CS 3354 Software Engineering
Final Project Deliverable II

EV Comms

Bryce Canillo, James Dao, Aditya Guin, Salman Hossain, Nam Truong, Sergei Wong

Delegation of Tasks

Name	**Task**
Nam Truong	Software requirements and project scope (**deliverable 1**), Product scheduling and costs (**deliverable 2**)
Sergei Wong	Use case diagrams and sequence diagrams (**deliverable 1**), Comparison of work with similar designs and presentation slides (**deliverable 2**)
Bryce Canillo	Create class diagram **(deliverable 1).** Assist architectural design diagram, conclusion, and references **(deliverable 2)**
Salman Hossain	Sequence diagrams (**deliverable 1**). Comparison of work with similar designs, presentation slides (**deliverable 2**).
Aditya Guin	Project Description. Architectural Design description and explanation (**deliverable 1**). Junit Test for application (**deliverable 2**)
James Dao	Assist in class diagram **(deliverable 1).** Architectural design diagram along with conclusion and references (**deliverable 2**)

Project Deliverable 1 Content

Project Description

Title: EV Comms

Group Members
Bryce Canillo
James Dao
Aditya Guin
Salman Hossain
Nam Truong
Sergei Wong

Project Idea
Car communication app between Electric Vehicle drivers. Users would be able to create an account, add friends, and see where other Electric Vehicle drivers are. In addition, communication between users for different charging stations or if certain charging stations are down. Another feature would be the communication of any external car damages to drivers who may be unaware of the damage. For example, if debris on the highway causes a scratch or flat tire while driving. This would create more individualized communication and convenience between drivers.

Motivation
With new electric companies coming up and about, the US is on road to transition to electric vehicles. As of 2019, there were 20 electric car company manufacturers. In addition, there are currently more than 2 million electric cars in the USA. As this number increases, we believe that having this communication app will improve the driver experience. In addition, communication of potential damages will reduce the probability of any accident. Therefore we believe that this application can be used in commonly.

Our feedback for the project was to think about similar apps and ensure that our application has something different than those apps. After researching, we found two applications, FlexCharging and OptiWatt, that show the best prices for charging stations. However, our app also includes communication between electric vehicle drivers and not just information about vehicle charging.

GitHub Actions
Link: https://github.com/salmanhossain1/3354-EVComms
README.md update: Aditya Guin
Project Scope: Nam Truong

Delegation of Tasks

Name	**Task**
Nam Truong	Product scheduling and costs (part of deliverable 2), Software requirements
Sergei Wong	Provide a use case diagram for our project. Create the sequence diagrams for each use case of the project.
Bryce Canillo	Assist in the production of case diagrams and sequence diagrams for each case of the project
Salman Hossain	Set up the GitHub repository and make sure access levels are granted and information is secure. Committed first README.md document. Assist in sequence diagram creation
Aditya Guin	Project Description. Architectural Design description and explanation. Updated README.md.
James Dao	Responsible for architectural design and considering to use a client-server architecture pattern or repository pattern.

Software Process Model

We believe that the incremental model is the best model to use for our application. Every delivery would be built upon the previous ones, ultimately encapsulating all our desired functionalities. For example, the first increment can enable a user to create an account and sign in through different means such as Gmail, Facebook, or Apple. The second and third increments would be different users adding other users as friends, and being able to communicate with them. Following this, we would integrate charging station locations using a map API and implement functionality to list the closest charging stations for a user. We were thinking of the spiral model but saw that projects following the spiral model generally have a lot of risks, and we believe our project isn't as risky.

Software Requirements

Functional Requirements

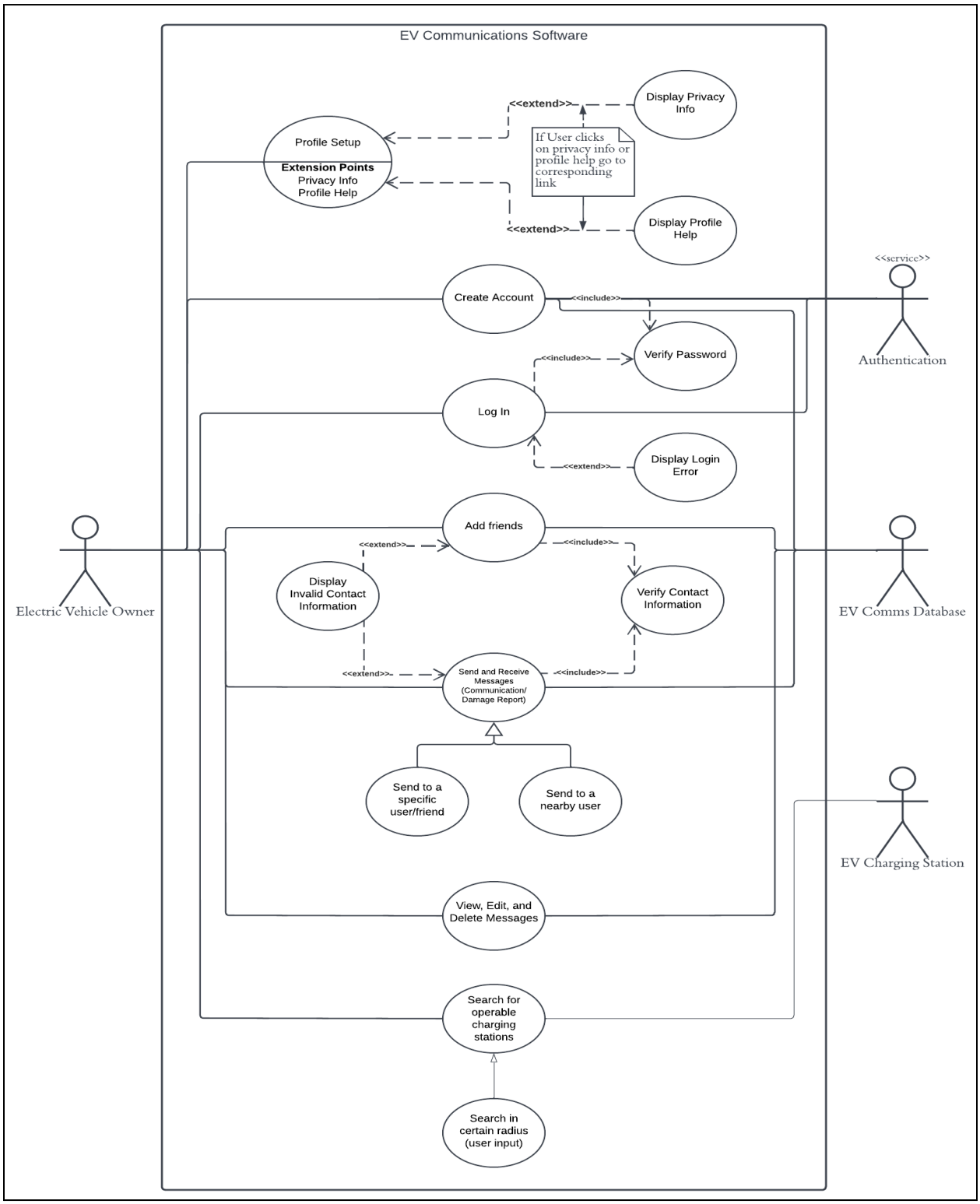
1. The user shall be able to do various actions on messages; view, edit, delete, send, and reply
2. The app shall be able to uniquely identify cars based on its given attributes and current geolocation.
3. The user shall be able to navigate to specific cars to message through the app
4. Each user shall be uniquely identified by their car and its attributes
5. The app shall notify users when they receive a message

Non-Functional Requirements

1. Usability
    a. The usability shall be easy to learn and navigate through.
2. Performance
    a. Messages shall be sent at most with 0.2 seconds delay.
3. Space
    a. The app should take less than 2gb of space on a phone.
4. Dependability
    a. The app shall accurately track cars and send messages to the right users.
5. Security
    a. The app shall ask for permission on accessing geolocation and disable location tracking when the user specifies not to.
6. Environmental
    a. The app development shall use the same iOS/android framework.
7. Operational
    a. The language used shall be that of the chosen framework.
8. Development
    a. The app will be written in a client-server architecture.
9. Regulatory
    a. The app shall adhere to location tracking standards.
10. Ethical
    a. The app shall not maliciously give out user data.
11. Accounting
    a. The app shall accurately keep a record of data it obtains.
12. Safety/Security
    a. The app will not take data of the user without permission and protect against other users and people.

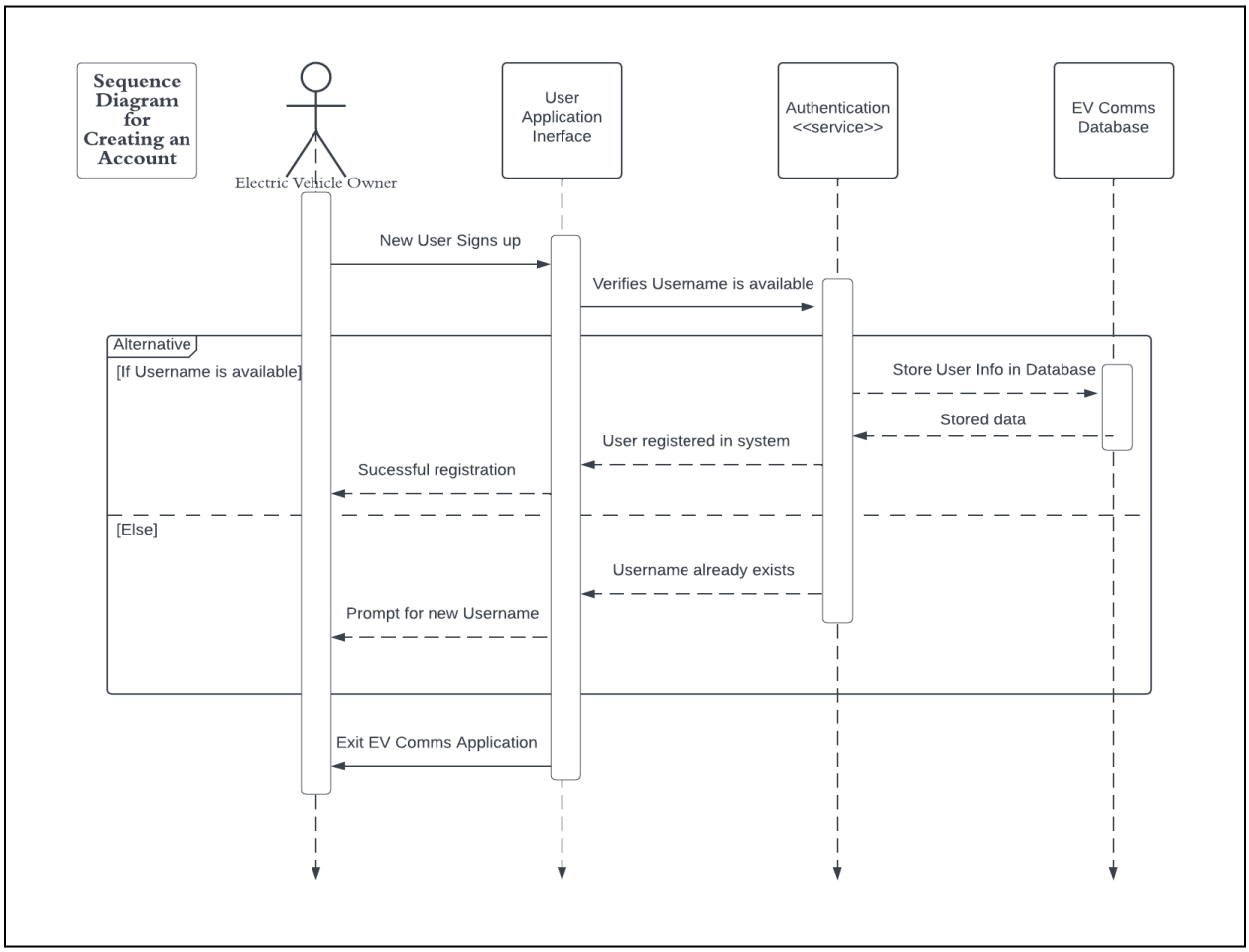
Use Case Diagram

Use Case Diagram for EV Comms

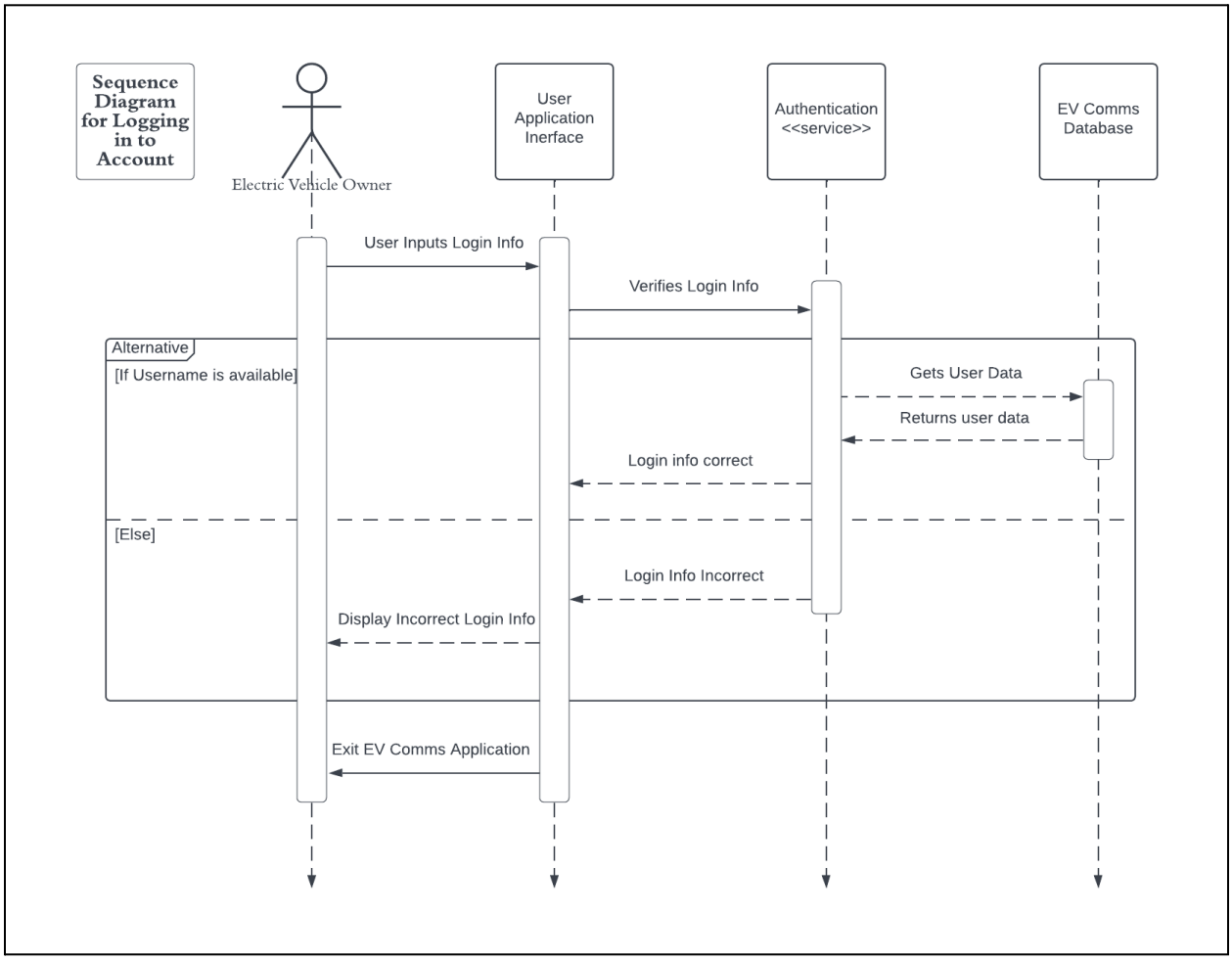


1. Sequence Diagrams

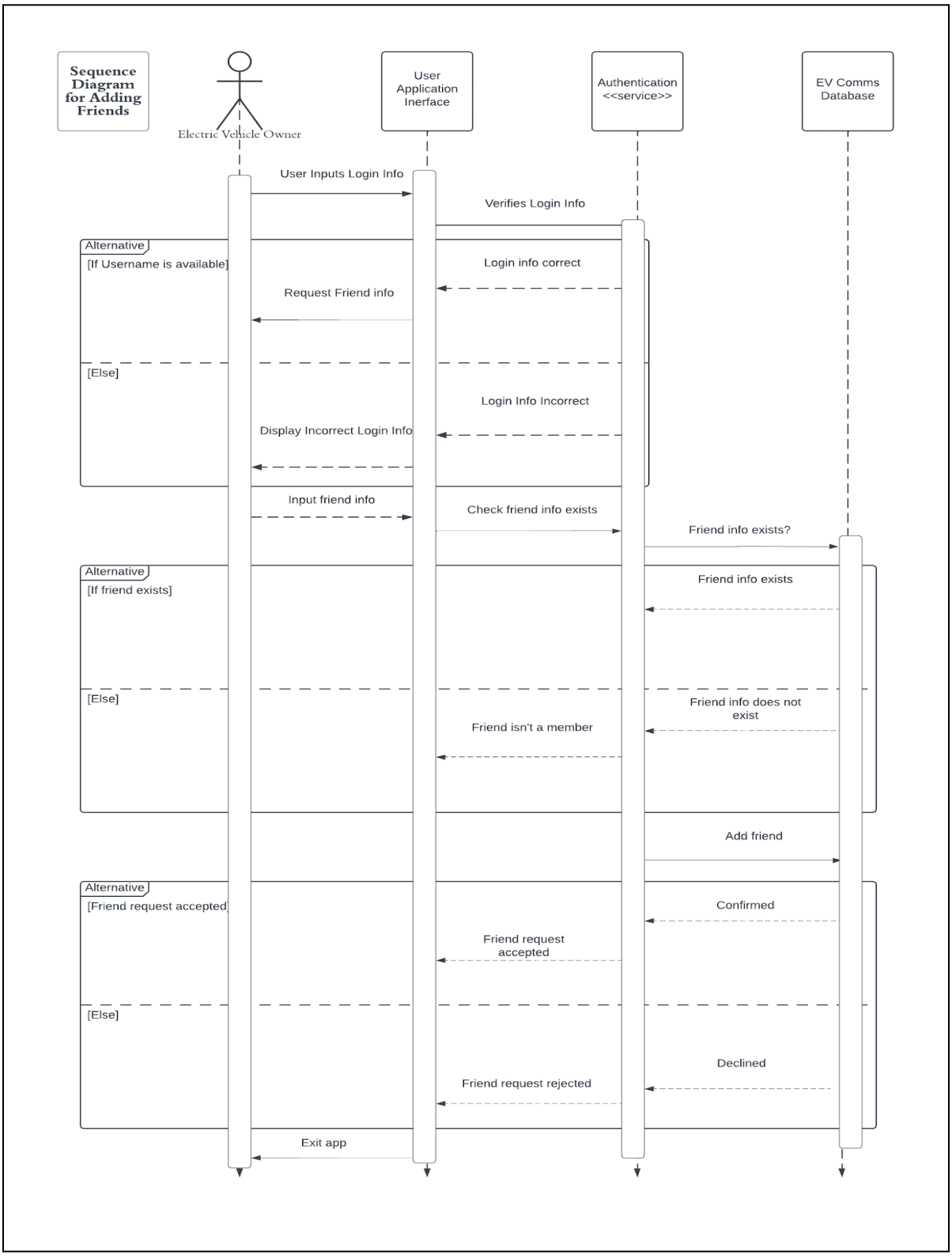
Sequence Diagram for Creating an Account



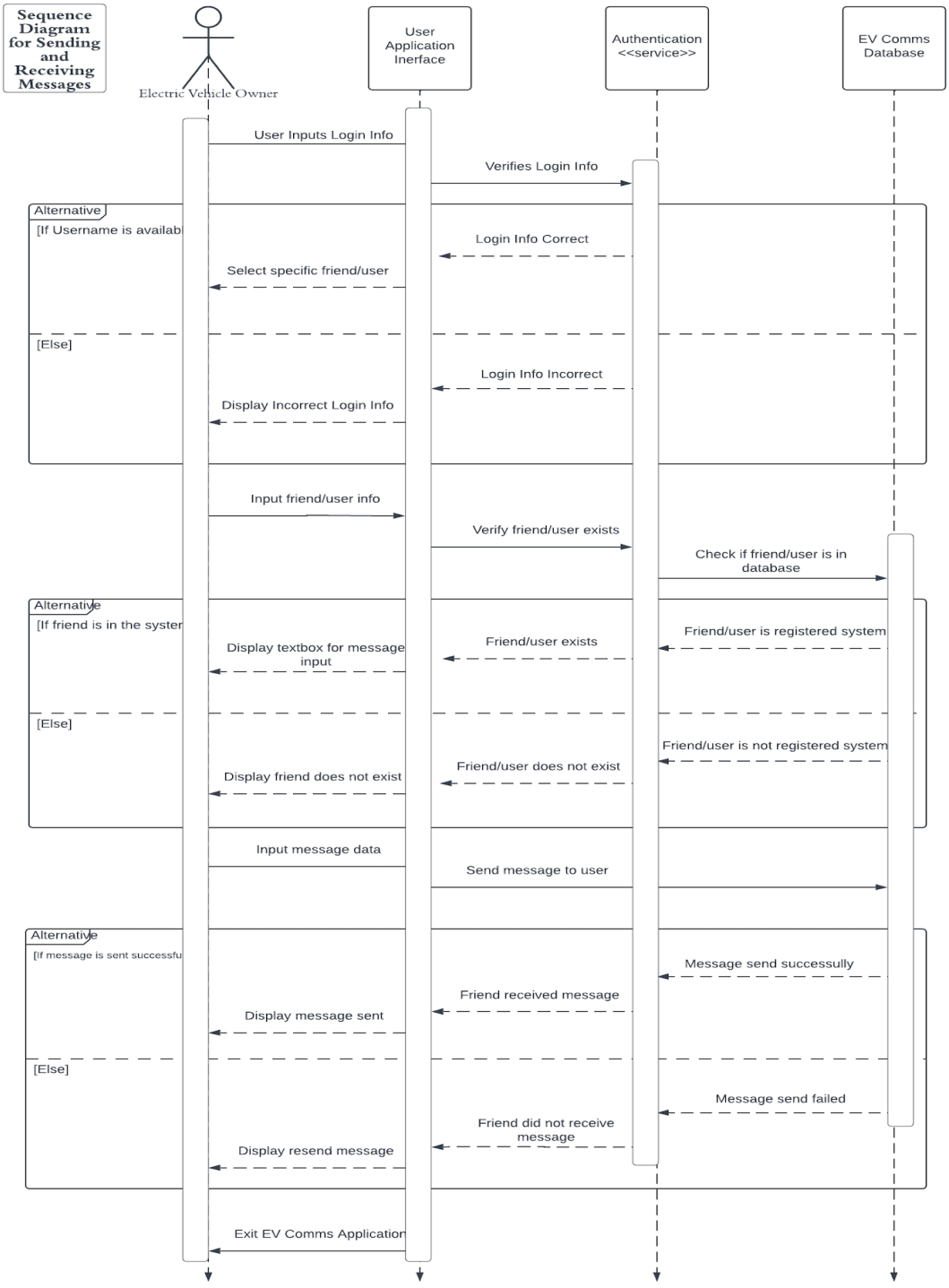
Sequence Diagram for Logging into account



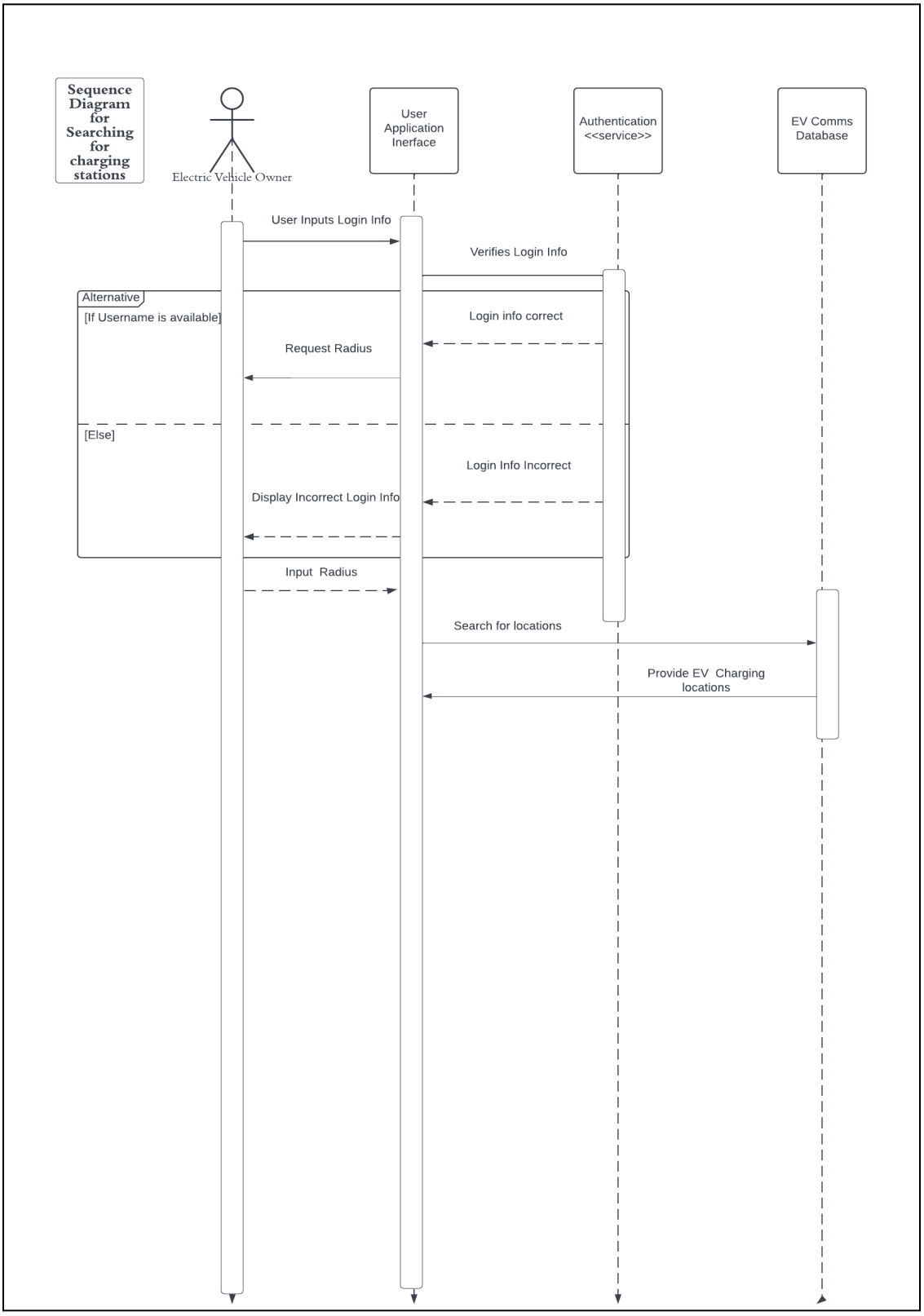
Sequence Diagram for adding friends



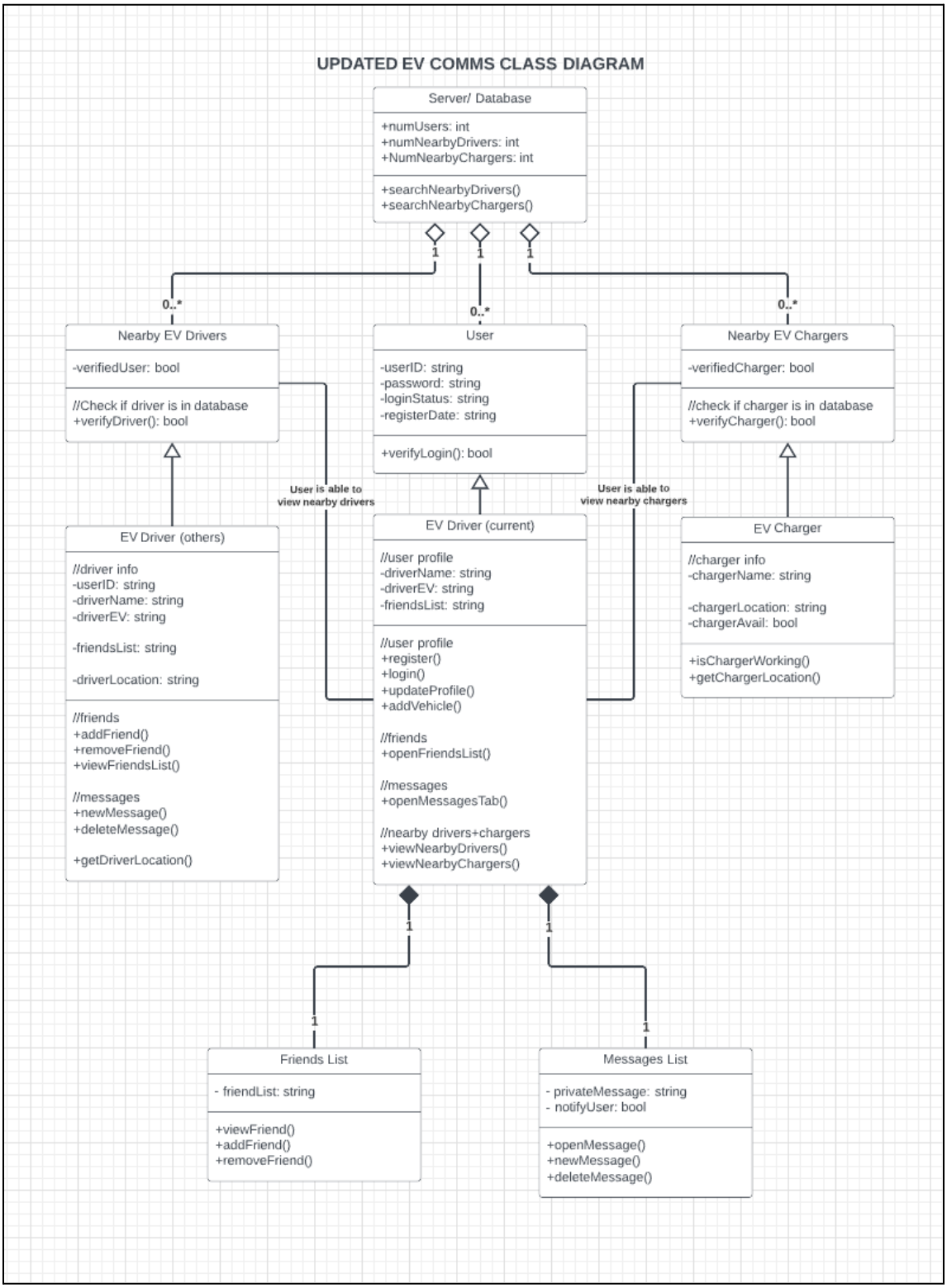
Sequence Diagram for adding sending and receiving messages



Sequence Diagram for Searching for charging stations

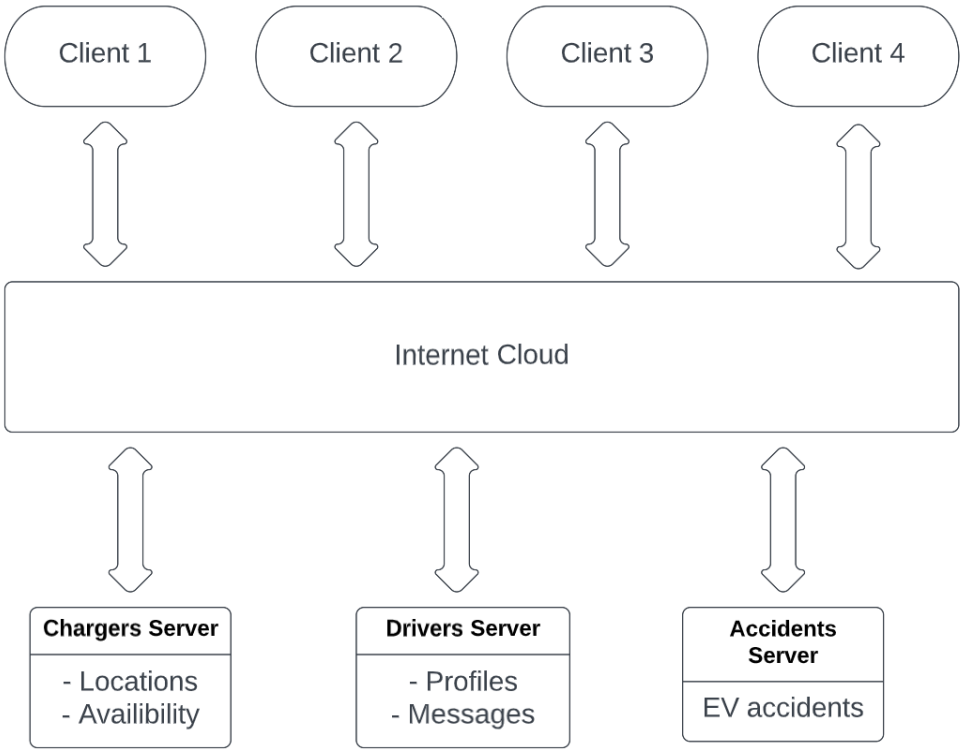


2. Class Diagram



Architectural Design

After a group discussion, we decided that the Client-server architecture is the best model for our application. In our application, different electric-vehicle drivers will create their own account, and be able to communicate with other drivers. In addition, if there are any updates on non-usable charging stations, or any major accidents involving electric vehicles, this message would need to be relayed. Therefore, having servers across the world (since electric vehicles are driven globally) would be necessary. In addition, because of the number of users (5+ million EVs currently), there needs to be multiple servers, as having a single server (repository model) would cause crashes.



Deliverable 2

3.1

Activities	Estimated Length of Completion (days)
Account System	2
Messaging UI	2
Notification System	3
Messaging Functionality(store and retrieve messages from database)	6
Location Tracker and locator algorithm	6
User Input UI	4
Scaling into Distributed Systems	7
Efficiency Improvements	4
Total	**34**

Estimation of days for each activity is taken from prior experience on how quickly certain functionalities can be coded. Creating a UI will usually take shorter periods of time such as 2 or 3 days. However, building more complex systems and an efficient backend system will take much more time.

Working schedules will not consist of weekends and assuming normal working hours, each developer is expected to put 6-7 hours a day towards project development.

Activity	**Start Date**	**End Date**
Account System	November 14, 2022	November 16, 2022
Messaging UI	November 14, 2022	November 16, 2022
User Input UI	November 15, 2022	November 21, 2022
Messaging Functionality	November 21, 2022	November 29, 2022
Location Tracker	November 24, 2022	November 32, 2022
Notification System	November 25, 2022	November 30, 2022
Scaling into Distributed Systems	November 30, 2022	December 9, 2022
Efficiency Improvements	December 10, 2022	December 15, 2022

Most of the simple activities such as UI can be worked on simultaneously as a team. However, as the project develops into the backend portion, most backend functionalities cannot start at the same time but must build on each other.

3.2

We used the FP algorithm.

	Function Category	Count	Complexity			Count X Complexity
			Simple	Average	Complex	
1	Number of user input	10	3	4	6	30
2	Number of user output	8	4	5	7	24
3	Number of queries	15	3	4	6	45

4	Number of relational tables	22	7	10	15	154
5	Number of external interfaces	8	5	7	10	40

GFP = 293

All PC would be a 3 except 1, 4, 14, 10, 3 which would be 5.

PCA = 0.65 + 0.01 * 52 = 1.17

FP = 293 * 1.17 = 342.81
Productivity = 30
Estimated Effort = 342.81/30 = 11.406 -> 12
Duration = 12/6 = 2 weeks

3.3
The project plans to use AWS to provide the services for distributed systems and backend functionalities. Services such as EC2 instances, lambda functions, and cloud watch events would be estimated to be around $200-$400 a month. Additional cost of publishing the app to users will add on an additional $150.

3.4
Data analytics software from 3rd party companies will be used. Companies such as Confluent will provide a good resource on data tracking throughout the app.

3.5
This will usually be a team of 6-7 people developing the project. The cost of the team would be salary based of around $25/hr with around a 35-hour workweek. In regard to training, there will not be any training cost after installation as this is aimed to be a simple public use application that would resemble most social media people are used to.

4 - Test Plan

One unit for our application would be retrieving the names of people who are within a certain radius of you. Given the location of a person and his/her friend (on the app), we calculate the euclidean distance between the two and return true if the friend is closer than a radius R (which we pass as a parameter). The Junit zip file is attached, and classes for Car and Person were created.

Comparison with Similiar Designs

PlugShare

PlugShare, the leading app for EV drivers, is a community-based tool that directs users to public charging stations worldwide. PlugShare is a free EV driver app for iOS, Android, and the web that allows users to locate charging stations, leave reviews, and connect with other owners of plug-in vehicles [2]. In comparison to EV Comms it has some differences in functionalities including the capabilities of securely paying for charging services via the app at participating charging stations and the ability to map out their routes for a trip to stay reliably charged throughout the drive [2]. However, with the EV Comms application, we have similarities in terms of identifying and providing charging stations based on users' location preferences. In terms of the User Interface with plug share, it provides a map providing all of the available charging stations in the designated area. With each location the user has the option of providing the directions, adding a photo, leaving a review, and bookmarking the location. The interface is similar to Google Maps in terms of structure and functionality. Since our application has no implementation, we don't have much physical comparison, however, the plan was to design an application that allows similar functionalities as PlugShare but also allows for communications between electric vehicle owners in terms of damage reports. It combines information for charging stations as well as a communication app for all-electric vehicle owners.

ChargeHub

ChargeHub, an application for EV drivers free on iOS & Android, is a charging-based app that points users to EV stations across North America [1]. ChargeHub is free on app stores. The application allows users to plan trips according to EV chargers, communicate with other members, coordinate charging times, & review EV charging stations [2]. ChargeHub is very similar to our application. Our application, EV Comms, seeks to provide charging stations on a map interface akin to Google Maps & allow communication to users. ChargeHub further allows users to map road trips with EV chargers in mind and let users pin point EV charging stations while also leaving reviews including photos [2]. One similarity between the apps is communication between other members. The difference is the type of communication. ChargeHub lets users communicate with other EV drivers to ask for help, coordinate charging, & request another EV driver to unplug their charger to utilize it [1]. Our app freely allows our members to communicate between members similar to a messaging type app, the goal in mind being to inform other drivers if they see anything wrong with their vehicle such as a flat tire. ChargeHub also has an additional functionality that EV Comms does not have in that it notifies users when a new charging station opens nearby [2]. Altogether, our application is very similar to ChargeHub in terms of functionality and purpose, with the caveat of ChargeHub having extra functionalities to facilitate relevant communication between members.

Conclusion

The era for electric vehicles is inevitable. Many tech companies are working toward a more sustainable form of transportation and EV's have exponentially taken over the industry. We want to provide a seemless transition from petrol to electric cars by providing information that would be convenient for EV users. As of today, EV's are still the first of its kind and are far from perfection. We hope that in the future our app would become more refined and include a larger database for users to access. This would include quicker load times, less server crashes, and more charging locations. Some changes that could be implemented would be the duration of how long our app would be completed. The FP algorithm that was used calculated that it would take 2 weeks to finalize the app, but we believe that it would take much longer. Otherwise, there were minimal changes that needed to be made in our development phase.

References

[1] ChargeHub, "ChargeHub App Features," *ChargeHub*, 2012. [Online]. Available: https://www.crunchbase.com/organization/chargehub. [Accessed Nov. 9, 2022].

[2] Rhythm Energy, "The 7 Best EV Charging Apps," *gotrhythm.com*, Dec. 6, 2021. [Online]. Available: https://www.gotrhythm.com/blog/electric-vehicles/the-7-best-ev-charging-apps. [Accessed Nov. 10, 2022].