User navigates to the "Tax Calculator" feature within PursePro. | 2. The system displays a summary of the user’s income and categorized deductible expenses. |
| 3. User selects the tax year they wish to calculate for (e.g., the current or previous year). | 4. The system calculates estimated tax liabilities based on the user's income, applicable deductions, and tax brackets. |
| 5. User reviews the estimated tax liability and has the option to modify or add any missing deductible expenses (e.g., business expenses, charity donations). | 6. The system updates the calculation based on any adjustments made by the user. |
| 7. User generates a detailed tax report for filing purposes. | 8. The system displays the report summary, including all income, deductions, and the final tax estimate. |
| 9. User can proceed to pay the tax amount through PursePro. | 10. The system deducts the tax amount from the user’s account. |

**i. Extensions:**

* **2a. No Income Data or Deductible Expenses Available:**  
  The system displays a message saying, "No income or deductible expense data found. Please enter," prompting the user to input missing financial data.
* **4a. System Cannot Calculate Tax Due to Incomplete Data:**  
  The system alerts the user that additional income or deduction data is required to perform the calculation.
* **7a. User Cancels Report Generation:**  
  The system stops the report generation process and discards the file.
* **9a. User Does Not Have Enough Balance:**  
  The system stops the payment of tax.

**8. Share Finance Account**

**a. Use Case Name:**

Share Finance Account

**b. Scope:**

PursePro System (Shared Finances Management Feature)

**c. Level:**

User-Level Goal

**d. Primary Actor:**

User (Manages or participates in shared expenses)

**e. Stakeholders and Interests:**

* **User:** Wants a convenient way to manage shared expenses and ensure contributions from participants are tracked accurately.
* **Other Participants:** Expect accurate notifications regarding their portion of the expenses and transparency in shared financial matters.
* **System Administrator:** Ensures secure and efficient operation to maintain user trust.

**f. Preconditions:**

* User is logged into PursePro and has access to the Shared Finances Management feature.
* Participants being added to the shared group have active PursePro accounts.

**g. Postconditions:**

* All shared expenses are accurately recorded, split among participants, and contributions tracked in real time.
* Participants receive notifications regarding their portion of the expense and any outstanding payments.

**h. Main Success Scenario:**

| **User Action** | **System Response** |
| --- | --- |
| 1. User logs into PursePro and navigates to the "Shared Finances Management" section. | 2. System displays an overview of existing shared finances (if any) or options to create new ones. |
| 3. User adds participants to the shared finance group. | 4. The system verifies the participants’ PursePro account status. |
| 5. User assigns expense amounts and splits to participants. | 6. System updates shared expenses and notifies participants of their contributions. |
| 7. User confirms the shared finance details. | 8. System begins real-time tracking of expenses and updates participants as contributions are made. |

**i. Extensions:**

* **3a. Participant Without a PursePro Account:**  
  The system notifies the user that the participant is not registered and prompts the user to invite them or manage their share offline.
* **5a. Incorrect Expense Amount:**  
  The system alerts the user that the input is invalid and prompts for correction.

**9. Request Payment**

**a. Use Case Name:**

Request Payment

**b. Scope:**

PursePro System (Payment Request Feature)

**c. Level:**

User-Level Goal

**d. Primary Actor:**

User

**e. Stakeholders and Interests:**

* **User:** Wants an easy and efficient way to request payments, especially for shared expenses or bill splitting.
* **Recipients:** Expect to receive clear and accurate payment requests with the ability to fulfill them easily through the app.
* **System Administrator:** Ensures secure and error-free processing of payment requests.

**f. Preconditions:**

* User is logged into PursePro and has access to the Payment Requests feature.
* The recipient has a valid PursePro account capable of receiving payment requests.

**g. Postconditions:**

* The payment request is successfully sent to the recipient.
* The requested amount is accurately reflected in the system for the recipient to pay.

**h. Main Success Scenario:**

| **User Action** | **System Response** |
| --- | --- |
| 1. User logs into PursePro and navigates to the "Payment Requests" section from the dashboard. | 2. System displays the option for the user to create a new payment request. |
| 3. User creates a new payment request and inputs the details (e.g., recipient information, amount). | 4. System validates the input to ensure the recipient’s information is correct and the amount is valid. |
| 5. System confirms the recipient’s account is active and capable of receiving the payment request. | 6. User reviews and confirms the payment request details before sending. |
|  | 7. System sends the payment request to the recipient and notifies the user that the request was successfully sent. |
|  | 8. Recipient can view the payment request within the app. |
|  | 9. System displays the requested amount and payment options for the recipient. |
|  | 10. Recipient can proceed to pay the amount requested. |

**i. Extensions:**

* **4a. Invalid Recipient Information:**  
  The system notifies the user that the recipient's details are incorrect or incomplete.
* **6a. Invalid Amount:**  
  The system alerts the user that the entered amount is invalid (e.g., negative or too high) and prompts for correction.
* **10a. Recipient Rejects the Payment Request:**  
  The system updates the status of the payment request to “Rejected” and notifies the user.

**10. Issue Virtual Card**

**a. Use Case Name:**

Issue Virtual Card

**b. Scope:**

PursePro System (Virtual Debit Card)

**c. Level:**

User-Level Goal

**d. Primary Actor:**

User (Who requests a virtual debit/credit card)

**e. Stakeholders and Interests:**

* **User:** Wants a secure and convenient virtual card for online purchases, linked to their digital wallet.
* **System Administrator:** Ensures secure card issuance that complies with financial regulations.

**f. Preconditions:**

* User has a verified and active PursePro account.
* User has sufficient funds in their digital wallet (for debit cards).
* The system has access to the bank or financial institution responsible for issuing virtual cards.

**g. Postconditions:**

* A virtual debit/credit card is successfully issued, linked to the user’s digital wallet, and available for use in online transactions.
* The system securely stores card details and provides access when required for purchases.

**h. Main Success Scenario:**

| **User Action** | **System Response** |
| --- | --- |
| 1. User navigates to the "Virtual Card" section in PursePro. | 2. System presents options to request a virtual debit card. |
| 3. User selects the desired card type and provides necessary details (e.g., billing address, card limit). | 4. System verifies the user’s account status, available funds, and checks for existing virtual cards. |
|  | 5. System processes the request and communicates securely with the issuing financial institution. |
|  | 6. System generates the virtual card, assigns a unique card number, expiration date, and CVV, and links it to the user’s wallet. |
| 7. User reviews the virtual card details (card number, expiration date, CVV). | 8. System securely stores the virtual card details and provides them when needed for transactions. |
|  | 9. User can use the virtual card for online purchases. |

**i. Extensions:**

* **4a. Insufficient Funds (Debit Card):**  
  The system notifies the user that they do not meet the required criteria for issuing the card.
* **6a. Issuing Institution Denies Request:**  
  The system informs the user that the virtual card request was denied.
* **8a. Cancel or Block Virtual Card:**  
  The system provides an option to cancel or block the card in case of misuse or fraud.

**11. Track Income**

**a. Use Case Name:**

Track Income

**b. Scope:**

Wallet Application

**c. Level:**

User-Level Goal

**d. Primary Actor:**

User

**e. Stakeholders and Interests:**

* **User:** Wants to monitor income from various sources and track trends over time.
* **PursePro:** Aims to provide accurate insights into income data and incorporate it into budgeting and financial health analysis.

**f. Preconditions:**

* The user is logged into the application.
* User income categories are predefined or can be added.

**g. Postconditions:**

* The income is recorded and categorized.
* Income is reflected in financial summaries and budgeting calculations.

**h. Main Success Scenario:**

| **User Action** | **System Response** |
| --- | --- |
| 1. User navigates to the "Income Tracker" section in PursePro. | 2. The system displays current income sources and provides an option to add new income. |
| 3. User selects "Add Income" and enters details (e.g., source, amount, date). | 4. System validates the input and records the income under the appropriate category. |
|  | 5. System updates financial summaries, reflecting the new income. |
|  | 6. User receives a confirmation message that the income was successfully recorded. |

**i. Extensions:**

* **3a. Invalid Input:**  
  The system notifies the user of invalid or incomplete data and prompts for corrections.

**12. Make Bill Payments**

**a. Use Case Name:**

Make Bill Payments

**b. Scope:**

Manage and pay recurring bills through the app.

**c. Level:**

User-Goal Level

**d. Primary Actor:**

User

**e. Stakeholders and Interests:**

* **User:** Wants a convenient and efficient way to manage and pay bills to avoid late fees and ensure timely payments.
* **Utility Providers:** Require accurate billing information and timely payments to ensure uninterrupted service for their customers.
* **System Administrator:** Aims to create a user-friendly interface that simplifies the payment process, enhancing user experience and satisfaction.

**f. Preconditions:**

* The user has linked bills and payment methods.

**g. Postconditions:**

* Bills are paid and marked as such in the system.

**h. Main Success Scenario:**

| **User Action** | **System Response** |
| --- | --- |
| 1. User navigates to the 'Bill Payments' section. | 2. System loads the bill payment interface. |
| 3. User selects a bill to pay. | 4. System retrieves the bill details. |
| 5. User confirms payment. | 6. System processes the payment and confirms success. |

**i. Extensions:**

* **3a. Bill Not Found:**  
  If the user selects a bill that cannot be retrieved (e.g., due to a server error), the system displays an error message:  
  *"Unable to retrieve bill details. Please try again later."*
* **5a. Insufficient Funds:**  
  If the user tries to confirm payment without sufficient funds in their account, the system alerts the user:  
  *"Payment failed due to insufficient funds. Please add funds to your account."*

**13. Reward Loyalty Points**

**a. Use Case Name:**

Reward Loyalty Points

**b. Scope:**

Redeem loyalty points for discounts, vouchers, or cashback.

**c. Level:**

User-Goal Level

**d. Primary Actor:**

User

**e. Stakeholders and Interests:**

* **User:** Wants a straightforward process to redeem loyalty points for rewards, making it easier to save money on future purchases.
* **System Administrator:** Focuses on creating a user-friendly interface and smooth interactions that enhance the user experience and increase satisfaction with the app.

**f. Preconditions:**

* User has linked loyalty accounts and accumulated points.

**g. Postconditions:**

* Points are redeemed, and rewards are provided.

**h. Main Success Scenario:**

| **User Action** | **System Response** |
| --- | --- |
| 1. User navigates to the 'Loyalty Points' section. | 2. System displays available points and associated stores. |
| 3. User selects an offer to redeem points for. | 4. System checks the point’s balance and offer availability. |
| 5. User reviews the selected offer details, including terms and conditions. | 6. User confirms the redemption. |
|  | 7. System processes the redemption and generates a voucher or discount. |

**i. Extensions:**

* **4a. Terms and Conditions Not Accepted:**  
  If the user does not accept the terms and conditions for the selected offer, the system displays a message:  
  *"You must accept the terms and conditions to proceed with the redemption."*
* **6a. Redemption Processing Error:**  
  If there is an error while processing the redemption (e.g., due to a system failure), the system informs the user:  
  *"Redemption could not be processed at this time. Please try again later."*

**14. Manage Subscription**

**a. Use Case Name:**

Manage Subscription

**b. Scope:**

Manage and cancel active subscriptions, set reminders.

**c. Level:**

User-Goal Level

**d. Primary Actor:**

User

**e. Stakeholders and Interests:**

* **User:** Wants a convenient platform to manage subscriptions, track payments, and set reminders to avoid late fees.
* **Service Providers:** Aim to enhance user retention by providing clear subscription information and easy cancellation options.

**f. Preconditions:**

* User has active subscriptions linked to their profile.

**g. Postconditions:**

* Subscription details are updated, and reminders are set.

**h. Main Success Scenario:**

| **User Action** | **System Response** |
| --- | --- |
| 1. User navigates to the 'Subscription Management' section. | 2. System displays the list of active subscriptions. |
| 3. User selects a subscription to update or cancel. | 4. User updates or cancels a subscription. |
|  | 5. System processes the changes and updates subscription information. |
| 6. User sets reminders for upcoming payments. | 7. System schedules reminders and notifies the user when payment is due. |

**i. Extensions:**

* **3a. Subscription Not Found:**  
  If the user selects a subscription that cannot be retrieved (e.g., due to a server error), the system displays an error message:  
  *"Unable to retrieve subscription details. Please try again later."*

**15. Convert Currency**

**a. Use Case Name:**

Convert Currency

**b. Scope:**

Handle multi-currency transactions with real-time conversion rates.

**c. Level:**

User-Goal Level

**d. Primary Actor:**

User

**e. Stakeholders and Interests:**

* **User:** Wants a simple and efficient way to convert currencies, ensuring they get accurate real-time conversion rates for their transactions.
* **App Developers:** Focuses on creating an intuitive user interface that facilitates easy currency conversions and ensures smooth user experience while integrating real-time data feeds.

**f. Preconditions:**

* User has access to the currency conversion feature.

**g. Postconditions:**

* The transaction is processed with the correct conversion rate applied.

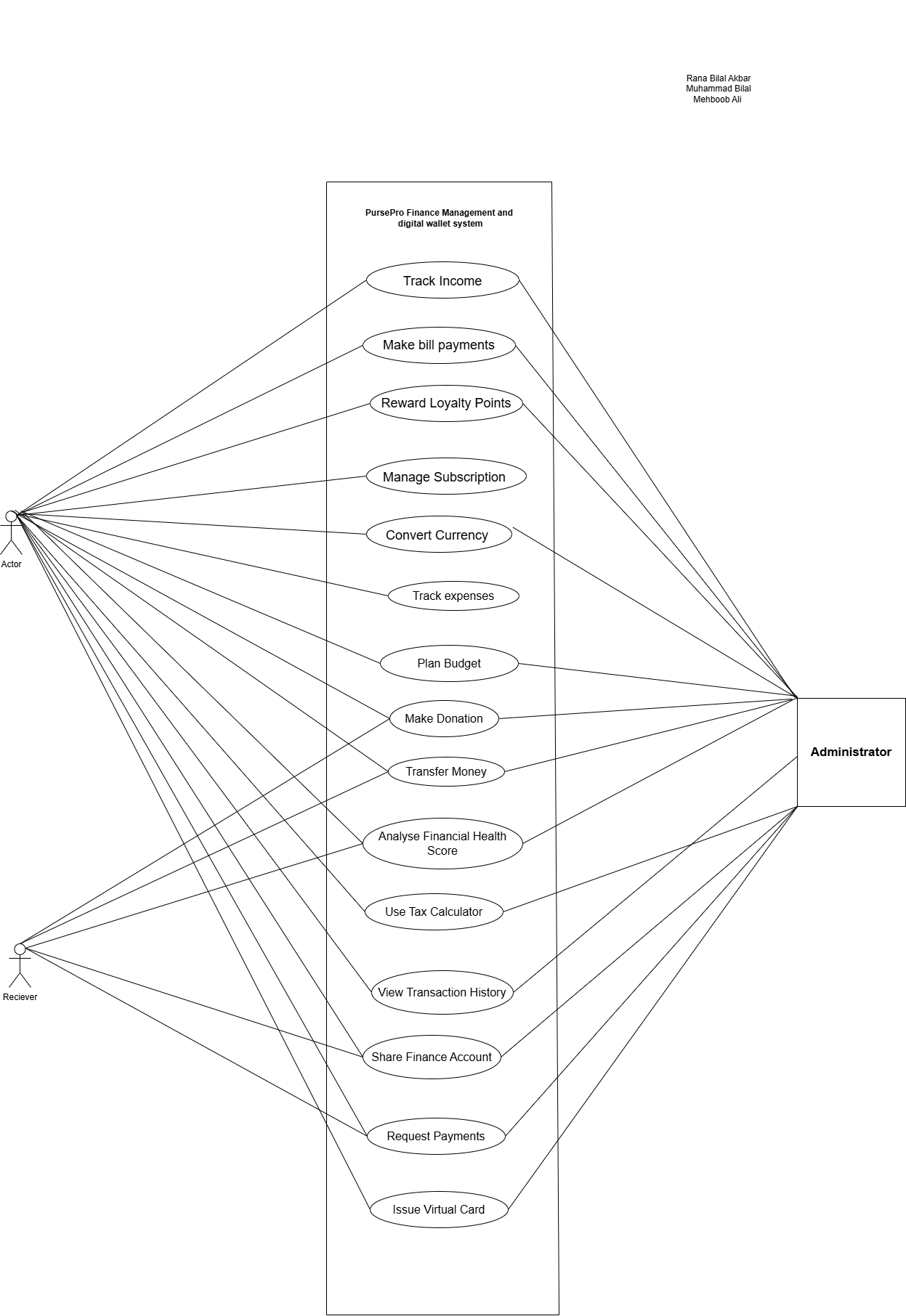
**h. Main Success Scenario:**

| **User Action** | **System Response** |
| --- | --- |
| 1. User navigates to the 'Currency Conversion' section. | 2. System displays available currencies and current exchange rates. |
| 3. User selects the currency to convert from and to. | 4. System retrieves the real-time conversion rates. |
| 5. User enters the amount to convert. | 6. System calculates the equivalent amount in the target currency. |
| 7. User confirms the transaction. | 8. System processes the transaction and applies the conversion rate. |

**i. Extensions:**

* **4a. Real-Time Rate Retrieval Failure:**  
  If the system encounters an error while retrieving real-time conversion rates, it displays a message:  
  *"Unable to retrieve conversion rates at this time. Please try again later."*

**2.5 Use Case Diagram**



**3. Other Nonfunctional Requirements**

**3.1 Performance Requirements**

The *PursePro* application will adhere to the following performance benchmarks to ensure smooth user experience and system reliability:

* The application should respond to user actions (e.g., logging expenses, making payments) within **2 seconds** under normal operating conditions.
* The system must handle up to **1,000 concurrent users** without performance degradation.
* Real-time features like **transaction processing** must retrieve and display data within **3 seconds** for standard network conditions.
* The application database must support querying and updating up to **1 million records** while maintaining response times under **500 milliseconds**.

**Rationale:**

* These performance requirements ensure that users experience minimal delays, even during peak usage, and that the system can scale with user demands.

**3.2 Safety Requirements**

* The application must include safeguards against data corruption during transactions (e.g., power loss or network failure). Any incomplete transaction must be rolled back or flagged for administrative review.
* The system will provide warnings and alerts when users are about to exceed budget or overdraft their account to prevent financial errors.
* All transactions must have a confirmation mechanism to prevent accidental payments.

**External Policies:**

* The application will comply with financial regulations, such as PCI DSS (Payment Card Industry Data Security Standard) and GDPR (General Data Protection Regulation) where applicable, ensuring the safety and privacy of financial data.

**3.3 Security Requirements**

The application must ensure the following security measures are in place:

1. **User Authentication:**
   * All users must log in with secure credentials, supported by two-factor authentication (2FA).
   * Session timeout occurs after 15 minutes of inactivity.
2. **Data Encryption:**
   * All sensitive user data (e.g., transactions, account details) must be encrypted using **AES-256** encryption standards, both in transit and at rest.
3. **Access Control:**
   * Only authorized users should be able to access or modify financial information.
   * Administrative functions (e.g., refunds, disputes) will be restricted to designated system administrators.
4. **Transaction Security:**
   * Every transaction must include validation mechanisms to detect and prevent fraud or duplicate payments.
   * Suspicious activities (e.g., unusually large transactions) should trigger alerts for both the user and administrators.

**3.4 Software Quality Attributes**

The following attributes will guide the development and evaluation of *PursePro*:

* **Adaptability:** The application should support integration with third-party APIs, such as bank systems, loyalty programs, and payment processors.
* **Reliability:** The system must maintain **99.9% uptime**, ensuring users have access to their financial data and services without interruptions.
* **Usability:** The application must provide a simple, intuitive interface for users of varying technical proficiency.
* **Maintainability:** The codebase will follow modular programming practices to allow for easier debugging and future feature additions.
* **Scalability:** The architecture should accommodate increased user load and transaction volume without requiring major rework.
* **Testability:** All critical features must include unit and integration tests to ensure functionality across edge cases.

**3.5 Business Rules**

The application will operate under the following principles:

1. **User Roles:**
   * Only registered users with verified identities will be allowed to conduct financial transactions.
   * Administrative users will manage escalated disputes, refunds, and system audits.
2. **Budgeting Rules:**
   * Users can create monthly or annual budgets with predefined categories. Budget limits cannot exceed the user's account balance or anticipated income.
3. **Loyalty Redemption:**
   * Loyalty points redeemed for rewards must comply with individual provider terms, such as minimum point thresholds or expiration dates.
4. **Data Retention:**
   * Financial records will be retained for a minimum of **5 years** to comply with regulatory requirements and provide users with access to historical data.

**3.6 Operating Environment**

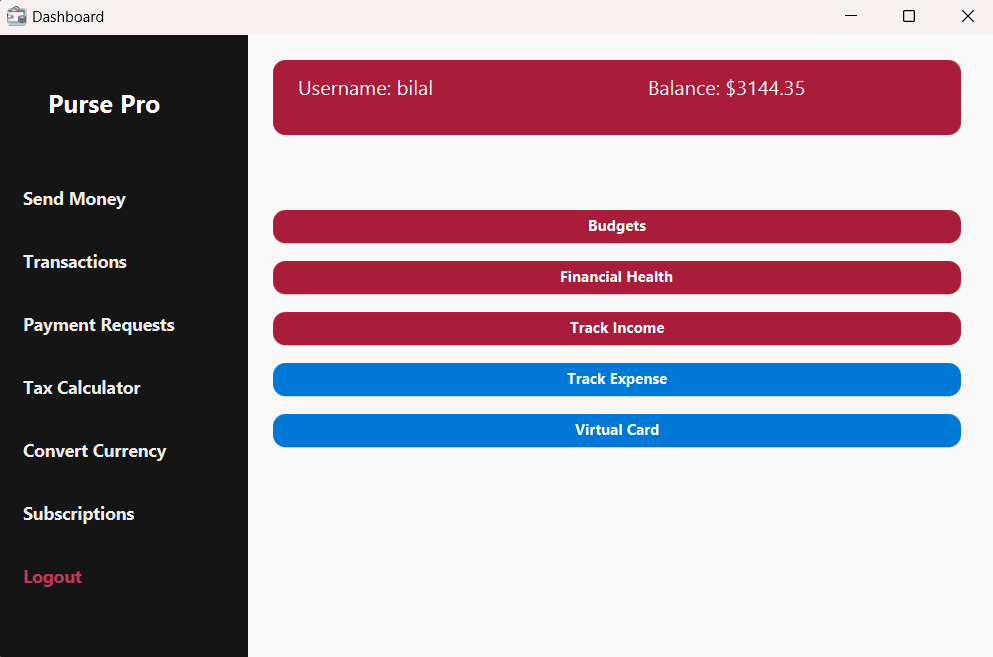
The *PursePro* application will operate in the following environments:

* **Hardware Platforms:**
  + Minimum Requirements:
    - Processor: Intel Core i5 or equivalent.
    - Memory: 4GB RAM.
    - Storage: 2GB available space.
* **Operating Systems:**
  + Windows 10 or later, macOS 10.15 (Catalina) or later, and Linux distributions (e.g., Ubuntu 20.04 or higher).
* **Network Requirements:**
  + Stable internet connection with a minimum speed of **1 Mbps** for real-time features like currency conversion or tax calculations.

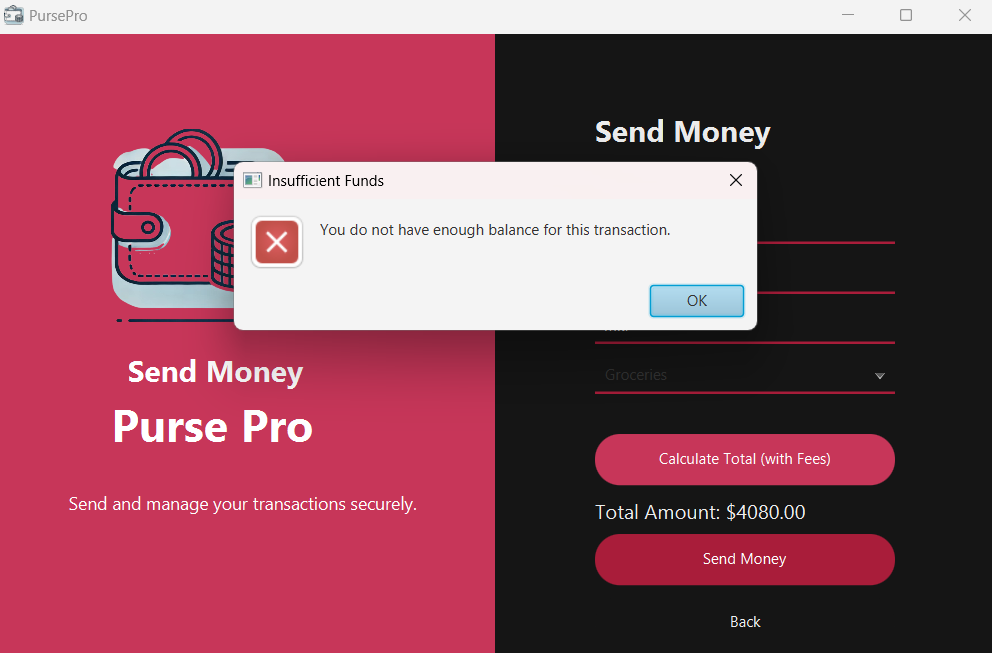
**3.7 User Interfaces**

The user interface will be designed with the following guidelines:

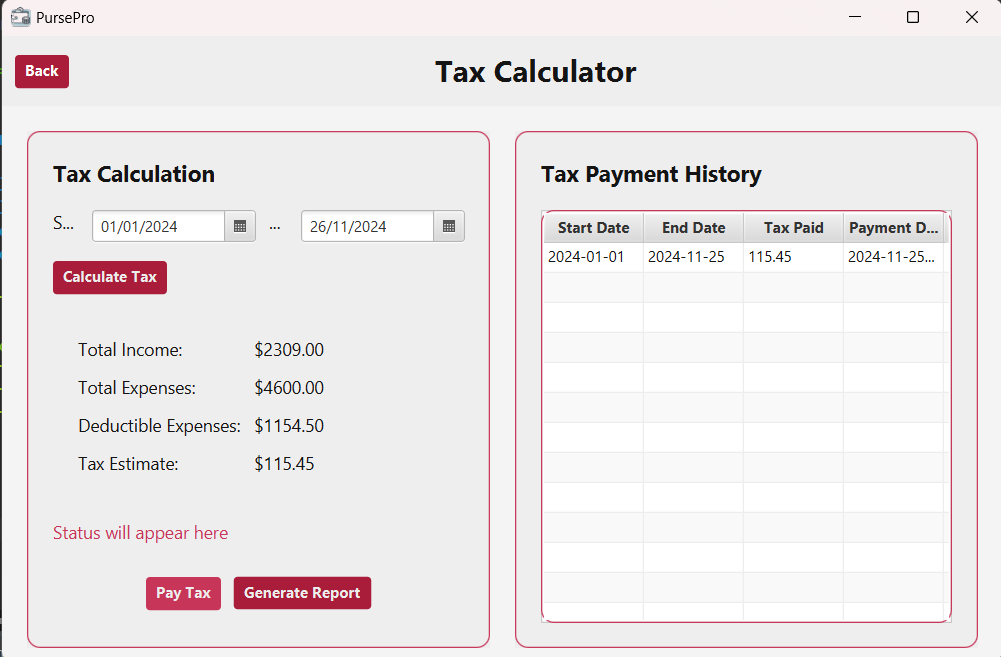
1. **Dashboard Overview:**
   * Provides users with a summary of their financial health, including income, expenses, savings goals, and upcoming payments.



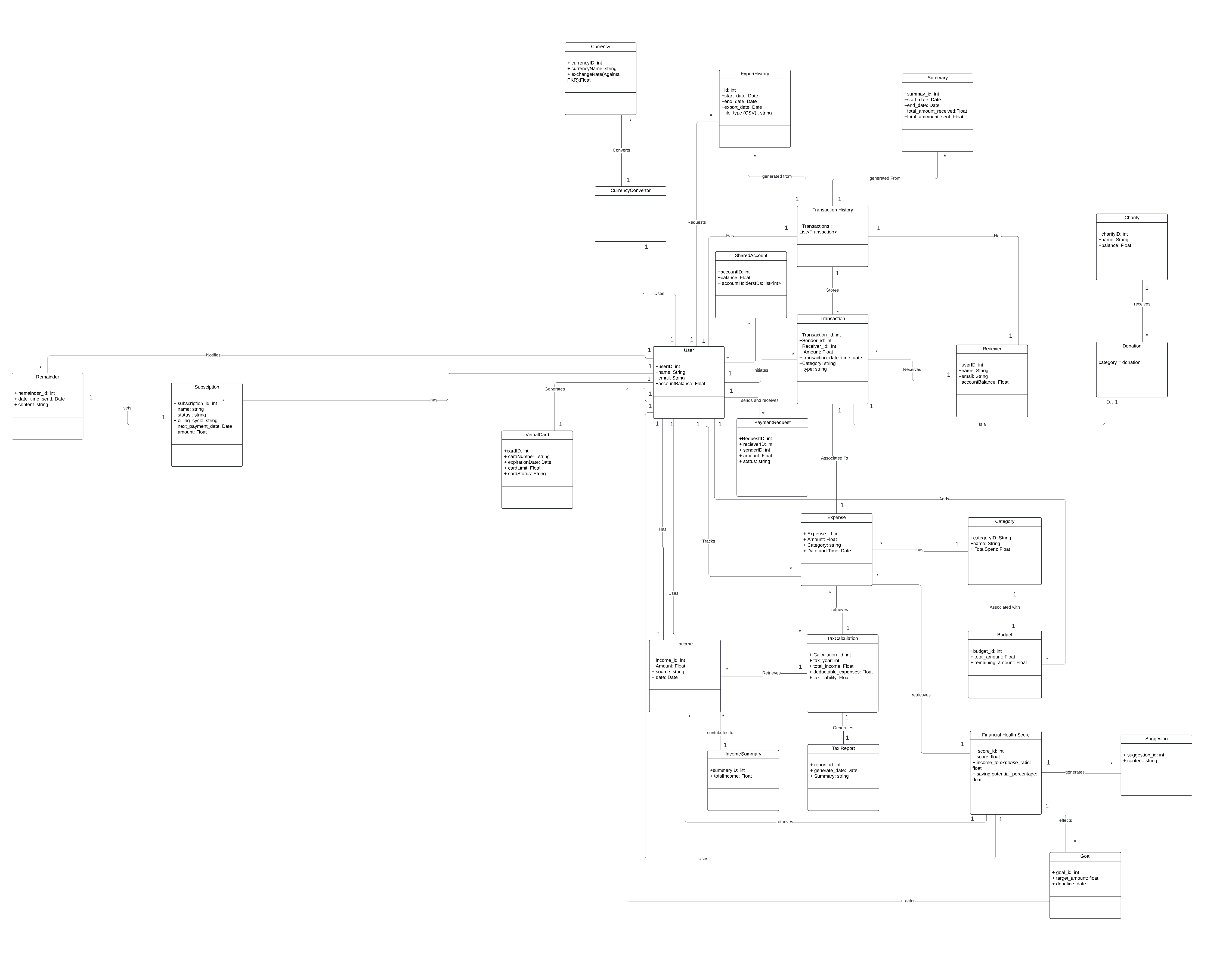
1. **Navigation and Consistency:**
   * All features will be accessible via a main navigation bar, ensuring ease of access.
   * Standard GUI elements, such as buttons, dropdowns, and icons, will follow consistent styling.
2. **Accessibility:**
   * Font sizes and contrast levels will comply with **WCAG 2.1 (Web Content Accessibility Guidelines)** to support users with visual impairments.
3. **Error Messaging:**
   * Clear and actionable error messages (e.g., "Insufficient funds for transaction") will guide users to resolve issues.



1. **Sample Screens:**
   * A consistent layout will be maintained across features like tracking expenses, managing budgets, and redeeming rewards.

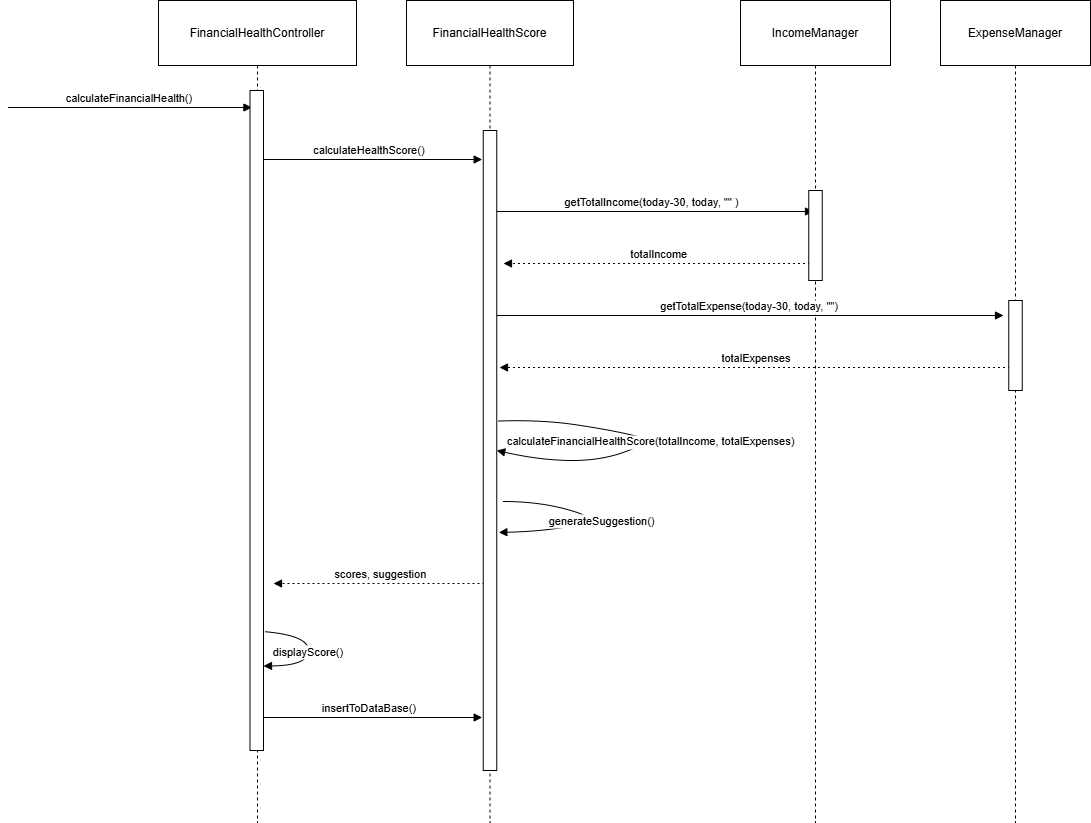
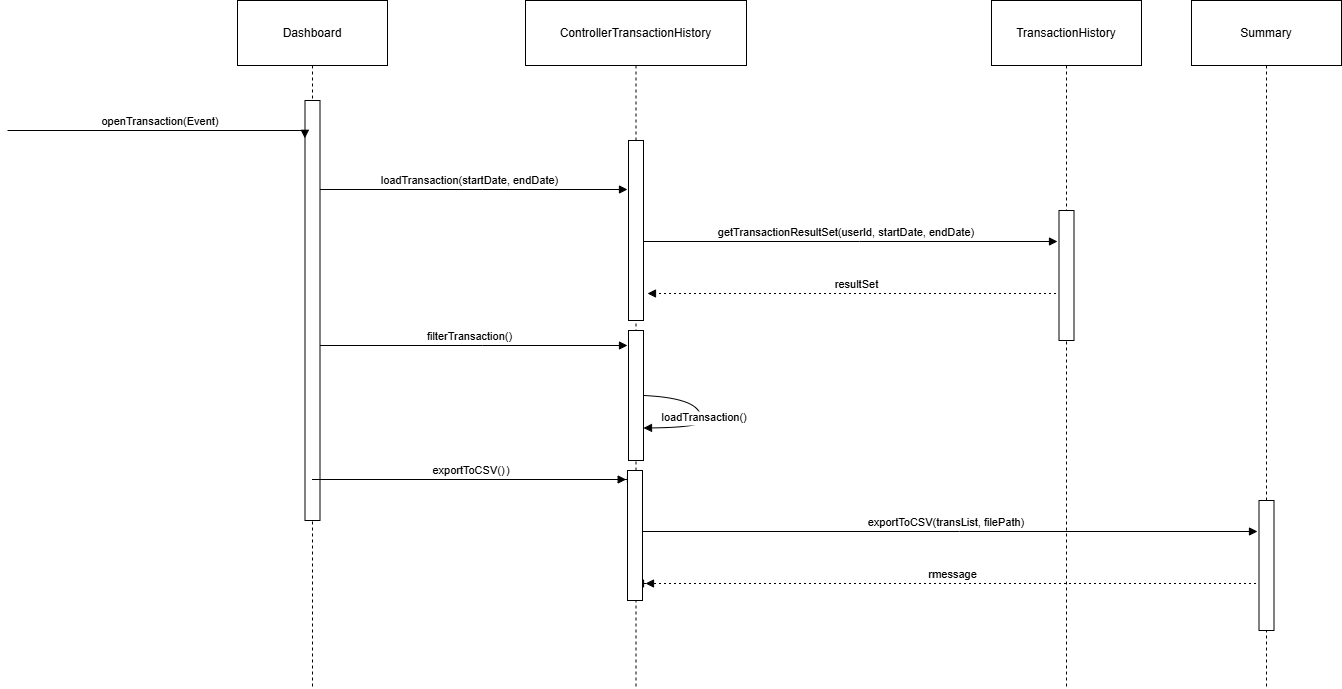
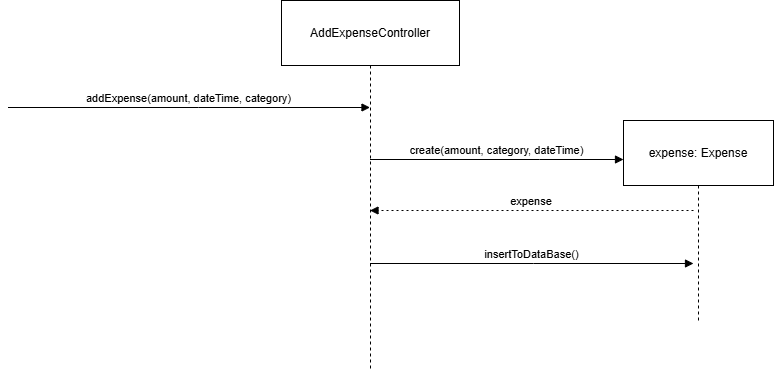
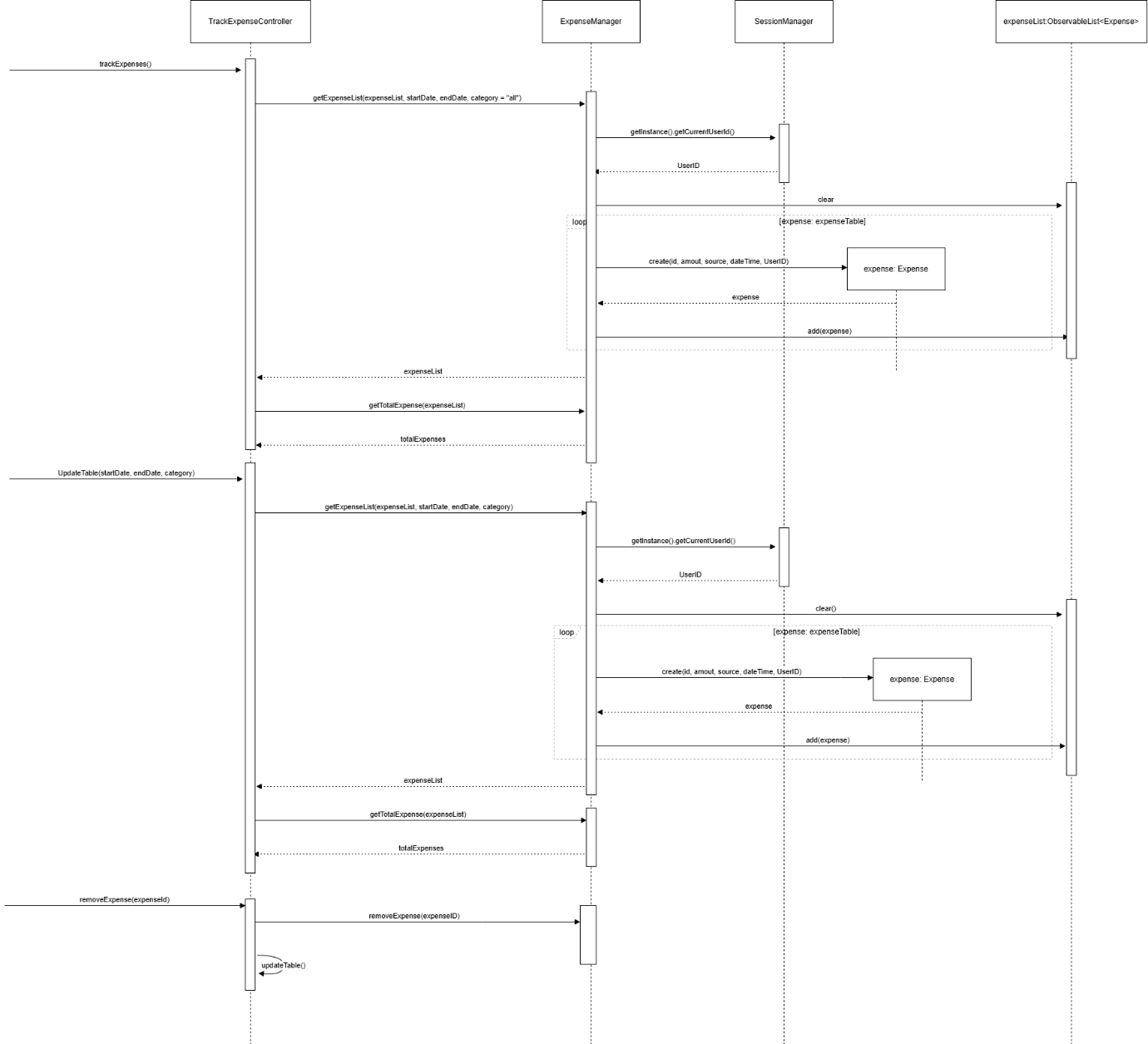
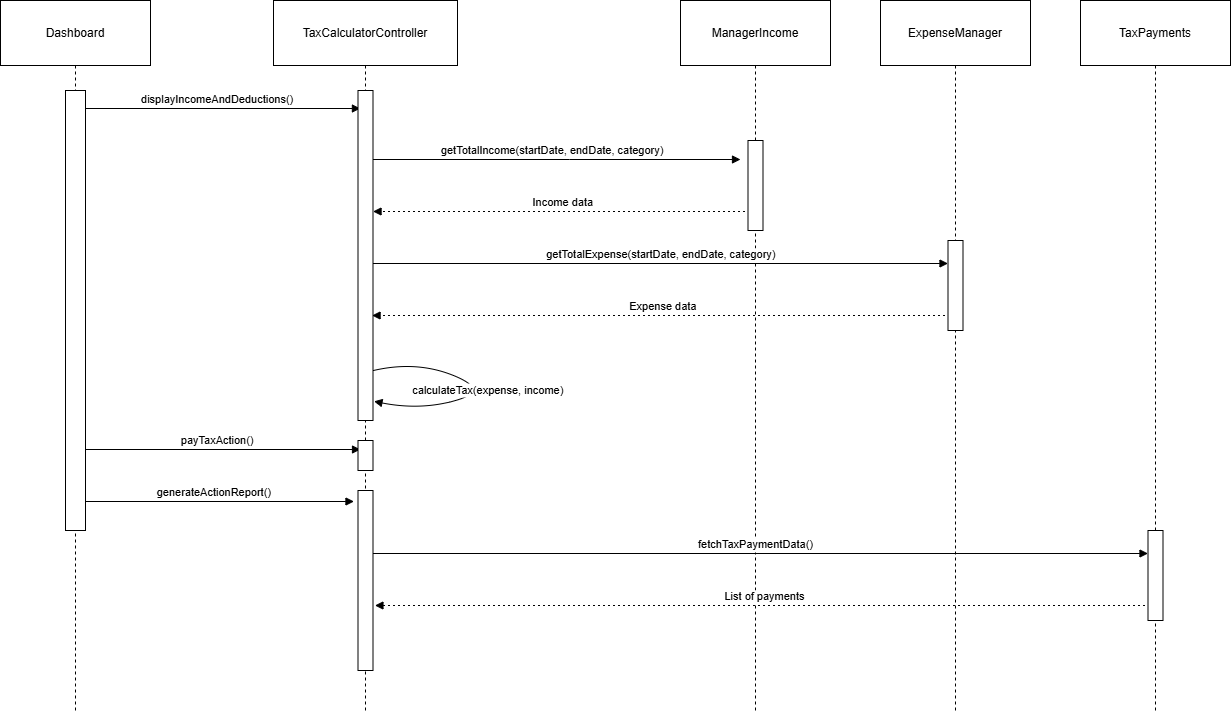
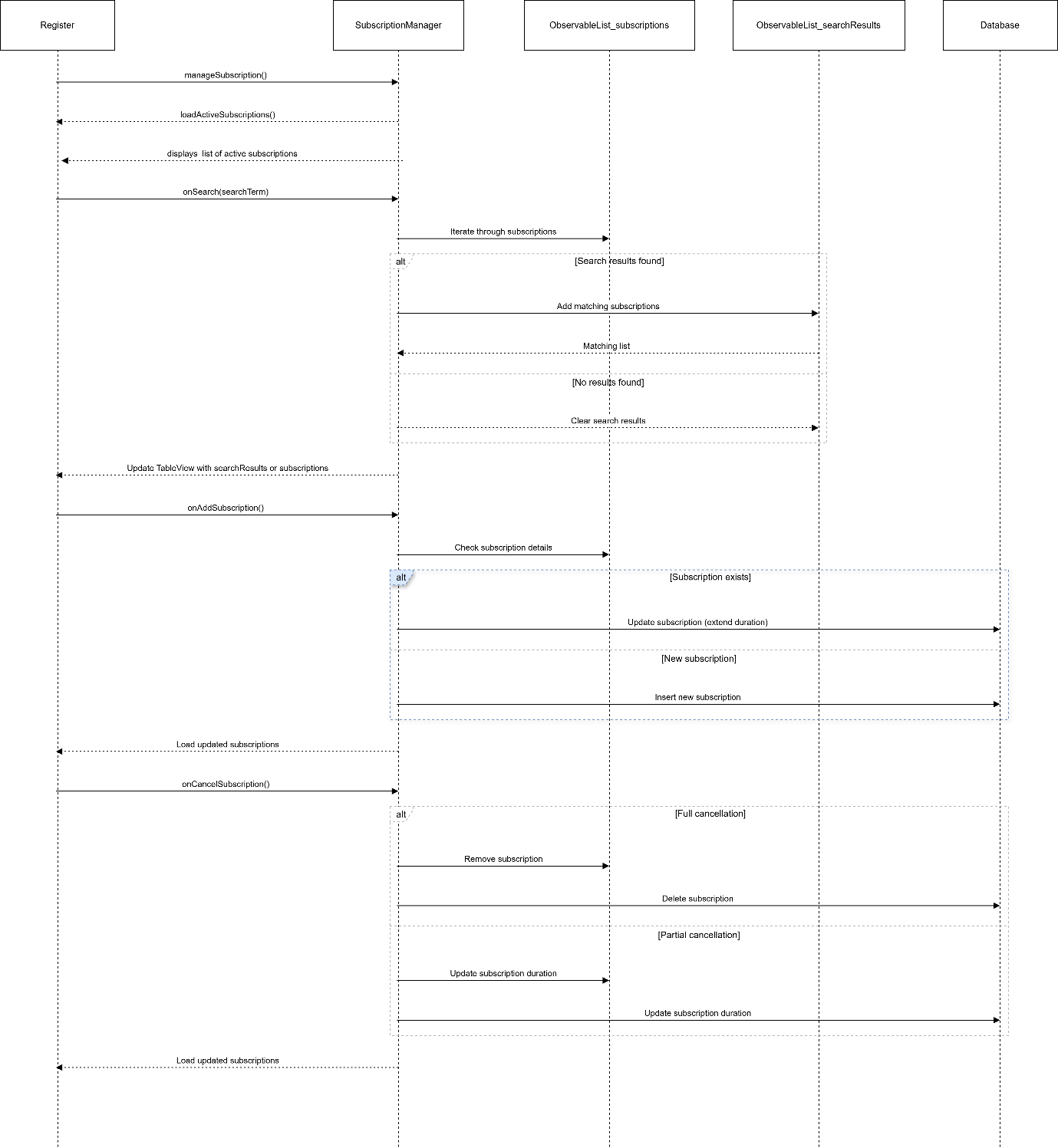
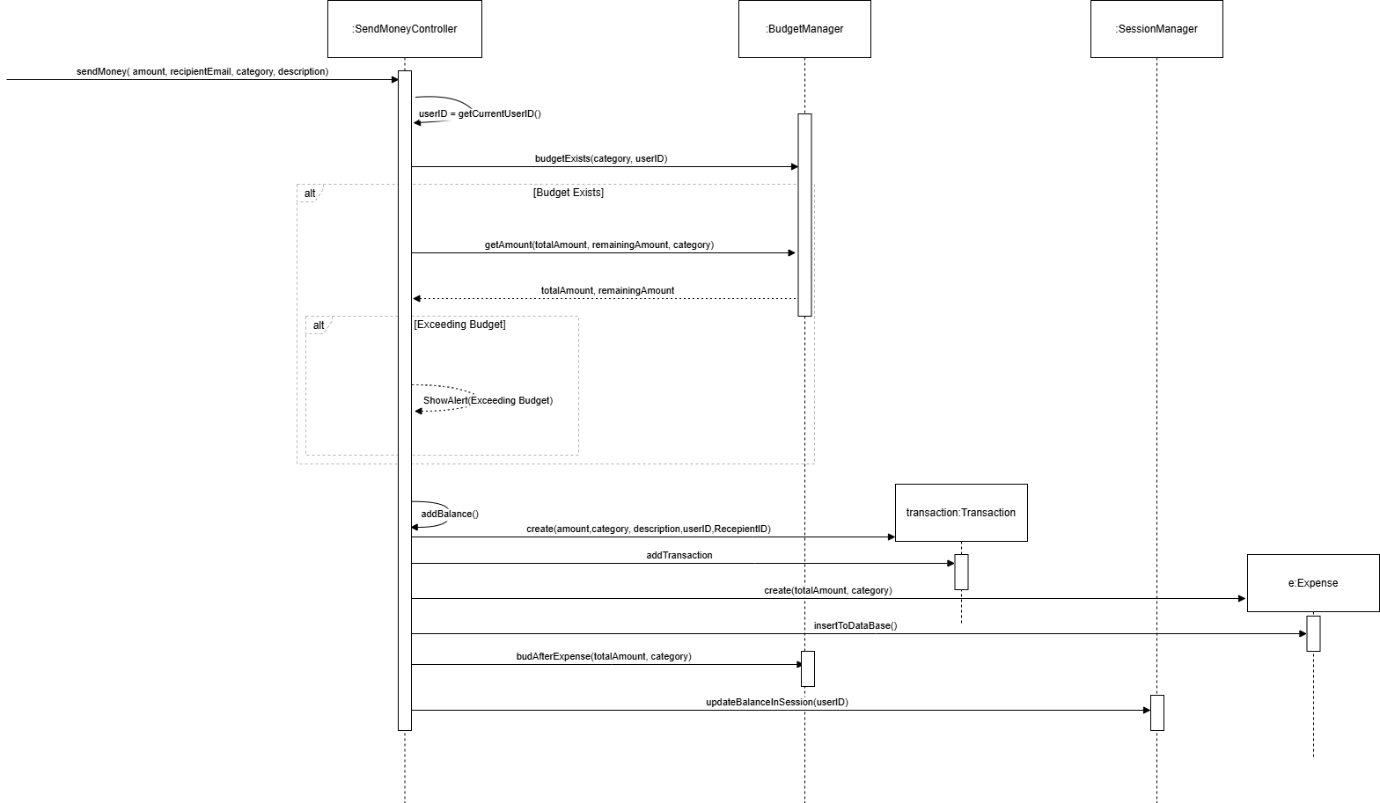


**4. Domain Model**

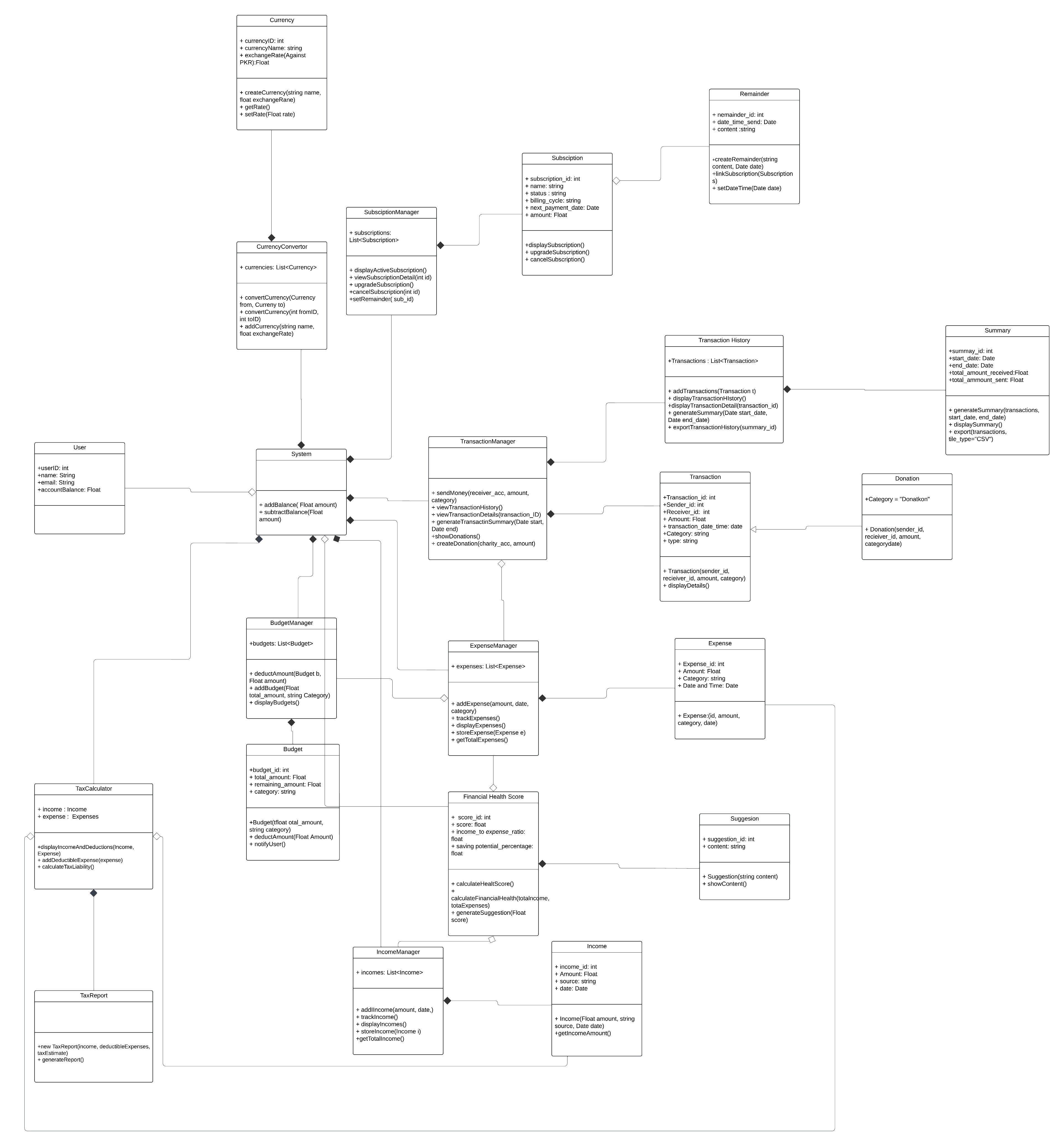


**5. System Sequence Diagram**

**6. Sequence Diagram**



**7. Class Diagram**



**8. Component Diagram**

* Illustrate the software components, their interactions, and how they contribute to the system's functionality. Include details like modules for finance tracking, budget planning, and transaction management.

**9. Package Diagram**

* Show the logical grouping of the system’s components into packages. This should include modules like **User Interface**, **Transaction Management**, and **Data Access**.

**10. Deployment Diagram**

* Depict the physical deployment of the software on hardware nodes, showing servers, databases, and end-user devices. Include connections and data flows.