SENIOR DESIGN
COSC/ECE 401

Project Proposal
AI Ledger

November 3, 2021

Group 1

Tyler Beichler, Emanuel Chavez, Blake Childress, Lillian Coar, Andrei Cozma, Jacob Leonard, Hunter Price

Advised by Dr. Bradley Vander Zanden

Executive Summary

The goal of this project is to create an application to effortlessly support shared spending within multiple groups that support the recording of spending between multiple parties. This application will allow users to prevent the loss of money through dropped or forgotten transactions within the group. The web application should provide support for groups such as families, friends, roommates, couples, and others who want to audit their spendings, settle balances, and pay other members outside of the application.

The application will be hosted via AWS cloud services which allows for easier scalability as needed. The front end will be built with React.js. The receipt OCR will be achieved through Tessaract.js. The machine learning models to classify item categories will be written with Tensorflow.js.

The application is also built around four major engineering characteristics, which were determined by analyzing the project goals and requirements. These four requirements are: accessibility, reliability, response time, and scalability. Accessibility and scalability are constraints, whereas the rest are variables that can be measured.

Table of Contents

Problem Definition & Background

When people do things together, usually there are expenses that cover multiple people or groups, and the parties involved want to keep track of these transactions so that they can pay or be paid for the expenses. Currently, there is no easy and free way to split up and keep track of expenses between multiple people or groups of people. The few products that were found that allow these features are not multiplatform and are not comprehensive in that there are limitations or missing features that the group intends on implementing in the application. There are some applications that are good at one aspect of expense sharing, such as restaurant bill splitting, and others that excel in another. Some applications are relatively good, but they are exclusive to IOS or Android, require monthly subscriptions for essential features, or do not have easy and intuitive interfaces.

One of the basic functions of the product is the ability to create groups of people. Within these groups, purchases can be split up among the members. Purchases will also be able to be split among multiple groups of people for instance: two families on a trip together. Another function is the manual insertion of purchases so that they can be recorded or split between people. The main priority is to get these functions down and make a simple and effective user interface. The group will start by making a web app so that from there the group can create android and IOS apps by creating a WebView of the web app. Later, the group plans on adding receipt scanning to speed up the process of adding multiple purchases, user accounts, and cloud storage for access on multiple platforms. We also plan on adding features that involve automation, such as being able to scan a receipt and automatically bill the users in the group, or even send direct Venmo and PayPal requests to prevent people from being required to use our app directly. More advanced features would be premium features, while retaining the core functionality of the application for free users. The background needed to complete this is knowledge of databases and managing them, web apps and creating a WebView for them, interpreting financial statements, and knowledge of expense tracking.

Venmo and CashApp are two apps that have been made that can perform similar functions as the project. These apps can be used but the users must keep track of the purchases and request/pay the other parties on their own after deciding what is owed. While this is an option, it is up to one of the parties involved to pay or request the money, where the application will keep track of it for both parties.

Our final product will be able to be utilized by both individuals and businesses. Individuals can use it for splitting expenses such as groceries, rent, and restaurants. Businesses will be able to use the app to keep track of expenses in certain departments and how much is being spent monthly or yearly. Groups such as Homeowners Associations can also use the final project Both people and businesses can use the recorded data to plan for their future expenses and change where their funds are being put into.

Requirements Specification

The final product will be a shared ledger implemented as a web application. This application will facilitate shared spending within groups along with allowing for tracking and review of these purchases. The customer requirements of the project are listed in the table below.

Works seamlessly on all major platforms and devices (Windows, Linux, macOS, Android, iOS): By virtue of being an application for shared expenses, different users will likely use different platforms. Ensuring compatibility and consistency between devices is crucial. The current minimum target for browsers is the previous release of Mozilla Firefox, Apple Safari, Google Chrome, and Microsoft Edge. Testing will need to be done to ensure compatibility with these browsers as well as the operating systems listed.
Must be easy and intuitive to use, and function in a logical manner for the target audience: In light of the goal of having mobile platforms usable, the application must be responsive and resized to maintain functionality on both mobile and desktop platforms. Even though this is a web application, it must be usable in the same manner as a native application. This means that the user should be able to navigate within the application without using the back button or any other browser navigation functionality. To keep the interface minimalistic and intuitive, form controls that are not available must be hidden.
Must adhere to industry best practices: Form fields should be validated to ensure that required forms are completed. Numeric forms should contain numeric data. Any forms which require special formatting should be validated to ensure user input follows that format. Exception handling should be done through server-side code. Exceptions must be handled in a way that is user-friendly, showing a custom error page and not leaking any information such as database object names or source code.
Must use styles that are consistent throughout the application and within the associated website: Elements of style throughout the application must be kept consistent including the use of capitalization, the use of punctuation, the location and appearance of error messages, and layout and spacing (barring any differences which result from responsive design).

Table 1. Customer requirements for the shared ledger application, listed from most important to least important.

It is important to note that the concept of shared ledgers has wide appeal. The convenience applies to people from many different demographics. All of the requirements were picked with this in mind. Keeping the barrier of entry low is important for getting people within a group to adopt the use of this application. Being available on all major platforms allows for anybody to start using this application. Putting emphasis on making the interface intuitive allows for users to use the application regardless of prior technical literacy. Validating forms curbs user error. Using custom error pages prevents stress on part of the user that would be caused by a technical error message. Consistency prevents confusion.

Although there are no physical engineering characteristics for the project, there are several variables and constraints that the team must consider. Four engineering characteristics are shown in the table below, which were determined by examining the intended implementation and specifications of the product. Furthermore, those characteristics are listed in descending order of importance.

Engineering Characteristic	Units/Options	Significance
Accessibility	n/a	Constraint: Must be available on all major platforms and have an intuitive user experience
Reliability	uptime/downtime	Variable: Determined by the dependability of the project and servers
Response Time	ms	Variable: Influenced by the efficiency of the application and quality of the servers/services
Scalability	n/a	Constraint: Relies on the product architecture and services chosen

Table 2. Engineering requirements; from most to least important.

Deliverables

A. Overview

The project aims to deliver a working prototype of a web application that will help customers audit their spendings within groups. A number of deliverables are required in pursuit of the final product as listed below:

- A frontend framework/library to create an appealing and functioning user interface for customers to perform tasks
- A backend database to store user and group information as well as their respective ledger
- A web hosting service that supports a full stack and dynamic application

- Optical Character Recognition (OCR) to utilize image conversion of a picture to machine-encoded text for receipts and other related documents
- Artificial Intelligence (AI) tools to help users optimize savings based on the history of their spendings and budgets. These tools will focus on machine learning models to categorize spendings as well.

B. Key Features

Component	Tools/Libraries
Frontend	React.js
Backend	Amazon EC2 Server
Hosting Service	Amazon Web Services (AWS)
Optical Character Recognition (OCR)	Tesseract.js
Artificial Intelligence (AI)	Tensorflow.js

Table 2: Tools Utilized For Each Component

This table specifies the key tools and libraries needed to produce the application with the goal to deliver a clean, minimalistic, and easy-to-use user interface. The group will utilize the JavaScript library, React.js, which will provide a simple framework to design the frontend. The Amazon EC2 Server provides a free-to-try subscription for twelve months which will support all of the needs regarding storing the information. Amazon Web Services (AWS) is able to deploy React.js applications while also implementing the Amazon EC2 Server. A simple Python program that utilizes the Tesseract library will provide the project with Optical Character Recognition (OCR) to read receipts and other related documents.

Availability is a key feature as the group aims to cover most devices and platforms such as establishing an integrated native application on iOS and Android. However, it was advised the group should focus on one or the other which is still to be decided. The design will be available on desktop and mobile devices which will be accessible to the web. If the user does not have an internet connection, the group aims to provide offline capabilities so that the user can still record information whenever they need it.

Project Management

Each major component of the application is assigned to a member of the team to oversee its progress and development and to provide regular updates. The project manager will be in charge of overseeing the overall front-end and back-end development teams to ensure that all the requirements, deliverables, and milestones are met.

As a team, the group will be working very closely throughout the entire process to ensure that the group are all on the same page and that everything is going as planned. The group will be holding standup meetings every week on Friday from 11:45 to 12:35 to discuss each team's progress and roadblocks. This will also be a way for us to discuss any concerns or issues that any of the members may have, and to ensure that the group is on track for the completion of the project.

The goal for the Fall semester of 2021 is to get a head start on planning, research, discovery, and creating a very basic architecture and prototype of the application. This prototype will consist of a front-end and back-end that will allow us to get a more concrete idea of the overall development of the application. The goal for the second semester is to develop all of the core components of the application further and add the rest of the features to expand the scope of the application according to the project deliverables and objectives.

The following schedule has been developed as a proposed timeline for the development of the proposed application for the Spring 2022 semester. It is largely contingent on the success of the research and development efforts as well as the availability of resources and key stakeholders.

Week	Goal
1	Planning, Research, and Discovery
2	Research and Briefs for Front-end and Back-end Architectures
3	Overall Design Planning and Wireframe
4	Homepage/Front Page Design & Development
5	Internal Pages Design & Development
6	Content Creation, Features, and SEO
7	Overall Development and Coding
8	Beta Testing, Feedback, and Improvements

9	Implementing Fixes & Improvements from Beta Testing
10	Launch & Working Demo
11	Final Improvements & Documentation
12	Working on Final Reports & Documents
13	Finalizing Reports & Presentations
14	Wrapping Up

Table 3: Project Schedule

Budget

Item	Estimated Cost
AWS Hosting Services - Free Tier	$0
Jira - Student	$0
Discord	$0
GitHub Student Developer Pack	$0
Domain	$15/Year
Total	$15

Table 4: Anticipated Spending

This project will require minimal spending while in development. The application will be hosted with Amazon Web Services (AWS)[1]. Specifically, it will use Amazon EC2 Compute, which is cheaper than purchasing hardware to host the website and database (free for 12 months). EC2 will give us up to 750 hours of Linux instances per month, which will suffice for the purposes. As a worst-case scenario, if the project work exceeds 750 hours per month, the budget will have to be modified to pay $0.051 per compute hour that is used (which can be funded by the premium tier of Smart Ledger). Once the project is completed, paid computation hours will be required to support a growing user base; however, this is out of scope for this course. Jira will be used as the primary project management platform [2]. Jira is free for students and it also has a free tier for up to 10 users. Discord will be the primary vehicle of communication for the team. Discord provides all of its essential features for free. Github will be used as the primary source control platform. Github provides a free student developer pack that gives us all the features that

are required for the development of Smart Ledger [3]. Lastly, a domain is required for the customer audience to be able to access the application externally, which typically costs around $15 per year. This brings the total budget to approximately $15 dollars for the project.

References

[1] Amazon EC2. (2021, 11 01). Amazon Web Services. https://aws.amazon.com/ec2/?did=ft_card&trk=ft_card&ec2-whats-new.sort-by=item.additionalFields.postDateTime&ec2-whats-new.sort-order=desc

[2] Jira Pricing. (2021). Atlassian. https://www.atlassian.com/software/jira/pricing?&aceid=&adposition=&adgroup=89541890342&campaign=9124878120&creative=461842798799&device=c&keyword=jira%20pricing&matchtype=e&network=g&placement=&ds_kids=p51241495620&ds_e=GOOGLE&ds_eid=70000000155 8501&ds_e1

[3] GitHub. (2021). GitHub Education. https://education.github.com/pack

Appendix

Appendix A:

Business Matrix Canvas