ENRoute

Application to find electric vehicle charging stations

Kirsten Boyles, Joe Corona, Richard DeYoung, Craig Turnbull, and Lucas Keizur.

envorso
Transforming Productivity

Table of Contents

The Development Team

Kirsten - UX / UI design

Craig - Maps API implementation

Joe - Voice Command

Richard - Database + User Accounts

Lucas - Charger Data

Our Client

Envorso - a modern solutions firm

- Bob Rapp
  - Business Consultant
    - Experience consulting top executive
    - Experience Building large scale projects
- Parker Jones
  - Technical Consultant
    - Experience with software development
- University Advisor - Professor Vajda

Problem Statement

Non-Tesla electric vehicle drivers have it rough.

They must use Third-Party Charging Station

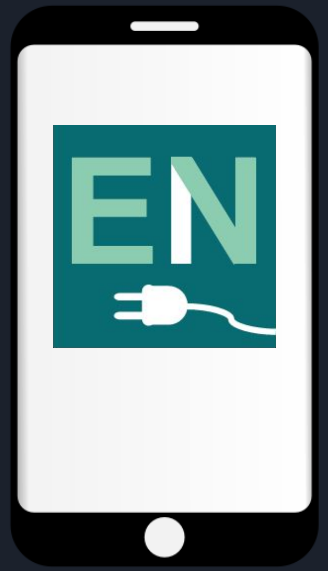
- Each station has different
  - Charge speeds
  - Pricing
  - Plugs
- Each company instigates charging differently
  - Some require a membership
  - Some require an app
- Each car is different
  - Different plugs
  - Different charging capabilities

Problem Statement - Solution

Our App - ENRoute

A solution to non-Tesla EV owners

- Almost instant list of car chargers
- Based on the user's:
  - Speed, Distance, Plug type preference
  - GPS location
- Highly Usable
- Highly Functional
- Currently covering some cities in Eastern Washington



Tools

Front-End: Flutter + Dart

Back-End: Google Firebase

Google Maps API

Other Car Charger APIs

Python - Web Scraper

iOS Development: CodeMagic

iOS Development

- Code Magic required us to run on Mac with XCode
- Samuelson Macs didn’t have XCode
- Needed Admin Privileges

- Got XCode to run on another Mac
- Ran into issues with CocoaPods

Why Flutter + Dart?

- Flutter has native support for iOS and Android
- None of us had experience in it, presented challenge
- App fully works on Android devices

Why Firebase?

- Easy to set-up
- Free
- Flutter and Firebase are Google Products
- NoSQL allows for flexibility of data

Software Solution

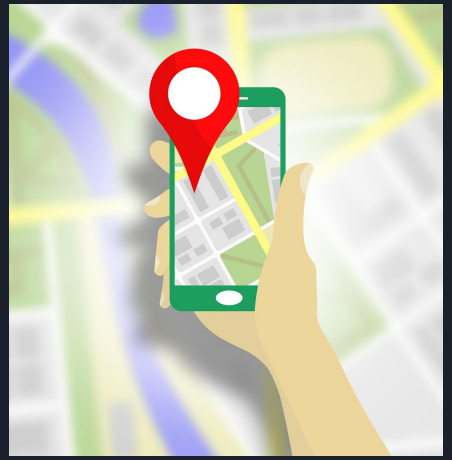
- VSCode
- Git
- Packages
  - Maps
  - Voice packages
  - Speech Recognition
  - Firebase
- Android Studio Emulators
  - Android SDK

Software Solution (cont.)

- Geohashing
  - 0.2 lat. by 0.2 lon. is a single chunk
  - Lat. and lon. act as Axes
- Firebase Queries
- Bubble Sort
  - Sorted chargers

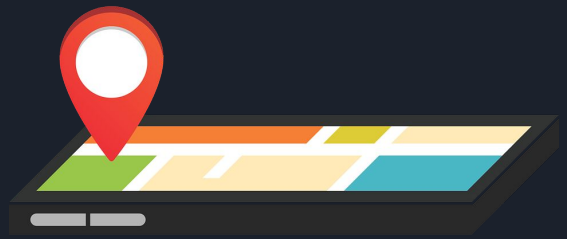
Maps API and DeveloperNetwork.gov

- Maps API
  - Also Google
  - Works well
  - Well known layout
- DeveloperNetwork.gov
  - Non-API method of locating charger info
  - Free
  - Easy to parse



Maps API

- Leaflet API
  - + Most lightweight
  - - Fewer features
- Mapbox API
  - + More customization
  - - Navigation broken after update
- Google Maps API
  - + Well documented
  - - Less customization



Functional Requirements

First Time Login

1. Prompt users for PID info and their car’s charge port type. - **I**
2. Allow users to sign up with a new account. - **I**
3. Users can sign up using Google or Apple. - **NI**
4. Ask users for their ZIP-code to generate a list of nearby charging services. - **I**

Functional Requirements

Locating a Nearby Charger (Fastest or Cheapest)

1. Match the vehicle's charge port with chargers nearby (will not present users with a charger they are not compatible with). - **I**
2. Determine the speed of chargers located. - **I**
3. Determine the price of chargers located. - **NI** (only Free and not Free)
4. Determine how long it takes for a car to fully charge. - **NI**

Functional Requirements

Locating a Nearby Charger - cont.

5. Determine which charger is the best based on speed or price. - **I**
6. Get the current charge of the vehicle from an API call or from user entry. - **NI**
7. Determine if a charger is currently in use. - **NI**
8. Report feature to allow users to report on the status and quality of a charger. - **NI**

Functional Requirements

Planning a Trip with Stops for Charging

1. App will function like a mapping software (Google Maps) - I (navigation external)
2. Users will be able to add stops in a trip based on when they want to charge. - **NI**
3. Presents users with the best charger near the requested stop. - **I** (city dependant)
4. Modify travel time based on how long each charge takes. - **NI**
5. Allow users to add stops while they are on the trip. - **NI**

Functional Requirements

Connecting to Existing Subscription Services

1. Users can enter in pre-existing subscription services. - **I** (does nothing yet)

2. User’s services will be stored on the database. - **I**

Functional Requirements

Voice Command Navigation

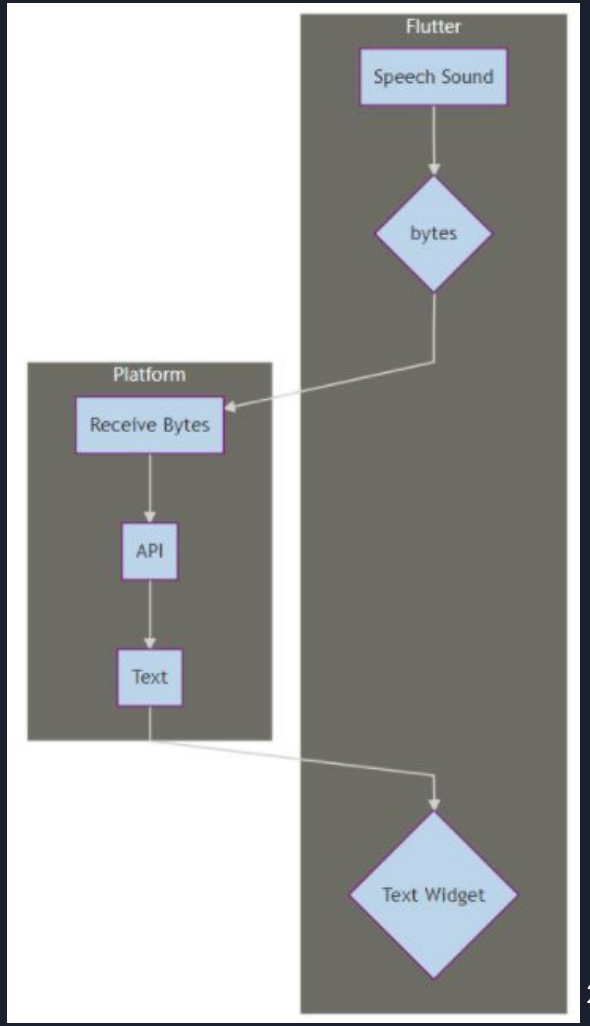
1. The app will respond to voice commands. - **I** (for city)

2. The app will navigate users based on their voice prompt. - **NI**

Voice Recognition

- Software able to recognize users voice.
  - Voice commands.
  - Aids the handsfree vision we have.

- Easy manage and search.
  - Name location or destination.
  - App will direct to the best possible route.



Voice Recognition (cont.)

- Flutter_Speech library.
  - Will listen to the user and display it onto the screen.

Non-Functional Requirements

1. The charger locator must return results to the users in a quick and efficient manner. - **I**
2. User Credit Card information will be encrypted. - **NI**
3. Users will be prompted with a first-time sign in. - **I**
4. The software will store all user PID in a secure database. - **NI** (not secure)

Non-Functional Requirements (cont.)

5. The software will be developed for iOS, but will be developed to make porting Android easy. - **NI**

6. The software must be usable by people of all technical skill levels. - **I**

7. Help menus and tutorials must be present in the software. - **I**

8. The software must automatically pay for the charger once the user confirms they wish to start a charge. - **NI**

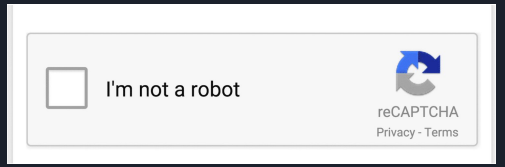
Account Creation with Other Services

- Proof of Concept: Python Web scraper
  - Python is good for web scraping
  - BeautifulSoup + Mechanize

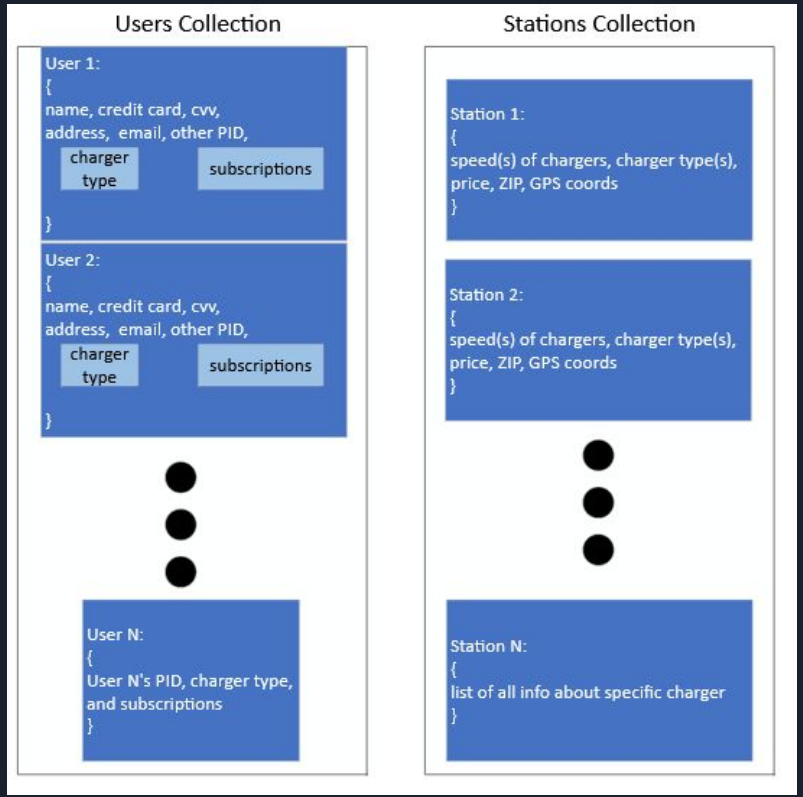
- Locate sign-in page and extract forms
  - Fill form data with PID from user

Account Creation with Other Services (cont.)

- Pitfalls
  - Every sign-up process is different
  - Not all services have webpage for sign-up
    - Solution: Download their app
    - Issue: Defeats purpose of requirement
  - Captchas
    - Solution: Partner with service
    - Issue: Other services not want to partner with competitors



Database Design



Testing

- White-box Testing
- Integration
  - Connecting pages as we go
  - Fix bugs as they appeared
- Android Emulator and Physical phone

“Future plans”

- Full / finished iOS integration
- Google and Apple Login
- Trip planner
- The amount of time to charge the vehicle
  - API calls to the users car (smartcar API)
- Have the voice driven system fully implemented

Known Bugs

- Navigation crashes app on startup occasionally
  - Unable to reliably reproduce

Demo!

Conclusion

The client, Envorso requested an app to solve the hassle of EV car charging.

Solution: enRoute

- A car charger finder
- Filters chargers by the user preference
- Built with the Flutter Framework
- Implements Google Maps APIs and Firebase
- Highly functional and user friendly.