Team 02 - M1 Technical Documentation

Limóney Technical Documentation

SW Engineering CSC648-848-05 Summer 2025

Limóney (Financial Budgeting WebApp)

Team 02
Milestone 1

7/2/25

Name	Role
Emily Perez (eperez@sfsu.edu)	Team Lead
Ishaank Zalpuri	Database Administrator
Andrew Brockenborough	Technical Writer
Dani Luna	Github Master, Database Administrator, Frontend Lead
Jonathan Gonzalez	Software Architect
Gene Orias	Backend Lead

Milestone	Version	Date
Milestone 1	Version 2	7/2/25
Milestone 1	Version 1	6/17/25

Table of Contents

Executive Summary

Limóney is an AI-enhanced budgeting web application designed to help users manage their finances all in one platform. We live in a time where many people struggle with managing recurring expenses, forgotten subscriptions, and inconsistent saving habits. Because of this, there's a growing need for financial tools that are not just useful, but also simple and easy to use. Our motivation is to keep the application both functional and accessible so everyday users can stay on top of their finances without feeling overwhelmed by data or complex interfaces.

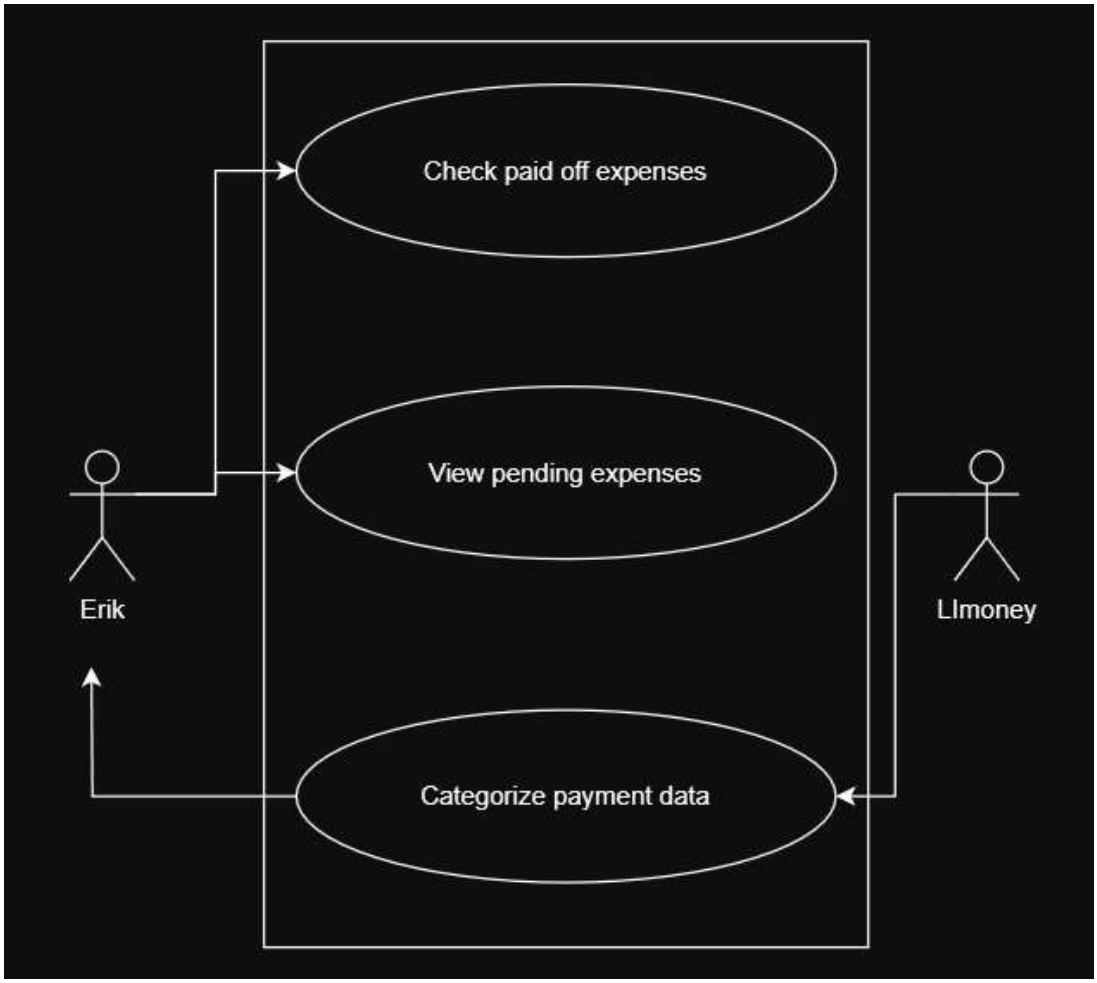
Inspired by *EveryDollar* and *Rocket Money*, what sets *Limóney* apart is that it combines the best features and usability of both apps into one platform, while also integrating AI to provide personalized, intelligent budgeting insights, which is something neither of those apps currently offer. The built-in assistant provides tailored spending suggestions, identifies unnecessary expenses, and recommends ways to save based on the user's financial habits. The result is a smarter, more intuitive budgeting experience that helps users improve their financial well-being over time.

Main Use Cases

Main Use Case 1

Create Budget

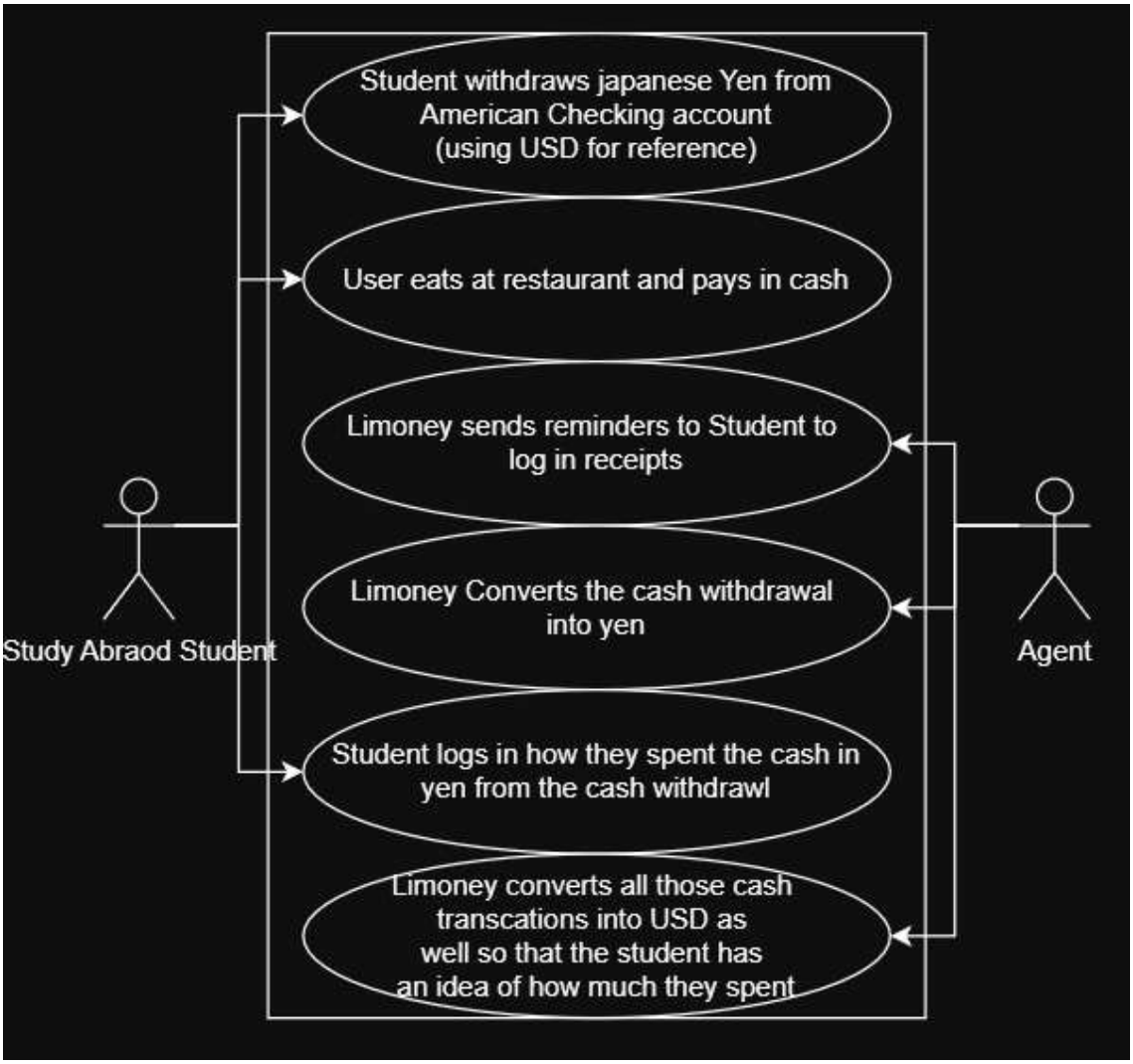
- **Actors:**
  - Erik
  - *Limóney*
- **Assumptions:**
  - Erik has multiple credit cards.
  - He has made recent payments, but is unsure of which expenses got paid.
  - Erik has connected his bank info to *Limóney* to keep track of his expenses.
- **Use Case:**
  - Erik made a credit card payment, but is unsure of what was actually paid. He opens up *Limóney*, and the app provides a clear list of which charges were paid off. For ex: his textbooks, school supplies, and what is still needed. Being able to see which expenses have been paid has given Erik peace of mind and helps him plan out his next payments with confidence.
- **Benefits for ...:**
  - Users:
    - Users gets to see which expenses have been paid.
    - Users gets help to decide which expenses to pay off next.
    - Users feel more in control of their credit and their spending


Check paid off expenses
View pending expenses
Categorize payment data
Erik
LImoney

Main Use Case 2

Tracking multiple reimbursements and predicting balances

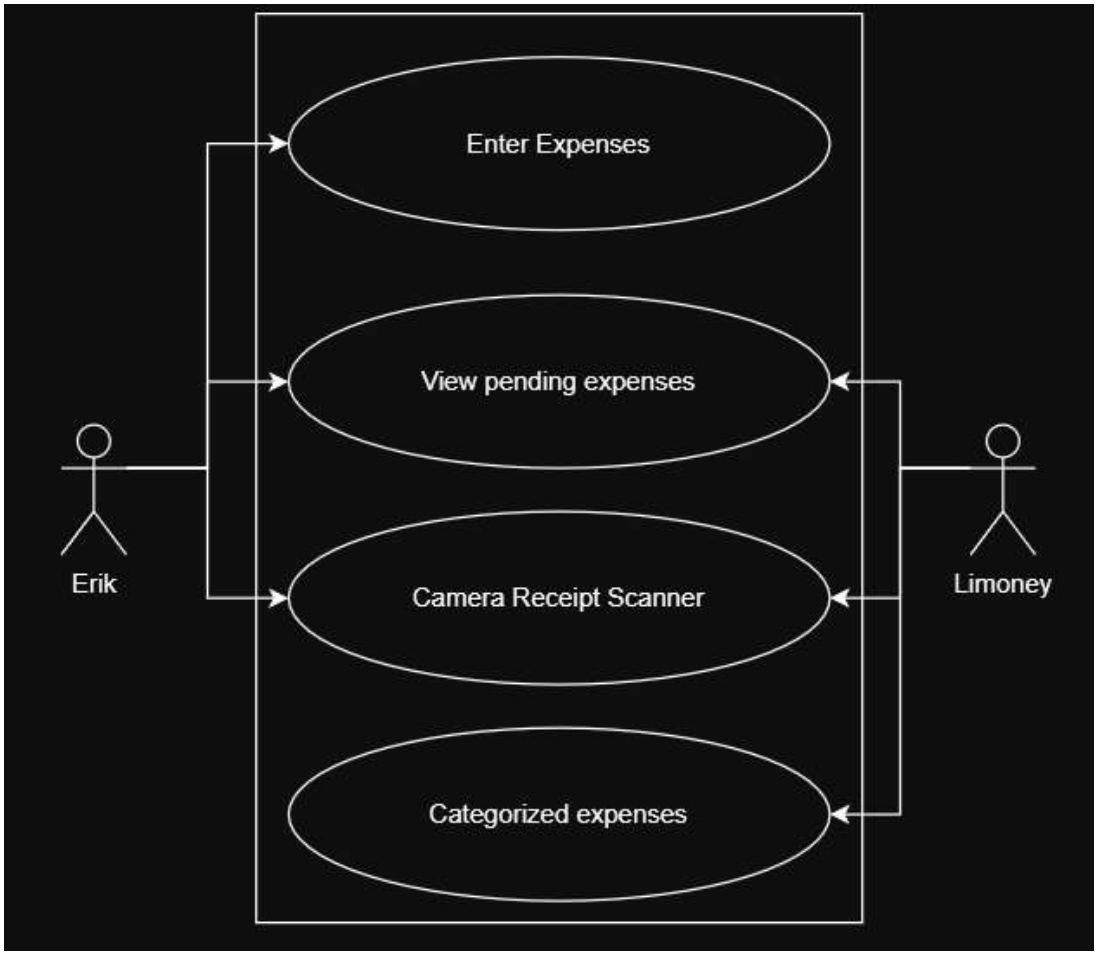
- **Actors:**
  - Student A
  - Student A's group of friends
- **Assumptions:**
  - Student A is somewhat financially stable
- **Use Case:**
  - Student A goes eating out with a group of friends once a month. Since they are a large group, it is usually difficult to split the bill among all of them. He pays the bill and expects his friends to reimburse him later, but everyone gets busy and can't keep track of how much needs to be paid back. The next time they go eating out, he puts the meal on his card. The student goes to the app right away and logs in the transaction. He also has the option of splitting the bill by a number of friends or splitting it by what they order and adding their names. These splits are called reimbursement requests, and the student can even send them to his friends as reminders. The request will be active until it is complete, and the money paid back will reflect on his account balances.
- **Benefits for...:**
  - Users
    - Users can keep track of who owes money back to him
    - Users are held accountable and become more financially responsible


Student withdraws japanese Yen from American Checking account (using USD for reference)
User eats at restaurant and pays in cash
Limoney sends reminders to Student to log in receipts
Limoney Converts the cash withdrawal into yen
Student logs in how they spent the cash in yen from the cash withdrawl
Limoney converts all those cash transcations into USD as well so that the student has an idea of how much they spent
Study Abraod Student
Agent

Main Use Case 3

Log Transactions

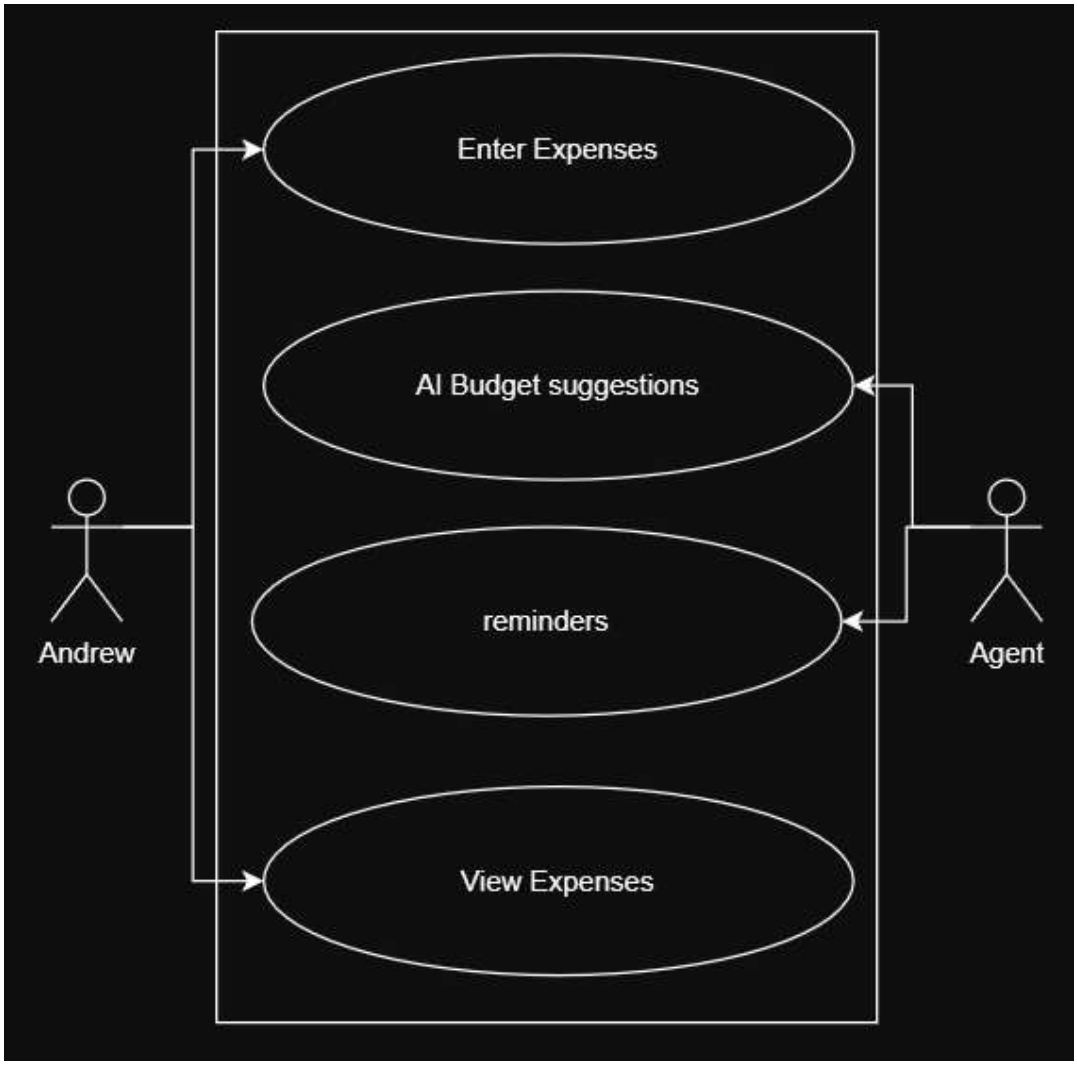
- **Actors**:
  - Erik (Busy College Student)
  - *Limóney* (Company)
- **Assumptions:**
  - Erik juggles school and work on a packed schedule.
  - He stores physical receipts but forgets to log them at the end of the day.
  - Erik prefers to manually enter his expenses for privacy reasons.
  - He does not use updated bank tools.
- **Use Case:**
  - Erik tries to keep track of his spending, but with work and school, he tends to forget about his receipts, which pile up. Logging the receipts by hand feels like too much. One day, *Limóney* sends out a reminder and suggests that he can scan the receipts via his camera instead. Erik tries it and is surprised by how quickly and efficiently it works. Erik is now logging expenses through *Limóney* because it provides a simple way to keep track of his receipts and has become a nightly routine.
- **Benefits for ...:**
  - Users:
    - Users have an easier time logging receipts with the camera scanner.
    - Users get helpful reminders help Erik stay on track.
    - Users with no technical skill have an easier time managing their finances.


Enter Expenses
View pending expenses
Camera Receipt Scanner
Categorized expenses
Erik
Limoney

Main Use Case 4

Security concern

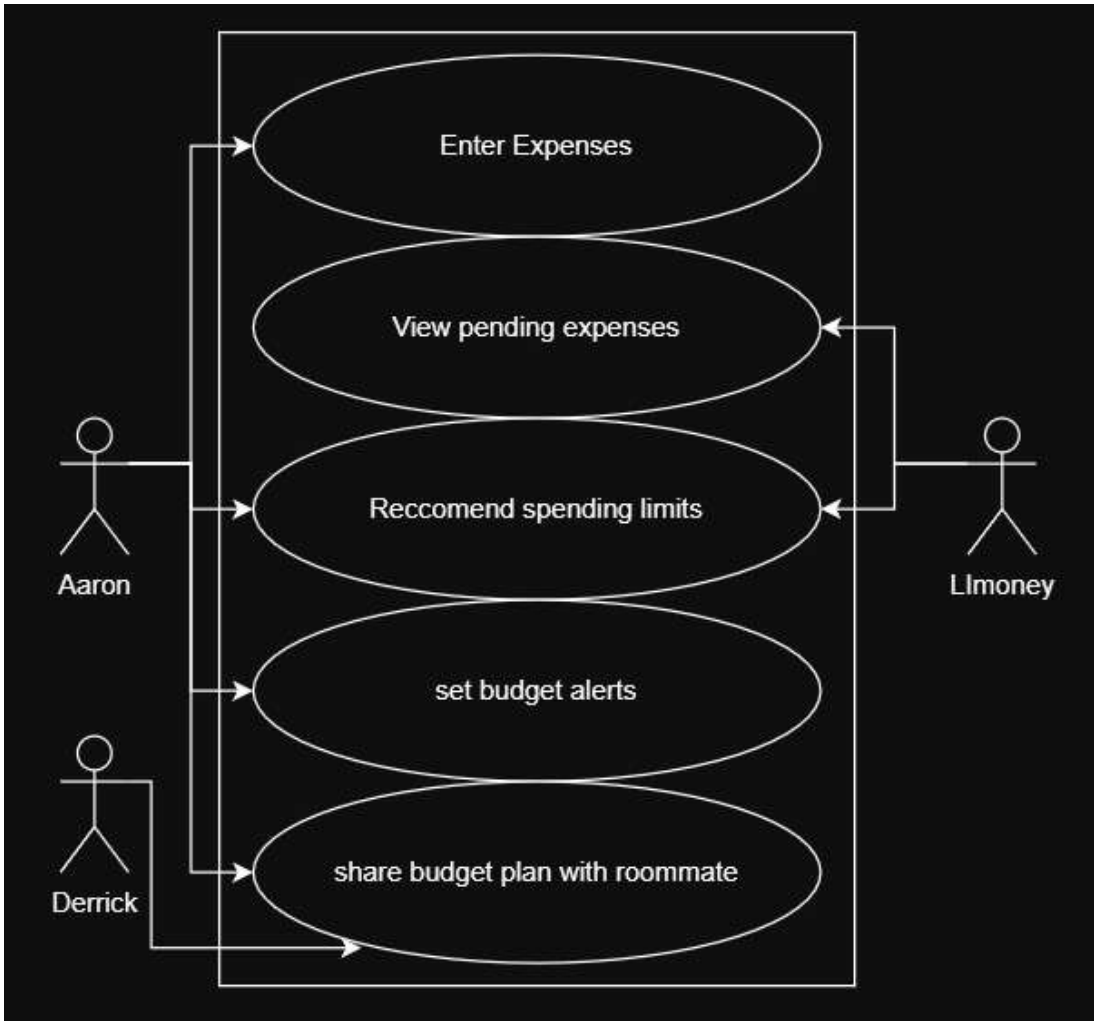
- **Actors:**
  - Andrew
  - *Limóney*
- **Assumptions:**
  - The user manually enters receipts
  - They don’t have the latest technologies
  - They don’t want to use apps that access their financial information
- **Use Case:**
  - Andrew is serious about his privacy and refuses to use financial apps that require access to his bank account or store data externally. For years, he has been using spreadsheets to track his spending manually. The process has become overwhelming to do. He discovers *Limóney* , a secure, AI-enhanced budgeting app designed for users like him. *Limóney* allows users to enter transactions manually and get personalized budgeting insights without ever syncing financial accounts. Andrew starts using *Limóney* interface to log purchases. The built-in AI analyzes his patterns and suggests simple ways to save like avoiding frequent late-night food orders. Andrew gets skeptical on AI and has the choice of turning off the AI suggestions. What surprises Andrew most is how nice it feels: he stays in control, but still benefits from smart financial planning tools
- **Benefits for …:**
  - Users:
    - Users can manage finances without linking sensitive financial accounts.
    - Users get smart recommendations based on spending habits.
    - Users know their data stays local and private.


Enter Expenses
AI Budget suggestions
reminders
View Expenses
Andrew
Agent

Main Use Case 5

Receive AI Suggestions

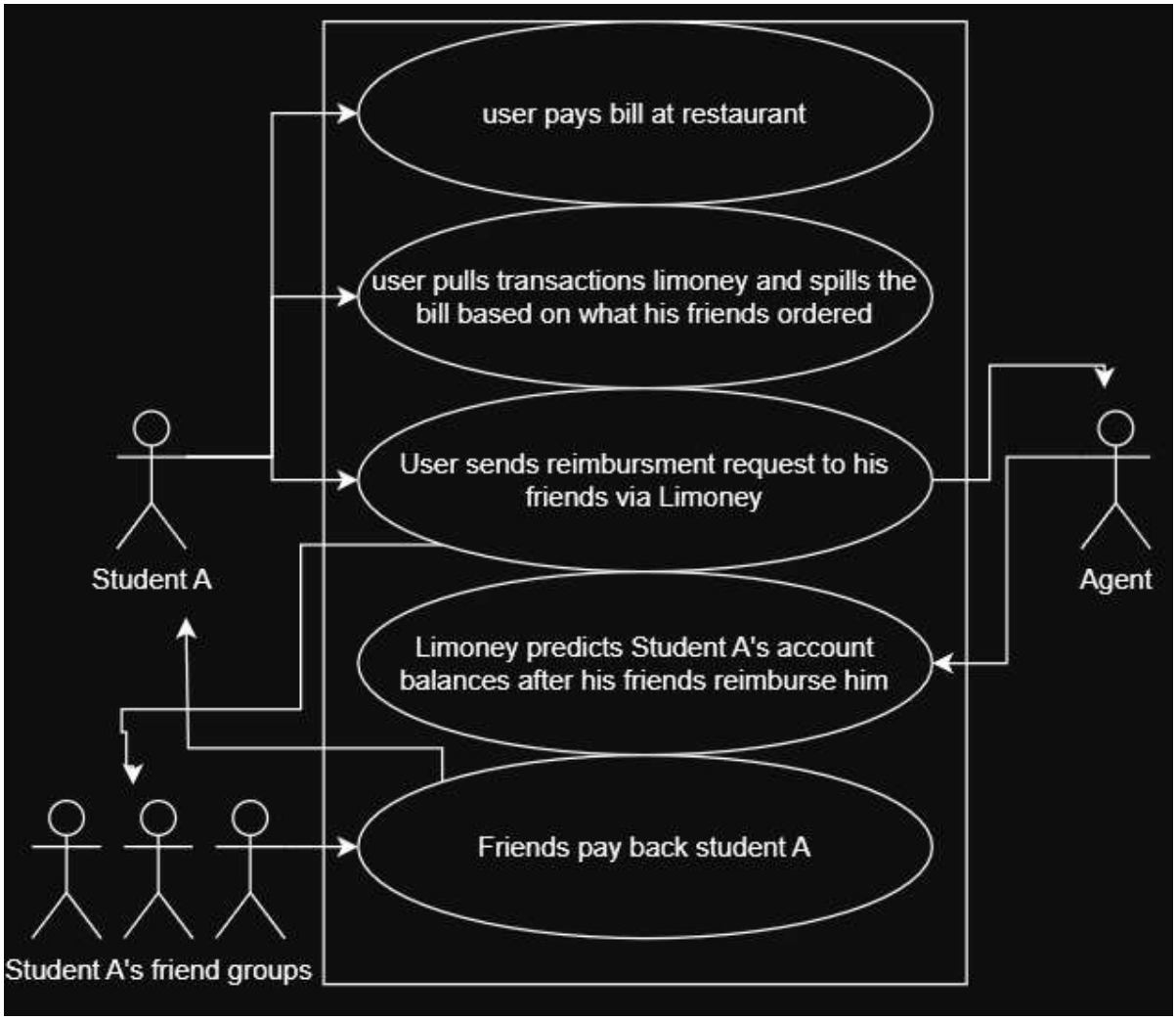
- **Actors:**
  - Aaron
  - Derrick (Aaron's Roommate)
  - *Limóney*
- **Assumptions:**
  - Aaron and Derrick have separate bank accounts and budgets in *Limóney*
  - Aaron has connected his banking info to the *Limóney* App
- **Use Case:**
  - On a Friday evening, Aaron opens the *Limóney* App to figure out why he keeps running low on money every month. The *Limóney* app, quickly points out that his dining expenses are higher than usual and suggests a closer look. Aaron notices that he has several delivery charges that had been adding on. The *Limóney* App recommends that Aaron sets up a weekly limit which will allows him to see how much he can potentially be saving. Relieved by this, Aaron sets a spending alert and feels in control of his budget and shares it with Derrick. Aaron and Derrick decide that cutting back on dining expenses can save them a lot so they begin cooking meals at home together and decrease their spending.
- **Benefits for ...:**
  - Users:
    - Users can understand exactly where their money is going.
    - Users can set spending alerts and adjust their budget in real time.
    - Users feel confident and are more proactive with their savings.


Enter Expenses
View pending expenses
Reccomend spending limits
set budget alerts
share budget plan with roommate
Aaron
Derrick
LImoney

Main Use Case 6

Currency Exchange

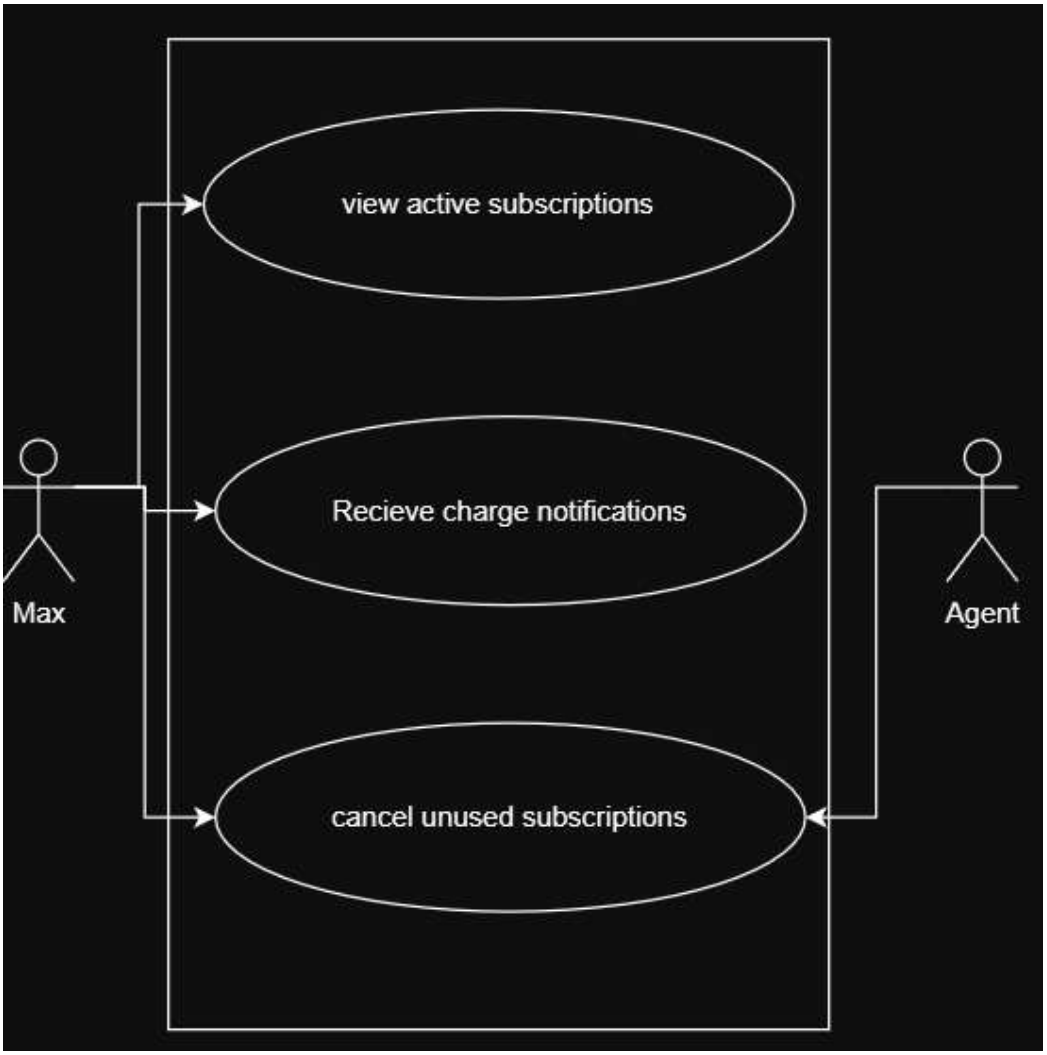
- **Actors:**
  - Study Abroad Students
- **Assumptions:**
  - Some Students struggle to adjust to foreign currency and want to get used to the system
- **Use Case:**
  - Study abroad student goes to Japan. The country paper based, most restaurants only take cash. Whether the student opts to manually track expenses or link his bank account, he needs to withdraw cash and keep a track of how he spends it. It gets confusing however as he collects receipts and the currency rate changes every day. By the time he sits down to enter his cash transactions, he has to spend extra time converting currency. Instead of stressing out, he chooses to switch the currency on the app to Japanese Yen so he can get used to the currency abroad and accurately log how he used his cash withdrawal. At the end, he also converts all those transactions back into USD to get an idea of how much he is spending and what the Japanese economy is like
- **Benefits for...:**
  - Users:
    - Users can keep a track of their expenses abroad and get used to foreign currency.


user pays bill at restaurant
user pulls transactions limoney and spills the bill based on what his friends ordered
User sends reimbursment request to his friends via Limoney
Limoney predicts Student A's account balances after his friends reimburse him
Friends pay back student A
Student A
Agent
Student A's friend groups

Main Use Case 7

Tracking ongoing expense

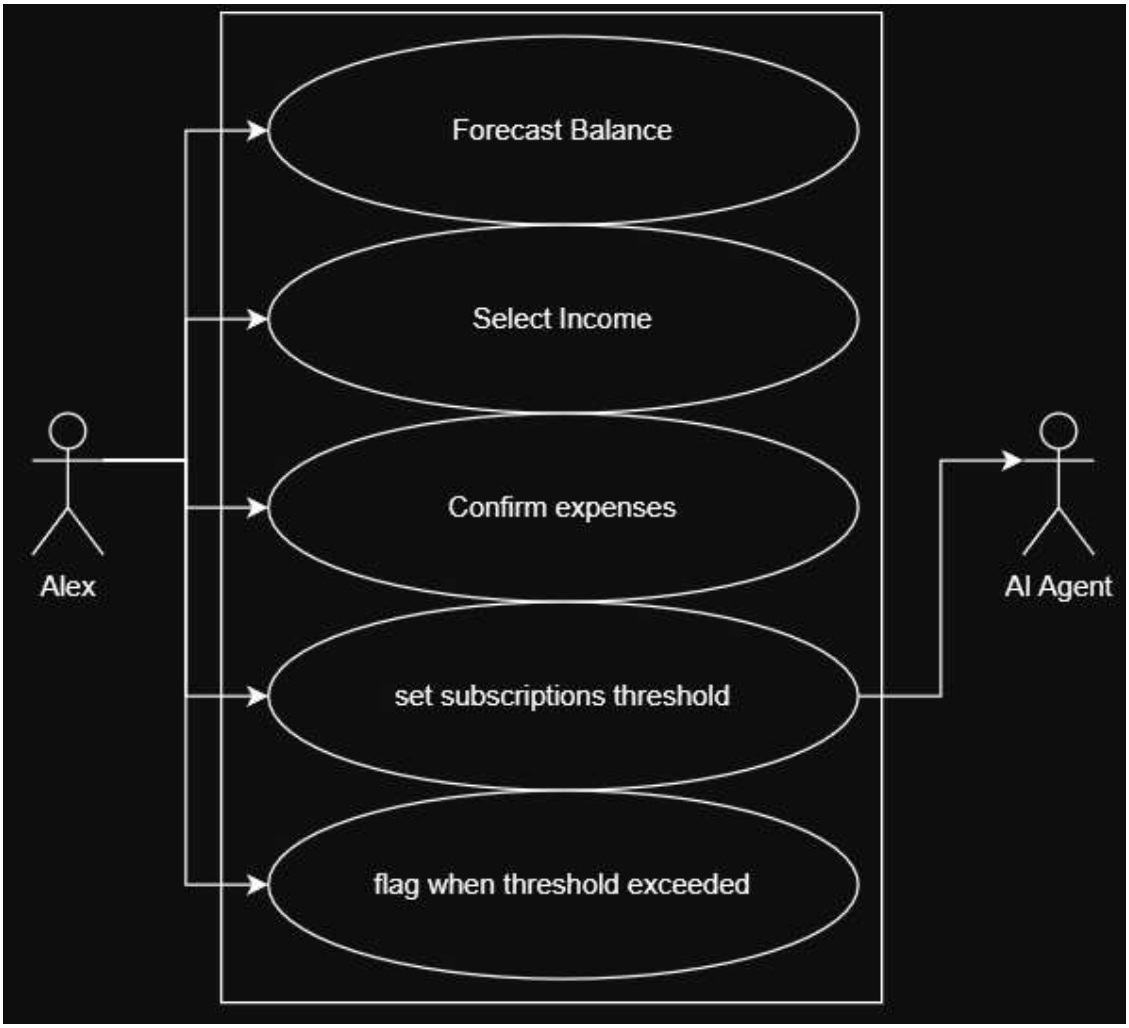
- **Actors:**
  - Max who has a lot of active subscriptions
  - Max doesn’t really know much about the digital landscape
- **Assumptions:**
  - Max has a lot of active subscriptions
- **Use Case:**
  - Max is a busy college student juggling classes, a part time job, and taking care of his siblings. Lately he has been feeling very overwhelmed and doesn’t have time to check his bank account. Max suddenly got a notification on his phone that he has been charged 20$ for a subscription he didn’t even know he had. Max then downloaded the *Limóney* app which can help him manage his subscriptions. Instantly he gets greeted by the active subscriptions he has on right now! With a few taps those useless subscriptions have been cancelled!
- **Benefits for ...:**
  - Users:
    - Users can prevent ongoing subscriptions as to easily see everything.
    - Users have the ability to see unwanted or unused subscriptions.


view active subscriptions
Recieve charge notifications
cancel unused subscriptions
Max
Agent

Main Use Case 8

Predicting balances end of month

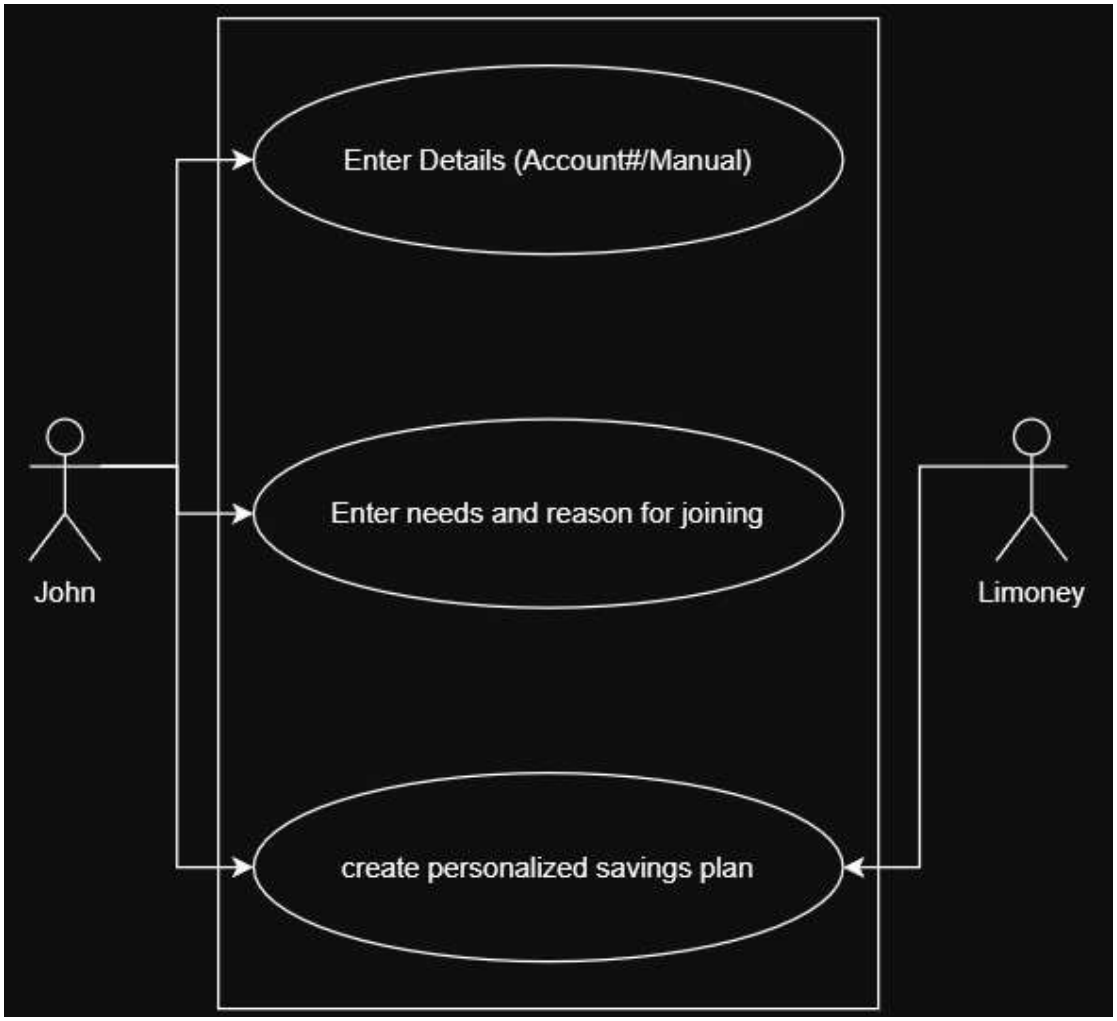
- **Actors:**
  - Alex who loves planning ahead
- **Assumptions:**
  - Alex has stability within expense and income
  - Alex updates their transactions regularly
- **Use Case:**
  - Alex, who is a planner by nature, wants to know if he will be financially comfortable by the end of the month. He opens the *Limóney* app and selects “Forecast Balance”. The app prompts him to select his income and confirms his current expenses like subscriptions. Alex then sets a threshold of 100$ within subscriptions so the app can flag Alex whenever he goes above that mark.
- **Benefits for ...:**
  - Users:
    - Users can completely avoid disturbance within the stability of expenses
    - User can be aware of any balances that could surprise then and can see their subscription bills easily and adjust accordingly.


Forecast Balance
Select Income
Confirm expenses
set subscriptions threshold
flag when threshold exceeded
Alex
AI Agent

Main Use Case 9

Signing up/Intro UI

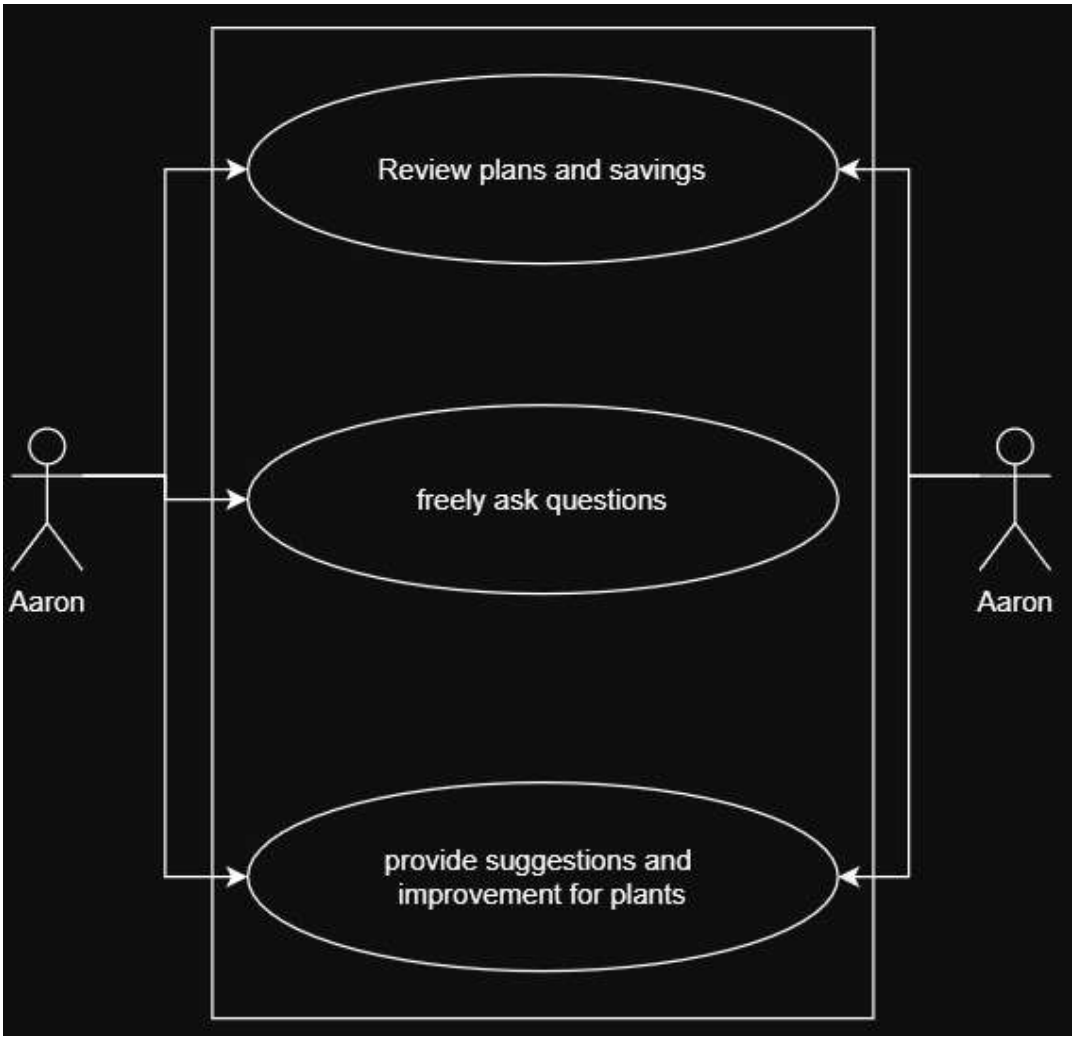
- **Actors:**
  - John
  - *Limóney*
- **Assumptions:**
  - John wants help with budgeting his expenses
  - John finds *Limóney* ad interesting and clicks
- **Use Case:**
  - John is worried about his finances. He is concerned about going broke. Enter *Limóney*, a budgeting app he found in an ad. He is now signed up, and it is now asking him questions. It begins with why John is on the app. What does he need help with? (budgeting, managing expenses, investment, tracking subscriptions) Afterwards, it asks if John wants to share his data using his SS and banking info, or if he prefers to manually enter his info. If he chooses the latter, it takes him to fill out the information he needs help with (income, average expenses, current bills, etc.). He then receives a message welcoming him to *Limóney* and a well-wishing of “Happy Savings!”, with an immediate option to work with *Limóney* to start saving money.
- **Benefits for…:**
  - Users
    - Users can feel welcomed and impressed by the application while giving them confidence that they will be taken care of.
  - Company
    - The company will benefit from the increase of the application' traffic through word of mouth.


Enter Details (Account#/Manual)
Enter needs and reason for joining
create personalized savings plan
John
Limoney

Main Use Case 10

Chatbot/AI assistance

- **Actors:**
  - John
  - LemonAid
- **Assumptions:**
  - John is familiar with the concept of a Chatbot and feels comfortable using it, understanding it is only for reference and will not be 100% accurate
  - John likes the idea of having his entries analyzed and would like to have an outside perspective on his savings without sacrificing his privacy
- **Use Case:**
  - John has successfully signed up and is enjoying using *Limóney*. However, all the technical jargon and the routine of inputting and reading numbers gets tiring. He wants to take a bit of a break and wants to be able to discuss his finances with others. He opens LemonAid, the company's financial aid assistance, and asks it for help. To streamline the process, LemonAid asks John how exactly it can help him today, while providing several recommendations that he can click on or directing him to enter his own question in the text field.
- **Benefits for ...:**
  - Users:
    - Users shall be able to discuss their finances with an AI assistant, allowing them not only to receive feedback from it, but also allowing them to think more deeply on the subject matter they are asking about. All while maintaining their privacy and not being required to share their finances with others.
  - Company:
    - The company shall be able to streamline the feedback process and make communication much easier. If users choose to share their chat information, the company can use it to further develop the AI and train it to be prepared for more situations.


Review plans and savings
freely ask questions
provide suggestions and improvement for plants
Aaron
Aaron

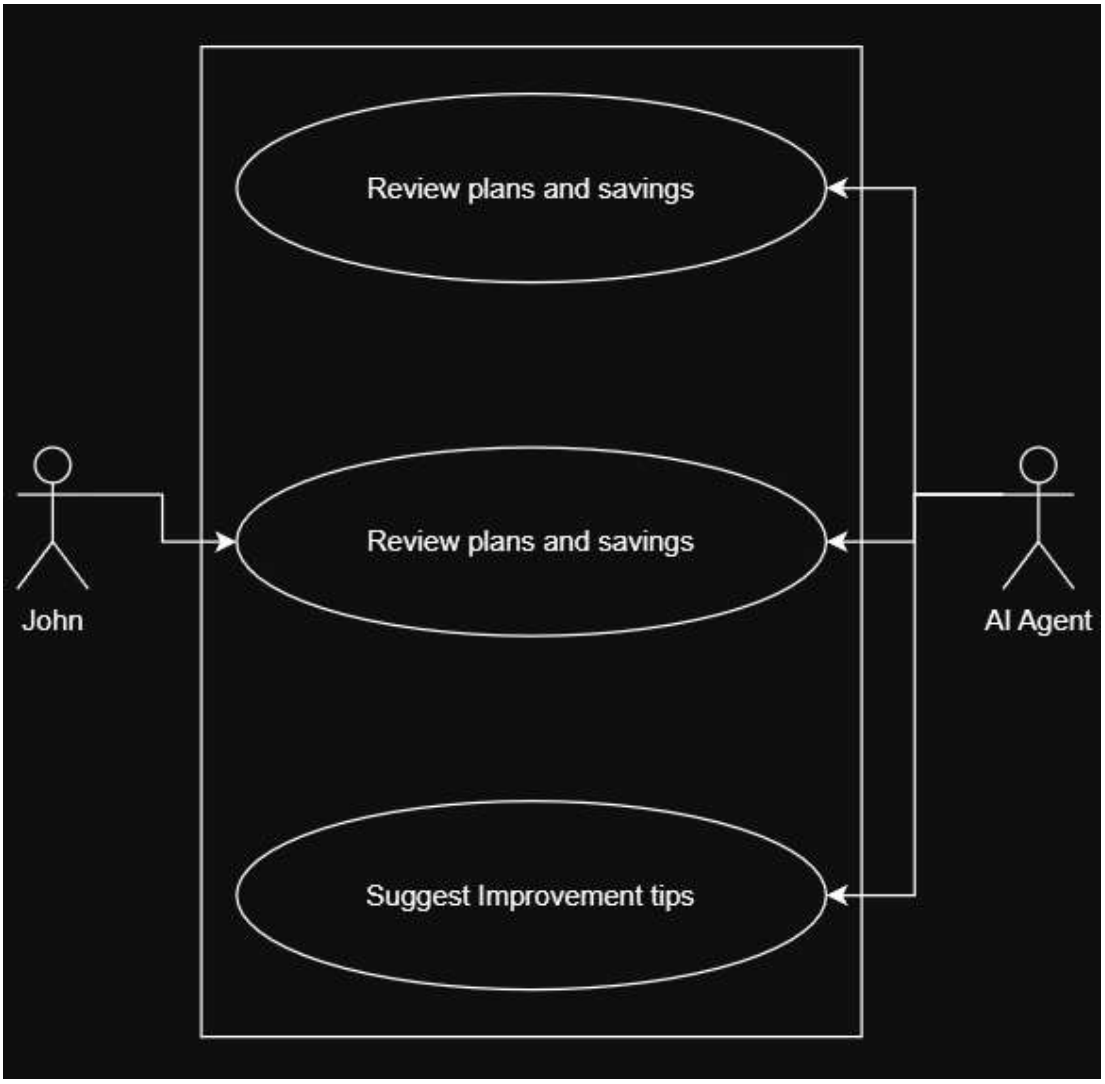
Main Use Case 11

Credit Score Inquiry

- **Actors:**
  - John
  - LemonAid
- **Assumptions:**
  - John has linked his credit tracking service to his *Limóney* account.
  - John understands the basics of a credit score and knows it will fluctuate over time.
  - John is comfortable sharing just enough personal data with LemonAid to receive his score, and trusts that no full SSN or sensitive PII data is stored.
- **Use Case:**
  - John logs into *Limóney* and clicks the “LemonAid” chat icon. LemonAid greets John and offers a list of quick actions: “Check my current credit score.” ,“Explain what hurts my score.”, “Show me tips on how I can improve my score.” John clicks “Check my current credit score.” LemonAid prompts: “Please confirm the last four digits of your SSN to fetch your score securely.” John enters the digits; LemonAid calls the credit API. LemonAid displays John’s FICO score (ex., 740), the date pulled, and a simple grade tier (Good, Fair, etc.). John asks a follow-up: “What factors pulled me down this month?” LemonAid lists the top 3 factors (ex., “Credit utilization rose to 42%,” “One late payment 30 days past due,” “New hard inquiry”). LemonAid then asks: “Would you like personalized tips to boost your score?” and offers buttons like “Lower utilization,” “Automate payments,” or “Order a sec. trade-line.” John selects “Automate payments,” and LemonAid walks him through setting up auto-pay on his credit cards.Session ends; John closes the chat, feeling informed about his current score and concrete next steps.
- **Benefits for …:**
  - Users:
    - Users can instantly see your up-to-date credit score and major contributing factors without navigating a complex dashboard.
    - Users can get targeted, personalized tips that tie directly to your credit report data.

        - Users are promised to have their privacy preserved**.** Only minimal, encrypted data (last four of SSN, account tokens) is used—no full PII is stored or shared.

    - Company:
        - The company will have users spend more time in *Limóney* and perceive higher value in the platform's AI assistance.
        - The company has cross-sell opportunities. When LemonAid suggests paid services (ex., identity monitoring and credit builder loans), conversion rates rise.
        - The company can refine LemonAid's credit-score recommendations with opt-in anonymized transcripts and expand to new credit products.


Review plans and savings
Review plans and savings
Suggest Improvement tips
John
AI Agent

List of Main Data Items and Entities

This page is a list of Item/Object/Entity concepts that will be used to define interactions on the Limóney web-app as well as the direction of the budgeting process

---

1. Main Entity: Users

Defines all types of users within the application.

1.1 Role Type: Customer

- **Definition:** Basic user type who utilizes the app's core features.
- **Behavior:** Can view finances, complete tasks, interact with AI, and earn rewards.

1.2 Role Type: Premium

- **Definition:** Enhanced user role with access to exclusive financial features.
- **Behavior:** Includes all customer features, plus extras.

1.3 Role Type: Tester

- **Definition:** Temporary or experimental users who test new features or versions.
- **Behavior:** Provides feedback based on testing sessions.

1.4 Role Type: Administrator

- **Definition:** Pinnacle of users with control over platform settings and user management.
- **Behavior:** Can manage users, resolve tickets, and oversee operations.

---

2. Main Entity: Tasks

- **Definition:** Actions users can perform to manage or improve finances.
- **Behavior:** Users can complete, receive, and view tasks, may be may be recommended by LemonAid.

---

3. Main Entity: Account

Central system for tracking money flow, balances, and budgeting

3.1 Role Type: Income

- **Definition:** Tracks the sources and amounts of funds received.
- **Behavior:** Adds to account balance and informs budgeting and progress metrics.

3.2 Role Type: Expense

- **Definition:** Records money spent by the user.
- **Behavior:** Subtracts from account balance, contributes to budgeting, and is analyzed for trends.

3.3 Role Type: Saving

- **Definition:** Entity related to saving money.
- **Behavior:** Tracks actions taken, or that can be taken, by user in order to improve this in a stat based fashion.

---

4. Main Entity: Bank

Banks connected to user accounts

4.1 Role Type: Credit

- **Definition:** Amount of money that the User is in debt.
- **Behavior:** Tracks user debt from using cards, taking loans, etc.

4.2 Role Type: Total

- **Definition:** Total of user bank assets.
- **Behavior:** Can determine User net-worth, and be used with other entities to determine savings and other things.

---

5. Main Entity: Transaction

- **Definition:** Represents the movement of money within an account.
- **Behavior:** Provides data to other entities to help improve user savings.

---

6. Main Entity: Budget

- **Definition:** Helps users control and plan spending.
- **Behavior:** Tracks and provides limitations on user spending.

---

7. Main Entity: Subscription

- **Definition:** Payments for installments towards different services and plans.
- **Behavior:** Used to calculate user spending and improve spending by eliminating subscription over-population.

---

8. Main Entity: Receipt

- **Definition:** Uploaded photo of receipt.
- **Behavior:** Helps with reimbursement and tracking finances.

---

9. Main Entity: Reimbursement Request

- **Definition:** Request for financial compensation.
- **Behavior:** Is created by user and reviewed by Admin that will take action to aid user.

---

10. Main Entity: Rewards

Unique aspect of Limoney that increases user engagement by turning saving money into a game.

10.1 Role Type: Level

- **Definition:** Value tracking user progress.
- **Behavior:** Unlocks privileges and visualizes achievements.

10.2 Role Type: Rating

- **Definition:** Ranks users competitively against each other.
- **Behavior:** Encourages engagement by adding a competitive layer for those intrested.

10.3 Role Type: Redeem

- **Definition:** Converts points into tangible or virtual rewards.
- **Behavior:** Provides incentives and gratification for app usage.

---

11. Main Entity: Notifications

- **Definition:** In-app messages that alert users to important events or tasks.
- **Behavior:** Informs users about tasks, budget limits, etc.

---

12. Main Entity: Support Ticket

- **Definition:** A message submitted to administrators for technical or financial assistance.
- **Behavior:** Sent to Admins for resolutions.

---

13. Main Entity: LemonAid (AI Assistant)

- **Definition:** AI Assistant integrated into the app.
- **Behavior:** Interacts with users, logs actions, and improves user guidance using AI.

Initial List of Functional Requirements

1. User

1.1. A user shall be able to create an account.
1.2. A user shall be able to securely log in to their account.
1.3. A user shall be able to recover or reset their password.
1.4. A user shall be able to select between manual or bank-linked financial tracking.
1.5. A user shall be able to customize their interface (e.g., dark mode).
1.6. A user shall be able to receive AI-generated financial guidance.
1.7. A user shall be associated with one or more accounts.
1.8. A user shall be classified as a customer, premium, tester, or administrator.
1.9. A user shall be able to interact with the AI chatbot for financial assistance.
1.10. A user shall be able to view and edit their financial data history.

---

2. Account

2.1. An account shall store income entries for a user.
2.2. An account shall store expense entries for a user.
2.3. An account shall belong to one and only one user.
2.4. An account shall track a balance value.
2.5. An account may be connected to one or more banks.
2.6. An account shall be able to categorize transactions.
2.7. An account shall have a transaction history log.
2.8. An account shall support spending limits and alerts.
2.9. An account shall track subscriptions and bill payments.
2.10. An account shall be able to forecast end-of-month balances.

---

3. Bank

3.1. A bank shall be linked to one or more user accounts.
3.2. A bank shall provide transaction data to the system.
3.3. A bank shall store balance and credit data for users.

4. Transaction

4.1. A transaction shall be classified as income or expense.
4.2. A transaction shall belong to exactly one account.
4.3. A transaction may be manually entered or synced from a bank.
4.4. A transaction shall be categorizable by the user or AI.
4.5. A transaction shall be editable and deletable by the user.
4.6. A transaction shall be linked to one or more receipts.
4.7. A transaction may be associated with a reimbursement request.

5. Subscription

5.1. A subscription shall be tied to a recurring transaction.
5.2. A subscription shall be viewable in a centralized dashboard.
5.3. A subscription shall trigger periodic spending alerts.

6. Budget

6.1. A budget shall be created for one or more spending categories.
6.2. A budget shall belong to one and only one account.
6.3. A budget shall allow setting and tracking of goals.
6.4. A budget shall support notifications based on spending limits.
6.5. A budget shall be able to generate monthly recaps.

7. Task

7.1. A task shall be assigned to a user account.
7.2. A task shall be of one type: savings, investing, setup, admin, testing.
7.3. A task shall be marked completed upon user action.
7.4. A recommended task shall be generated based on user activity or AI analysis.

8. Reward System

8.1. A user shall be able to earn rewards for completing tasks.
8.2. A user shall be assigned a level based on completed actions.
8.3. A user shall be able to view rankings if competitive mode is enabled.
8.4. A user shall be able to redeem rewards through the system.

---

9. AI Assistant (LemonAid)

9.1. The AI assistant shall analyze user transactions and spending habits.
9.2. The AI assistant shall recommend saving opportunities and budget changes.
9.3. The AI assistant shall forecast recurring charges and balances.
9.4. The AI assistant shall provide suggestions for uncategorized transactions.
9.5. The AI assistant shall simulate different financial scenarios for planning.

---

10. Notification System

10.1. The system shall send reminders for logging receipts.
10.2. The system shall notify users of overspending events.
10.3. The system shall notify users of upcoming bills or subscriptions.
10.4. The system shall deliver daily, weekly, or monthly financial summaries.
10.5. The system shall alert users when savings goals are off track.

---

11. Administrator

11.1. An administrator shall be able to access user account data.
11.2. An administrator shall be able to view flagged transactions.
11.3. An administrator shall be able to respond to support tickets.
11.4. An administrator shall be able to manage and moderate platform use.
11.5. An administrator shall be able to suspend or maintain site operations.

List of Non-Functional Requirements

1. Performance

1.1. The system shall respond to user actions within 2 seconds under normal load.
1.2. Backend APIs shall respond to simple requests in under 200 milliseconds.
1.3. The dashboard shall load within 3 seconds over a stable 10 Mbps connection.
1.4. AI recommendations shall be delivered within 5 seconds of user input.
1.5. The system shall support real-time syncing of transactions without freezing the UI.

---

2. Security

2.1. All communications shall be encrypted using HTTPS and TLS protocols.
2.2. All sensitive data, including user credentials, shall be encrypted at rest using AES-256.
2.3. Passwords shall be hashed using bcrypt or equivalent secure hashing algorithms.
2.4. Role-based access control shall be implemented to separate admin and user privileges.
2.5. The system shall prevent common web vulnerabilities such as SQL injection, XSS, and CSRF.
2.6. Only authenticated and authorized users shall be able to change, delete, or insert sensitive data.

---

3. Scalability

3.1. The system shall support up to 500 concurrent users without performance degradation.
3.2. The database shall support horizontal scaling across distributed servers via Docker and AWS.
3.3. The system shall efficiently scale with a 10x increase in user data and

transaction volume.
3.4. Database schema shall support sharding for large financial datasets.

---

4. Usability

4.1. The UI shall be intuitive and require no documentation for key tasks such as setting up budgets or tracking expenses.
4.2. The system shall support both dark mode and light mode themes.
4.3. The AI assistant shall use natural, conversational language to support non-technical users.
4.4. Navigation, task completion, and interaction shall follow established UX best practices.

---

5. Reliability

5.1. The system shall maintain at least 99.5% uptime over a 30-day period.
5.2. Transaction data shall be backed up every 12 hours automatically.
5.3. Background processes such as syncing and reminders shall automatically retry on failure.
5.4. Error handling shall provide feedback to users without crashing the system.

---

6. Maintainability

6.1. The codebase shall follow JavaScript linting rules using ESLint and Prettier.
6.2. Docker containers shall be used to ensure parity between local and production environments.
6.3. The codebase shall follow a modular architecture to isolate and decouple components.
6.4. Critical modules shall maintain a minimum of 70% unit test coverage.
6.5. Version control shall be managed through Git and GitHub with frequent commits and reviews.

---

7. Portability

7.1. Docker containers shall allow the app to run across local development, staging, and AWS cloud environments.
7.2. The app shall run identically across Windows, macOS, and Linux-based systems.

---

8. Compatibility

8.1. The system shall integrate with major U.S. banks using secure open banking APIs.
8.2. The AI engine (for ex: ChatGPT API or Gemini API) shall be modular and replaceable without impacting core functionality.
8.3. The system shall support the following browsers:

- Google Chrome v80+
- Mozilla Firefox v75+
- Microsoft Edge v80+
- Safari v11.1.1+

---

9. Privacy

9.1. The app shall allow users to manually enter financial data without linking any bank accounts.
9.2. User data shall not be shared or sold to third-party services.
9.3. All stored data shall remain local unless explicit user consent is given.
9.4. Privacy mode shall restrict AI access to only local anonymized data.

---

10. Compliance

10.1. The system shall comply with GDPR regulations for data privacy and user consent.
10.2. Users shall be able to export or permanently delete their data on request.

10.3. Consent prompts shall be clearly visible during signup and financial data linking.

---

11. Supportability

11.1. System logs shall track user behavior, feature usage, and AI errors for debugging.
11.2. An admin dashboard shall allow staff to review flagged transactions and feedback.
11.3. The system shall support browser versions listed under Compatibility.
11.4. Support documentation shall be maintained for installation, setup, and deployment.

---

12. Efficiency

12.1. The system shall operate at no more than 60% CPU usage under normal load conditions.
12.2. Memory usage shall not exceed 75% of allocated memory on the EC2 instance.
12.3. Efficient logging shall be implemented to prevent log flooding and performance drag.

---

13. Look and Feel

13.1. The system shall maintain a clean, minimal aesthetic with responsive design.
13.2. UI elements shall follow a cohesive design language across all pages and devices.
13.3. Accessibility standards (WCAG 2.1 AA) shall be considered for inclusive design.

---

14. Coding Standards

14.1. All backend code shall conform to Node.js and Express.js best practices.
14.2. Frontend code shall follow React component structure and state management conventions.
14.3. Continuous Integration tools shall run linting and testing on every merge request.

---

15. Environmental Sustainability

15.1. The system shall optimize backend server usage to reduce idle time and energy waste.
15.2. Docker containers shall be configured to auto-scale down during periods of low usage.
15.3. The app shall minimize data transfer size (e.g., compress JSON responses, lazy-load images) to reduce bandwidth and energy consumption.
15.4. The frontend shall use efficient rendering practices in React (e.g., avoiding unnecessary re-renders, using memoization).
15.5. System architecture shall consider the use of renewable-energy-based data centers (e.g., AWS sustainability zones).
15.6. Logs and backups shall be archived using energy-efficient cold storage where applicable.
15.7. Documentation shall be provided digitally only (no paper printing by default).
15.8. The system shall avoid unnecessary background polling or constant sync unless essential for real-time use.

Competitive Analysis

We analyzed top finance apps competing with Limóney to understand their strengths and gaps. These insights shaped a smarter, more intuitive product.

High-Level Competitor Overview

Competitor	URL	Primary Audience	Core Offering	Mobile Support
RocketMoney	https://www.rocketmoney.com ↗	Budget-conscious individuals	Manages subscriptions, tracks spending, automates savings, negotiates bills.	Excellent
YNAB	https://www.youneedabudget.com ↗	Proactive planners	Zero-based budgeting, emphasizes forward-looking control.	Strong
Mint (Intuit)	https://mint.intuit.com ↗	Broad consumer financial users	Tracks spending, credit scores, budgeting, account syncing in one free app.	Good
Copilot	https://www.copilot.money ↗	iOS users seeking modern UX	AI-powered categorization, real-time insights, sleek design.	iOS-only
Emma	https://emma-app.com ↗	Younger, tech-savvy users	Subscription tracking, budgeting, crypto	Cross-platform

			support, social sharing.		
EveryDollar	https://www.everydollar.com ↗	Envelope-budgeting enthusiasts	Zero-based budgeting with envelope-style allocation, expense tracking, debt payoff focus.	Excellent	

Research Table (Qualitative Findings)

Competitor	Key Differentiators	Weaknesses	User Target Fit
RocketMoney	Subscription canceling; bill negotiation	Limited deeper planning; lacks personalized AI guidance	Passive optimizers
YNAB	Goal-based, zero-sum budgeting	Steep learning curve; higher subscription cost	Active planners
Mint	Free, long-established platform	Being phased out; inconsistent updates	Casual overview seekers
Copilot	Premium UX; AI-based category suggestions	iOS-only; premium price	Modern insight seekers
Emma	Crypto integration; social features	Feature overload; can overwhelm	Gen Z/Millennials
EveryDollar	Envelope-style zero-based budgeting; debt focus	No free bank sync for envelopes; advanced features behind paywall	Users who follow Ramsey's methodology

Feature Comparison Table

Feature	RocketMon	YNAB	Mint	Copilot	Emma	Ev

	ey					
Budgeting Tools	+	+	+	+	+	
Bill Negotiation	+	-	-	-	-	
Subscription Cancellation	+	-	+	-	+	
Real-Time Spend Categorization	+	+	+	+	+	
Credit Score Monitoring	-	-	+	-	+	
Smart Saving Suggestions (AI)	+	-	-	+	+	
Crypto Asset Integration	-	-	-	-	+	
Bill Reminders / Notifications	+	+	+	+	+	
Financial Goals Tracking	+	+	+	+	+	
Envelope-Style Allocation	-	-	-	-	-	

Legend:

- - = Not implemented
- + = Implemented
- ++ = Planned enhancement by our team

Summary of Competitive Insights

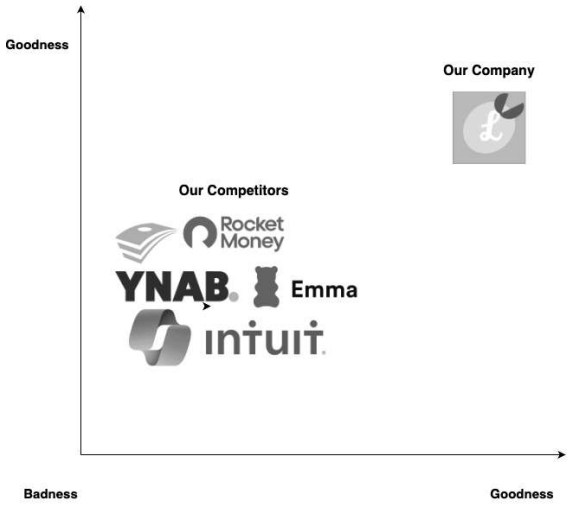
- **Automation & AI-driven guidance are underutilized.** RocketMoney and Copilot do well with smart suggestions, but none combine seamless AI advice with active bill negotiation and cancellation in one place.
- **Zero-based vs. passive approaches.** YNAB and EveryDollar champion envelope and zero-based budgeting, which appeals to planners, but casual users find those models too rigid.
- **Platform reach gaps.** Copilot's iOS exclusivity and Emma's feature overload highlight a need for a balanced, cross-platform experience that's powerful yet simple.
- **Legacy vs. modern churn.** With Mint's phase-out, there's an opportunity to capture lapsed users looking for a familiar but improved solution.

Our Unique Advantage

- **Unified AI & automation.** We blend proactive AI-driven savings tips with negotiated bill reductions and one-click subscription cancellations.
- **Flexible budgeting.** Support both envelope-style allocation for planners (à la EveryDollar) and passive tracking for casual users, all in one app.
- **Cross-platform clarity.** Available across web, iOS, and Android with an intuitive interface that scales from quick overviews to deep dives.
- **Stress-free control.** Built for clarity: users get actionable recommendations without the noise of endless toggles or paywalls.

Competitive Visual

On both axes, we measure overall “goodness” (ease of use, helpful features, delight). Competitors cluster in the middle: RocketMoney, YNAB, Emma, Intuit (Mint), and EveryDollar each excel in pockets but have notable trade-offs. Limoney sits in the top right, reflecting its unique blend of AI automation, envelope flexibility, and seamless cancellation tools, delivering the most user-centric experience.



Checklist

- The team needs to find timeslots outside of class time to meet
  - Done - Met on 06/07/25
- GitHub Master has been chosen.
  - Done - The GitHub master is Dani
- The team has collectively decided on and agreed to use the listed software tools and deployment server.
  - Done
- The team is ready to use the chosen front-end and back-end frameworks, and those who need to learn are actively working on it.
  - Done - Team is mostly familiar with things we are using, currently learning/reviewing React and Sequel
- The Team Lead has ensured that all members have read and understand the final M1 before submission.
  - On-track
  - Will be done as tech doc is turned in and explained to everyone
- GitHub is organized as discussed in class (e.g., master branch, development branch, folder for milestone documents, etc.).
  - ~~Issue - Need to speak with professor regarding how to submit work and how credit is looked at.~~
  - ~~On-track - Met with professor, have discussed how it works, currently working on cohesion~~
  - Done - Have determined how to pull and create branches for the project, as well as settled on using Gitbook
- Members have sent an email reviewing all teammates contributions
  - On - track
  - Done

High-Level System Architecture and Technologies Used

- CIoud Server Host: AWS EC2 – t2.micro (1 vCPU, 1 GB RAM)
- Operating System: Ubuntu 22.04 LTS
- Database: MySQL 8.0.42
- Web Server Framework: Express.js 4.19.2
- Backend Language: JavaScript
    - Backend Framework: Node.js
- Frontend Language: HTML, CSS, JavaScript
    - Frontend Framework: React
- Containerization: Docker
- SSL Certificate: Let’s Encrypt with Certbot
- Version Control & Collaboration: Git, GitHub
- Optional Tools:
    - Firebase
    - AI-Based Budgeting Assistant: Powered by ChatGPT API or Gemini API for personalized financial suggestions

List of Team Contributions

Name	Contributions (for C2&3)	Score (1-10)
Emily Perez	Wrote executive summary, wrote high level system..., set up cloud server, helped with credentials md, helped with main use cases	10
Ishaank Zalpuri	Wrote list of main data...,wrote main use cases, wrote checklist, helped set up permissions with cloud server, did what was assigned	10
Andrew Brockenborough	wrote competitive analysis, helped w main use cases, did what was assigned	9
Dani Luna	made the title page, wrote main use cases, set up aboutus page, did what was assigned	10
Jonathan Gonzalez	wrote main use cases, wrote functional req, wrote non functional req, did what was assigned	10
Gene Orias	wrote table of contents, wrote main use cases, helped with credentials md, did what was assigned	10