**Final Project Report: SmartSpender**

**Project Direction and Overview**

SmartSpender remained faithful to the vision presented within our first project proposal. The core concept was to create a generic budgeting website where people can scan receipts and have expenditures calculated automatically for them. As we developed the project, we remained committed to this concept and incorporated features in order to have the platform smart and informative in a better way for a quality budgeting experience.

**Achievements and Drawbacks**

Our application successfully allows users to add receipt items and distribute them among different individuals to share expenses rightfully. This is useful in mere bill sharing as well as enables enhanced financial disclosure among users. We could not get sufficient time to fully implement the AI-based receipt scanner on the frontend as was originally planned to be a part of user experience. Such functionality is currently a top choice for future development.

**Schema and Data Source Evolution**

The project schema evolved drastically during development. Initially, in development, the schema consisted of four tables: account information, items, receipts, and donations. As our understanding of user needs evolved, we included two additional tables: a budget table and an average expense table. We pulled the average expense data from the U.S. Bureau of Labor Statistics, so we could compare spending user habits to national averages by income levels. This added complexity to the app's analytical functions.

**ER Diagram and Design Changes**

Our early ER design was relatively simple, with only four major tables. As we progressed, we understood that more capability layers were necessary. We added budget and average expense tables to facilitate more advanced budgeting capabilities and personalized monetary information. These capabilities allowed users to set spending limits in various categories and see how their financial activity stacks up against others in the same income levels. The expanded design better fulfills our goal of offering a comprehensive system for financial management.

**Changes to Functionality**

Certain functions initially planned were changed or removed. The AI-driven receipt scanner was ultimately not employed due to its complexity and susceptibility to high error rates. The structure

and composition of every receipt are extremely diverse, and with or without the aid of AI, the margin of error proved too great for production at this time. An interactive dashboard functionality was also eliminated due to time constraints. While it would have added to the app's visual appeal and usability, we decided to implement core backend and budgeting features instead.

One of the key features we implemented was a monthly reset of spending data. This innovative database feature makes user budgets time-sensitive, thereby making them more useful for ongoing personal money management.

**Advanced Database Features**

SmartSpender has a number of advanced database applications that significantly contribute to its functionality:

A spending comparison engine contrasts user spending with average spending from the U.S. Bureau of Labor Statistics and provides feedback on whether the user is spending above or below average.

A monthly budget monitor alerts users when they exceed their budget for a particular category, keeping users fiscally in line.

A fair split calculator ensures that bill splitting is carried out according to actual items and not equal division, providing fairness when the expenses are being split by multiple users.

**Technical Challenges**

Each team member faced some technical challenges that resulted in worthwhile learning:

Kevin had issues with linking the server to the Google Cloud Platform (GCP) database and resolved them by making configuration adjustments and handling permissions.

Wen Hao spent time integrating the Gemini API into the AI receipt scanner.

David spent time addressing formatting differences between backend output and frontend input to ensure that data transfer between the two systems is smooth.

CJ faced and resolved problems surrounding application state management across various frontend pages, which improves the overall user experience.

**Future Work**

There are a number of priority areas for future development:

AI Receipt Scanner: With error handling and editable fields, this could significantly reduce user data input effort.

Interactive Dashboards: The inclusion of bar charts, pie graphs, and historical data visualization would make financial insights and user interaction more robust.

Social Splitting Network: Introducing a feature to maintain records of frequent collaborators would increase the speed of bill splitting by eliminating frequent searching for user IDs.

**Teamwork and Division of Labor**

The division of labor was fair and we worked well together as a team.

Kevin was mainly in charge of backend functions and managing the database on gcp. CJ was responsible for the frontend and user interface. David set up the backend server and converted the backend functions to the data format needed for the frontend. Wenhao was responsible for creating the advanced procedures and the Receipt Scanner AI before that was scrapped.

Managing teamwork was fairly smooth as we all agreed to meet on weekends and also work separately. We would book a study room in Grainger every weekend to get together and discuss next steps, assign roles, and simply work together. Since we used github we also worked offline if needed to complete our agendas before the next meetup. We also required pull requests to change the main branch of github to prevent merge conflicts and streamline the collaboration on coding.