CS 4850, Fall, 2024

SP-6 Green, Time Mileage Tracker;

Jack Simrill, Jonathan May, Matt Stewart, Donald Grooms;

Sharon Perry, November 17, 2024;

https://sp6timemileage.github.io/;

https://github.com/SP6Green/TimeMileageTracker/

Number of Code Lines: 486 (Estimate as we are still adding code, but will have a final number for the full final report package)

Number of Tools: 6 Google Maps API, 3 Environment Tools, 1 Version Control Tool, 1 Database Tool: 11 Tools

Total Man Hours: 458

Table of Contents

Introduction

The purpose of the application is to track the mileage of employees while they're working. The application will collect data in real time and generate reports from the user's travels. This will aid employees in receiving credit for their travel from their employer. The application takes the guesswork out of tracking milage which will improve the experience of traveling for both employees and employers.

Requirements

GPS Display

Display Map with User's current location

Add search function to allow users to search for an address

Using routing function of maps API to direct the user to their selected location

Track Mileage

Login

Create Account feature – Import info into the database

Login page with Username and Password

Forgot Password feature

Stats

Create stats page to show the user mileage data

Add section for users to view past reports

Generate a Mileage Report

Settings Page

Add features that allow the user to personalize the app

Add the ability to choose the Language

Monetization

Create subscription tiers that include increasing benefits for the user

Add pricing information

Alternative: Implement Ads that appear when a Report Generation is requested

Analysis

Requirements Analysis

- GPS Display: For the GPS display the tools needed for implementation are Google Maps API, the Haversine library and the Expo Location library. These tools will allow us to successfully implement navigation, maps display, and Track Mileage.
- Login / Stats: For the Login and stats page we will need to use Firebase to track user information in real time, SQL, user authentication, and React Native and JavaScript design methods.
- Settings: We will use React Native libraries and functions to implement app customization and language selection.
- Monetization: We will either implement a subscription tier-based monetization system or use Ads to generate revenue using our App.

Design

Assumptions and Dependencies

The assumptions and dependencies are that the operating system will be used is either iOS or the Android operating system. The hardware requirement that will be needed is the phone's location module. So, we can track the user's location will they move around. The software would be the Google Maps API and the cloud database system, Firebase, used to store the data.

General Constraints

The only general constraint is that Google Maps API is not entirely free, and they operate on a subscription/resource-based payment system. This limits the amount of testing we can do. The performance requirement would be that your phone runs on either one of the two operating systems that the app will be used on.

Development Methods

The development method we are using for this software design is SDLC. This method is a process that allows the production of software in the shortest possible production time. The goal of SDLC is to create software that meets and hopefully exceeds the user's expectations. For this project, we have a few architectural strategies in place. We will be using JavaScript along with React Native and Expo to develop this application. We are using GitHub and VS Code to write and update our code during this process. Also, to keep track of user information and other metrics within our application, we will set up a SQL database

to store and pull information. This will, at the start, not be a cloud-based database, however, we hope to import the database into the cloud for security and ease of access. For the cloud-based database we will be using Firebase.

System Architecture

Our app, the Time Mileage Tracker, is engineered to efficiently monitor user mileage through GPS technology while generating detailed reports based on trip data. To achieve this functionality, the system has been broken down into subsystems. The main components include the GPS Module, which tracks the user's location, the Database system, which collects data and stores the user's information, and the Report Generator, tasked with taking in the database information and creating a report. The GPS Module feeds trip data to the Database system, which then communicates with the Report Generator to produce comprehensive reports. This structured approach not only enhances the overall performance and reliability of the app but also simplifies future enhancements and maintenance, delivering a robust application that meets users' needs effectively.

Definition

The GPS module will track the user's location and display routing information for the trip function. This will be used to collect the trip data for report processing. Most of the location data will be gained using geocoding and reverse geocoding. All information gained by this module will be sent to the Database System and recorded in the database. The Database system will keep track of all user information. This includes login information, trip data, location data, etc. The database will be created and handled using Firebase. This will also be used to retrieve information for the Report Generator. The Report Generator is a module

that takes user trip data and forms it into an easy-to-read report. The reports will include mileage data, travel time, and routing information (if we can figure this out).

Constraints

The user will have to enable their location to be used while using the app. Another constraint is that Google Maps API is not entirely free, which limits the amount of testing due to the need to remain within the given budget. The budget you must stay within to avoid being charged is $200.00.

Resources

The resources we will manage include user trip data and database-related resources. We will also have to manage the phone's resource use so as not to drain more resources than needed. Lastly, we must manage the number of API calls and requests so as not to go over budget.

Interface/Exports

- GPS Module: Tracks user data and exports data to the database system.
- Database System: Manages user data imported from the GPS module. Exports this data to the Report Generator.
- Report Generator: Imports user data and generates user reports.

Glossary

SDLC: Software development Life Cycle

API: Application Programming Interface

Geocoding: The process of converting addresses or locations to coordinates which can be used to pinpoint a user's location on a map.

Reverse Geocoding: The process of changing a location on the map into a readable address for humans' use.

Routing: The process of calculating the optimal route from the current location to the target location. This will be used to calculate GPS routes for our users to follow.

Firebase: A mobile and web app development platform backed by Google.

Development

The creation of the project required weekly development meetings. Our two developers meet weekly to implement and review each feature that our App requires. The development approach that we used was the Waterfall approach. This means our developers completed app features in the order set within the requirements. Once a feature was completed, they moved on to the next feature on the list. The tools we used to develop our app were Google Maps API, React Native with Expo, GitHub, Visual Studio Code, and Firebase. These tools allowed us to implement each feature with ease as well as record data and issue code updates. Also, we used the Haversine Library to track the mileage using the haversine formula. This formula allows us to use the latitude and longitude numbers we grab from the user to calculate the distance they travel whilst tracking the trip.

Test Plan

Requirements	Pass	Fail	Severity Level
GPS Display	Pass		
Directions	Pass		
Navigation	Pass		
Mileage Tracking	Pass		
Real Time Database	Pass		
Login Page	Pass		
User Authentication	Pass		
Password Recovery		Fail	Low
Report Generation	Pass		
Stats Tab		Fail	Low
Language Change		Fail	Low
App Theme		Fail	Low

Version Control

During the development of our Time Mileage App, we used GitHub for our version control. Every new feature was pushed to GitHub, and each member was able to access the full code from there.

Summary

The Time Mileage App is mainly used for tracking mileage for an employee while working. The app will track user data in a real-time database and generate a mileage report. It will be beneficial for the employee when receiving credit for miles traveled during a work trip. The solution the app is attempting to solve is to eliminate the guesswork of tracking the miles on your own. Future goals would be improving design/layout, generating reports, and user authentication.

Appendix

References:

React Native Course - https://www.youtube.com/watch?v=ANdSdIlgsEw

Setting up Expo - https://www.youtube.com/watch?v=1_WXNvK_tjs