ENRoute

Application to find electric vehicle charging stations

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

Table of Contents

The Development Team

Kirsten - GUI design

Craig - Maps API

Joe - Voice Command

Richard - Database + User Accounts

Lucas - Charger Data

Our Client

envorso

Transforming Productivity

Envorso - a modern solutions firm

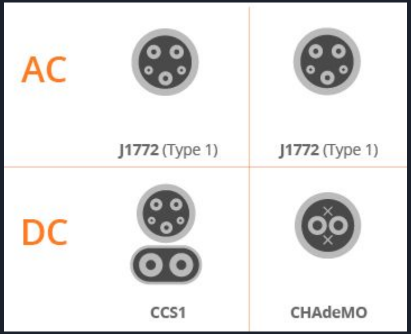
- Bob Rapp
  - Business Consultant
    - Experience consulting top executive
    - Experience Building large scale projects
- Parker Jones
  - Technical Consultant
    - Experience with software development
- University Advisor - Professor Vajda

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:
  - Price, Brand, Speed, Distance, Plug type preference
  - GPS location
- Highly Usable
- Highly Functional

Tools

Front-End: Flutter + Dart

Back-End: Google Firebase

Google Maps API

Other Car Charger APIs

iOS Development: CodeMagic and other iOS solutions!

Functional Requirements

First Time Login

1. The software will prompt users for PID info and their car’s charge port type. - **H**
2. The software will allow users to sign up with a new account. - **H**
3. The software will allow users to sign up using Google or Apple. - **L**
4. The software will ask users for their ZIP-code to generate a list of nearby charging services that they can sign up for. - **L**

Functional Requirements

Locating a Nearby Charger (Fastest or Cheapest)

1. The software can match the vehicle's charge port with chargers nearby. The software will not present users with a charger they are not compatible with. - **H**
2. The software will be able to determine the speed of chargers located. - **H**
3. The software will be able to determine the price of chargers located. - **H**
4. The software will be able to determine how long it takes for a car to fully charge. - **H**

Functional Requirements

Locating a Nearby Charger - cont.

5. The software will be able to determine which charger is the best based on speed or price of the charger (determined by user) - **H**
6. The software will get the current charge of the vehicle from an API call or from user entry. - **L**
7. The software will be able to determine if a charger is currently in use. - **O**
8. The software will have a report feature to allow users to report on the status and quality of a charger. - **O**

Functional Requirements

Planning a Trip with Stops for Charging

1. The software will be able to function like a mapping software (Google Maps or Waze) - **H**
2. Users will be able to add stops in a trip based on when they want to charge. - **O**
3. The software will present users with the best charger near the requested stop. - **O**
4. The software will modify travel time based on how long each charge takes. - **O**
5. The software will allow users to add stops while they are on the trip. - **O**

Functional Requirements

Connecting to Existing Subscription Services

1. The software will have a place for users to enter in pre-existing subscription services. - **H**
2. The software will need to store each user’s subscription service on a database. - **H**

Functional Requirements

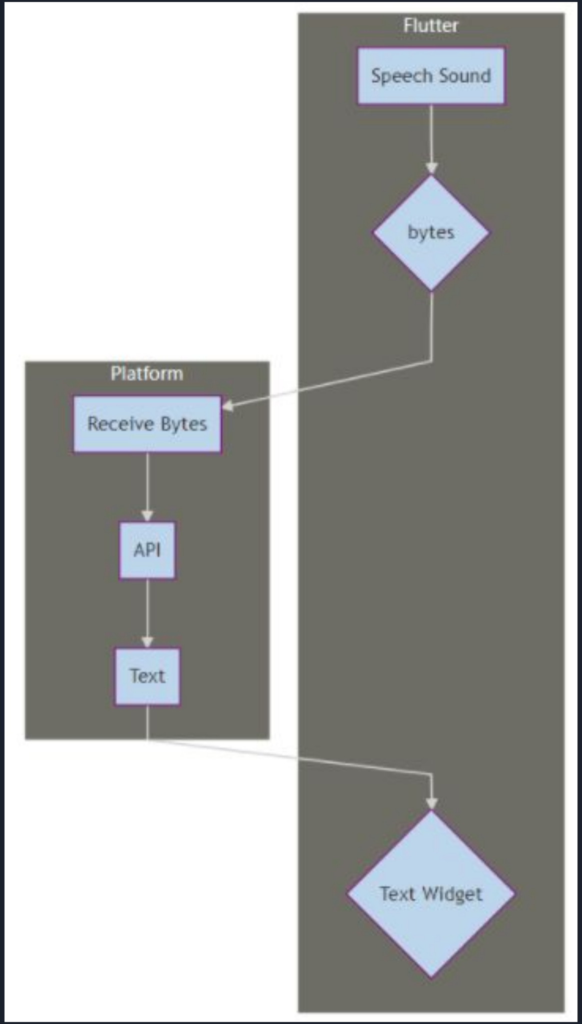
Voice Command Navigation

1. The software will respond to voice commands. - **H**
2. The software will be able to navigate users to a charger based on what a user asks for in the voice prompt. - **H**

Voice Recognition

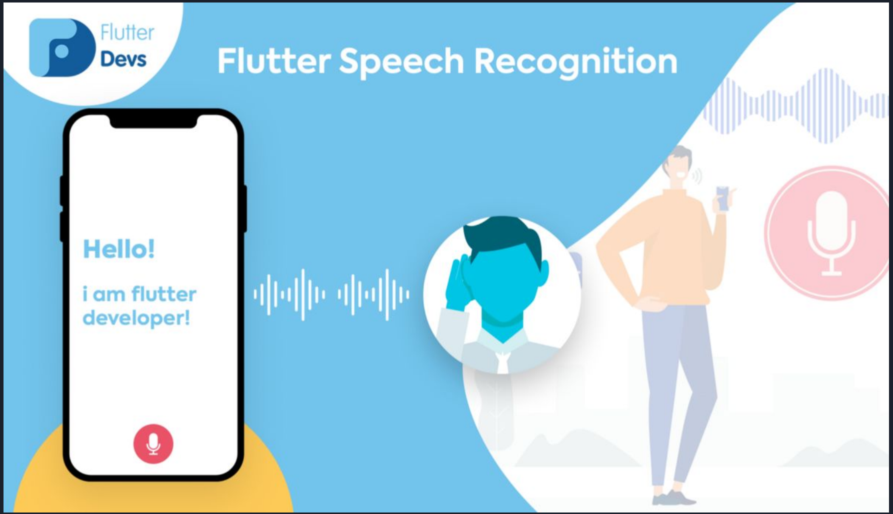
- Software that is able to recognize users voice.
    - Voice commands.
    - Aids the handsfree vision we have.

- Easy to manage and search.
    - Name a location or destination.
    - App will be directed 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

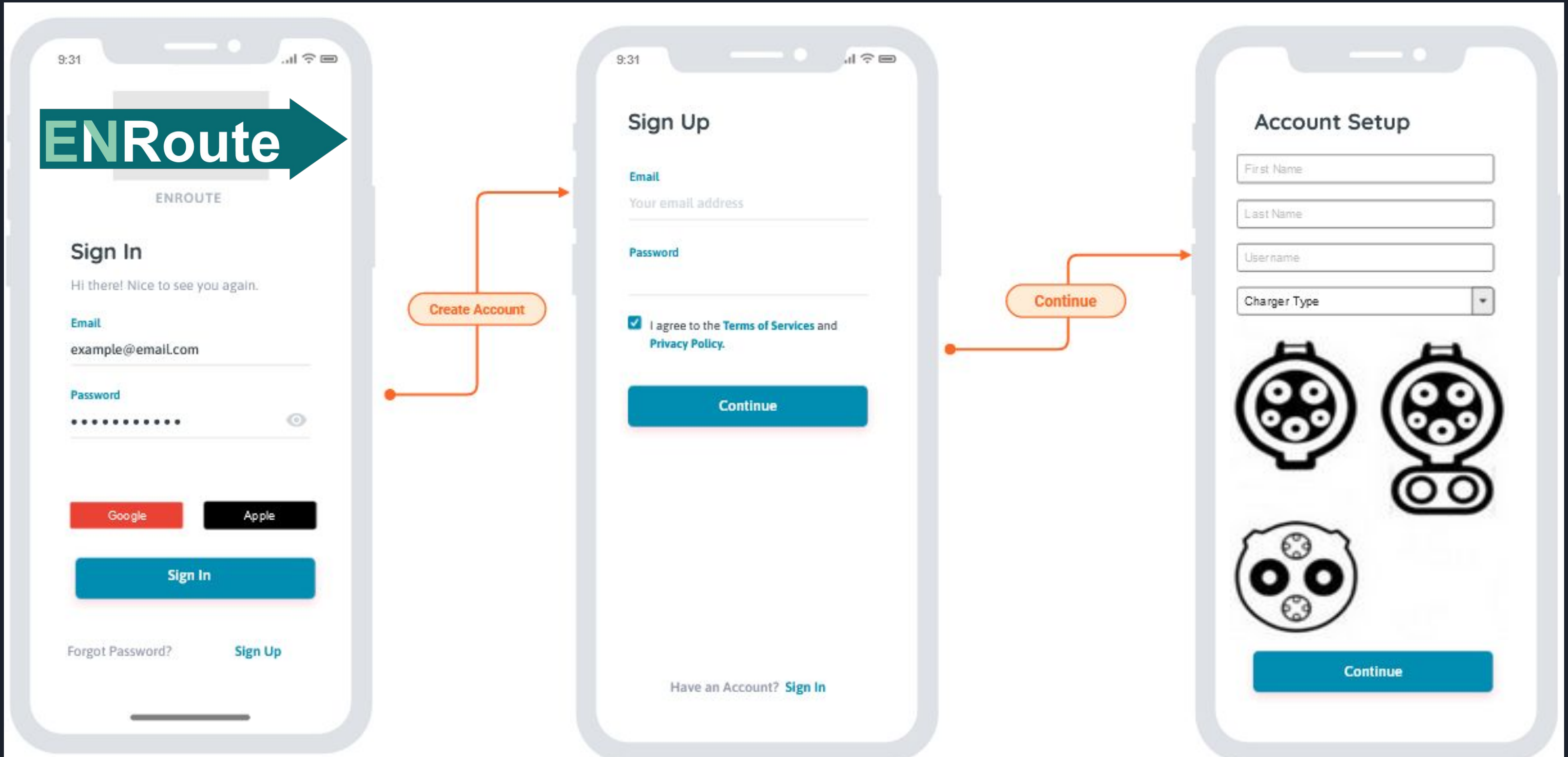
1. The charger locator must return results to the users in a quick and efficient manner. - **H**
2. User Credit Card information will be encrypted. - **O**
3. Users will be prompted with a first-time sign in to create an account and enter their subscription services. - **H**
4. The software will store all user PID in a secure database. - **O**
5. The database that stores PID must be protected to prevent user data from leaking to external sources. - **O**

Non-Functional Requirements (cont.)

6. The software will be developed for iOS, but will be developed to make porting Android easy. - **H**
7. The software must be usable by people of all technical skill levels. - **H**
8. Help menus and tutorials must be present in the software. - **H**
9. The software must automatically pay for the charger once the user confirms they wish to start a charge. - **L**

User Interface Design

First sign in



Sign In

Hi there! Nice to see you again.

Email: example@email.com

Password: ●●●●●●●●●●●

Sign Up

Email: Your email address

Password

☑ I agree to the Terms of Services and Privacy Policy.

Have an Account? Sign In

Account Setup

First Name

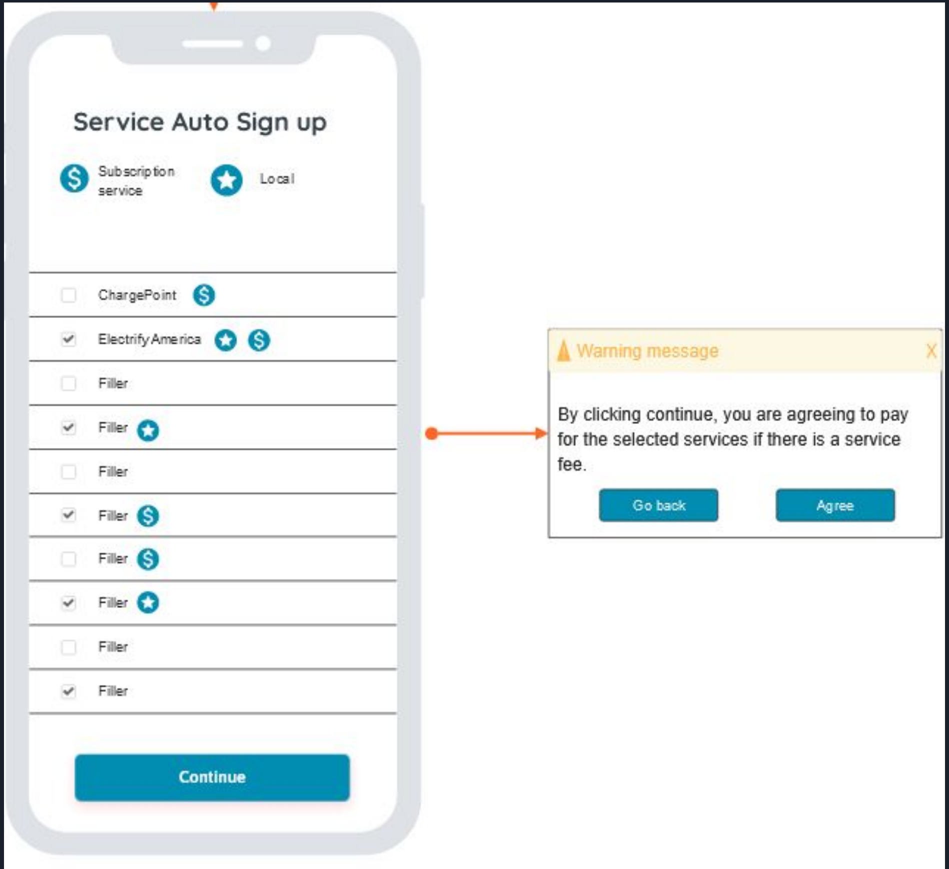
Last Name

Username

Charger Type

User Interface Design

First sign in - cont.



Service Auto Sign up

$ Subscription service ★ Local

- □ ChargePoint $
- ☑ ElectrifyAmerica ★ $
- □ Filler
- ☑ Filler ★
- □ Filler
- ☑ Filler $
- □ Filler $
- ☑ Filler ★
- □ Filler
- ☑ Filler

Continue

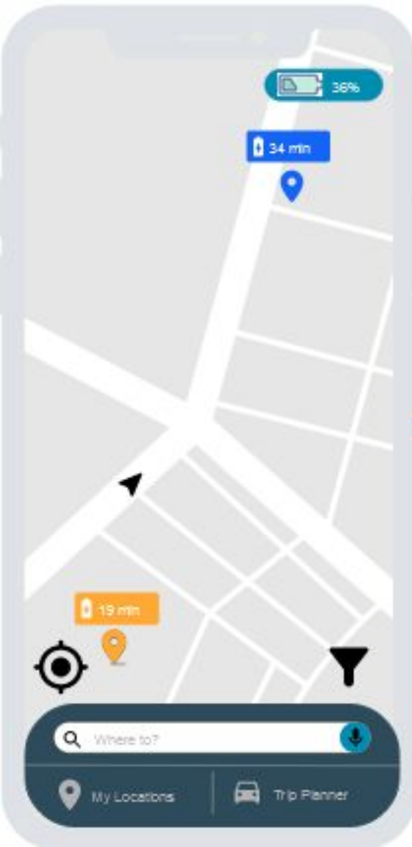
Warning message X

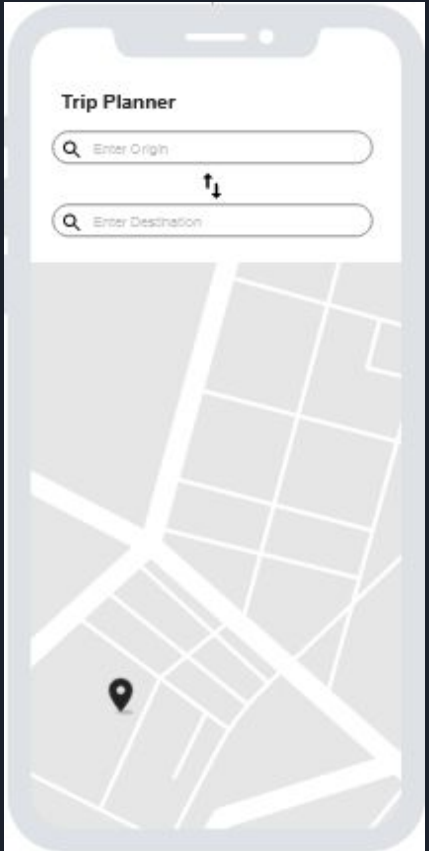
By clicking continue, you are agreeing to pay for the selected services if there is a service fee.

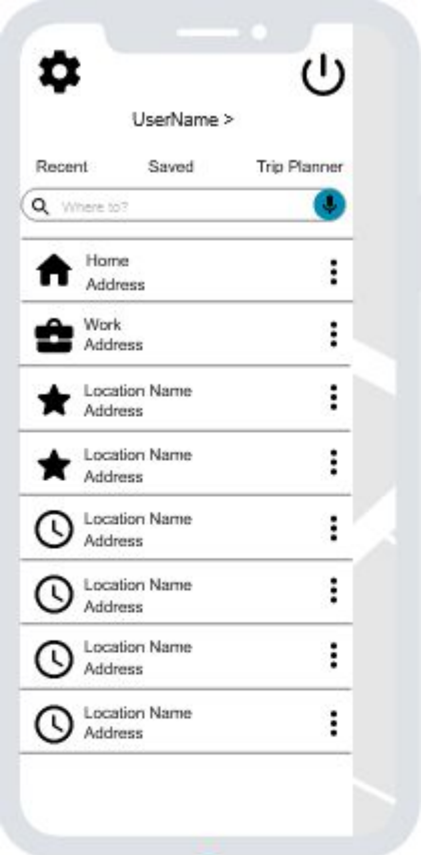
Go back Agree

User Interface Design

Home

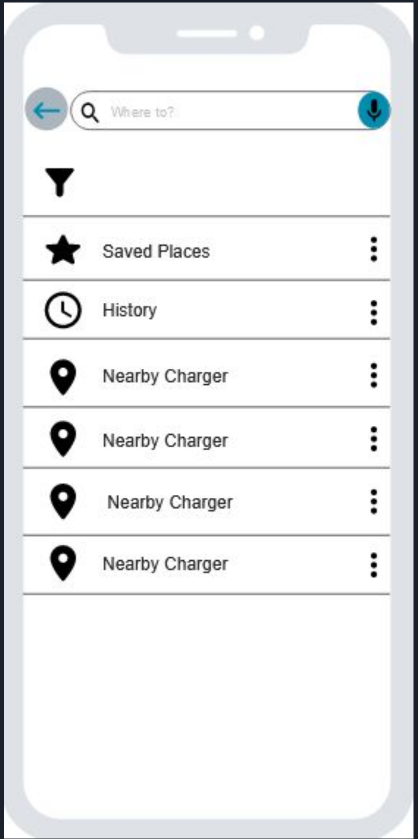






User Interface Design

Home



Where to?

Saved Places

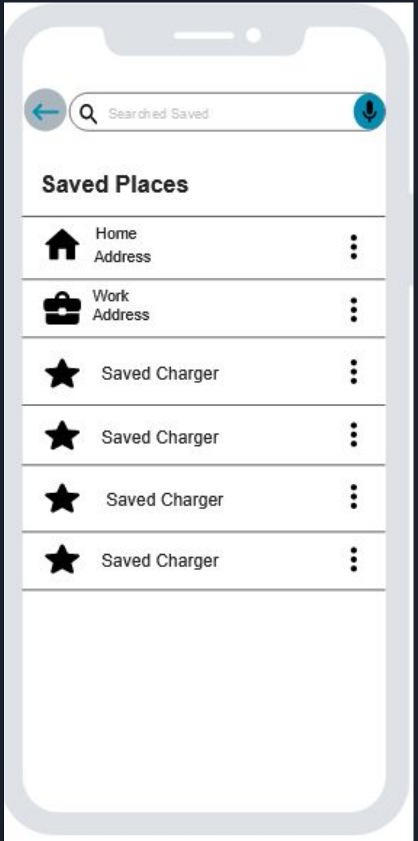
History

Nearby Charger

Nearby Charger

Nearby Charger

Nearby Charger



Searched Saved

Saved Places

Home
Address

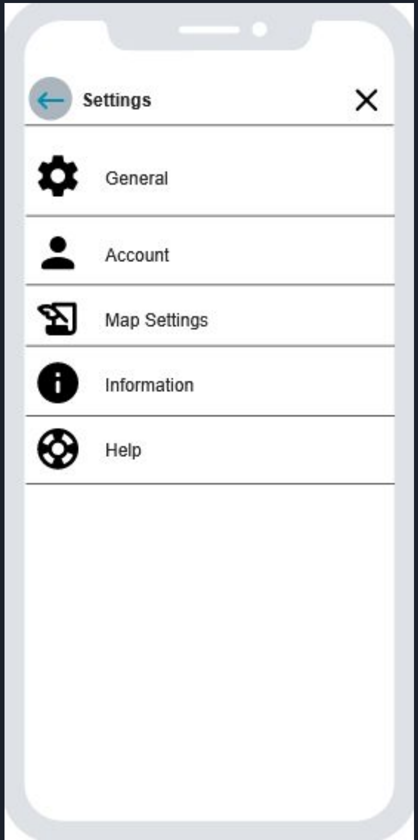
Work
Address

Saved Charger

Saved Charger

Saved Charger

Saved Charger



Settings

General

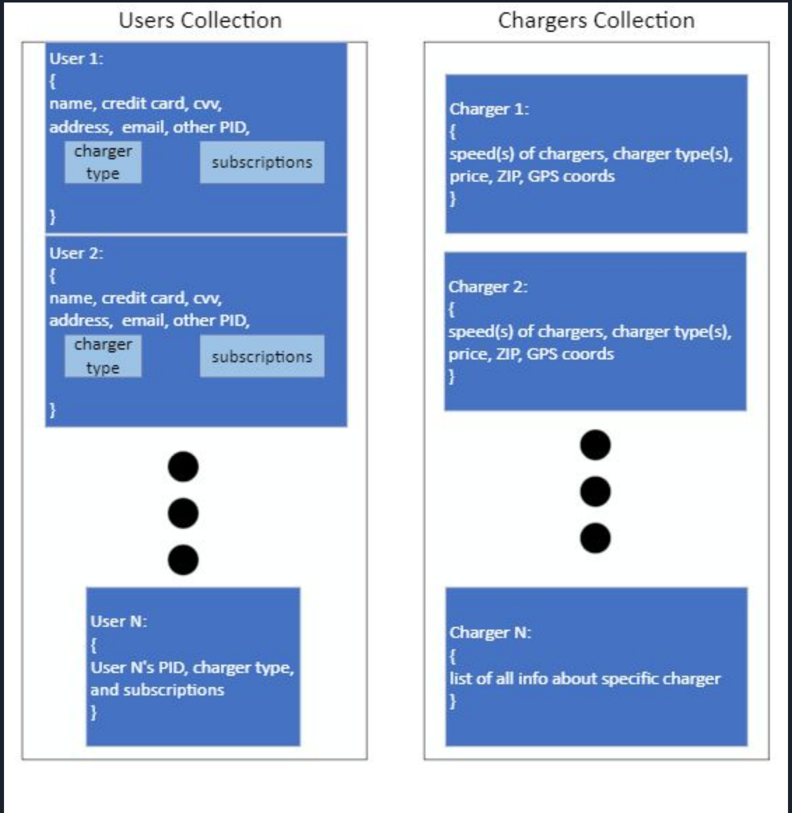
Account

Map Settings

Information

Help

Database Design



Users Collection

User 1:
{
name, credit card, cvv,
address, email, other PID,
charger type
subscriptions
}

User 2:
{
name, credit card, cvv,
address, email, other PID,
charger type
subscriptions
}

User N:
{
User N's PID, charger type, and subscriptions
}

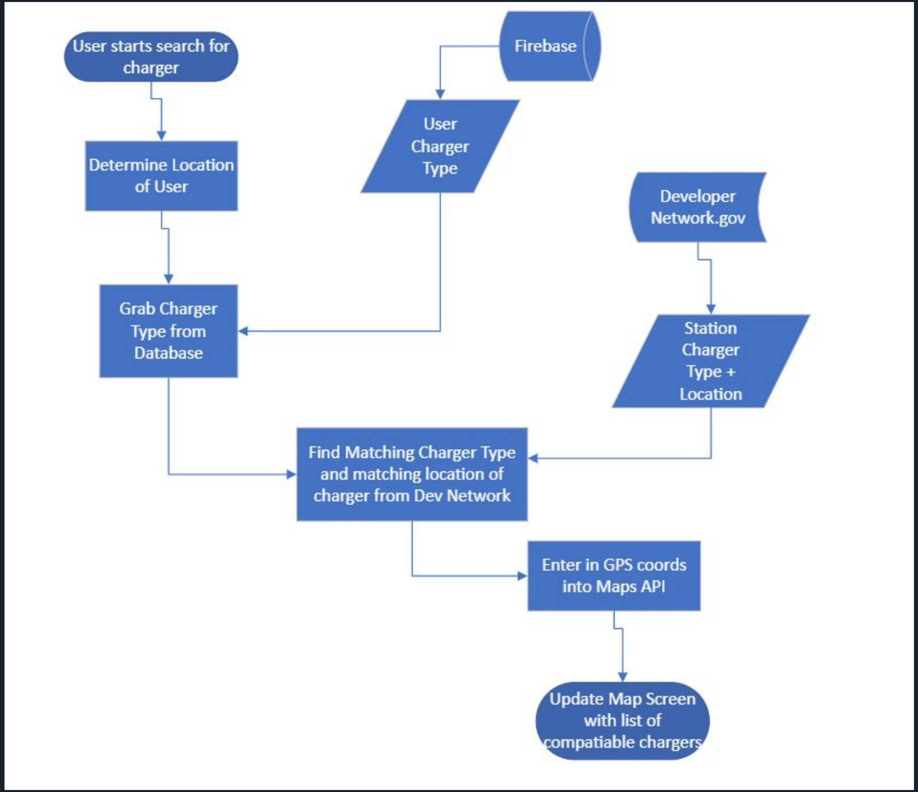
Chargers Collection

Charger 1:
{
speed(s) of chargers, charger type(s),
price, ZIP, GPS coords
}

Charger 2:
{
speed(s) of chargers, charger type(s),
price, ZIP, GPS coords
}

Charger N:
{
list of all info about specific charger
}

Database to Maps API Communication



User starts search for charger

Determine Location of User

Grab Charger Type from Database

Firebase

User Charger Type

Developer Network.gov

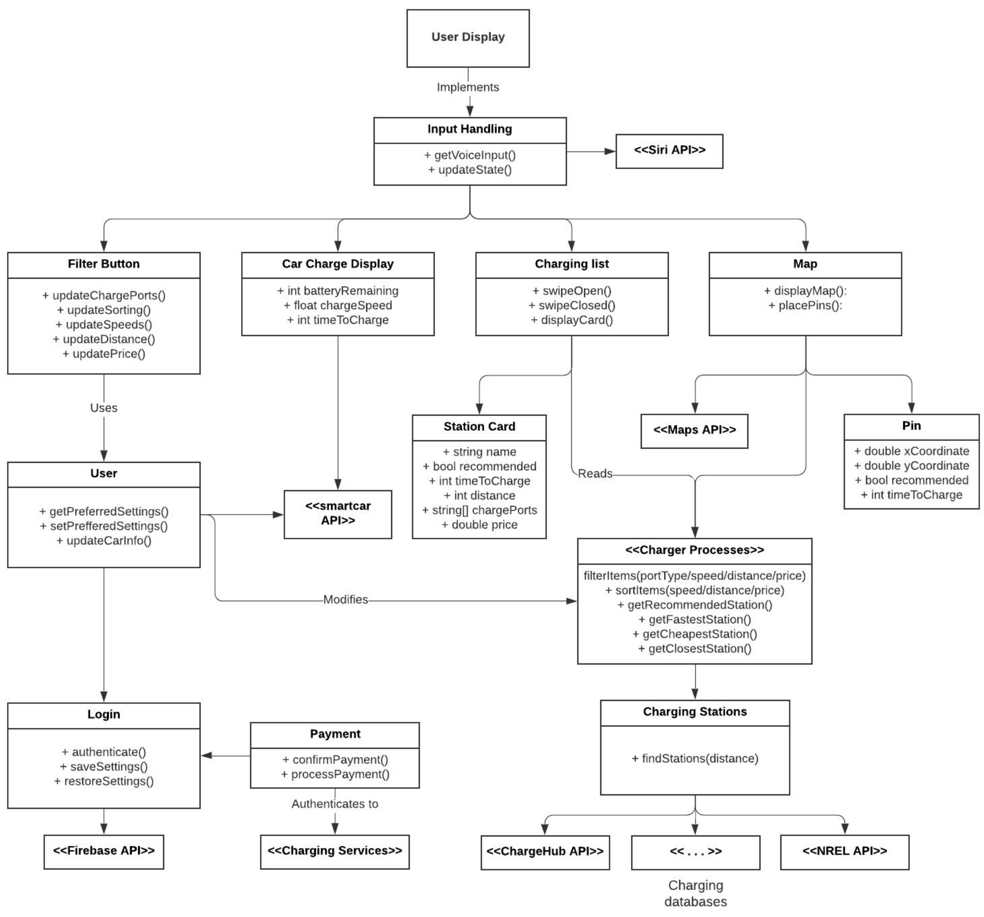
Station Charger Type + Location

Find Matching Charger Type and matching location of charger from Dev Network

Enter in GPS coords into Maps API

Update Map Screen with list of compatiable chargers

Class relationship UML Diagram



User Display

Implements

Input Handling

+ getVoiceInput()
+ updateState()

<<Siri API>>

Filter Button

+ updateChargePorts()
+ updateSorting()
+ updateSpeeds()
+ updateDistance()
+ updatePrice()

Car Charge Display

+ int batteryRemaining
+ float chargeSpeed
+ int timeToCharge

Charging list

+ swipeOpen()
+ swipeClosed()
+ displayCard()

Map

+ displayMap():
+ placePins():

Uses

User

+ getPreferredSettings()
+ setPrefferedSettings()
+ updateCarInfo()

<<smartcar API>>

Station Card

+ string name
+ bool recommended
+ int timeToCharge
+ int distance
+ string[] chargePorts
+ double price

<<Maps API>>

Pin

+ double xCoordinate
+ double yCoordinate
+ bool recommended
+ int timeToCharge

Reads

<<Charger Processes>>

filterItems(portType/speed/distance/price)
+ sortItems(speed/distance/price)
+ getRecommendedStation()
+ getFastestStation()
+ getCheapestStation()
+ getClosestStation()

Modifies

Login

+ authenticate()
+ saveSettings()
+ restoreSettings()

Payment

+ confirmPayment()
+ processPayment()

Authenticates to

Charging Stations

+ findStations(distance)

<<Firebase API>>

<<Charging Services>>

<<ChargeHub API>>

<< . . . >>

<<NREL API>>

Charging databases

Why Firebase?

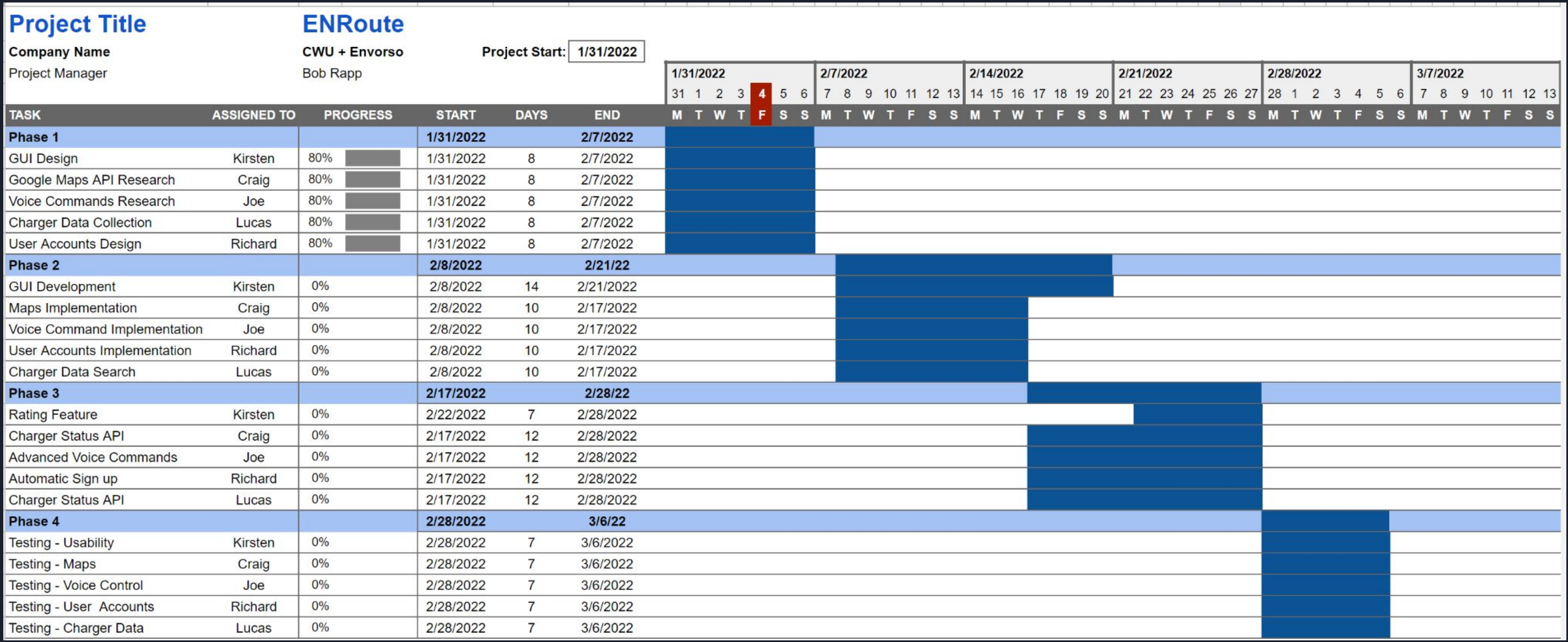
- Easy to set-up
- Free
- Flutter and Firebase are both Google Products
- NoSQL allows for flexibility of data

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

Demo!

Gantt Chart



Project Title **ENRoute**

Company Name **CWU + Envorso** **Project Start:** 1/31/2022

Project Manager Bob Rapp

1/31/2022 2/7/2022 2/14/2022 2/21/2022 2/28/2022 3/7/2022

31 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 1 2 3 4 5 6 7 8 9 10 11 12 13

M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S S

TASK	ASSIGNED TO	PROGRESS	START	DAYS	END
Phase 1			**1/31/2022**		**2/7/2022**
GUI Design	Kirsten	80%	1/31/2022	8	2/7/2022
Google Maps API Research	Craig	80%	1/31/2022	8	2/7/2022
Voice Commands Research	Joe	80%	1/31/2022	8	2/7/2022
Charger Data Collection	Lucas	80%	1/31/2022	8	2/7/2022
User Accounts Design	Richard	80%	1/31/2022	8	2/7/2022
Phase 2			**2/8/2022**		**2/21/22**
GUI Development	Kirsten	0%	2/8/2022	14	2/21/2022
Maps Implementation	Craig	0%	2/8/2022	10	2/17/2022
Voice Command Implementation	Joe	0%	2/8/2022	10	2/17/2022
User Accounts Implementation	Richard	0%	2/8/2022	10	2/17/2022
Charger Data Search	Lucas	0%	2/8/2022	10	2/17/2022
Phase 3			**2/17/2022**		**2/28/22**
Rating Feature	Kirsten	0%	2/22/2022	7	2/28/2022
Charger Status API	Craig	0%	2/17/2022	12	2/28/2022
Advanced Voice Commands	Joe	0%	2/17/2022	12	2/28/2022
Automatic Sign up	Richard	0%	2/17/2022	12	2/28/2022
Charger Status API	Lucas	0%	2/17/2022	12	2/28/2022
Phase 4			**2/28/2022**		**3/6/22**
Testing - Usability	Kirsten	0%	2/28/2022	7	3/6/2022
Testing - Maps	Craig	0%	2/28/2022	7	3/6/2022
Testing - Voice Control	Joe	0%	2/28/2022	7	3/6/2022
Testing - User Accounts	Richard	0%	2/28/2022	7	3/6/2022
Testing - Charger Data	Lucas	0%	2/28/2022	7	3/6/2022

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.