ÇANKAYA UNIVERSITY

FACULTY OF ENGINEERING

COMPUTER ENGINEERING DEPARTMENT

Project Report

Version 1

CENG 407

Innovative System Design and Development I

202412

ChargeMind: AI Based E-charge-Management and Planning Application

Selin Deniz 202011051

Meleksu Özdoğan 202011054

Lorin Melek Vural 202011035

Hanife Müge Karakaya 202011044

Kaan Baydemir 202011070

Advisor: Dr. Asst. Prof. Serdar Arslan

Table of Contents

Abstract

The rapid increase in demand for electric vehicles (EVs) has brought problems such as range anxiety, inefficient route planning, lack of real-time charging station data, and approaches that neglect environmental sustainability. Existing solutions often fall short of comprehensively addressing these issues.

To overcome these problems, ChargeMind was developed as an AI-powered e-charging management and planning application. The system offers users machine learning algorithms, real-time data integration, and user-centered design to deliver personalized, efficient, and sustainable charging experiences. ChargeMind creates user-specific route plans with cost, distance, carbon emission tracking, environmental friendliness, and real-time charging station availability updates. It also provides high performance and security with cloud-based scalability, encrypted communication, and multi-factor authentication.

The project provides more user-friendly results compared to existing solutions, with a scalable, sustainable system architecture that can support better route optimization, better user experience, and increased user demands with real-time responsiveness and a more environmentally friendly solution through advanced machine learning models.

Through this work, we have deepened our awareness of the importance of integrating sustainable practices into technology solutions while balancing user needs and technical efficiency.

Keywords: Electric Vehicle, E-Charge, Artificial Intelligence, Big Data, Sustainability, Route Optimization, Carbon Emission

Özet

Elektrikli araçlara (EV) olan talebin hızla artması, menzil kaygısı, verimsiz rota planlaması, gerçek zamanlı şarj istasyonu verilerinin eksikliği ve çevresel sürdürülebilirliği göz ardı eden yaklaşımlar gibi sorunları beraberinde getirdi. Mevcut çözümler genellikle bu sorunları kapsamlı bir şekilde ele almakta yetersiz kalıyor.

Bu sorunların üstesinden gelmek için ChargeMind, yapay zekâ destekli bir e-şarj yönetimi ve planlama uygulaması olarak geliştirildi. Sistem, kullanıcılara kişiselleştirilmiş, verimli ve sürdürülebilir şarj deneyimleri sunmak için makine öğrenimi algoritmaları, gerçek zamanlı veri entegrasyonu ve kullanıcı merkezli tasarım sunuyor. ChargeMind, maliyet, mesafe, karbon emisyonu takibi, çevre dostu olma ve gerçek zamanlı şarj istasyonu kullanılabilirliği güncellemeleri ile kullanıcıya özel rota planları oluşturur. Ayrıca bulut tabanlı ölçeklenebilirlik, şifreli iletişim ve çok faktörlü kimlik doğrulama ile yüksek performans ve güvenlik sağlar.

Proje, gelişmiş makine öğrenimi modelleri aracılığıyla daha iyi rota optimizasyonu, daha iyi kullanıcı deneyimi ve daha doğa dostu bir çözüm için gerçek zamanlı yanıt verme ve artan kullanıcı taleplerini destekleyebilecek ölçeklenebilir, sürdürülebilir bir sistem mimarisi ile mevcut çözümlere kıyasla daha kullanıcı dostu sonuçlar sağlar.

Bu çalışma sayesinde, kullanıcı ihtiyaçları ve teknik verimliliği dengelerken sürdürülebilir uygulamaları teknoloji çözümlerine entegre etmenin önemi konusundaki farkındalığımızı derinleştirdik.

Anahtar Kelimeler: Elektrikli Araç, E-Şarj, Yapay Zekâ, Büyük Veri, Sürdürülebilirlik, Rota Optimizasyonu, Karbon Emisyonu

1. Introduction

1.1 Company Information

To transform the electric vehicle (EV) charging ecosystem, a group of candidate computer engineers launched the ground-breaking ChargeMind project. Combining advanced knowledge in data science, software engineering, and artificial intelligence, the project was started at Çankaya University as a senior undergraduate capstone project. In order to provide a smooth, effective, and eco-friendly EV charging solution, ChargeMind integrates state-of-the-art technologies to address urgent issues in sustainable transportation.

1.2 Problem Statement

EVs are becoming a more sustainable option for transportation than conventional combustion-engine vehicles on a worldwide scale. However, serious weaknesses in the current charging infrastructure have been made clear by the quick uptake of EVs. Among these limitations include ineffective charging station use, lengthy charging durations, insufficient real-time monitoring, and a lack of grid stability support. Further impeding efforts to quantify and reduce environmental effects is the lack of advanced carbon emission tracking systems for e-charging stations. Achieving a sustainable and scalable EV ecosystem requires addressing these issues, as recent research on smart grid and green energy systems has shown.

1.3 Background

In recent years, there has been a lot of interest in the necessity of smart charging solutions. While Ionity's charging stations and Tesla's Supercharger network provide proprietary fast-charging services, they do not integrate artificial intelligence to improve charging schedules and grid interaction. Brambilla et al.'s research on "The Smart Grid Semantic Platform" from 2021 highlights how crucial it is to combine energy and big data technologies to improve grid efficiency. Furthermore, research on carbon emission tracking, as done by Liu et al. (2020), highlights how AI-driven methods can be used to measure and lessen the environmental impact of EV charging. Even with these developments, there is still much space for innovation because the existing solutions are still disjointed.

1.4 Solution Statement

ChargeMind offers an intelligent, self-sufficient charging system that makes use of big data analytics, artificial intelligence, and Internet of Things (IoT) technology. Demand-response management, predictive maintenance, and real-time scheduling are all part of the solution to maximize grid interaction and charging efficiency. Additionally, the platform incorporates a carbon emission tracking module that monitors and reduces environmental effects using sensor data and neural network models. A scalable, dependable, and user-focused solution catered to the requirements of EV owners, charging station operators, and grid management is guaranteed by ChargeMind's multi-layered approach.

1.5 Contribution

ChargeMind sets itself apart by fusing several cutting-edge technologies into a single, integrated solution. In contrast to current solutions, it expands on existing capabilities by offering:

1. AI-driven Optimization: Machine learning methods are used to improve grid stability and dynamic scheduling.

2. Comprehensive Carbon Tracking: A special function that makes it possible to track carbon emissions at charging stations in real-time.

3. Scalable architecture is the design of a distributed system that guarantees high availability and dependability in a variety of geographical locations.

ChargeMind makes a substantial contribution to the advancement of smart grid and sustainable transportation technologies by resolving the shortcomings of existing charging methods and implementing cutting-edge features. The initiative is a vital step toward a more environmentally friendly and effective future for EV charging infrastructure.

2. Literature Review

2.1 Introduction

The global shift is towards promoting sustainable energy solutions and reducing carbon emissions, reinforcing the usage of electric vehicles (EVs) due to the limited natural resources. As the adoption of EVs is increasing, user-friendly and efficient charging solutions become paramount since traditional fueling infrastructures can not directly fit in with EV requirements, which creates new challenges in the context of technical issues and user experience.

This study examines the integration of cutting-edge AI-based solutions, such as real-time station availability monitoring, carbon emission measurement, and tailored route optimization, to address typical issues related to EV charging. Furthermore, to provide a seamless charging experience, this study aims to provide a personalized experience by optimizing time, cost, and environmental friendliness based on the user's priorities.

Technical developments and various research in the EV ecosystem are inspected to set the groundwork for this project. That process started with a market investigation to see current infrastructures, the underlying technical and architectural basis used, and the features provided. This investigation highlights crucial points specific to EV charging, such as crowded or invaded stations, incorrect route directions, and non-available but available stations.

In light of these processes and findings, solutions produced from that study aim to close gaps in the market by utilizing artificial intelligence to provide cost-effective, eco-friendly, and personalized solutions to each individual's needs, contributing to a more user-friendly and environmentally friendly EV charging experience. As the team carries out this study, we aim to give a thorough overview of the breakthroughs and solutions that drive ChargeMind’s development.

2.2 Market Analysis

2.2.1 Plugshare

PlugShare is a free EV driver app for mobile (Android and iOS) and web that allows users to find charging stations, leave reviews, and connect with other plug-in car owners. As of August 2023, it is available in 29 languages [1]. It has a worldwide public charging map with stations from every major network in North America, Europe, and most of the world.

PlugShare uses the user’s location or provided address to list charging stations for electric cars and show their locations on the map. The app also has certain filtering features. The user can filter by the charging station they want to go to, the station’s rating, services near the station (food, shopping, parking, accommodation, grocery stores, etc.), the number of electric vehicle charging stations in a given area, stations currently in use but coming soon, charging stations provided by companies or brands that offer a specific charging service (e.g., Supercharger, Blink, Fastned), and the type of socket. PlugShare also allows users to add stations to share their home chargers with other users. Users can register to provide information about their electric vehicles at theapp's startp. Registered users can also submit new stations, update information on existing stations, and add notes and photos for other processes. Electric vehicle charging stations are usually displayed in different colors on the map according to their access status and charging power levels. This color coding makes it easier for users to quickly find suitable stations.

The “Pay with PlugShare” feature in PlugShare allows users to pay for certain charging stations directly from the app. This feature does not require a membership, balance, or recurring payment plan and allows you to pay only for what you use. At supported stations, the “Pay with PlugShare” option is displayed in the station details. By clicking on this option, users can view charging fees, complete payment with a credit card or other payment method, and start charging sessions. When the charging process is completed, the payment summary and receipt are provided through the application.[2]

Although PlugShare offers good options for filtering users, the travel planner feature is visible in the application, but this feature does not meet the expectations of the users. The user cannot see the chargers along the route they will travel but only sees a road map. At the same time, the user interface is criticized for being complicated and difficult to understand. This may cause problems for users or those who do not have technical knowledge.

2.2.2 Lixhium

Lixhium is a national charging station application that combines the electric vehicle charging ecosystem on a single platform, brings together licensed charging station operators and electric vehicle users, and displays all stations on the same map. It is a 'neutral' marketplace with various features like smart route planning, commenting, calculating the appropriate charging amount and fee for the vehicle, illustrating the availability of stations, paying using a virtual card with wallet integration, and getting points [3].

Even though we cannot publicly access their technological infrastructure, we can say that they use big data analytics and may include machine learning algorithms [4].

However, through the feedback evaluations received from users; It is also inadequate due to problems such as slow updates, delays in adding newly opened stations, misleading station suggestions, screen freezes during route creation, and inability to produce alternative routes [5].

2.2.3 Open Charge Map

Open Charge Map is a free and open-source application launched in 2011 that helps electric vehicle (EV) owners find the best charging stations. It allows users to easily find charging stations near them or anywhere they want on a map. It also provides users access to station details (working hours, charging connection types, pricing information, etc.) and also has a rating system that is supported by user reviews. The main aim of this application is to make that process easier for electric vehicle owners when they plan their travel routes and provide the most suitable charging options.

This program offers filtering options that help users determine the speed and type of connection they want. Users can get a more customized experience with features such as visualization on the map, live occupancy status, directions, and user reviews. It also allowed users to update station information and add new charging points constantly.[6]

In addition to having many useful features, OpenChargeMap has some drawbacks. Because the platform is based on data that is observed through user support, some station information may be incorrect or out of date. These directions given incorrectly can be dangerous enough to affect users' roadmaps for travel, especially due to information on fullness status or speed of charging. Users report that the fullness information of stations, especially those that are full during peak hours, is not updated in a timely manner, which can be a significant problem for EV drivers traveling long distances. However, issues such as lack of information or lack of data about charging stations in certain areas shown on the map reduce the reliability of the application.

2.2.4 Voltla

Voltla is an application for electric vehicle owners to find all charging stations, learn vehicle charging prices from a single point, create routes to charging stations and start charging & payment. It is a platform that supports sustainable transportation and contributes to Turkey's EV infrastructure. It can be used on iOS and Android devices.

Volta offers charging stations in the vicinity to users based on location information. As in other EV applications, Voltla also has certain filtering options. This filtering can be done according to the charging type, AC (Alternating Current) or DC (Direct Current), socket power, socket kWh price range, and brand (Turkey and non-Turkey brands). At the same time, the user is also offered the opportunity to save their filters in order to prevent the user from re-entering the filtering options they frequently use. Electric vehicle charging stations are shown in different colors on the map according to the charging type. The user can send a request to the company and save their car and address to the system. The most striking feature of the application is that it provides information about the unit prices of electric vehicle charging companies according to the socket type and the amount of energy charged (Kwh).

Voltla offers its users three different methods to start the charging process. The first of these is to select the socket you want to start charging from the menu that opens when you click on the pins where you can start charging and receive payment via Voltla with an icon symbol on the map. Another method is to start the charging process by scanning the QR code on the screen of the devices at the stations where the charging process can be started or by scanning it with the help of the camera. The third method is to start charging by clicking on the start charging button with the socket code on the QR code reading screen. Before starting the charging process, Volta checks whether the socket is placed correctly, its distance to the charging station, whether the user has any previous debts, and starts the charging process if everything is appropriate. The payment is automatically withdrawn from the selected credit card with the card information entered by the user.

One of the negative aspects of the Volta application is that it only allows Turkish phone numbers to be registered during registration. This situation restricts access for users outside of Türkiye. With the widespread use of electric vehicles, accessibility to international users is an important criterion for a charging infrastructure application.[7]

2.2.5 ChargePrice

ChargePrice is one of the largest platforms for independent price comparison for electric vehicle (EV) charging. Apart from this feature, it has various data sets, such as the real-time status of charging stations, availability, charging rates, and vehicle features. It helps operators optimize their tariffs and improve their charging infrastructure. It also allows users to select their favorite charging stations and share them with other users. ChargePrice ensures the quality of this data by constantly checking it with cross-queries[8]. ChargePrice also provides the opportunity to create and manage a portfolio of personalized recharge cards, such as "My Best Charging Card," to users.

Additionally, the capacity to deliver real-time data and continuous communication between users and charging stations is improved by combining API and OCPI (Open et al.) connections with various operators. As a result, consumers may access the most recent data regarding charging station performance, availability, and cost straight from the platform. To develop its platform and fix any operational problems, user feedback is also a key component of its service improvement process through reviews and ratings. Frequent quality checks are carried out based on user experiences and invoice accuracy to guarantee further the platform's reliability and correctness of the information. Price alerts are another feature of ChargePrice that helps consumers minimize and avoid unforeseen expenses by providing proactive and automatic notifications if charging rates change. Lastly, ChargePrice encourages direct engagement with operators, enabling consumers to easily submit input to enhance the charging infrastructure, ask for help, or address issues.

Despite the advantages, realince of the real time data one of the struggles due to the possible challenges such as delayed or inaccurate data caused by connectivity problems or discrepancies between operators and charging stations.

2.3 EVRP

EVRP, or Electric Vehicle Routing Problem, is a variation of Vehicle Routing Problem that studies the concept of finding an optimal set of routes for an electric vehicle such that it conforms to various constraints, including but not limited to the battery percentage of the electric vehicle and traffic density. While the adoption of electric vehicles has seen a significant increase over the past decade, electric vehicles still need recharging and consequently necessitate elaborate route planning. Due to this situation, studies about EVRP have started to emerge at an exponential rate in recent years [9].

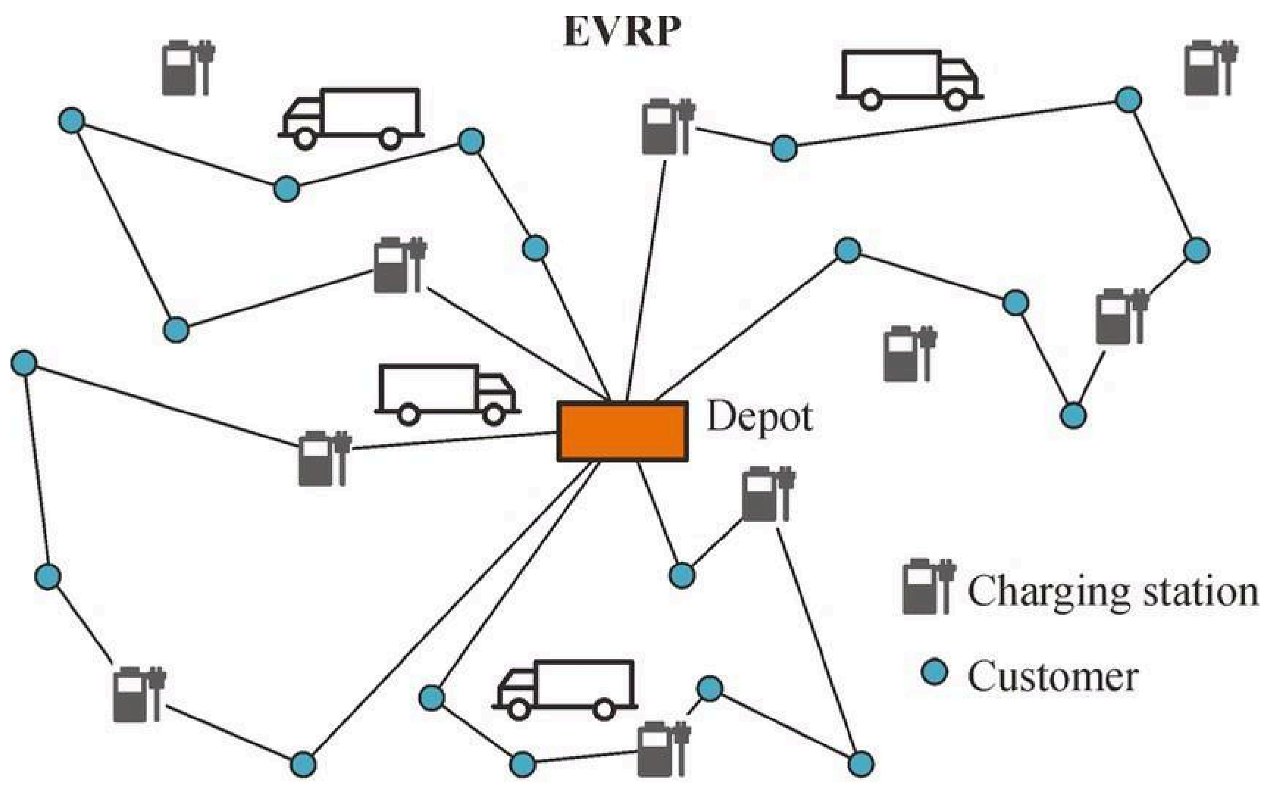


Figure 1 - EVRP Graph

[Source: https://www.researchgate.net/figure/An-example-solution-for-EVRP_fig2_351700506]

While EVRP focuses on fleets of vehicles, it can also be used for end-user electric vehicles. In our project, we aim to facilitate electric vehicle usage via not only using machine learning, but also EVRP methodologies to ease the electric vehicle driver experience, eliminating long-range anxiety and the adoption of the heavy load vehicles.

2.4 Machine Learning

With numerous parameters, comes the great challenge to predict what comes next in real-world scenarios. This includes foreseeing whether a charge station will be available when the driver arrives at the location. To avoid such situations, we aim to leverage the power of machine learning to find the paths that will ultimately optimize the paths that will be taken by electric car drivers.

For the possible case, features that will be selected will include both discrete and continuous data, such as hours being numerical and weather being either rainy or sunny. Handling both values of both types at the same time requires a sophisticated machine learning model, which in turn has driven us to compare existing machine learning models.

2.4.1 Linear Regression

It is the statistical methodology that models the relationship between a dependent and an independent variable. It makes the assumption that the variables have a linear correspondence and places a line through the available data. It is easily understandable and easy to implement, computationally efficient in small datasets, and each feature coefficient is easily interpretable; however, it is inefficient for complex data and prone to outliers. Foreseeing the availability of charging stations necessitates a non-linear understanding of features. Linear regression cannot address this situation, being a not accurate choice. On the other hand, using this model effectively still benefits our project.

2.4.2 Decision Trees

A decision tree splits at the decision points repeatedly based on feature values. Accuracy improves at each decision split. It is suited for non-linear relationships, easily representable visually, and allows both continuous and discrete data to be used, but it can result in completely different trees even when a small change occurs in data, it may yield poor performance on complex datasets, and they have a tendency to overfit which in turn makes them perform poorly in scenarios where there is unseen data, making it harder to predict charging station availability accurately.

2.4.3 Random Forest

The Random Forest method combines multiple decision trees. A random subset is used to train each decision tree for random features and random data. Ultimately, predictions are averaged to reduce overfitting and increase accuracy. This method has some pros, such as that it can handle both interactions and non-linear relationships, and it is less prone to overfitting compared to single decision trees. However, they can be computationally expensive for large datasets. Due to the ensemble nature, they can become less interpretable. They may also face challenges in making fine-grained predictions in highly dynamic environments. While Random Forests can handle tabular data considerably well, they art fine-tuneless compared to Gradient Boosting Machines(GBM). It may also require a significant amount of computational resources even without being much more efficient than GBM. This model can still be preferred for our project.

2.4.4 Gradient Boosting Machines

The gradient Boosting Machines (GBM) method gathers simple models together to form a better, more powerful single model. It can handle non-linear and linear relationships, and it is exceptionally superior on structured tabular data; on the other hand, it can be computationally expensive for very large datasets, necessitates careful setup of hyperplanes, and is less interpretable than Linear Regression or single Decision Trees. While GBM offers high performance on tabular data, there may be training and scaling problems because of high computational needs. Still, creating a preferable choice for the project.

2.5 Real-Time Operations

Obtaining instant data using Google Maps has a great advantage for real-time operations. Google Maps anonymously collects the location and speed information of millions of users worldwide and instantly analyzes this data to provide up-to-date information about traffic conditions. This information obtained is of great importance for the logistics sector, etc., which is sensitive in operational terms. For example, if someone who wants to travel using electric vehicles uses traffic data to plan a route, delivery times are optimized, and fuel costs are reduced. At the same time, alternative routes are suggested in the event of an unexpected traffic jam, thus minimizing time loss.

Thanks to this integration in applications optimized for electric vehicles, it is important to plan charging stops and breaks efficiently, especially on long-distance journeys. An EV application equipped with instant traffic data suggests suitable charging stations that the driver can reach in the shortest time, helping him avoid routes with traffic congestion. In addition, it can estimate possible waiting times at charging stations by taking into account traffic data during rush hour and adjusting the driver's route accordingly.

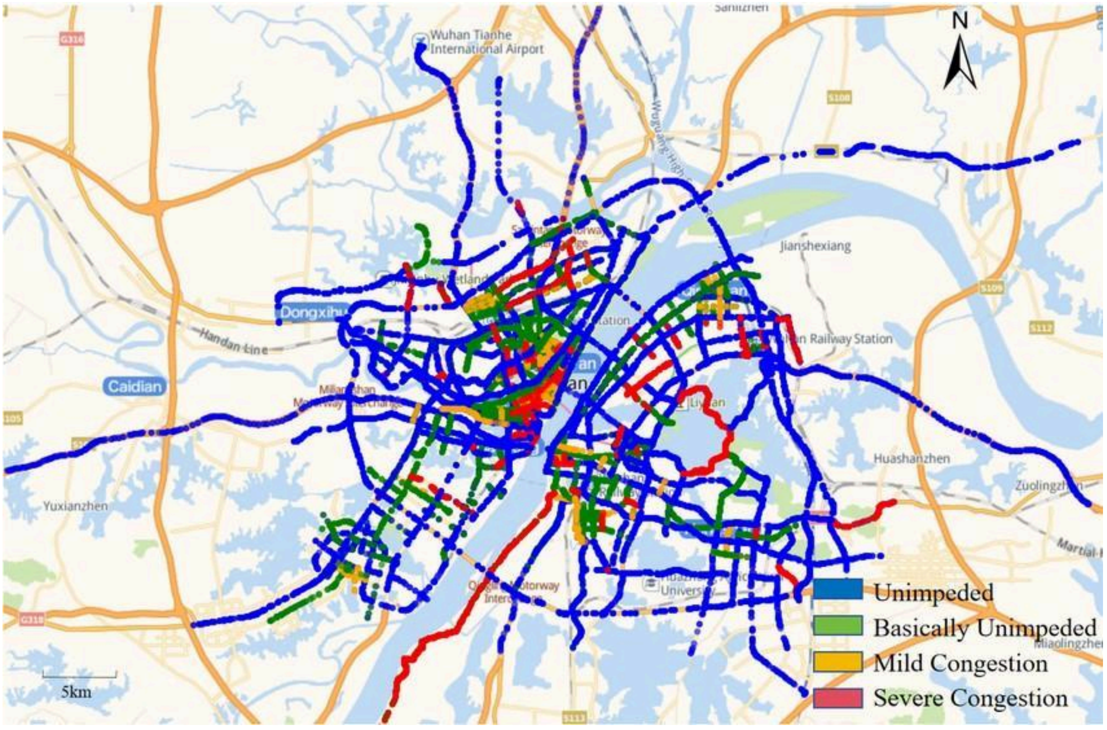


Figure 2 - Traffic Density Map

[Source: https://www.researchgate.net/figure/Real-time-traffic-data-collection-area_fig1_382702699]

The fullness data that comes from charging stations allows users to see in real-time whether the charging points at the stations are full or empty. This allows EV drivers to see if there is a suitable charging point before arriving at a station and evaluate alternative stations if necessary. Using full information provides a great advantage for users, especially during busy times or in areas with limited stations. This data from charging stations plays a major role in estimating how busy a station is, average waiting times, and which charging connections are available. This data also helps to minimize both waiting times and unnecessary routes.

2.6 Carbon Emission

The transportation sector is one of the largest sources of greenhouse gas emissions in the European Union, and despite efforts such as increasing the deployment of electric vehicles (EVs), emissions have shown little reduction since 2005. Estimates for 2023 show only a slight drop of 0.8% compared to 2022, with projections indicating that domestic transport emissions will not fall below their 1990 levels until 2032 [10]. By transitioning to electric vehicles (EVs), carbon emissions can be reduced globally, with EVs offering a cleaner alternative to traditional combustion engines.

Charged EVs, however, have a larger carbon footprint due to the electricity used to charge them. In order to fully understand EV adoption's environmental impact, it is crucial to monitor carbon emissions from charging stations. It is traditional to use sensors to detect air quality and greenhouse gas emissions, but it is also possible to estimate these emissions without using sensors. An example is the use of energy consumption data, grid emissions factors, and other indirect methods.

The design of an energy big data and carbon emission monitoring system can benefit from machine learning models like the perceptron model, especially when targeting carbon neutrality and carbon peaking. By integrating real-time data from energy consumption and emissions sources, these systems can track carbon footprints effectively. For example, the perceptron model can classify and predict emissions based on inputs such as energy consumption and regional grid factors, providing more accurate emissions data for real-time decision-making [11].To fully understand EV adoption's environmental impact, it is crucial to monitor carbon emissions from charging stations. While it is traditional to use sensors to detect air quality and greenhouse gas emissions, it is also possible to estimate these emissions without using sensors. An example is the use of energy consumption data, grid emissions factors, and other indirect methods.

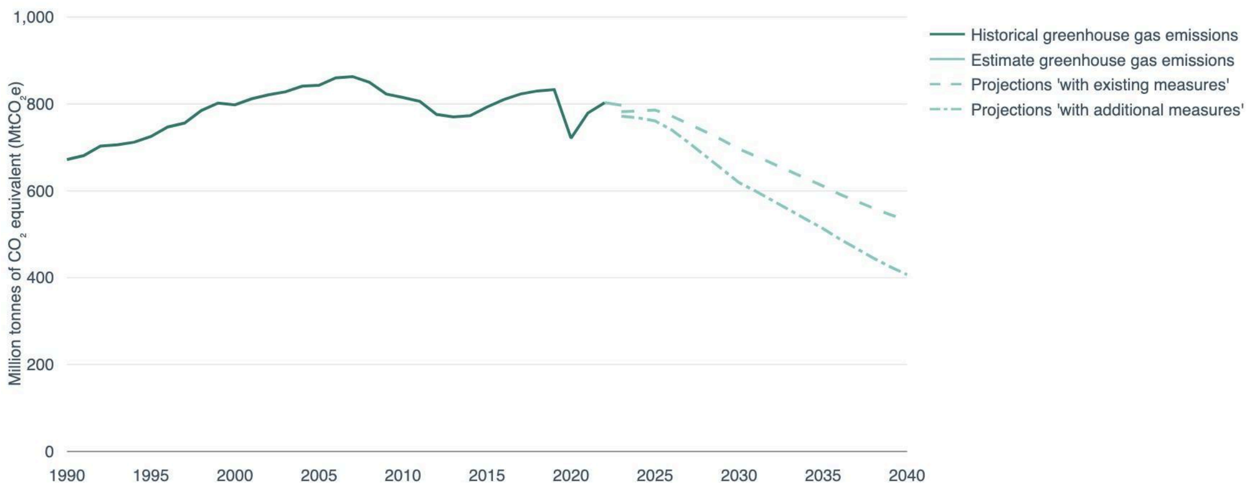


Figure 3 - Greenhouse gas emissions from transport in Europe

[Source: European Environment Agency,2023,https://www.eea.europa.eu/en/analysis/indicators/greenhouse-gas-emissions-from-tr ansport].

The energy consumption per charging session is a standard method of estimating carbon emissions at EV charging stations. It is possible to measure the carbon dioxide emissions related to electricity consumption during the charging process by monitoring the energy used (in kWh) and applying a region-specific emissions factor (kg CO_2 per kWh). As an example, in the case of an EV that consumes 10 kWh and the local grid's emissions factor is 0.5 kg CO_2/kWh, then the total carbon emissions would be 5 kg of CO_2. Besides, an alternative approach is to utilize real-time carbon intensity data provided by public APIs from regional or national grid operators. Emitted CO_2 per kWh of generated electricity can be provided by these APIs at any given moment. As a result, it is possible to estimate carbon emissions without the need for physical sensors by linking the duration of each charging session with the grid's carbon intensity.

Furthermore, carbon footprint can be provided from third-party services's APIs, which estimate the based on energy consumption, vehicle type, and regional factors. By the usage of these services, the accuracy of emission stigmatizations may be improved, and sensor-based monitoring systems may be placed with alternative solutions.

Additionally, installation and maintenance of the physical sensors can be cumbersome and quickly scaled to multiple charging stations. Therefore, the primary benefits of these alternative methods are cost-effectiveness, scalability, and simplicity. However, estimation accuracy depends on the quality and granularity of the emissions factors used, whereas the variability of the electricity grid's carbon intensity must also be taken into account. Besides, certain local factors, such as traffic or weather, may indirectly influence carbon emissions and are more challenging to account for without physical sensors.

In the implementation phase of these alternative solutions, which include a carbon emission monitoring feature, energy consumption-related data from charging sessions should be gathered. Then, this data must be combined with real-time grid carbon intensity data from public APIs so that emissions calculation can be based on the consumed energy and the carbon intensity at that time. The formula for emissions calculation would be:

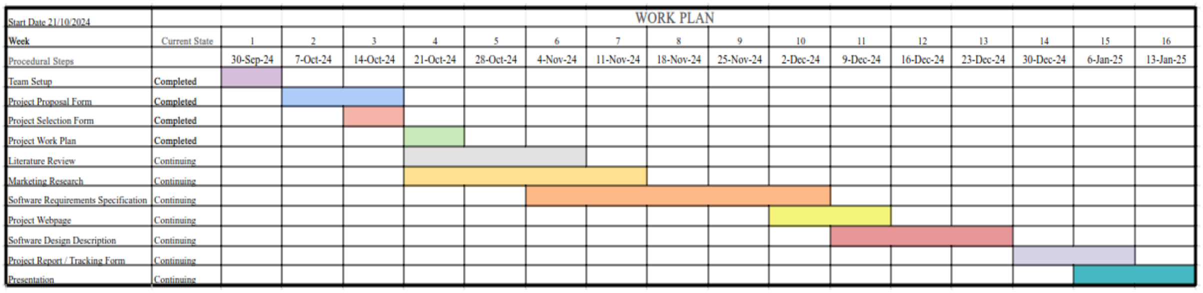
$$\text{Carbon Emission} = \text{Energy Consumed (kWh)} \times \text{Grid Carbon Intensity (kg } CO_2\text{/kWh)}$$

In conclusion, users will have the privilege of monitoring the carbon footprint of each charging session in real-time via this method. Moreover, daily, weekly, or monthly emissions trends can be displayed on the application, which would offer insights and help users make more environmentally friendly decisions.

2.7 Conclusion

This literature review gives an overview to the modern day problems of electric vehicles, current status of the market, and possible solutions to address problems with various methods ranging from EVRP and machine learning models. We aim to provide customized experience to electric vehicle drivers via utilizing aforementioned machine learning models where it is possible to apply.

3. Work Plan



Start Date 21/10/2024		WORK PLAN															
Week	Current State	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Procedural Steps		30-Sep-24	7-Oct-24	14-Oct-24	21-Oct-24	28-Oct-24	4-Nov-24	11-Nov-24	18-Nov-24	25-Nov-24	2-Dec-24	9-Dec-24	16-Dec-24	23-Dec-24	30-Dec-24	6-Jan-25	13-Jan-25
Team Setup	**Completed**	🟪															
Project Proposal Form	**Completed**		🟦	🟦													
Project Selection Form	**Completed**			🟥													
Project Work Plan	**Completed**				🟩												
Literature Review	Continuing				▨	▨	▨										
Marketing Research	Continuing				🟨	🟨	🟨	🟨									
Software Requirements Specification	Continuing						🟧	🟧	🟧	🟧	🟧						
Project Webpage	Continuing										🟨	🟨					
Software Design Description	Continuing											🟥	🟥	🟥			
Project Report / Tracking Form	Continuing														🟪	🟪	
Presentation	Continuing															🟦	🟦

4. Software Requirements Specification

4.1 Introduction

4.1.1 Purpose

This document provides an extensive description of the ChargeMind Mobile Application System. Its purpose is to define the system's objectives, essential functionalities, user interactions, scope of action, constraints, and responses to various inputs and situations. It also aims to reduce ambiguity, harmonize the project group and ensure consistency throughout the system lifecycle. Moreover, it serves as a clear and detailed guide for the system's development process. The intended audience includes stakeholders, project members, and developers, ensuring a common understanding of the application's requirements and design goals.

4.1.2 Scope of Project

Electric Vehicle [12] utilization has increased significantly over the last decade. This increased usage made the masses witness one specific problem electric vehicles brought to our lives: Vehicle batteries go down on longer trips. The ChargeMind project aims to provide personalized and optimized routes to electric vehicle drivers and electric vehicle fleet owners to overcome these longer trip concerns by finding suitable paths that satisfy the needs of its users.

Since there are many various parameters to consider for each electric vehicle, a sophisticated solution is needed for each scenario. The ChargeMind application shall fill this deficiency by leveraging the power of Artificial Intelligence to adapt each use case while keeping user preferences in mind. Keeping the heavy workload in the cloud and reaching its users with mobile platforms, ChargeMind intends to provide higher availability to a wider user base and become a significant player among electric vehicle applications.

4.1.3 Glossary

SRS	A formal document that describes in detail the functional and technical requirements of a software project.
GPS	Anywhere in the world, a satellite-based global navigation system can provide location, speed, and time information.
IEEE	A global technical association that creates standards and supports experts in a variety of engineering disciplines, including computer science, electrical engineering,

	electronics, telecommunications, and energy systems.

SMS	A protocol for communication that enables handheld devices to send and receive brief text messages.
LTE	Depending on the GSM/EDGE and UMTS/HSPA standards, LTE is a wireless broadband communications standard for mobile devices along with data terminals in the telecommunications sector. It is marketed as 4G LTE.
RAM	The short-term memory requirements of electronic devices are satisfied by the RAM system.
Clock Speed	An indicator of a processor's speed at which instructions are carried out, usually given in hertz (Hz), which is the number of cycles per second. It establishes the speed at which a processor may complete tasks.
Non-Volatile Storage	Storage that keeps data even after the power is switched off is referred to as non-volatile storage. Flash memory, HDDs, and SSDs (Solid State Drives) are a few examples.[13]
REST API	An application programming interface that accesses and utilizes data via HTTP requests is known as a RESTful API. The GET, PUT, POST, and DELETE data types which stand for reading, updating, creating, and removing resource related operations can be utilized with that data.[14]
JSON	A stand-alone data transfer format called JSON was created to express basic data structures.[15]
HTTPS	The Hypertext Transfer Protocol (HTTP) has an extension called HTTPS. It is extensively used on the Internet and employs encryption to provide safe

	communication across a computer network. [16]
SSL/TLS	Two networking devices or applications communicate securely through protocols designed to protect data during transmission.
	The updated SSL version that addresses current SSL vulnerabilities is called Transport Layer Security (TLS).[17]
TCP/IP	The protocol suite of the internet, is a framework used for organizing the set of communication protocols.[18]
OAuth 2.0	One established protocol for authorization procedures is OAuth.
MFA	MFA is a multi-step account sign-in process that requires users to enter more information than just a password. [19]
SSO	SSO is an authentication solution that allows users to sign in to multiple applications and websites with one-time authentication.[20]
React Native	A JavaScript Framework used to create cross-platform applications.[21]
RSA	Asymmetric encryption algorithms like RSA are frequently seen in a wide range of goods and services. Data is encrypted and decrypted using asymmetric encryption.
AES	AES is a symmetric block cipher chosen by the US government to protect classified information.[22]
KMS	It is a system that makes it possible to create, store, distribute, use, and destroy encryption keys in a safe manner.
MACs	MACs are security tools that check a message's authenticity and integrity.

HMAC	An HMAC is a particular kind of message authentication code (MAC) used in cryptography that combines a secret cryptographic key with a cryptographic hash function.[23]
RBAC	According to their responsibilities, users' authorizations and system access are managed using this access control model.

4.1.4 Definitions, Acronyms, and Abbreviations

- **SRS:** Software Requirements Specification
- **GPS:** Global Positioning System
- **IEEE**: Institute of Electrical and Electronics Engineers.
- **SMS:** Short Message Service
- **LTE:** Long-Term Evolution
- **RAM:** Random Access Memory
- **REST API:** Representational State Transfer Application Programming Interface
- **JSON:** JavaScript Object Notation
- **HTTPS:** Hypertext Transfer Protocol Secure
- **SSL:** Secure Sockets Layer
- **TLS:** Transport Layer Security
- **OAuth:** Open Authorization
- **MFA:** Multi-Factor Authentication
- **SSO:** Single Sign-On
- **AES:** Advanced Encryption Standard
- **MACs:** Message Authentication Codes)
- **HMAC:** Hash-based Message Authentication Code
- **RBAC:** Role-Based Access Control
- **KMS:** Key Managment System

4.1.5 Document Conventions

The document adheres to standard naming conventions, terminology, and formatting guidelines specified by IEEE Software Requirement Specification document standards for clarity and consistency.

4.1.6 Overview of the Document

This Software Requirements Specification (SRS) document is a thorough guide to the development and functionality of ChargeMind: AI-Powered E-Charge Management and Planning Application. The paper is designed to provide clarity and help a variety of stakeholders, including developers, project managers, and future maintainers. This paper includes the following sections:

- **Introduction:** Provides background information on the project, such as its aim, scope, glossary of terms, and references, ensuring that all readers have a solid knowledge of its foundation.
- **Overall Description**: Outlines the product's perspective, development approach, user characteristics, and general restrictions. This section establishes the contextual and technological groundwork required for understanding the system's objectives and limitations.
- **Specific Requirements:** Describes the functional and non-functional requirements of the ChargeMind application, including expected behaviors, performance criteria, and constraints for developers.
- **Model and Diagrams:** Uses visual representations like as use-case diagrams, class diagrams, and activity flows to demonstrate the system's design and operational flow.

Each part is designed to meet the demands of specific audiences:

- General stakeholders, such as clients and project sponsors, will discover descriptions of the product's purpose and benefits.
- Developers will obtain deep knowledge of the system's technical specs and implementation requirements.
- Users and testers will benefit from parts that describe user interfaces and system interactions.

The document follows standard principles to ensure clarity, consistency, and usefulness across ChargeMind's development lifecycle.

4.2 Overall Description

4.2.1 Product Perspective

ChargeMind: AI-Based E-Charge Management and Planning Application is a smart solution for optimizing and planning electric vehicle [16] (EV) charging. The project is separated into two major modules: management mode and planning mode.

- The Management Mode focuses on the monitoring and control of e-charging stations. It offers capabilities such as energy usage tracking, carbon emission monitoring, and predictive maintenance to ensure smooth operation.

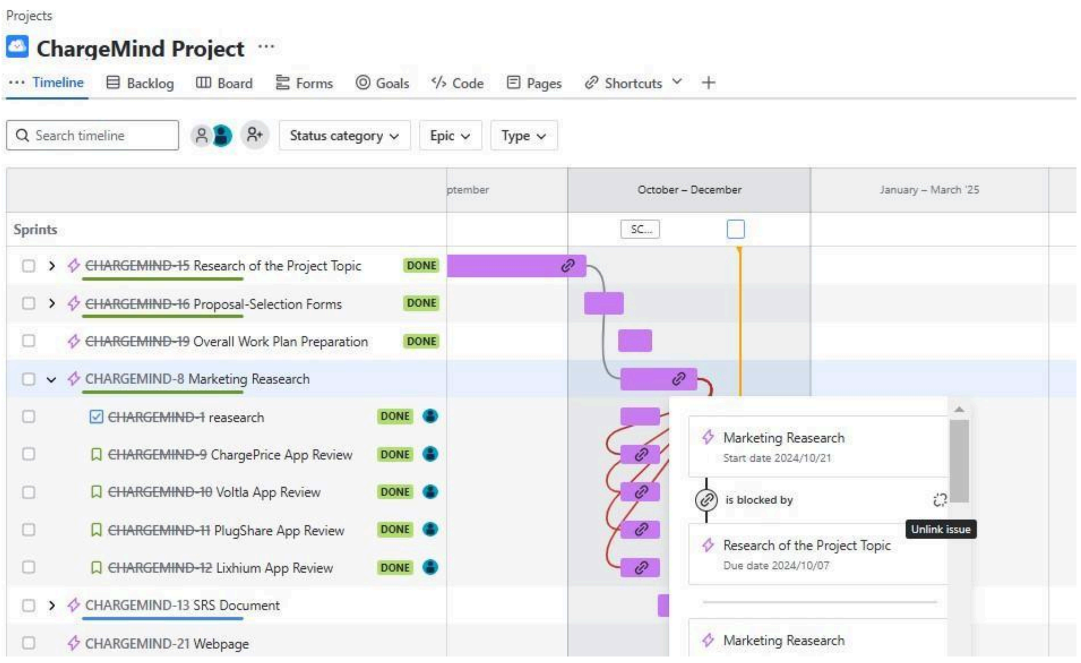
- Planning Mode offers advanced route optimization and scheduling capabilities to assist EV users in locating the most appropriate charging stations based on parameters such as distance, cost, and availability.

The program uses artificial intelligence and big data analytics to improve the performance of both modes, resulting in dependable and efficient charging solutions for both EV consumers and station managers.

4.2.1.1 Development Methodology

The platform's development methodology is based on the agile framework which is an iterative, collaborative, flexible and responsive development lifecycle approach that facilitates adaptation to evolving requirements and promotes a team environment. Agile methodology ensures ongoing developments, continuous feedback, and adjustments throughout each stage of the project.

For developing the project, Scrum is applied as the agile method, over the Jira [25]. Jira's task management features like Scrum boards, timeline table, creations of backlog, provide progress tracking, team organization, and collaboration throughout the development process. Over this tool, the project is split out into a series of continual sprints in sync with the timeline, with *Figure 1* representing a part of the timeline table designed for this project. Each sprint focuses on the different main work of the project, like real-time station availability, carbon footprint tracking, or design and architecture of the system (All will be explained in detail in the following sections of the document.). Each sprint concludes with the completion of a specific part of the project, which is then presented to the mentor and teacher for validation.

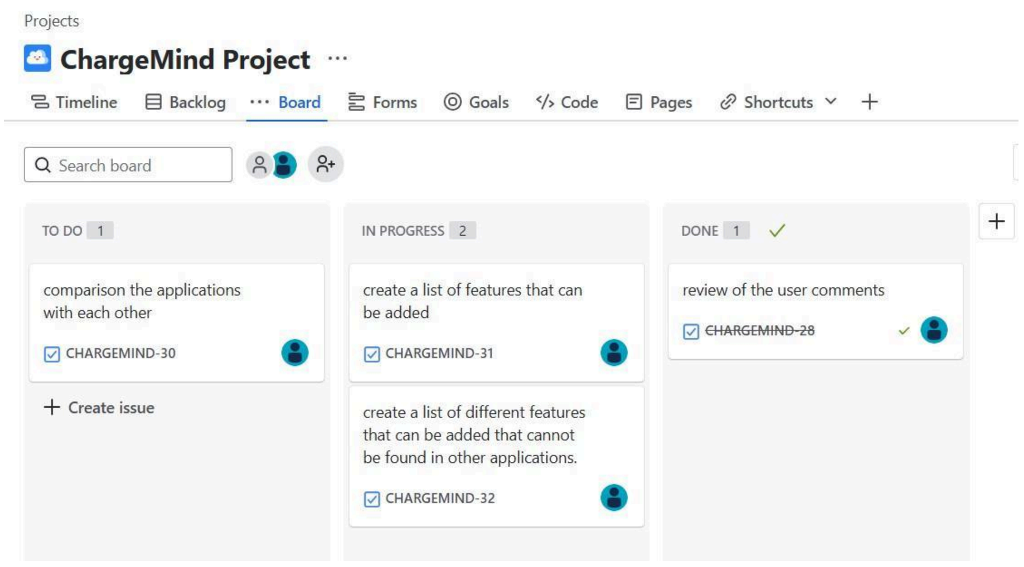


~~CHARGEMIND-15~~ Research of the Project Topic DONE
~~CHARGEMIND-16~~ Proposal-Selection Forms DONE
~~CHARGEMIND-19~~ Overall Work Plan Preparation DONE
CHARGEMIND-8 Marketing Reasearch
~~CHARGEMIND-1~~ reasearch DONE
~~CHARGEMIND-9~~ ChargePrice App Review DONE
~~CHARGEMIND-10~~ Voltla App Review DONE
~~CHARGEMIND-11~~ PlugShare App Review DONE
~~CHARGEMIND-12~~ Lixhium App Review DONE
CHARGEMIND-13 SRS Document
CHARGEMIND-21 Webpage

Marketing Reasearch
Start date 2024/10/21
is blocked by
Unlink issue
Research of the Project Topic
Due date 2024/10/07
Marketing Reasearch

Figure 1 A Part of the Project on the Timeline Table

Also, the Kanban-based task management system that guarantees smooth workflows using a drag-and-drop method, is used to provide visual charting of tasks. Task cards on the Kanban board allow each stage of the project to be followed step by step. Thus, the tasks to be done, in progress and completed tasks can be easily understood. This makes it simpler for team members to delegate, prioritize, and monitor tasks. *Figure 2* illustrates the Kanban board created with the backlogs of the market research, which is one of the sprints of the project. Backlogs contain all processes within a specific sprint. “To Do” phase includes which needs to be done with priority. "In Progress” phase contains tasks that are currently being constructed. “Done” phase represents processes that are being completed.


Projects

ChargeMind Project

TO DO 1

comparison the applications with each other
CHARGEMIND-30

IN PROGRESS 2

create a list of features that can be added
CHARGEMIND-31

create a list of different features that can be added that cannot be found in other applications.
CHARGEMIND-32

DONE 1 ✓

review of the user comments
~~CHARGEMIND-28~~ ✓

Figure 2 Marketing Research Sprint Backlogs on the Kanban board

4.2.2 User Characteristic

User Demographics

- **Age**: Participants over 18 years old.
- **Vehicle Usage**: Users who utilize electric vehicles (EVs) for traveling.
- **User Segments**:
  - **Individual EV Owners**:
    - Tech-savvy individuals seeking ease in route planning and charging station discovery. ○ **Corporate Users**:
    - Fleet managers focused on optimizing costs and scheduling efficient charging for multiple vehicles.
    - Maintenance staff of contracted firms

User Behavioral Traits

- Users may want to constantly check the charging status of their vehicles.
- Users gave importance to charging station occupancy information and ease of payment.
- Users that prefer fast and easily accessible charging points.
- Users motivated by environmental consciousness may prioritize green energy sources.

4.2.3 General Constraints and Assumptions

- A smart mobile device should be owned by the user.
- Familiarity with the basic mobile operating systems such as iOS or Android is required.

- Familiarity with the basic level of knowledge about mobile apps is required.
- The application is developed in English and Turkish, and it is assumed that all users can understand one of these languages.
- The application is designed to work all the time. The user will be able to log into his own account whenever he wants.
- Users must have a mobile operating system (Android, IOS etc.) and a version that is compatible with the application should be installed on their phones.
- The leveraged database system should not have any problems with data storage and the information can be synchronized at any time without any problems.
- Battery optimization algorithms must be adapted to diverse EV models.
- Users should have active internet connections for accessing real-time data.
- Up-to-date information should be provided by related APIs.
- Device resources (CPU, memory, battery) should not be excessively utilized by the application to maintain usability on mid-range smartphones.
- Data such as vehicle model, battery level, and destination should be provided by users for accurate recommendation generations.

4.3 Specific Requirements

The electric vehicle charging station mobile application is a comprehensive solution designed to enable users to easily find available charging stations and make payments seamlessly. This section outlines the specific requirements that drive the functionality and user experience of the application, as well as detailing the interface requirements and describing each of the functional and software system attributes (non-functional requirements) in a precise and testable manner that aligns with the project's goals.

4.3.1 External Interfaces

4.3.1.1 User Interface

Registration and Login Screens

Potential users of the ChargeMind application are greeted with a registration page which necessitates an e-mail address, password, and a valid phone number from the user. The user is obliged to follow the 2-factor authentication phase that demands e-mail and SMS verification by the user. After registration, the user is free to choose persistent session after a successful login or may login each time the user opens the application.

Home Screen

The Main Menu provides quick access to key app sections like Route Planning, Payment and Reservations, Reviews and Ratings, Settings, and Help, accessible through a navigation drawer or bottom navigation bar. Display the user's name, vehicle details, and battery level, with a prominent profile icon leading to the Profile Page, where users can manage their account, view recent charging activity, and update personal information is provided in the Profile section. Also, the homepage has a separate button to go to Search and Filter Page.

Filter Page

The Search Page allows users to filter based on charging speed, user comments, station type, station grade, and other criteria. Results are displayed in a list format, with the ability to sort by proximity or rating.

Route Planning Page

This page contains a map. Users can enter their destination, and the app calculates an optimal route, including charging stops. After entering destination choice and filtering options estimated travel time, charging stops, and total cost for charging are displayed on the screen and map. This page also has a button that directs users to the Charging Station Details Page for stations' details on the map.

Charging Station Details Page

This page provides detailed information about a selected charging station including availability (how many stations are free), pricing (per session or per kWh), charging speed, AC or DC, user reviews and ratings, station grades, etc.

User Profile Page

This page displays the user's personal information, settings, and preferences like vehicle details (model, battery capacity, type, etc.), history of charging sessions, stations visited, and payments made.

Reservation Page

This page provides the opportunity of booking a charging session at a compatible station with confirmation details (time, location, reserved slot) and an option to cancel.

Reviews and Ratings Page

This page displays users about stations' rates and provides feedback on their experiences. Users can give a star rating (from 1 to 5 stars) to reflect their overall satisfaction with the station. Users can also write thorough evaluations, offering their opinions on things like customer service, charging speed, and station use. The page will display all submitted reviews for each charging station, helping other users make informed decisions. Reviews can be sorted by relevance or most recent, and users can filter them based on star ratings.

Help and Support Page

The page provides users with directing users to ChargeMind web page for any issues or inquiries related to the app. It includes a frequently asked questions where users can find answers to common questions about the app's charging stations, payment methods etc. If users cannot find the information they need, they can directly contact customer support through email.

Settings Page

This page provides options to adjust app preferences like language selection (English or Turkish), notification settings, login and register settings, manage privacy settings and changing user details.

4.3.1.2 Hardware Interface

A typical piece of hardware equipment a user shall possess is a mobile smartphone. The smartphone lets the user interact with the application via touch screen display. This smartphone shall consist of a GPS component to track the user as the user relocates in the physical world. A stable internet connection must be provided to the smartphone via built in cellular chips, demanding approximately 20 Mbps transmission rate which can be achieved easily with 4G (LTE). The mobile application requires the smartphone to have at least 8 GB RAM, 3 GHz of processor clock speed, and 64 GB of non-volatile storage. The smartphone that is needed by the user is independent from the brand, meaning that both Android-based and iOS-based smartphones' hardware shall run the mobile application with no major modification to the code base.

4.3.1.3 Software Interface

The backend software, which is going to be decided as Java-based Spring Boot, will act as the central processing software piece, accepting requests made by the users of ChargeMind application. All the requests sent by its users will rely on o RESTful API and these requests are going to be parsed and interpreted by the backend software. These requests are made through the frontend of the application which is based on React Native with the body of the request being in the JSON format.

ChargeMind backend software shall interact with SQL and/or NoSQL database technologies via leveraging specialized libraries for Java that abstract low-level communication while providing a standard interface for database operations such as, creating, reading, updating, and finally deleting. Structured queries will be used to access SQL databases; on the other hand, object-based formats like JSON will be used for NoSQL databases. By using encryption, communications with the database will be secured. To address issues like connection failures or invalid queries, robust error handling will be implemented. To validate functionality under varying conditions, integration tests will be developed.

Java-based backend software interacts with Python-based scripts that are created for machine learning tasks, using external interfaces such as JPype [26] or Py4J [27], that enable seamless communication between Python and Java without depending on external APIs or inter-process communication. This integration lets the backend invoke Python functions explicitly, facilitating tasks like data preprocessing or training with frameworks such as PyTorch [28] or TensorFlow [29]. Data, or an argument, is passed from Java to Python in a predefined, structured format, and results are returned directly to the Java application for further processing. The integration is secured to protect data integrity and confidentiality, with error handling logics in place to handle script execution failures or invalid inputs. Generic testing ensures reliable and efficient operation under various workloads and scenarios.

4.3.1.4 Communication Interfaces

The communication interface is a physical or logical connection that allows the exchange of data between two devices, systems, or software. It contains methods and protocols used to make it possible for different systems to communicate with each other. Data formats, transmission rates, and physical connections are typically identified by this interface, which decodes the data flow between hardware or software components.

ChargeMind uses a REST API approach to integrate many data sources. REST API endpoints provide access to various services such as location information of charging stations, realtime occupancy rates, energy pricing data and carbon footprint calculations. These APIs collaborate with third-party platforms like Open Charge Map, Google Maps, and regional energy databases to consistently deliver the data that users need. JSON is the most preferred format for data exchange in REST APIs. The ChargeMind application uses the JSON format to receive and transmit dynamic data such as the real-time status of charging stations, pricing information and user account settings Data is sent over the HTTPS protocol. This protocol guarantees user privacy and security by ensuring that data is transmitted in an encrypted manner. Additionally, communication procedures use the Secure Sockets Layer/Transport Layer Security (SSL/TLS) protocols. By doing this, a decently safe environment is established against external threats while the client and server exchange data. For client-server communication, the application uses the TCP/IP protocol. This protocol offers a fundamental way of communication that ensures data delivery and validity. Likewise, the ChargeMind platform offers a range of authentication options to boost user security and offer a smooth user experience. OAuth 2.0 offers powerful authorization features and more flexible integrations. Using Google's OAuth 2.0 and OpenID Connect protocols, users can log in to the system securely. The ChargeMind platform uses the Multi-Factor Authentication (MFA) method to improve user security. This method requires more than one verification stage, such as a password and a one-time code sent to their mobile device ensuring that accounts are protected against unauthorized access. Furthermore, Single Sign-On (SSO) offers a system that does not require users to log in again for the duration of the session after logging in once.

4.3.2 Functional Requirements

Name	Register to App
Use Cases	UC-1
Purpose/description	Allow users to create a new account to access the app's features.
Input	• User email address • User password
Processing	· Validate the input (check if the email is in the correct format, the password meets complexity requirements).
	· Check for duplicate accounts using the provided email. · Store the user data securely in the database.
Output	· Confirmation of successful registration.

	· Error messages if validation fails ("Email already in use" or "Invalid password format").

Name	Login to App
Use Cases	UC-1
Purpose/description	Users may safely access the program thanks to this feature. The user may see charging stations, add favorite stations, and pay after signing in. An interface is displayed to the user and user verification is carried out throughout the login procedure.
Input	• Username/email • Password • Two-factor authentication (SMS code) • Security Tokens (Google Authentica-tor)
Processing	The login form is where the password and username/email are obtained. They are transmitted to the server after being encrypted using salting and hashing techniques. The hash kept in the database is contrasted with the password on the server. The user's account details are acquired if the password verification process is successful. A second verification code is requested from the user if twofactor authentication (MFA) is enabled. A further verification code is requested by the system. This code can be sent by SMS or produced by a security token application like Google Authenticator. Following verification, the user is given a session tokens and their session is formed.
Output	• After successfully logging in, the user is sent to the main page of the program. • The error message "Invalid username or password." is displayed for an invalid password or email in the event that the login attempt is unsuccessful.
	· "Verification code required." is displayed if two-factor authentication is not present. · A secure access key produced specifically for the user's usage in

	future transactions is called a session token.

Name	Plan Route For EV
Use Cases	UC-2
Purpose/description	Users will be able to view an optimal path from their current location to the destination location.
Input	• Current location of the user (Retrieved by GPS) • Destination location: Specified by the user or retrieved by GPS. • Filtering Choices. (Optional)
Processing	After geolocation of the current location and destination location is retrieved from the user, by using machine learning, the system will try to find an optimized path by both considering current location, destination location, pretrained machine learning model, and filtering choices.
Output	· Drawing of a route on the map showing current and destination location, alongside charging stations on the route.

Name	Filter
Use Cases	UC-3
Purpose/description	Allow users to filter charging stations based on specific criteria such as location, charger type, price, rating, and availability.
Input	• Location • Charger type • Rating • Availability • ₺ /kWh • Carbon Emission Rate
Processing	The system applies the selected filter criteria to the available charging stations database. It compares the attributes of each charging station (location, charger type, price, rating) against the user’s filter choices. The stations that do not meet the criteria are excluded
	from the results. After filtering, the system presents a refined list of charging stations that match the selected parameters.

Output	· List of filtered charging stations based on the selected criteria.

Name	Real-Time Availability Updates
Use Cases	UC-4
Purpose/description	While it provides users with up-to-date information about the availability of charging stations in real time, such as whether another user is currently using the station, it enables efficient planning and reduces waiting times.
Input	• User location or selected region • Desired time of use • Charger type (AC/DC)
Processing	The system retrieves real-time data from both users' GPS locations and the station APIs or databases. Stations' availability data is processed to determine which ones are operational and have available slots. The system dynamically updates status to reflect changes in availability.
Output	A real-time map view or list of charging stations showing availability status will be displayed for the selected station or across the map (e.g. "Available", "In Use").

Name	Stations Information Updates
Use Cases	UC-5
Purpose/description	Ensure that users have access to accurate and up to date information about charging stations, including general availability, pricing, and status, provided by contracted station operating companies.
Input	· Real-time data updates from contracted companies (general availability, pricing, maintenance status etc.) by API or manual submissions of them.
Processing	The system receives data updates from companies through secure APIs or manual inputs. The updated information is integrated into the app and reflected in real-time. Updates, such as station unavailability or pricing changes, will not negatively impact the user's charging process.

Output	• Real-time updates displayed in the app for each charging station.

Name	Carbon Emission Calculation
Use Cases	UC-6
Purpose/description	Allows users to view regional carbon emission data by filtering charging station's locations. This encourages environmental awareness among electric vehicle (EV) users and allows consumers to prioritize eco-friendly stations.
Input	• Location: The area name or coordinates of the charging station. • The E-Charge Station's name or ID: A unique station identification for accurate emission data retrieval.
Processing	The systems retrieve carbon emission information from nearby monitoring devices or environmental APIs. Afterwards, handle the data according to the input (station ID, location) for filtering purpose. Then utilize machine learning or pre-established algorithms to analyze and determine the proportion of carbon emissions for the designated area. Finally, give customers rated recommendations, if possible, comparing emissions from neighboring stations and the source of electricity produced (solar, wind, coal, etc.)
Output	• Carbon Emission Percentage: Displays the computed carbon emission percentage for the designated charging station and the neighborhood, taking into account the location of the station, emissions in the nearby areas • Electricity source: Displays the source of the electricity (e.g., renewable or non-renewable).

Name	User Reviews
Use Cases	UC-7
Purpose/description	Allow users to write and submit reviews for charging stations based on their experiences.

Input	• User Selected Station ID • Text review (comments)
Processing	First, users select the station to review and make a comment. The review is stored in the database and linked to the corresponding charging station. The system updates the station's profile with the new review.
Output	· Feedback for successful review submission.

Name	User Ratings
Use Cases	UC-8
Purpose/description	Users may write reviews and provide a star rating (from 1 to 5) to each charging station. Reviews provide charging station operators the chance to enhance their offerings and assist in educating other consumers.
Input	• User selected score • User Selected Station ID
Processing	The user provides a rating and remark on the charging station detail page. After that, the user input is validated (for instance, ratings may only be made by customers who have actually received the service). The star ratings are stored in the database. The charging station's average rating is recalculated. The other users can see the remark.
Output	• When a submission is successful, the user is notified: "Your review has been saved. I'm grateful. • The charging station detail page shows the most recent average rating. · If the submission is not successful: • "Your rating could not be sent," is the error message. Please give it another try.

Name	Payment to Charging Service Providers
Use Cases	UC-9
Purpose/description	It enables users to seamlessly make payment to all charging station providers to overcome the burden of using multiple provider applications.

Input	· Credit/Debit Card Credentials · User Profile Information
Processing	User credit/debit card information will be used to send a payment request to the contracted bank, the payment will be received,
	and it will be redirected to corresponding charge station provider's contracted bank.
Output	• If payment is successful, an acknowledgment notification will be shown to the user. • If payment is failed, a negative acknowledgment will be shown to the user.

4.3.3 Software System Attributes

This section specifies the basic characteristics of the ChargeMind mobile application, such as reliability, scalability, and performance. These characteristics include general standards that support the functional features of the application and meet user expectations.

4.3.3.1 Portability

ChargeMind has been developed to work seamlessly on different platforms (Android and iOS). The application appeals to a wide range of users by providing cross-platform support using React Native.

· The application offers a consistent user experience on devices with different screen sizes and resolutions. In addition, it offers fast update support to be compatible with different versions of iOS and Android. It offers a global platform by providing English and Turkish language support.

4.3.3.2 Performance

ChargeMind is designed to meet high performance standards to improve user experience and increase system efficiency. · The application's response time to user interface actions is a maximum of 1 second.

- Real-time loading of charging station information will take a maximum of 2 seconds.
- The annual uptime rate of the application is more than 99.7%.
- Map integration should load in 1.5 seconds at most.
- Users should complete the payment transaction in 4 seconds.
- The server should be capable of processing queries from 10,000 users at the same time. Load balancing techniques will be used for this.
- Machine Learning tasks shall not exceed the maximum execution time of 5 seconds for each user.

4.3.3.3 Usability

Real-time availability, easy session management, and rapid access to charging stations are all features of ChargeMind's simple interface.

- It takes 2-3 minutes for users who use the application for the first time to understand the basic functions and start using the application actively.
- Explanatory feedback is provided for each action taken by users during their interaction with the application. In case of an error, it will display meaningful error messages that guide the user.
- The application offers consistent user experience across different devices and screen sizes. It will be optimized to be comfortable to use on both small-screen smartphones and large-screen phones.
- To enhance the usability of the app, the color palette and font sized will be carefully selected to provide a user-friendly experience.

4.3.3.4 Adaptability

Chargemind will have a high level of adaptability to adapt to different user needs.

· It offers recommendations based on users' behaviors and preferences within the app. It will provide personalized recommendations by learning users' habits. For example, frequently used charging stations can be highlighted by the app as users' first choice.

4.3.3.5 Scalability

ChargeMind will have a high degree of scalability to adapt to increasing user demands and growing business needs.

- The application will be configured to support rapid user growth from 10,000 to 1,000,000. Server resources can be quickly increased as needed using cloud-based infrastructure (AWS).
- The database structure will be optimized to avoid performance loss as data size increases.
  New charging stations can be easily added without affecting the application's existing infrastructure. New station information will be added quickly thanks to API integrations and modular structures.
- Load balancing [30] techniques will be used to meet increasing user demands. Thus, the traffic of increasing user demands is managed.

4.3.3.6 Security

ChargeMind will be equipped with strong security measures to ensure the security of user data.

· Application will use hashing and salting techniques to securely store user passwords. Hashing and salting operations will ensure that passwords are safe even in the event of theft or leakage of user passwords, increasing the overall security level of the application.

4.3.3.7 Reliability

ChargeMind will be designed to have a high level of reliability.

- The application will be configured to provide high availability. The system will be designed to target at least 99.9% uptime at all times.
- Backup and fault tolerance will be provided using data replication and multiple data centers.
- The application will notify the user as quickly as possible when an error occurs, allowing steps to be taken to resolve the problem.
- The application will be configured to be updated regularly without affecting the user experience.

4.3.4 Safety and Fault Tolerance Requirements

This section explains the ChargeMind platform's safety criteria, ensuring that the system is secure, reliable, and capable of dealing with possible safety risks while in operation.

4.3.4.1 Data Privacy and Security

- **Confidentiality**

  All user-related and operational data, such as user information and energy consumption, are managed and stored in a secure manner to avoid unwanted access. Confidential data is encrypted during transmission and at rest with industry-standard encryption methods. HTTPS is utilized during transmission, with the TLS protocol and strong cipher suites (e.g., AES-256 and RSA with 2048-bit keys) ensuring safe communication between the client and the server. Data at rest is secured using AES with a key length of 256 bits, which is widely regarded as one of the most secure encryption methods available. Furthermore, encryption keys are securely handled using a Key Management System (KMS), ensuring that keys are periodically changed. This encryption method ensures that user data is kept secure throughout storage, transfer, and processing.

- **Integrity**

  Data integrity is protected by modern cryptographic techniques such as Digital Signatures, Message Authentication Codes (MACs), and Version Control. Digital signatures are used to verify the validity and integrity of essential data; the data is hashed and encrypted using the sender's private key, allowing the recipient to decrypt it using the sender's public key and validate the data. MACs, created with a secret key and a cryptographic hash function (e.g., HMAC), are used to authenticate and ensure the integrity of data sent between parties. Git version control is used to track changes and preserve historical accuracy, maintaining data integrity over time, particularly in systems that receive regular updates.

- **Authentication**

ChargeMind authentication is supposed to be strong and secure. Multi-Factor Authentication (MFA) is enabled, forcing users to submit at least two verification factors, such as a password and a one-time code sent to their mobile device. Google Authenticator tokens are utilized for this purpose. OAuth 2.0 is used for secure token-based authentication, which enables interaction with third-party services such as Google for user identity verification. Strong password policies are enforced, requiring users to create complicated passwords that are hashed using bcrypt and stored securely. Furthermore, Single Sign-On (SSO) is available to simplify user access to many connected services, reducing the need for multiple logins. These safeguards jointly protect user accounts and sensitive data, ensuring a high level of security throughout the platform.

- **Authorization**

ChargeMind employs Role-Based Access Control (RBAC) to control access to sensitive or important system functions by giving specific roles to users or services, each with distinct permissions that define their actions within the system. Roles include Administrator, User, and Maintenance Staff, with access privileges provided based on the least privilege principle, ensuring that users and services only have the minimum permissions required. Roles are hierarchical, with higher-level roles (e.g., Administrator) having greater permissions and lower-level roles (e.g., User) having more restricted access. Granular permissions are offered to provide more precise control over access to data (e.g., user data, transaction records) and operations (e.g., starting/stopping charging sessions). ChargeMind offers a role management interface, allowing administrators to edit and allocate responsibilities as needed, along with an access request and approval workflow for temporary privilege escalation. All changes must be evaluated and approved before they are given.

4.3.4.2 Fault Tolerance and Error Handling

ChargeMind uses a micro/macroservices [31] architecture and distributed system design, as well as fault-tolerant methods, to ensure that operations continue even when hardware or networks fail. The system smoothly handles server crashes and disruptions between e-charging stations and the central system by leveraging real-time data streaming and communication technologies such as Apache Kafka. Fault detection initiates automatic error handling operations, providing user-friendly error messages to alert users to any issues. In the event of a critical mistake, such as a malfunctioning charging station, the system quickly contacts authorized personnel of contracted companies to ensure a quick fix. ChargeMind also incorporates automated recovery capabilities, allowing the system to self-heal from small errors with minimal downtime. High availability and resilience are achieved by the use of redundant components, load balancing, and failover mechanisms.

4.3.4.3 Continuous Testing

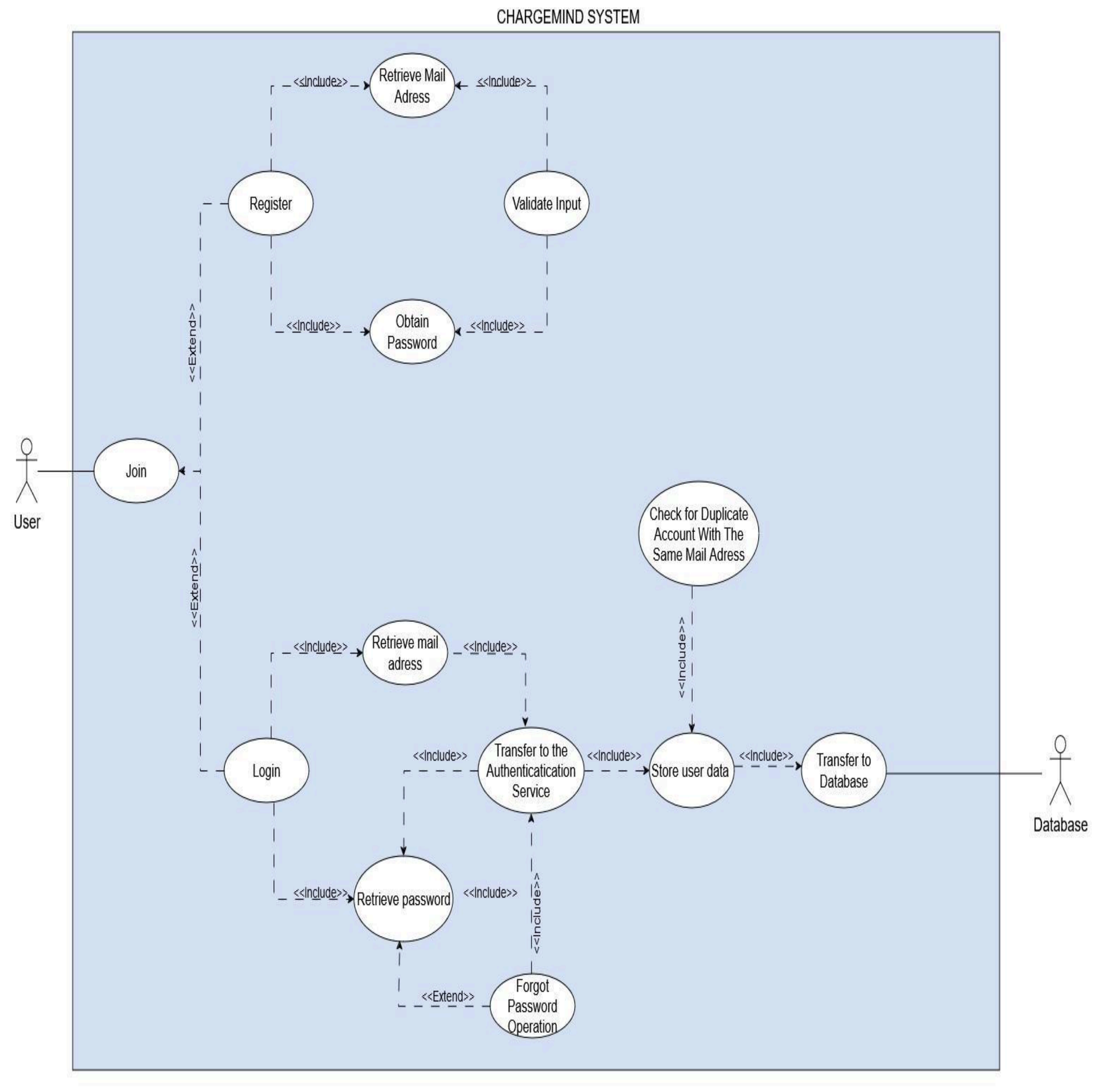
ChargeMind develops a comprehensive security testing strategy, which includes unit testing, integration testing, and end-to-end security testing, to verify the system's resilience against potential threats. Critical components such as user authentication, data encryption, and access control methods are tested unit by unit to ensure that they function properly under expected situations. Integration testing examines the interaction of various system components, such as e-charging stations, the central server, and the user interface, replicating real-world scenarios to find any integration flaws that could compromise security or dependability. Endto-end security testing includes penetration testing, vulnerability assessments, and simulations of various attack scenarios, assuring the platform's resilience to internal and external attacks.

4.4 Use Case Diagrams

4.4.1 UC-1

Name	Register/Login to App
Use Case	UC-1
Actor	User, Database
Description	It allows users to register and log in to the application. This process includes steps such as email address verification and password retrieval. The database is active in verifying and storing user data.
Precondition	The user must have a valid email address to access the application. If an account has already been registered, the login process is performed.
Scenario	1.The user wants to register and enters her/his email address. 2.The system verifies the email address. 3.User creates password. 4.The system checks whether there is a previous record with the same email (from the Database). 5.If the registration is successful, the user data is saved in the database. 6.The user logs in and the email verification process begins. 7.In case of forgotten password, the user completes the password recovery steps via email.
Postcondition	The user successfully logs in to the app or creates a new account. Registration information is stored securely in the database.

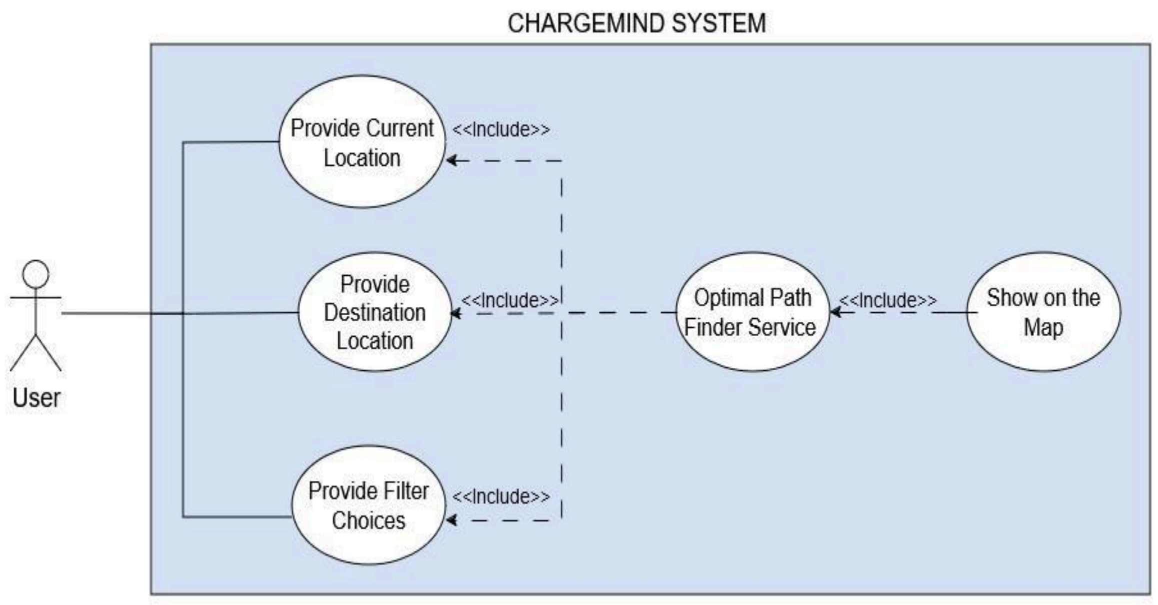
Exceptions	1.If the user enters an incorrect or invalid email address, the registration/login process will fail.
	2.If the password requirements are not met, the user is requested for a new password. 3.If the user tries to register with an already registered e-mail address, they are informed that the e-mail address already has an account.



4.4.2 UC-2

Name	Plan Route For EV
Use Case	UC-2
Actor	User

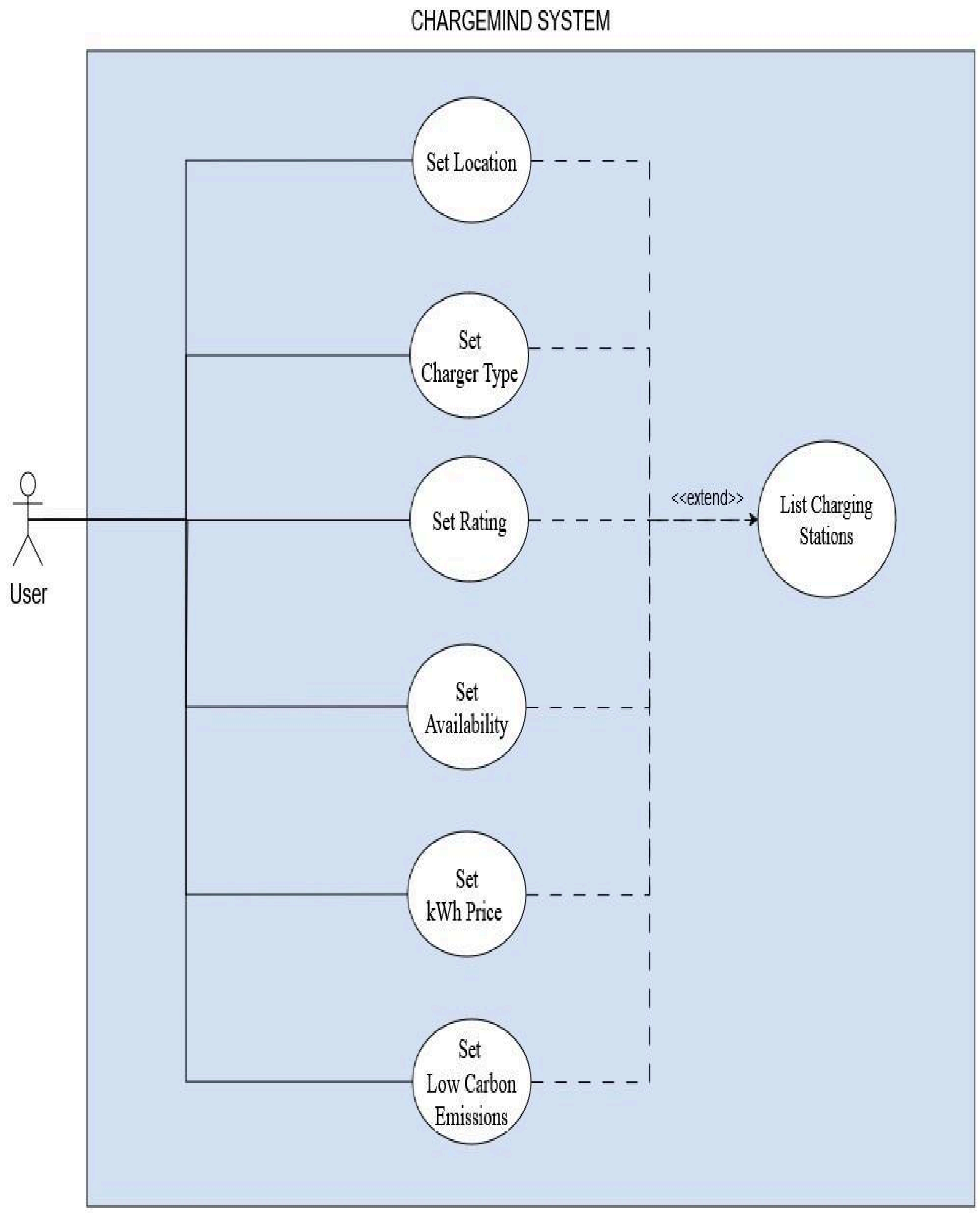
Description	The system finds the optimal path for the user based on their current location, destination, and filter choices.
Precondition	The user must provide their current location, destination location, and filter choices (if any).
Scenario	1.The user opens the system. 2.The user enters their current location. 3.The user specifies the destination. 4.The user optionally provides filter choices. 5.The system processes the input and calculates the optimal path. 6.The system displays the optimal path on the map.
Postcondition	The optimal path is displayed on the map, ready for the user to follow.
Exceptions	1.Invalid or missing location inputs (current or destination). 2.No charging stations available along the route. 3.System error during path calculation.



4.4.3 UC-3

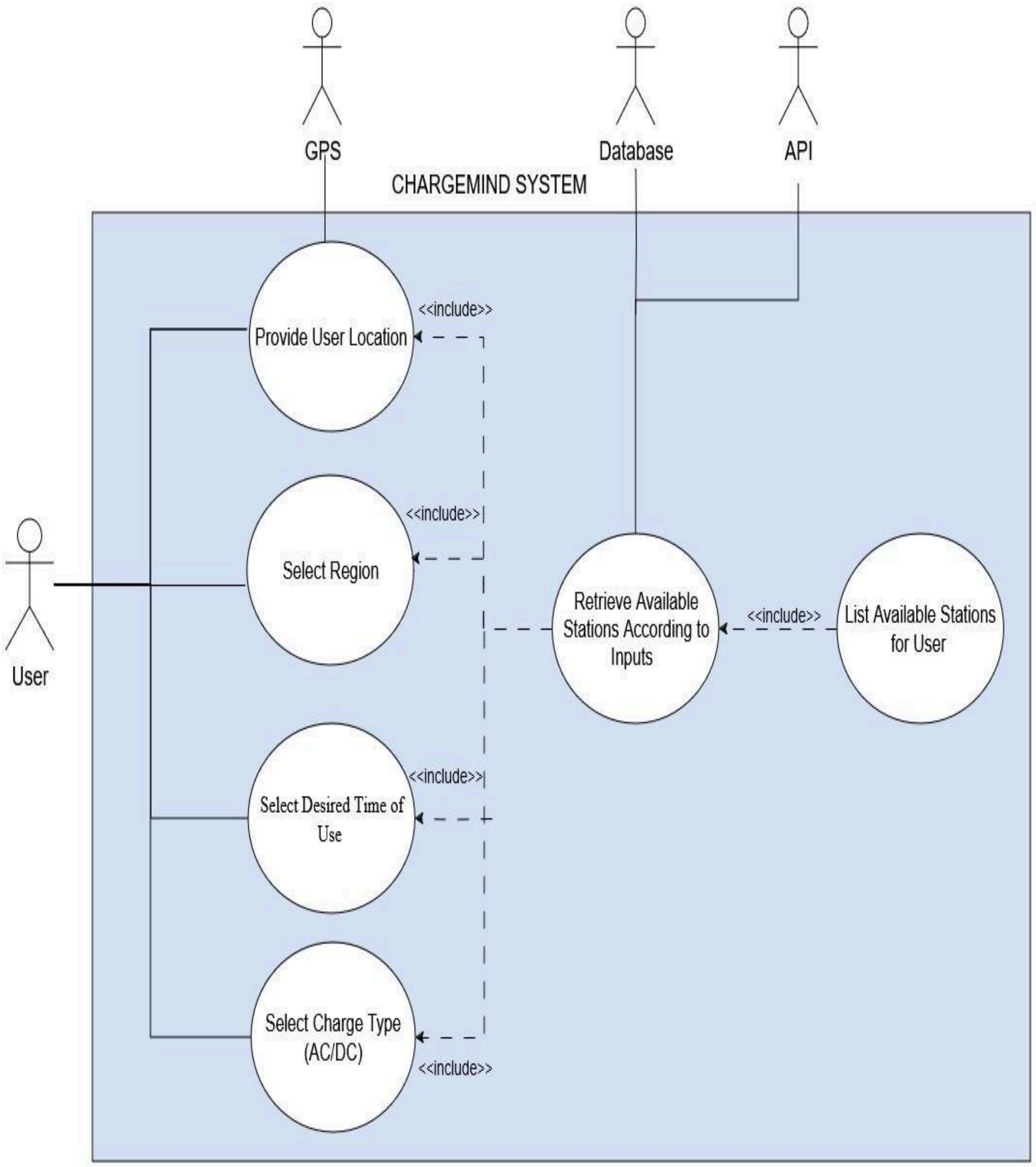
Name	Filter
Use Cases	UC-3
Actor	User

Description	The system displays a list of charging stations based on the user's location, charger type, and filter preferences.
Precondition	The user must provide their location and can optionally set additional filter criteria such as charger type, rating, availability, kWh price, and carbon emissions.
Scenario	1.The user opens the system. 2.The user sets their current location. 3.The user optionally provides filters: charger type, rating, availability, kWh price, or low carbon emissions. 4.The system processes the input and queries the database for charging stations matching the criteria. 5.The system displays the list of charging stations.
Postcondition	The list of charging stations is displayed to the user, ready for selection or further action.
Exceptions	1.Invalid or missing location input. 2.No charging stations matching the criteria. 3.System error during query or data retrieval.


CHARGEMIND SYSTEM
Set Location
Set Charger Type
Set Rating
Set Availability
Set kWh Price
Set Low Carbon Emissions
<<extend>>
List Charging Stations
User

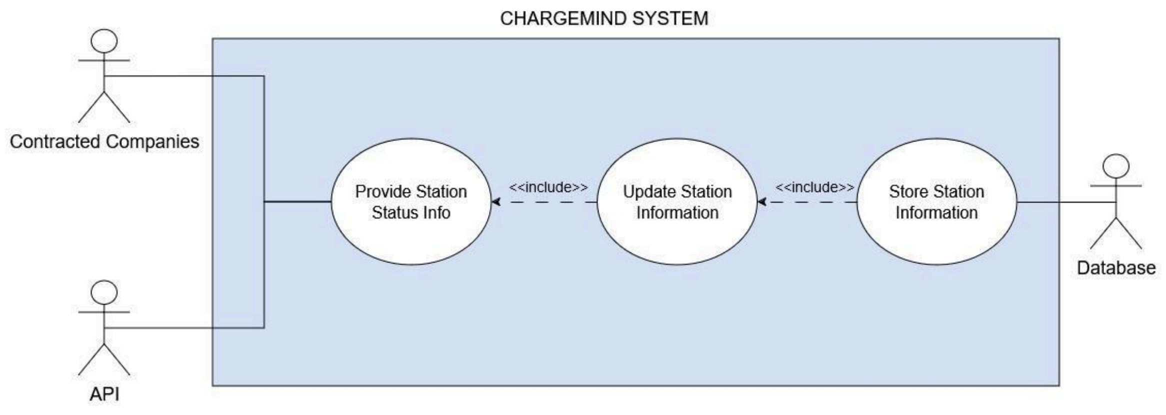
4.4.4 UC-4

Name	Real-Time Availability Updates
Use Cases	UC-4
Actor	User, GPS, Database, API
Description	The system lists available charging stations for the user based on location, region, desired time, and charge type.
Precondition	The user must provide their location (via GPS or manually) and optionally select filters such as region, desired time of use, and charge type (AC/DC). The system must have access to the database and external APIs for retrieving station information.
Scenario	1.The user opens the system. 2.The user provides their location (manually or via GPS). 3.The user selects additional filters, such as region, desired time of use, and charge type (AC/DC). 4.The system retrieves available stations from the database and APIs based on the provided inputs. 5.The system lists the available stations for the user.
Postcondition	A list of available charging stations is displayed to the user, allowing them to select a station for further action.
Exceptions	1.GPS fails to provide location data. 2.Invalid or incomplete inputs from the user. 3.No charging stations available matching the criteria. 4.System or database errors during data retrieval. 5.API connectivity issues.


GPS
Database
API
CHARGEMIND SYSTEM
Provide User Location
<<include>>
Select Region
<<include>>
User
Retrieve Available Stations According to Inputs
<<include>>
List Available Stations for User
Select Desired Time of Use
<<include>>
Select Charge Type (AC/DC)
<<include>>

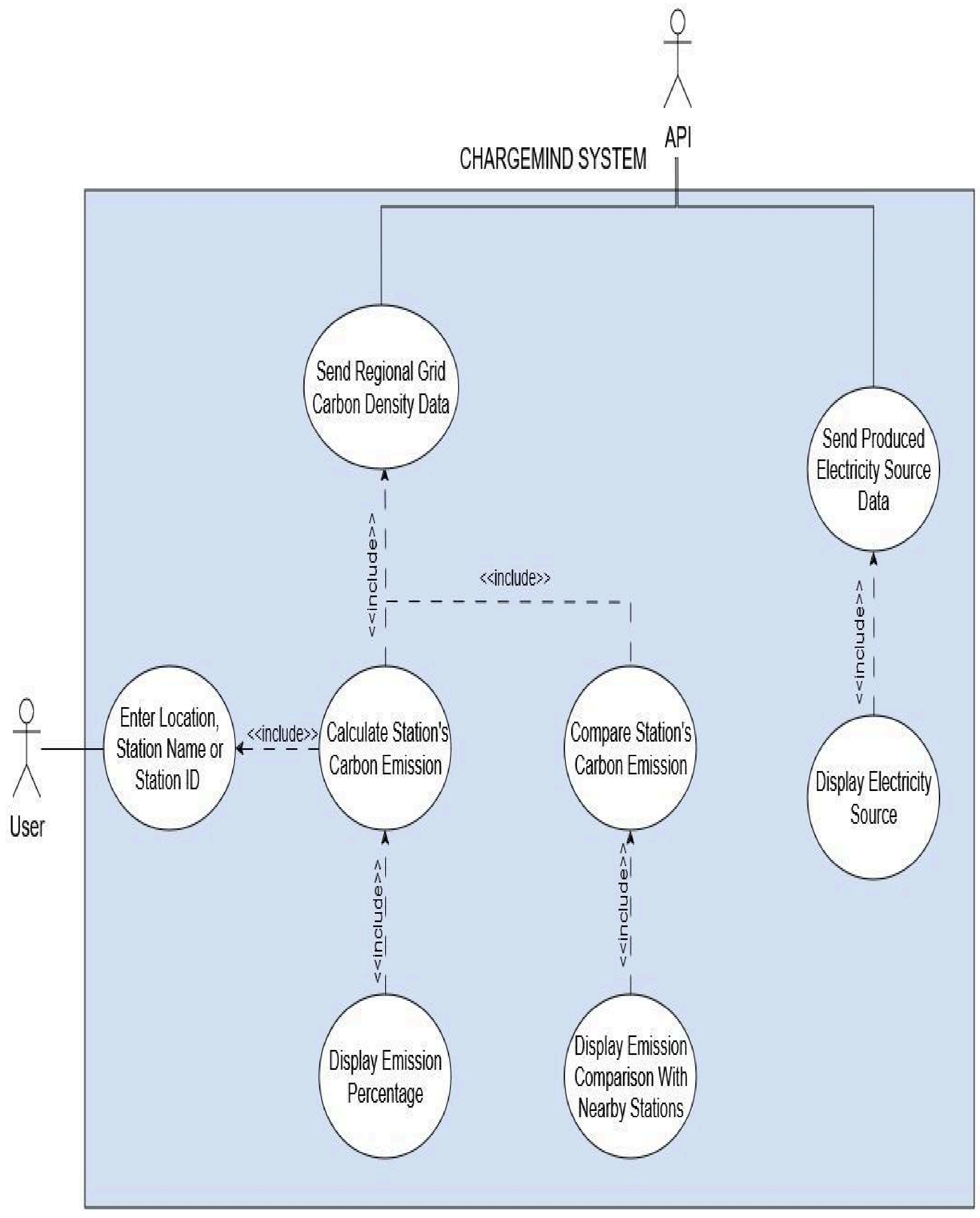
4.4.5 UC-5

Name	Stations Information Updates
Use Cases	UC-5
Actor	Contracted Companies, API, Database
Description	The system updates the status and information of charging stations based on data provided by contracted companies and APIs. The updated information is stored in the database.
Precondition	Contracted companies or APIs must provide accurate and up-to-date station status information. The database must be operational and accessible.
Scenario	1.Contracted companies or APIs send station status information to the system. 2.The system validates the received data. 3.The system updates the charging station information based on the input. 4.The updated information is stored in the database.
Postcondition	The charging station information in the database is updated and ready to be accessed by users or other processes.
Exceptions	1.Invalid or incomplete data provided by contracted companies or APIs. 2.Database connection or storage failure. 3.System error during the validation or update process.


CHARGEMIND SYSTEM
Contracted Companies
API
Provide Station Status Info
<<include>>
Update Station Information
<<include>>
Store Station Information
Database

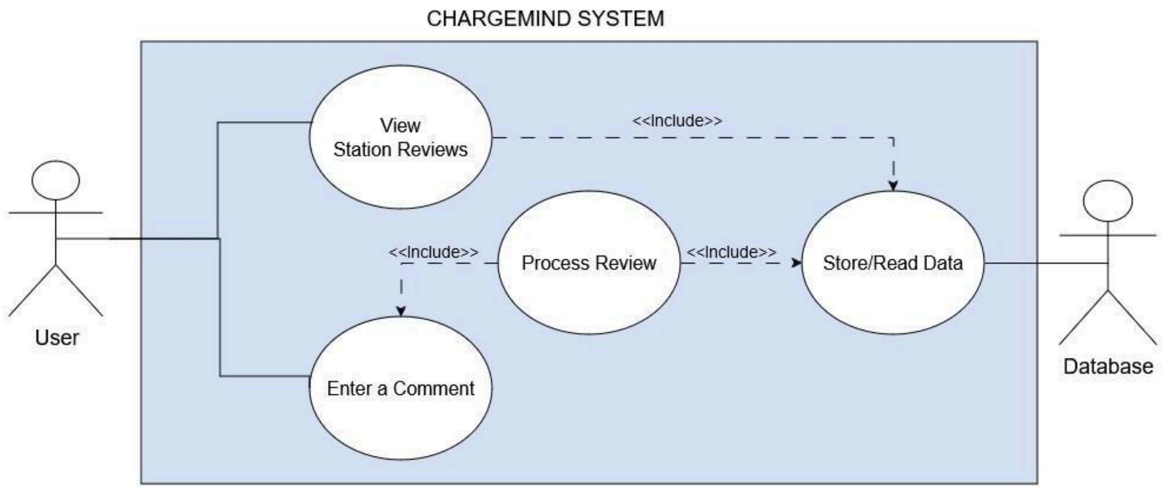
4.4.6 UC-6

Name	Carbon Emission Calculation
Use Cases	UC-6
Actor	User, API
Description	The system calculates the carbon emissions of a charging station based on the regional grid carbon density and the electricity source data. It also allows the user to compare the emissions with nearby stations and displays the electricity source used.
Precondition	The user must provide location, station name, or station ID. The API must supply regional grid carbon density data and electricity source data.
Scenario	1.The user provides the location, station name, or station ID. 2.The system retrieves regional grid carbon density data from the API. 3.The system calculates the carbon emissions for the specified station. 4.The system displays the carbon emission percentage. 5.Optionally, the user requests to compare the station's emissions with nearby stations. 6.The system displays the comparison results. 7.The system displays the electricity source used at the station.
Postcondition	The system provides detailed carbon emission data, comparison with nearby stations, and electricity source information to the user.
Exceptions	1.Invalid or incomplete input from the user (e.g., missing location, station name, or station ID). 2.Failure to retrieve data from the API (e.g., network error, API unavailability). 3. Inaccurate or missing data from the API. 4.Calculation errors due to data inconsistencies.


API
CHARGEMIND SYSTEM
Send Regional Grid Carbon Density Data
Send Produced Electricity Source Data
<<include>>
<<include>>
<<include>>
User
Enter Location, Station Name or Station ID
<<include>>
Calculate Station's Carbon Emission
Compare Station's Carbon Emission
Display Electricity Source
<<include>>
<<include>>
Display Emission Percentage
Display Emission Comparison With Nearby Stations

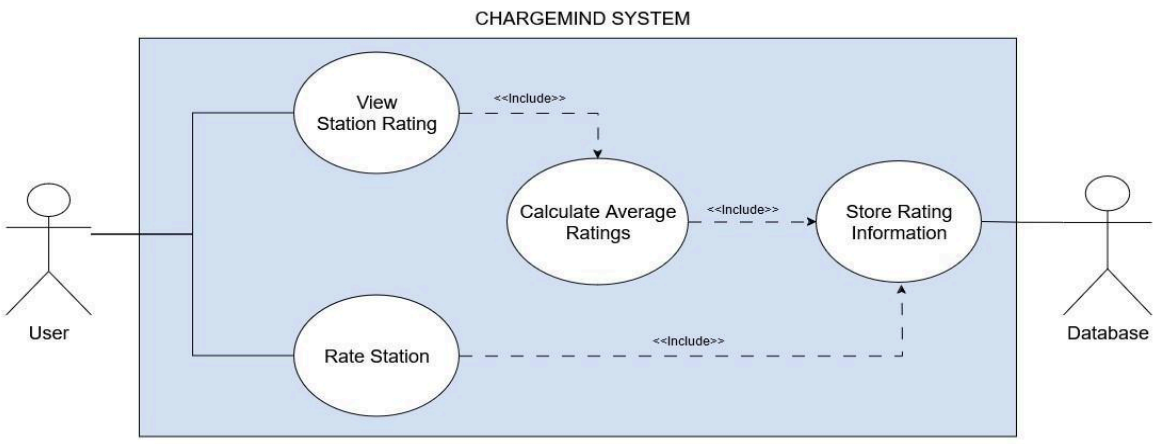
4.4.7 UC-7

Name	User Reviews
Use Cases	UC-7
Actor	User, Database
Description	The system allows users to view station reviews, add comments, and process the reviews. The data is stored and retrieved from the database as needed.
Precondition	The database must be operational and contain station review data. Users must be authenticated if required to leave comments.
Scenario	1.The user accesses the review section for a station. 2.The system retrieves station reviews from the database and displays them. 3.The user enters a new comment if desired. 4.The system processes the user's comment (e.g., validation, moderation). 5.The system stores the new comments in the database.
Postcondition	The user can view existing reviews, and any new comment is stored in the database.
Exceptions	1.Database connectivity issues prevent retrieval or storage of data. 2.User submits an invalid or inappropriate comment. 3.System errors during review processing (e.g., validation failure).



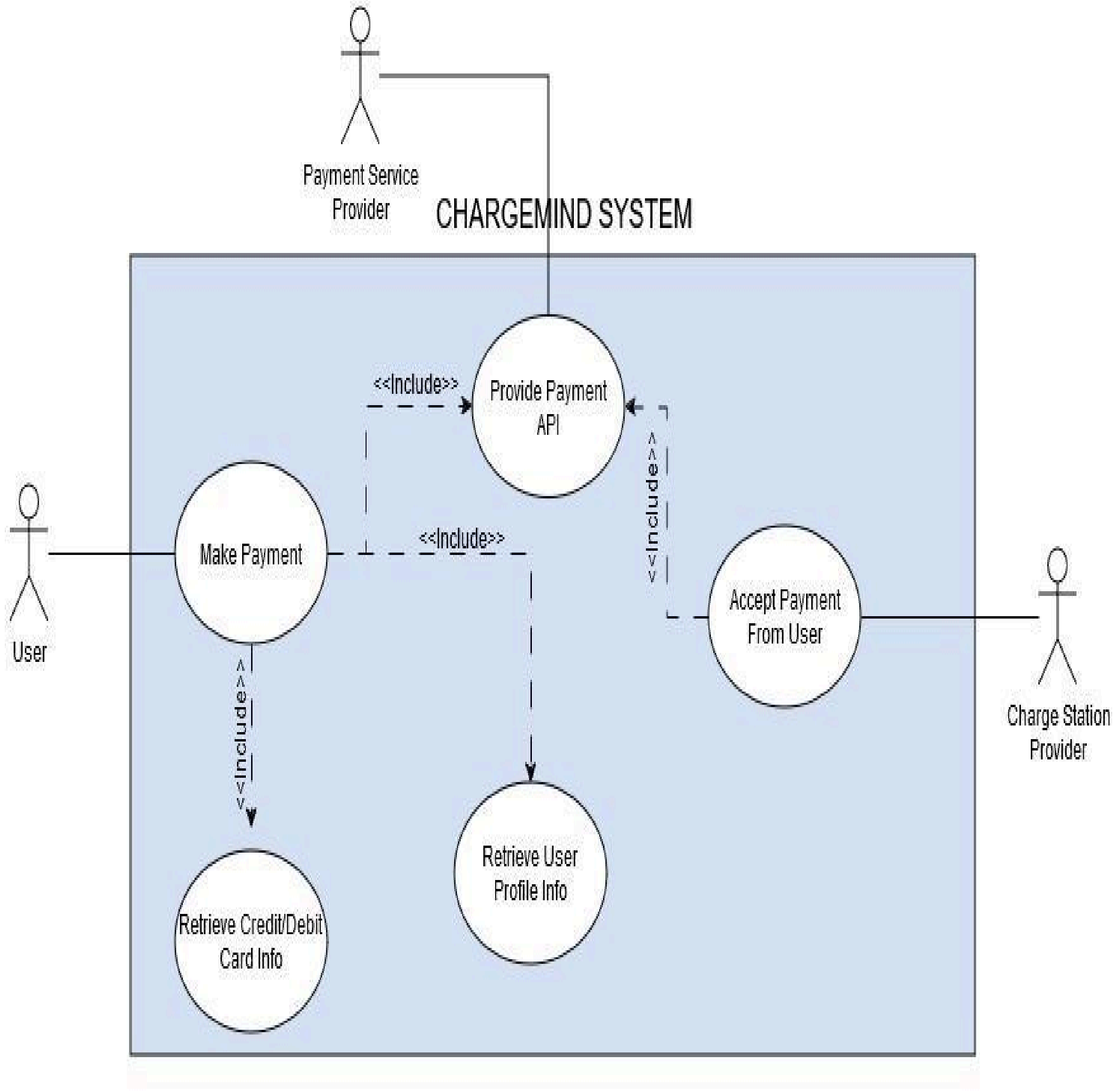
4.4.8 UC-8

Name	User Ratings
Use Cases	UC-8
Actor	User, Database
Description	The system allows users to rate charging stations, calculate average ratings, and view the ratings. The rating information is stored and retrieved from the database.
Precondition	The database must be operational and contain rating data for stations. Users must be authenticated if required to submit ratings.
Scenario	1.The user selects a station to rate or view its rating. 2.The system retrieves the station's rating data from the database and displays it. 3.If the user rates the station, the system validates the rating input. 4.The system calculates the updated average rating for the station. 5.The system stores the updated rating information in the database.
Postcondition	The station's updated rating is stored, and the user can view the latest average rating.
Exceptions	1.Database connectivity issues prevent retrieval or storage of data. 2.User submits an invalid rating (e.g., out of acceptable range). 3.System errors during rating validation or calculation.



4.4.9 UC-9

Name	Payment to Charging Service Providers
Use Cases	UC-9
Actor	User, Payment Service Provider, Charge Station Provider
Description	The system allows users to make payments for charging services by integrating with payment service providers. It retrieves user payment information and processes the transaction.
Precondition	The user must have an active profile with valid payment details. The payment service provider and charging station provider must be operational.
Scenario	1.The user selects the option to make a payment for charging services. 2.The system retrieves the user's profile information. 3.The system retrieves credit/debit card details or linked payment information. 4.The system provides the payment API to initiate the transaction. 5.The payment is processed and accepted by the charge station provider via the payment service provider.
Postcondition	The payment is successfully completed, and the user can proceed with charging services.
Exceptions	1.Invalid or expired payment information. 2.Connectivity issues with the payment service provider. 3.Payment declined by the user's bank or payment service provider. 4.System errors during payment processing.


Payment Service Provider
CHARGEMIND SYSTEM
<<Include>>
Provide Payment API
<<Include>>
User
Make Payment
<<Include>>
<<Include>>
Accept Payment From User
Charge Station Provider
Retrieve Credit/Debit Card Info
Retrieve User Profile Info

5.Software Design Document

List of Figures

5.1 Introduction

5.1.1 Scope

The ChargeMind technology is intended to address one of the most serious problems for electric vehicle (EV) owners: battery depletion during long trips, or more generally, “range anxiety”. ChargeMind ensures that EV drivers and fleet owners may locate appropriate paths that match their energy and trip requirements by providing tailored and optimal route planning.

5.1.2 Purpose

The purpose of this document is to provide detailed descriptions of design concerns and design decisions, recording design information and explanations for the stakeholders. This SDD will further elaborate and explain the functional and non-functional requirements mentioned in the Software Requirement Specification document.

5.1.3 Glossary

Term	Definition
Activity Diagram	A flowchart showing how one activity leads to another activity.
Artificial Intelligence	Technology enabling machines or computers to mimic human-like intelligence and learning capabilities.
Backend	The part of a computer system or application that is not directly accessed by the user, typically responsible for storing and manipulating data.
Class Diagram	A graphical notation for building and visualizing object-oriented systems.
Data Flow Diagram	It is a way of representing a flow of data through a process or a system
EVRP	The EVRP is an optimization problem that focuses on determining the most efficient routes for electric vehicles while considering constraints like battery capacity, charging station availability, and energy consumption.
Fleet Owner	It refers to an individual or company that owns and operates a fleet of vehicles for various purposes.

Frontend	Relating to or denoting the part of a computer system or application with which the user interacts directly.
Hardware	The machines, wiring, and other physical components of a computer or other electronic system.
Hashing	The practice of transforming a given key or string of characters into another value for the purpose of security like one-way encryption [32].
Java	Java is a multi-platform, object-oriented, and network-centric programming language and computing platform for coding web applications [33].
JPype	JPype is a Python module to provide full access to Java from within Python [26].
Machine Learning	The ability of computer systems to learn from data and make predictions.
NoSQL	NoSQL is a type of database management system that is designed to handle and store large volumes of unstructured and semistructured data [34].
OAuth 2.0	One established protocol for authorization procedures is OAuth.
Py4J	Py4J enables Python programs running in a Python interpreter to dynamically access Java objects in a Java Virtual Machine [27].
PyTorch	PyTorch is a machine learning library based on the Torch library, used for applications such as computer vision and natural language processing [28].
Range Anxiety	Worry on the part of a person driving an electric car that the battery will run out of power before the destination, or a suitable charging point is reached.
React Native	React Native is a popular JavaScript-based mobile app framework that allows you to build natively rendered mobile apps for iOS and Android [35].

REST API	An application programming interface that accesses and utilizes data via HTTP requests is known as a RESTful API. The GET, PUT, POST, and DELETE data types that stand for reading, updating, creating, and removing resource related operations can be utilized with that data [14].
SDD	A description of software created to facilitate analysis, planning, implementation, and decision-making.
Sequence Diagram	Shows process interactions in software engineering that are time-ordered.
Spring Boot	Java Spring Boot is an open-source tool that makes it easier to use Java-based frameworks to create microservices and web apps [36].
SSL	SSL is an encryption-based Internet security protocol for the purpose of ensuring privacy, authentication, and data integrity in Internet communications [37].
Stakeholder	Denoting a type of organization or system in which all the members or participants are seen as having an interest in its success.
Tensorflow	TensorFlow is a leading open-source library designed for developing and deploying stateof-the-art machine learning applications using data flow graphs [38].
TLS	TLS is a widely adopted security protocol designed to facilitate privacy and data security for communications over the Internet [39].
User Interface Design	Techniques used by designers to construct interfaces in software or technological devices, focusing on appearances or style.

5.1.4 Definitions, Acronyms, and Abbreviations

- **AI:** Artificial Intelligence
- **EV:** Electric Vehicle
- **EVRP:** Electric Vehicle Routing Problem
- **GPS:** Global Positioning System
- **IEEE:** Institute of Electrical and Electronics Engineers
- **iOS:** iPhone Operating System
- **JSON:** JavaScript Object Notation
- **LTE:** Long-Term Evolution
- **OAuth:** Open Authorization
- **RAM:** Random Access Memory

- **REST API:** Representational State Transfer Application Programming Interface
- **SDD:** Software Design Description/Specification
- **SQL:** Structured Query Language
- **SSL:** Secure Sockets Layer
- **TLS:** Transport Layer Security
- **UI:** User Interface
- **UML:** Unified Modelling Language
- **UX:** User Experience

5.1.5 Intended Audience

The audience (i.e. stakeholders) of this document is intended to be technical and management staff of the ChargeMind project. These stakeholders include software developers, UI/UX designers, system architects, programmers, maintainers, testers, project managers, and product owners.

5.1.6 Conformance

The document conforms to standard naming conventions, terminology, and formatting guidelines specified by IEEE Software Design Description document standards for clarity and consistency.

5.1.7 Overview of the Document

In this Software Design Document (SDD), the technical and architectural layout of the ChargeMind application, an AI-based e-charge route planning system, is described. It provides a general guide for stakeholders, developers, and maintainers by detailing the system's overall structure, components by UML diagrams, activity diagrams, class diagrams, sequence diagrams, database schema, and user interface design. The document also addresses key aspects such as security, fault tolerance, and scalability to ensure a robust and reliable solution. Additionally, it includes design considerations for system testing and validation, ensuring that the final implementation meets performance and usability expectations. By presenting a clear and structured overview of the system's design, this document serves as a foundational reference throughout the development lifecycle.

5.1.8 Motivation

The motivation of the ChargeMind project is to provide users with a more comfortable charging experience with the increasing use of electric vehicles today. The need for smarter charging solutions is also becoming increasingly important with the use of electric vehicles. Current charging applications may be inadequate in areas such as route planning, accessibility, and user interface design. This project aims to address these shortcomings with real-time data integration, advanced route planning algorithms, and user-friendly features. It also aims to provide a more environmentally friendly approach with carbon footprint tracking and real-time emission estimation. The goal of the project is to support sustainable transportation by not only providing users with a practical solution but also making them aware of their environmental impact.

5.2 System Overview

The ChargeMind project is built on top of various core components to fit the needs of electric vehicle drivers. These core components of the application are database, backend, frontend, machine learning model. React Native is preferred for the front-end part to make it platform independent (i.e., both works on iOS and Android). Users of the application are offered miscellaneous such as basic route planning, advanced filtering for different concerns, and realtime status retrieval from charging stations. Each user can change preferences on the app, interact with the map, visualize paths, leading to a customized experience for each user.

On the backend, Java and Spring Boot are desired to be used. The design behind these technologies is to handle critical parts of the app such as process of data, optimization of routes, and execution of advanced machine learning models. The backend also plays the role of a bridge between third party services such as Google Maps, and various charge station service providers. This design leads the system to stay up to date with the rest of the world. For the storage purposes, a combinational usage of SQL and NoSQL is preferred. While keeping, for example, user credentials in a structured manner in SQL databases, the system will adapt to fresh data with a much more flexible method with the help of NoSQL. This approach shall make the system future-proof, flexible, and scalable.

Artificial Intelligence has a crucial role in the ChargeMind system. It is the backbone of the route planning feature, essential for effectively finding available stations while optimizing the route for keeping the user preferences in mind. This is achieved via not only traditional filtering methodologies but also machine learning models that the ChargeMind system is going to use. Tensorflow and/or PyTorch will be the preferred libraries for the development of the model. The machine learning model that will be developed shall find the shortest route for a user while pinpointing potentially available charging stations along the route. The Java backend shall communicate with the machine learning model via inter-communication solutions such as Py4J or JPype.

To meet its security requirements, ChargeMind uses industry-standard protocols like OAuth 2.0 for secure authentication, ensuring strong, token-based verification for both users and third-party services. ChargeMind also uses TLS/SSL encryption for its REST API to protect server-client communications, using secure cipher suites such as AES-256 and RSA with 2048bit keys. User credentials are kept in hashed form using the bcrypt method, which includes a salt mechanism for enhanced security and forward secrecy, prohibiting unauthorized access even if the database credentials are compromised. Moreover, ChargeMind protects sensitive user and operational data by ensuring its confidentiality, integrity, and availability via extensive encryption. Data is secured at rest with AES-256, and encryption keys are handled via a Key Management System (KMS) to ensure regular key rotation. ChargeMind also implements Multi-Factor Authentication (MFA) and enforces strong password standards to improve user access security. Role-Based Access Control (RBAC) restricts access to important system functions while ensuring that users and services follow their assigned permissions. Overall, ChargeMind effectively mitigates security risks while maintaining system integrity and availability by continuously implementing advanced cryptographic techniques, adhering to best practices in data security, and incorporating regular security testing.

Finally, the mobile application is highly optimized for state-of-the-art smart phones. For network connections, a smartphone should have 4G LTE or higher cellular protocol implemented in it, and a high precision GPS module is expected for determining drivers' location. The hardware prerequisites for these end-user devices are 8 GB of RAM, 2.5-3 GHz processor, and 64 GB of non-volatile storage.

5.3 System Design

This section contains the architectural design of the system, the problem description, the user interface design, and the hardware design. Additionally, it includes certain diagrams, such as Sequence Diagram, Activity Diagram, Data Flow Diagram, and Class Diagram.

5.3.1 Problem Description

ChargeMind aims to provide better charging experience for electric vehicle users by planning personalized routes, considering factors such as city, location, region, destinations to be visited before reaching the destination, cost optimization, possible duration of charging for specified range, and the selection of environmentally friendly or top-ranked stations. These factors are supported by the carbon emission tracking and station ranking features. Through the implementation of these features, it seeks to promote increased electric vehicle usage by delivering eco-friendly and autonomous solutions for problems like EVRP (Electric Vehicle Routing Problem), and Range Anxiety in the e-charging domain.

5.3.2 Architectural Design

To provide better comprehension of our architectural design, we explained the problem, technologies used and included numerous diagrams. As previously indicated, our system will perform three key roles. As a result, we will develop architecture that incorporates these responsibilities. Figure illustrates the fundamental structure of the ChargeMind System.

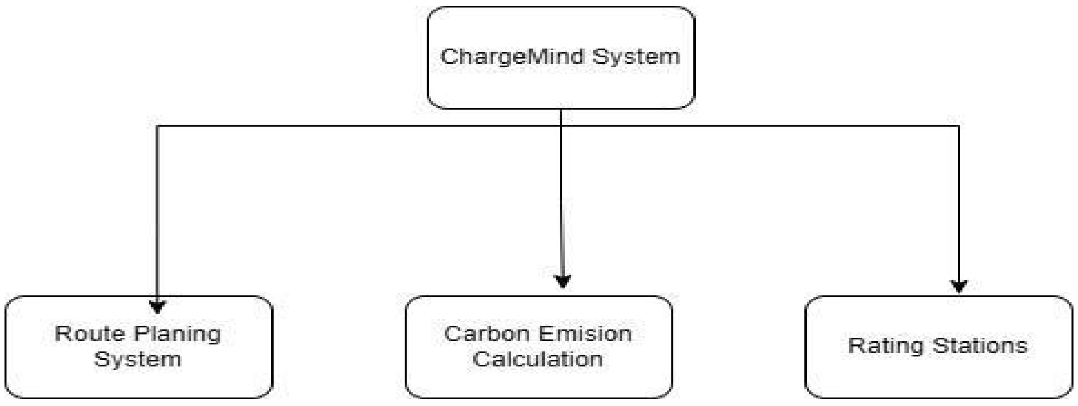


Figure 1: Brief of System Architectural Design

5.3.2.1 Data Flow Diagram

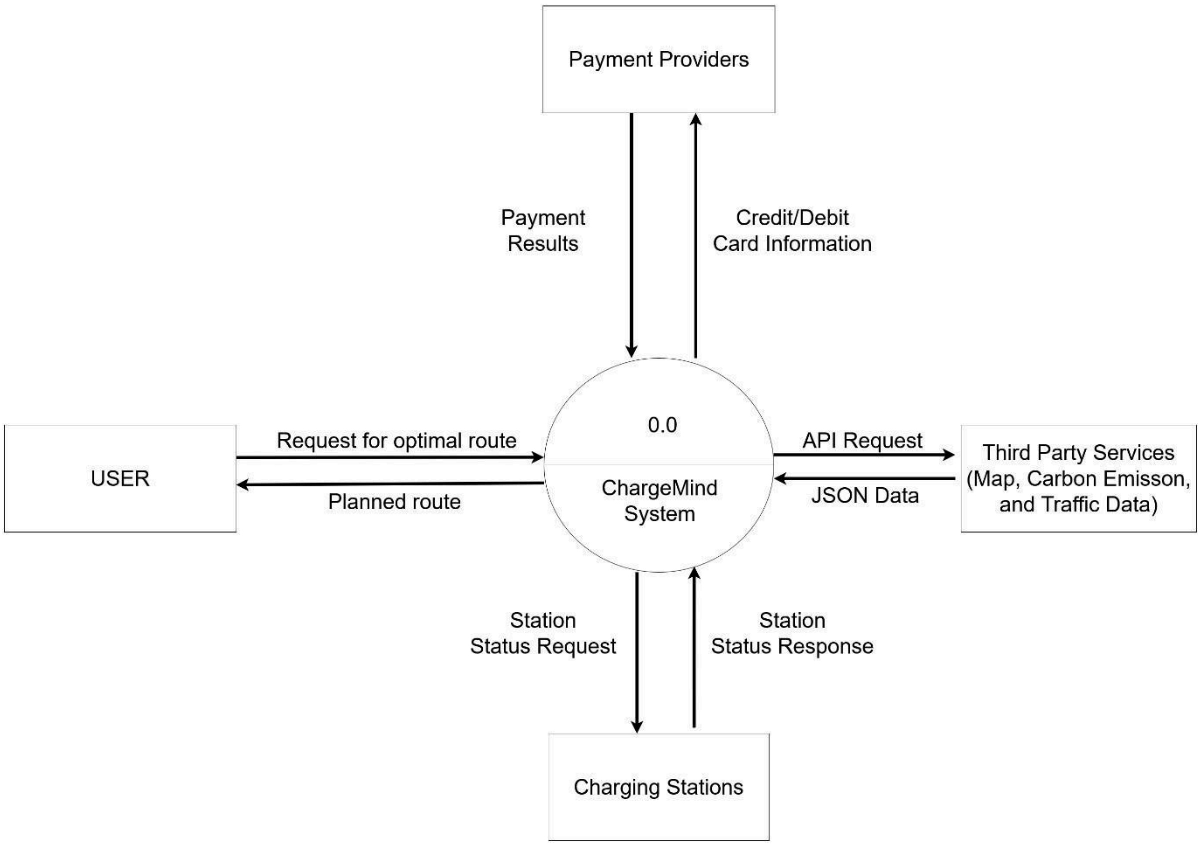


Figure 2: Context Diagram

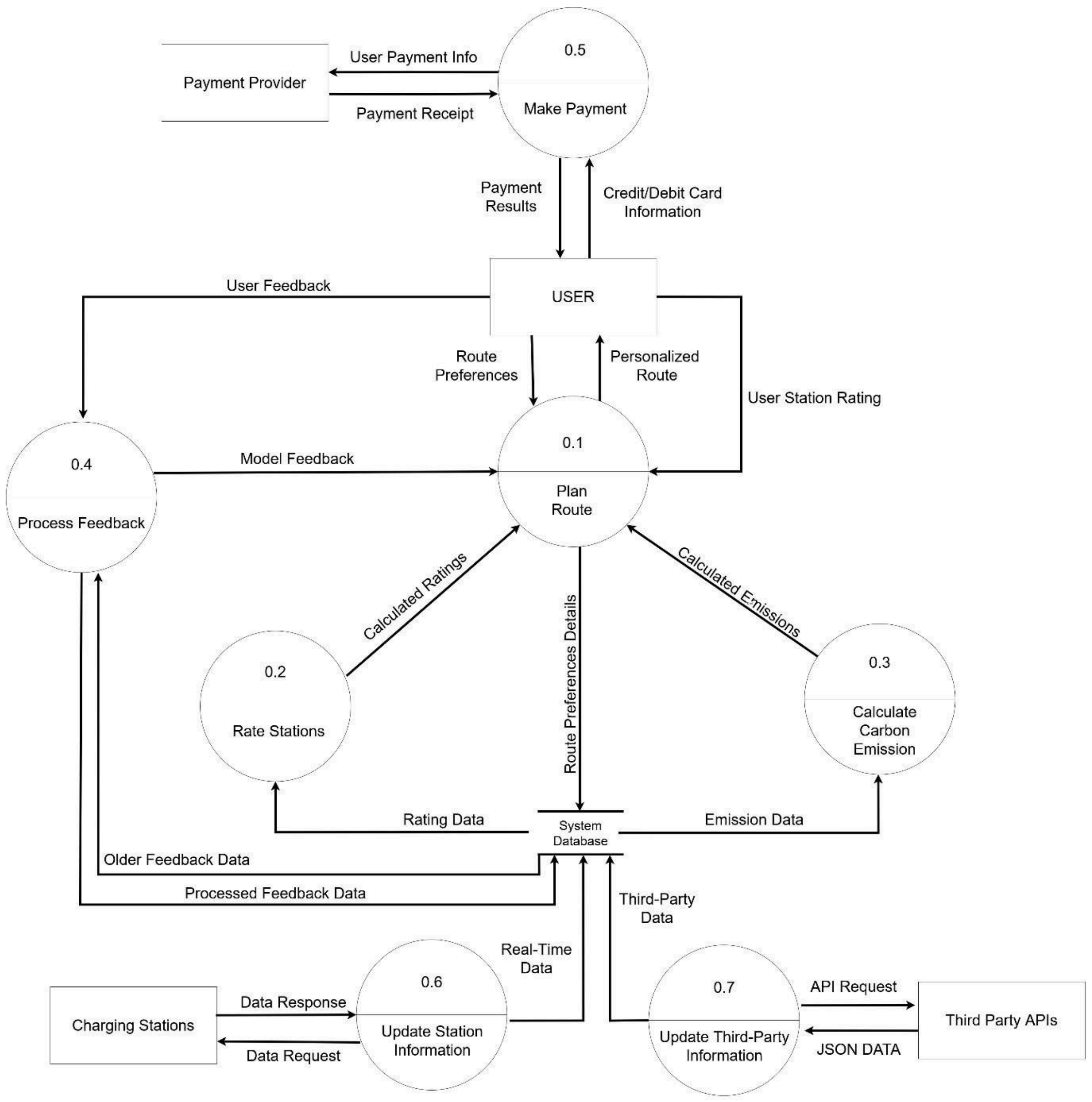


Figure 3: DFD Level - 1

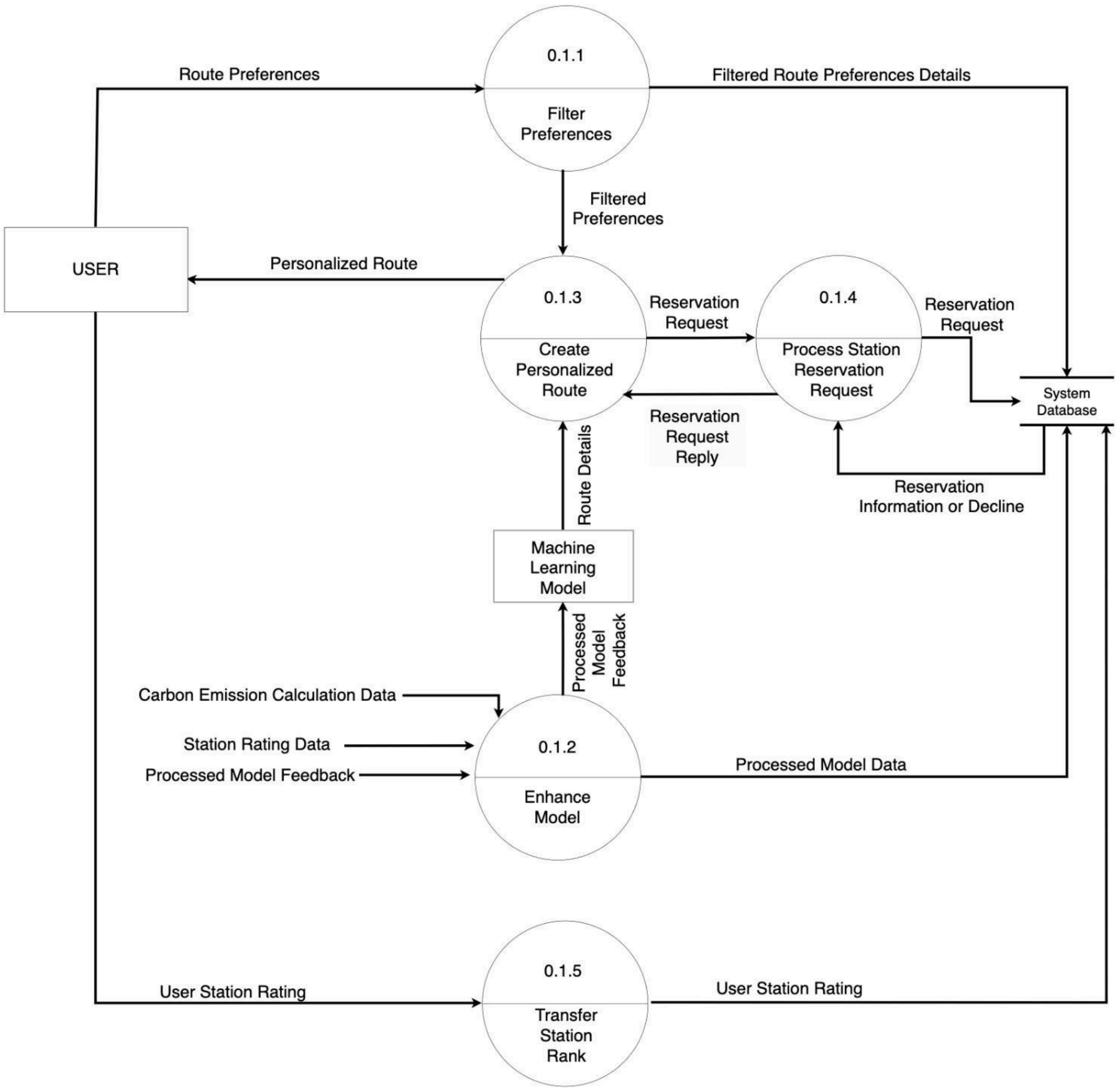


Figure 4: DFD Level - 2 for Personalized Route Creation

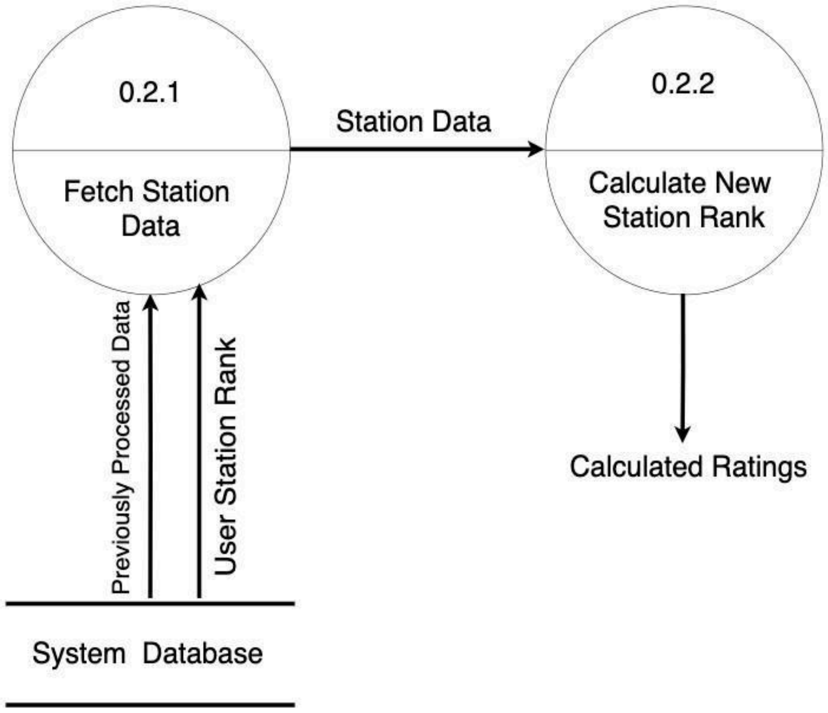


Figure 5: DFD Level - 2 for Station Rank Calculation

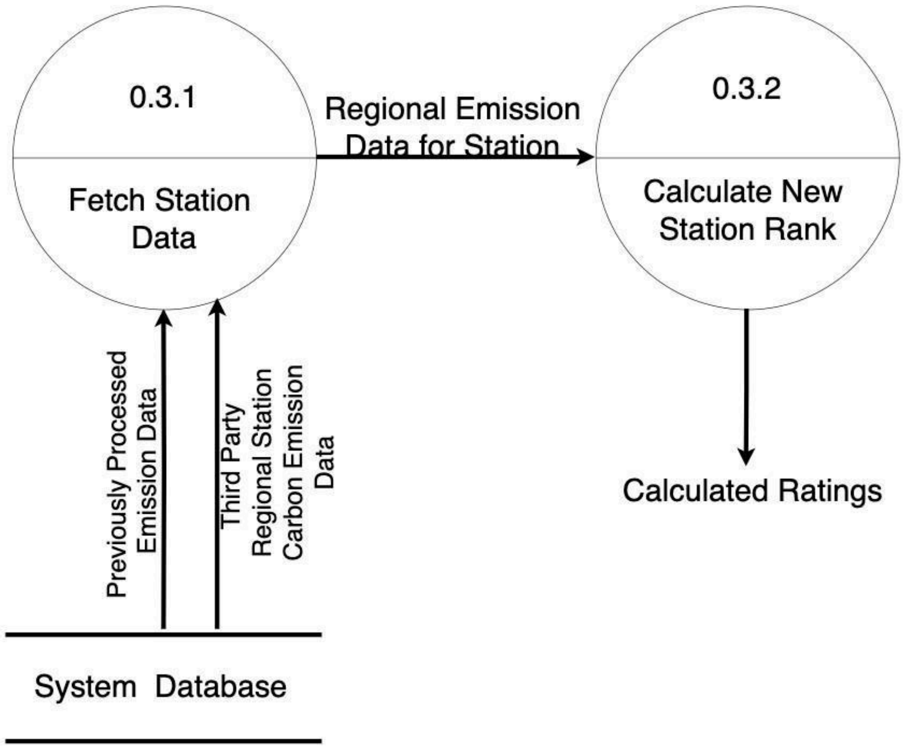

0.3.1
Fetch Station Data
Regional Emission Data for Station
0.3.2
Calculate New Station Rank
Calculated Ratings
Previously Processed Emission Data
Third Party Regional Station Carbon Emission Data
System Database

Figure 6: DFD Level - 2 for Carbon Emission Calculation

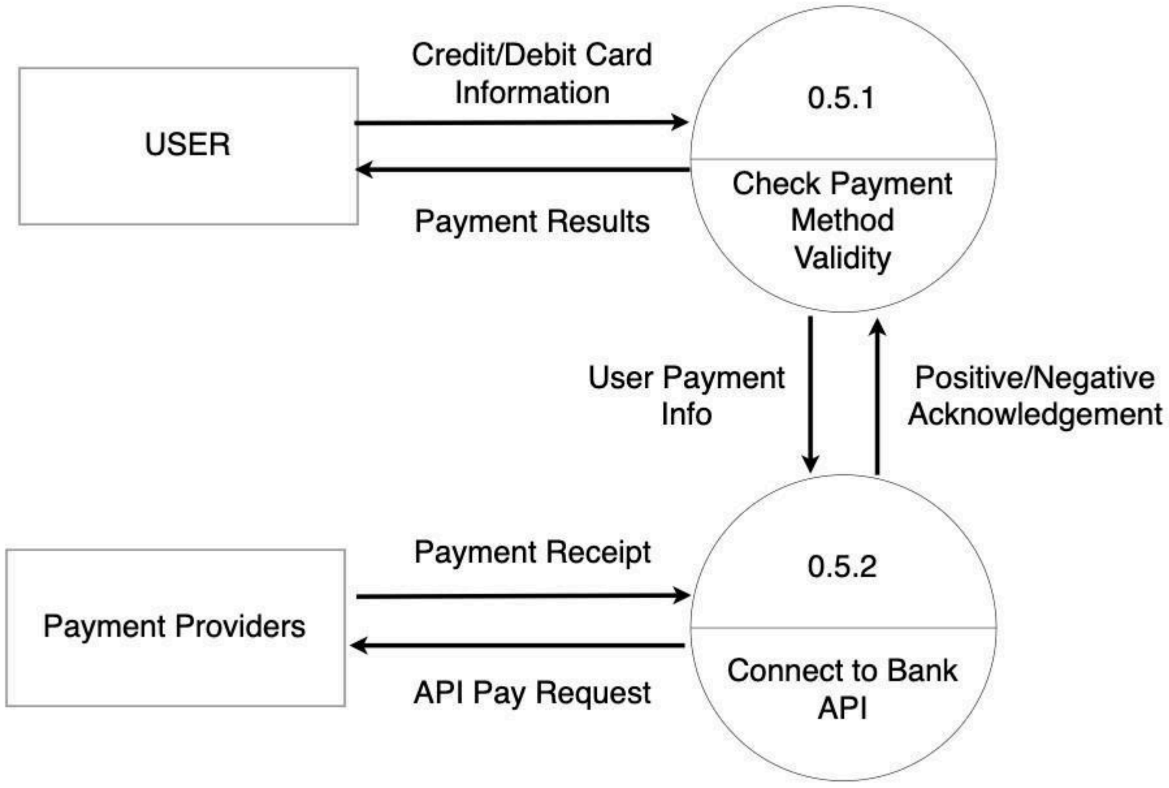


Figure 7: DFD Level - 2 for Payment

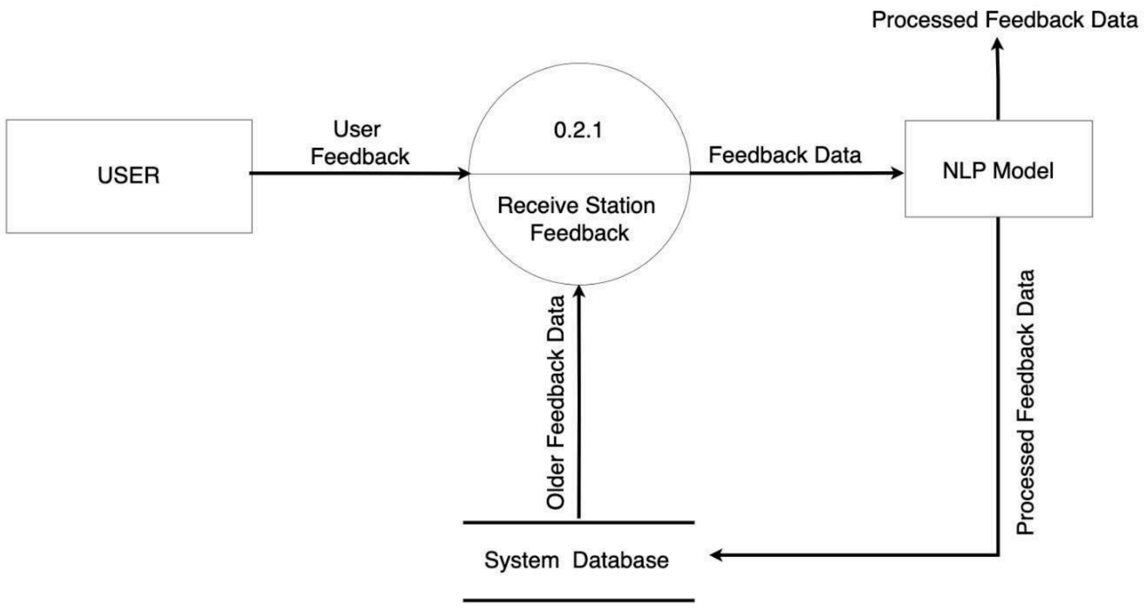


Figure 8: DFD Level - 2 for Feedback Processing

5.3.2.2 Activity Diagram

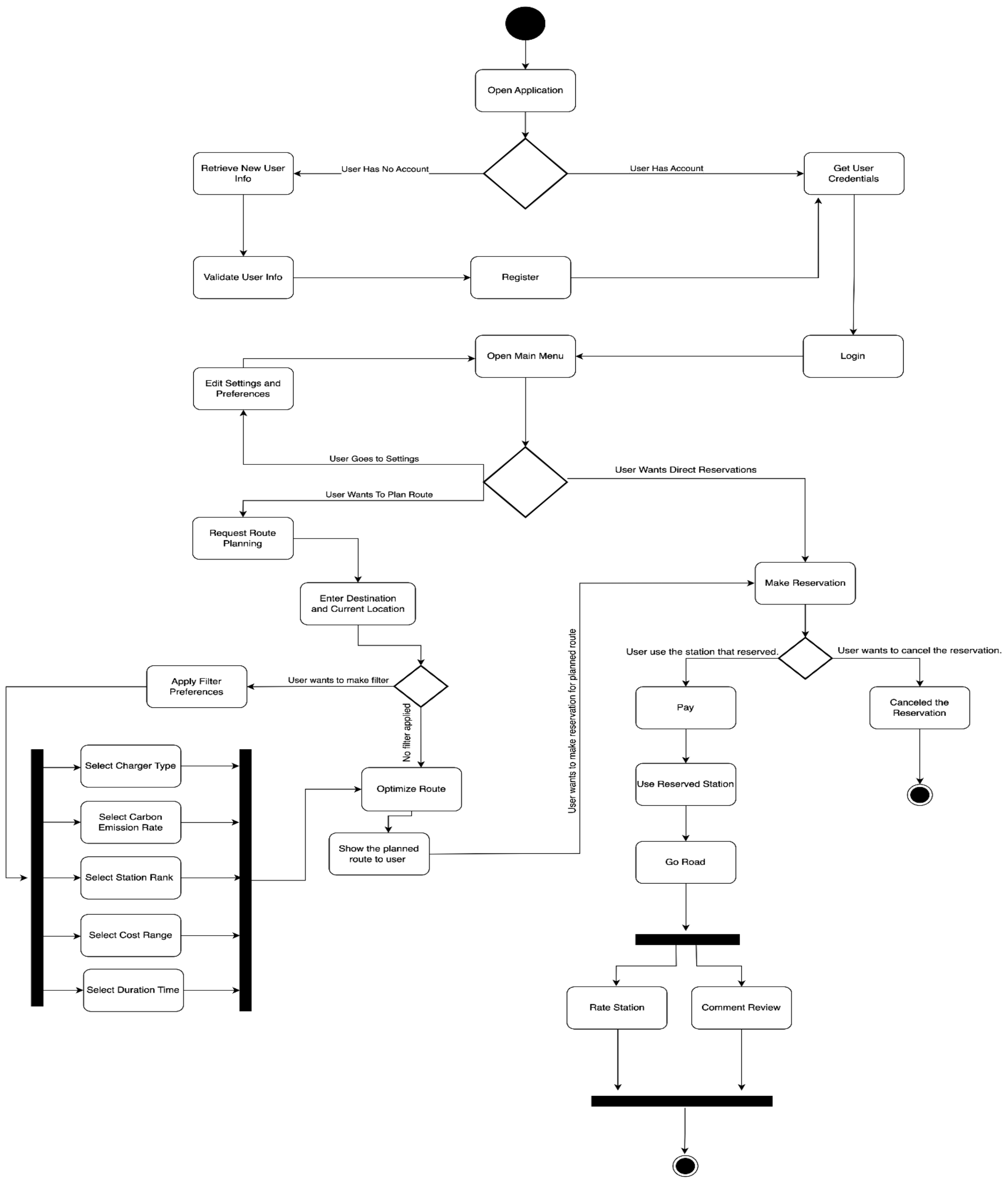
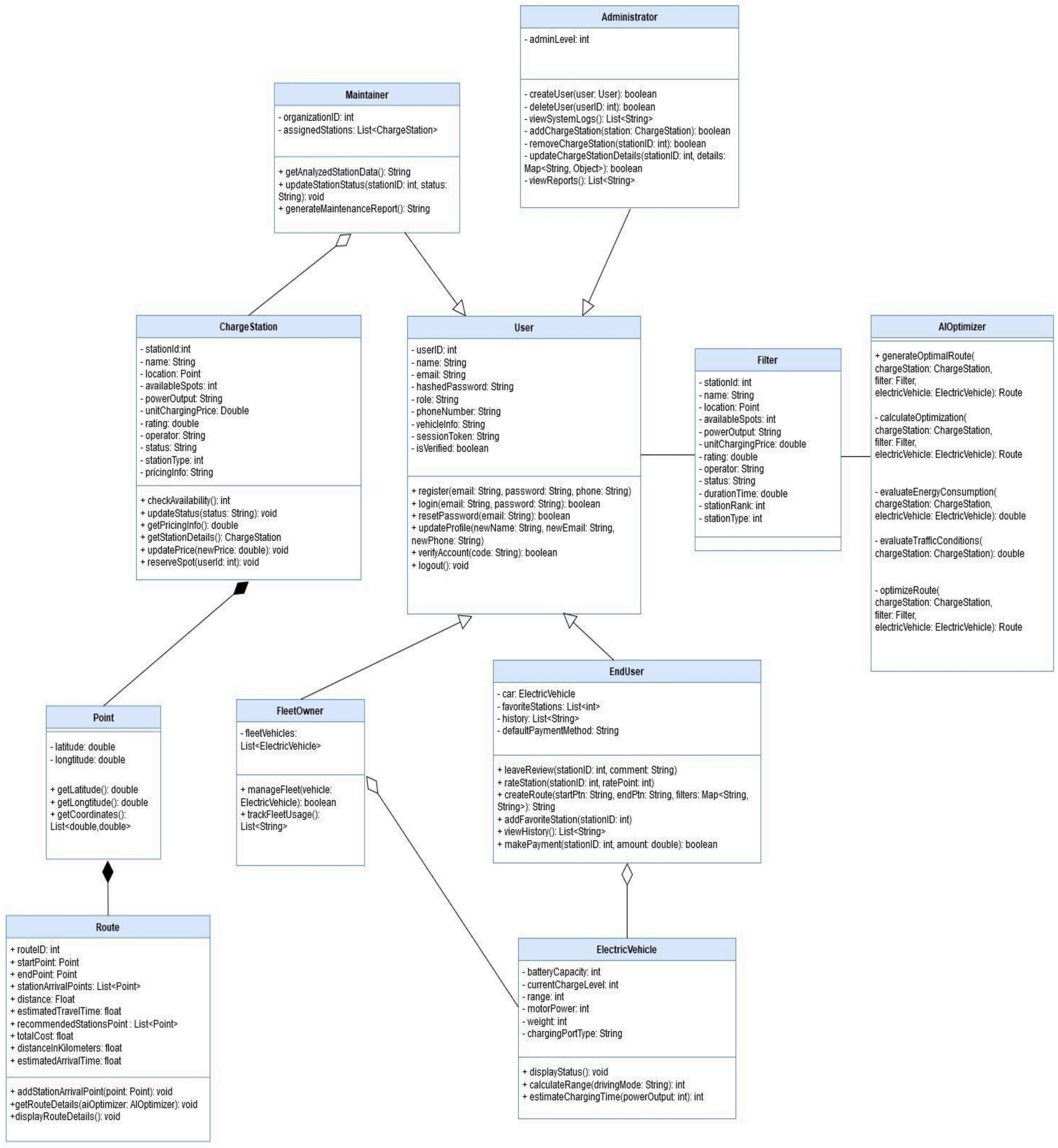


Figure 9: Activity Diagram for General System

5.3.2.3 Class Diagram



Administrator
- adminLevel: int
- createUser(user: User): boolean
- deleteUser(userID: int): boolean
- viewSystemLogs(): List<String>
- addChargeStation(station: ChargeStation): boolean
- removeChargeStation(stationID: int): boolean
- updateChargeStationDetails(stationID: int, details: Map<String, Object>): boolean
- viewReports(): List<String>
Maintainer
- organizationID: int
- assignedStations: List<ChargeStation>
+ getAnalyzedStationData(): String
+ updateStationStatus(stationID: int, status: String): void
+ generateMaintenanceReport(): String
ChargeStation
- stationId:int
- name: String
- location: Point
- availableSpots: int
- powerOutput: String
- unitChargingPrice: Double
- rating: double
- operator: String
- status: String
- stationType: int
- pricingInfo: String
+ checkAvailability(): int
+ updateStatus(status: String): void
+ getPricingInfo(): double
+ getStationDetails(): ChargeStation
+ updatePrice(newPrice: double): void
+ reserveSpot(userId: int): void
User
- userID: int
- name: String
- email: String
- hashedPassword: String
- role: String
- phoneNumber: String
- vehicleInfo: String
- sessionToken: String
- isVerified: boolean
+ register(email: String, password: String, phone: String)
+ login(email: String, password: String): boolean
+ resetPassword(email: String): boolean
+ updateProfile(newName: String, newEmail: String, newPhone: String)
+ verifyAccount(code: String): boolean
+ logout(): void
Filter
- stationId: int
- name: String
- location: Point
- availableSpots: int
- powerOutput: String
- unitChargingPrice: double
- rating: double
- operator: String
- status: String
- durationTime: double
- stationRank: int
- stationType: int
AIOptimizer
+ generateOptimalRoute(chargeStation: ChargeStation, filter: Filter, electricVehicle: ElectricVehicle): Route
- calculateOptimization(chargeStation: ChargeStation, filter: Filter, electricVehicle: ElectricVehicle): Route
- evaluateEnergyConsumption(chargeStation: ChargeStation, electricVehicle: ElectricVehicle): double
- evaluateTrafficConditions(chargeStation: ChargeStation): double
- optimizeRoute(chargeStation: ChargeStation, filter: Filter, electricVehicle: ElectricVehicle): Route
Point
- latitude: double
- longtitude: double
+ getLatitude(): double
+ getLongtitude(): double
+ getCoordinates(): List<double,double>
FleetOwner
- fleetVehicles: List<ElectricVehicle>
+ manageFleet(vehicle: ElectricVehicle): boolean
+ trackFleetUsage(): List<String>
EndUser
- car: ElectricVehicle
- favoriteStations: List<int>
- history: List<String>
- defaultPaymentMethod: String
+ leaveReview(stationID: int, comment: String)
+ rateStation(stationID: int, ratePoint: int)
+ createRoute(startPtn: String, endPtn: String, filters: Map<String, String>): String
+ addFavoriteStation(stationID: int)
+ viewHistory(): List<String>
+ makePayment(stationID: int, amount: double): boolean
Route
+ routeID: int
+ startPoint: Point
+ endPoint: Point
+ stationArrivalPoints: List<Point>
+ distance: Float
+ estimatedTravelTime: float
+ recommendedStationsPoint : List<Point>
+ totalCost: float
+ distanceInKilometers: float
+ estimatedArrivalTime: float
+ addStationArrivalPoint(point: Point): void
+getRouteDetails(aiOptimizer: AIOptimizer): void
+displayRouteDetails(): void
ElectricVehicle
- batteryCapacity: int
- currentChargeLevel: int
- range: int
- motorPower: int
- weight: int
- chargingPortType: String
+ displayStatus(): void
+ calculateRange(drivingMode: String): int
+ estimateChargingTime(powerOutput: int): int

Figure 10: Class Diagram for General System

5.3.2.4 Sequence Diagrams

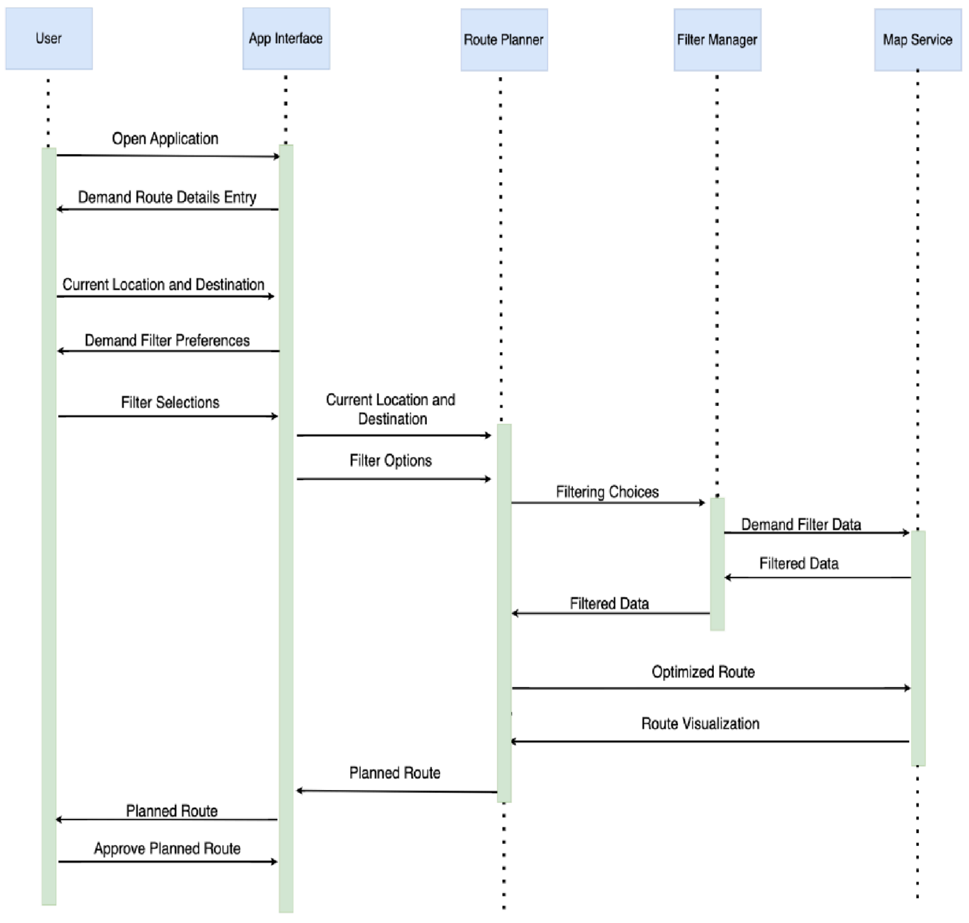


Figure 11: Sequence Diagram for Route Planning

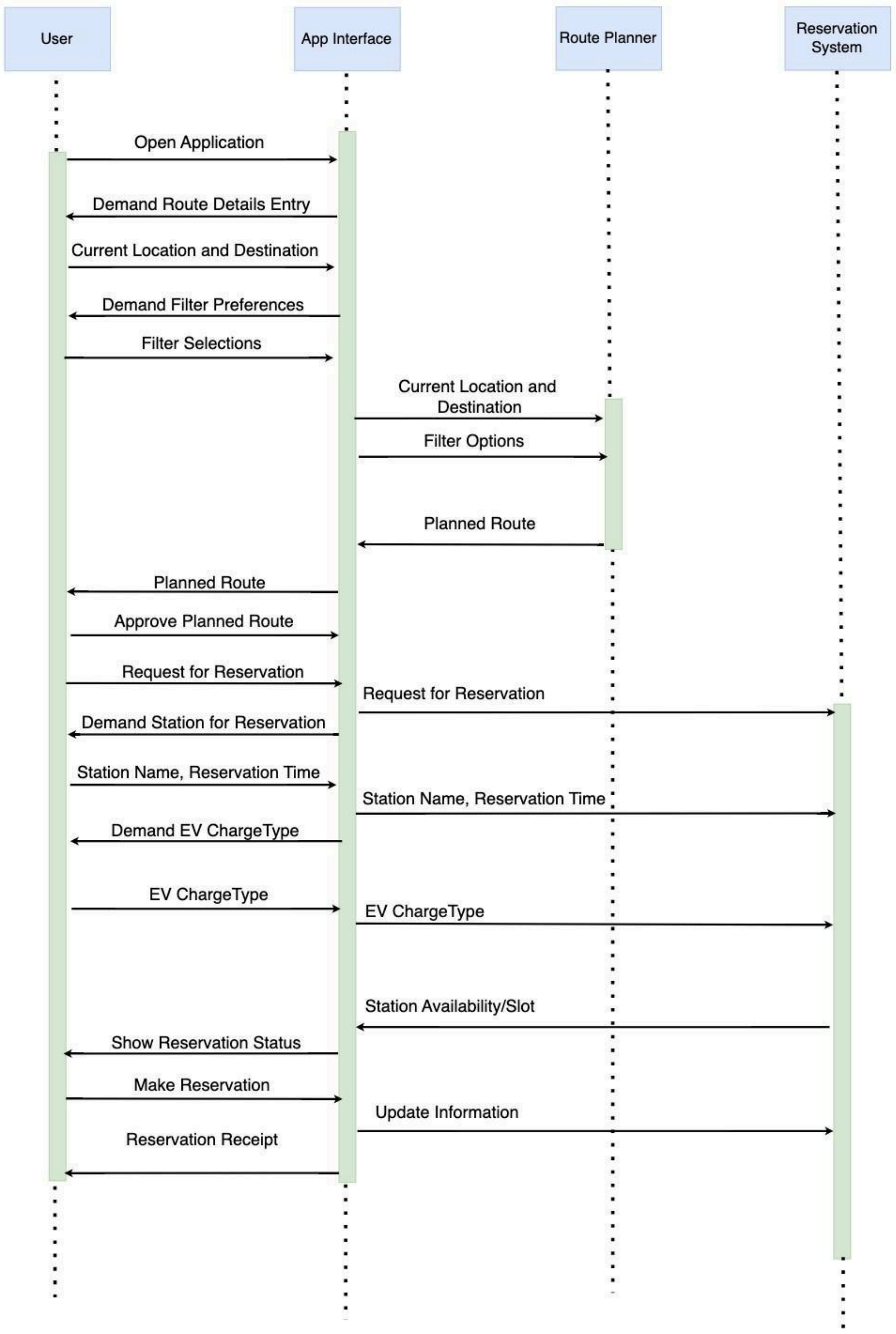


Figure 12: Sequence Diagram for Reservation System

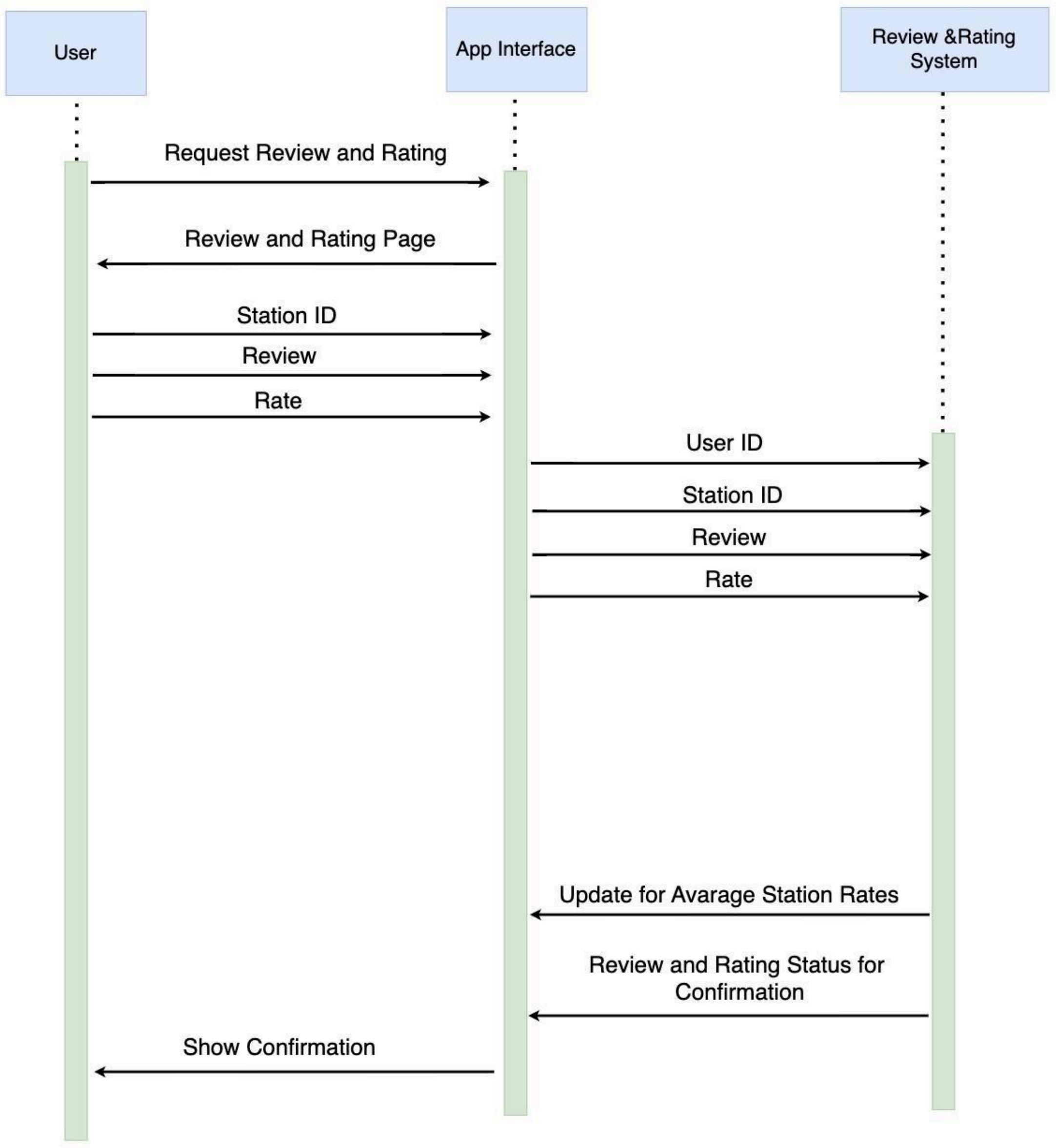


Figure 13: Sequence Diagram for Review and Rating Systems

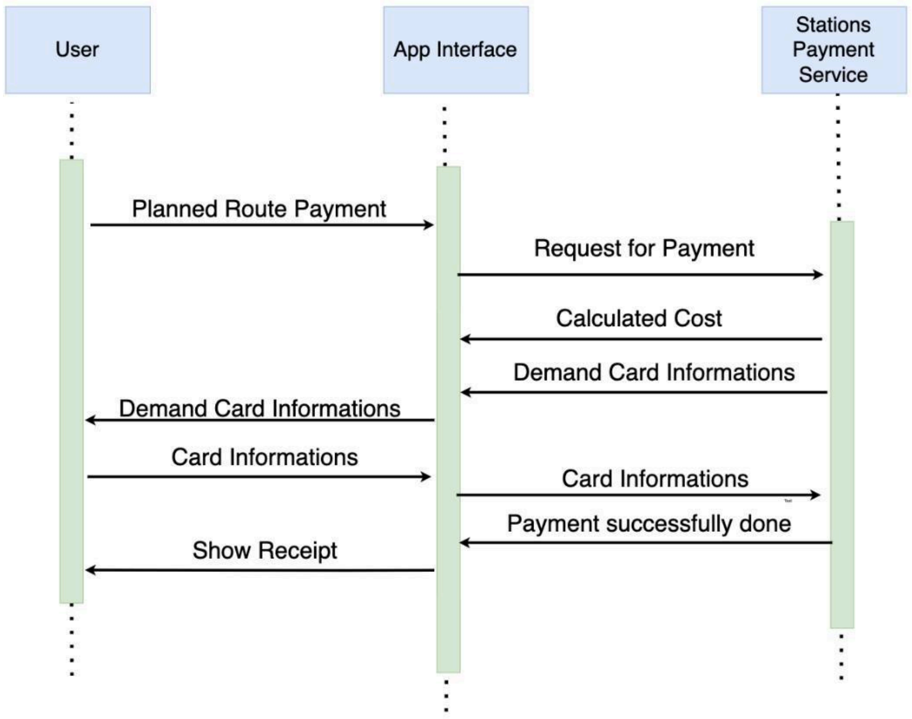


Figure 14: Sequence Diagram for Payment Service

5.4 User Interface Design

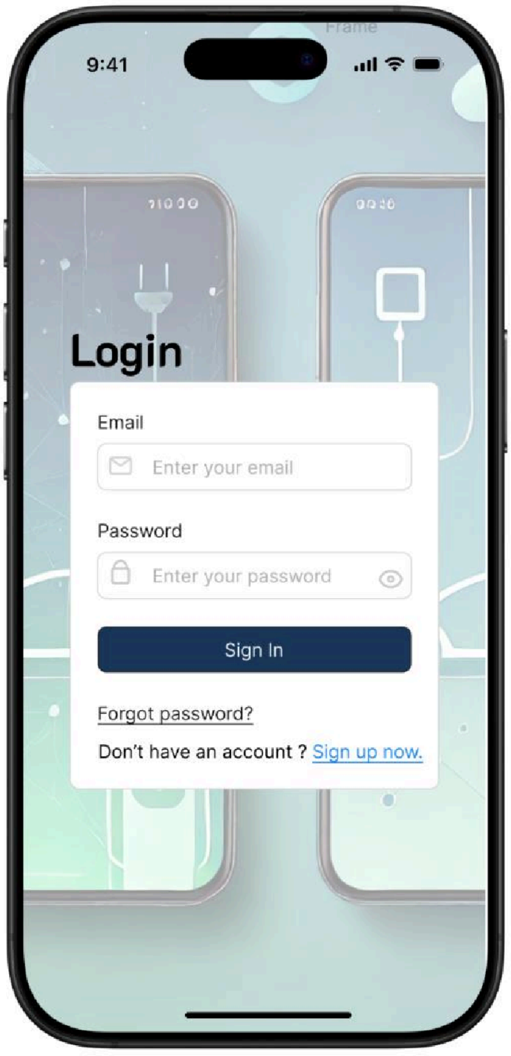


Figure 15: UI Design-1 for Login Page

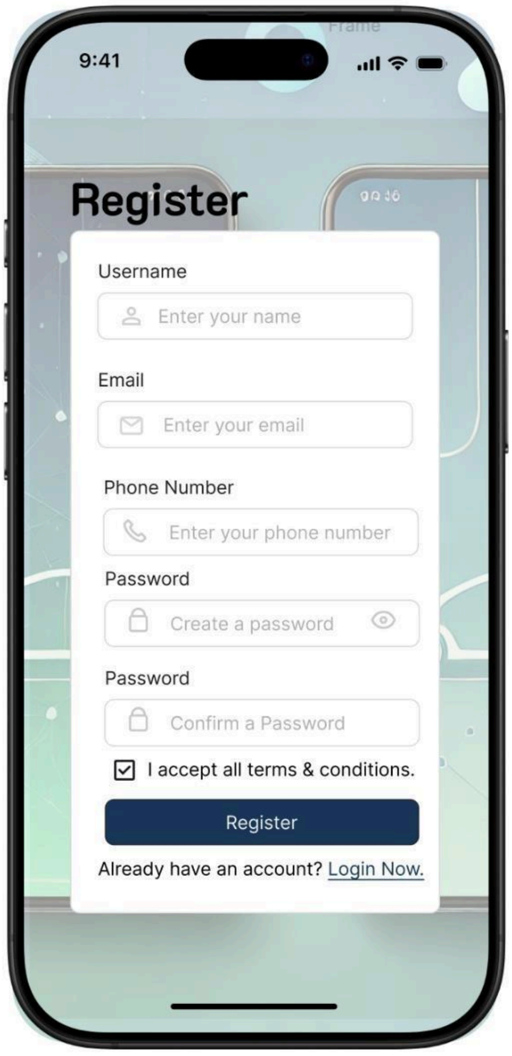


Figure 16: UI Design-2 for Register Page

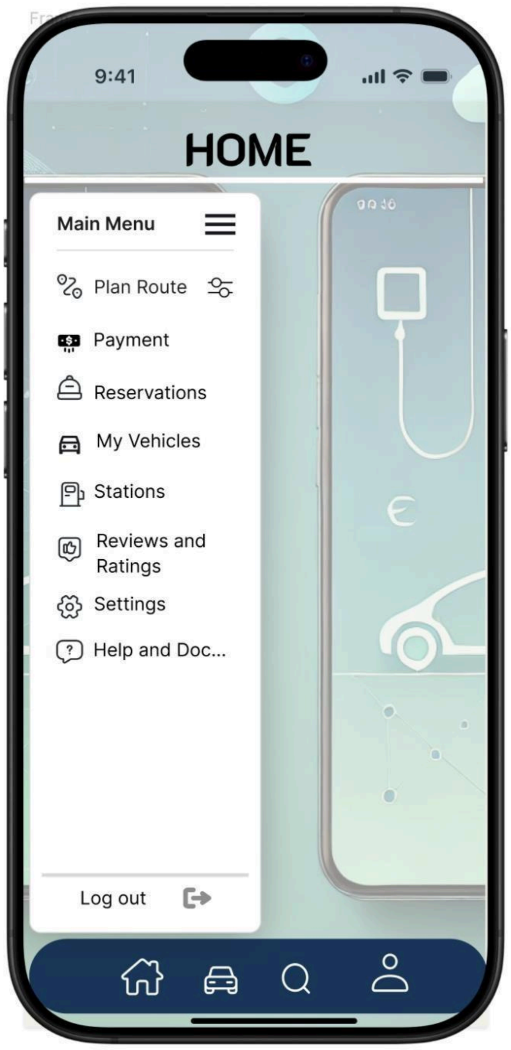


Figure 17: UI Design-3 for Home Page

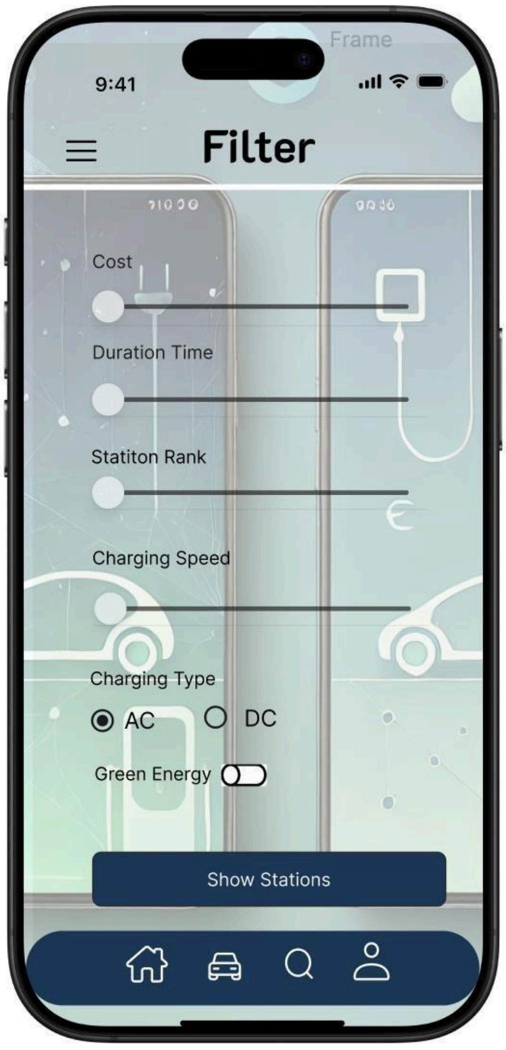
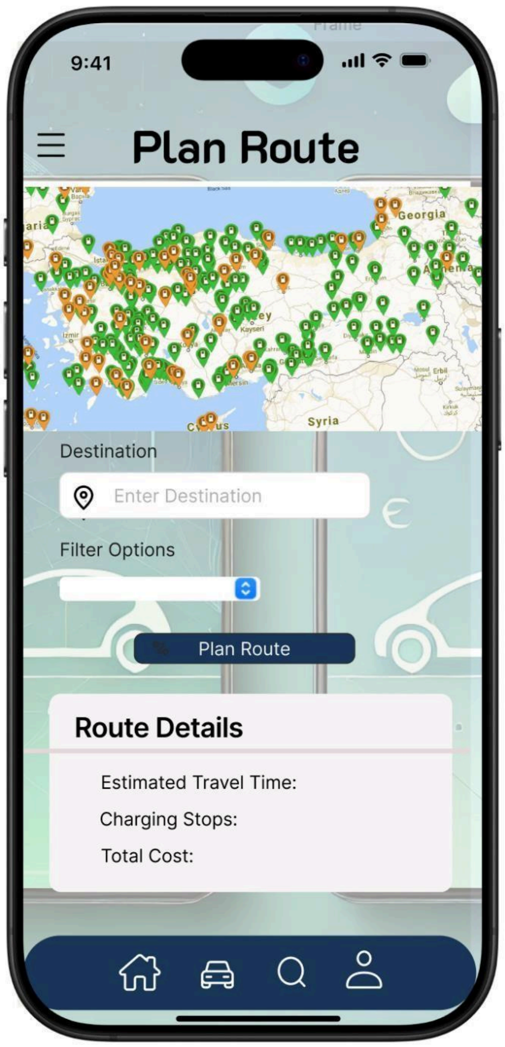


Figure 18: UI Design-4 for Filter Page



9:41

Plan Route

Destination

Enter Destination

Filter Options

Plan Route

Route Details

Estimated Travel Time:

Charging Stops:

Total Cost:

Figure 19: UI Design-5 for Plan Route Page

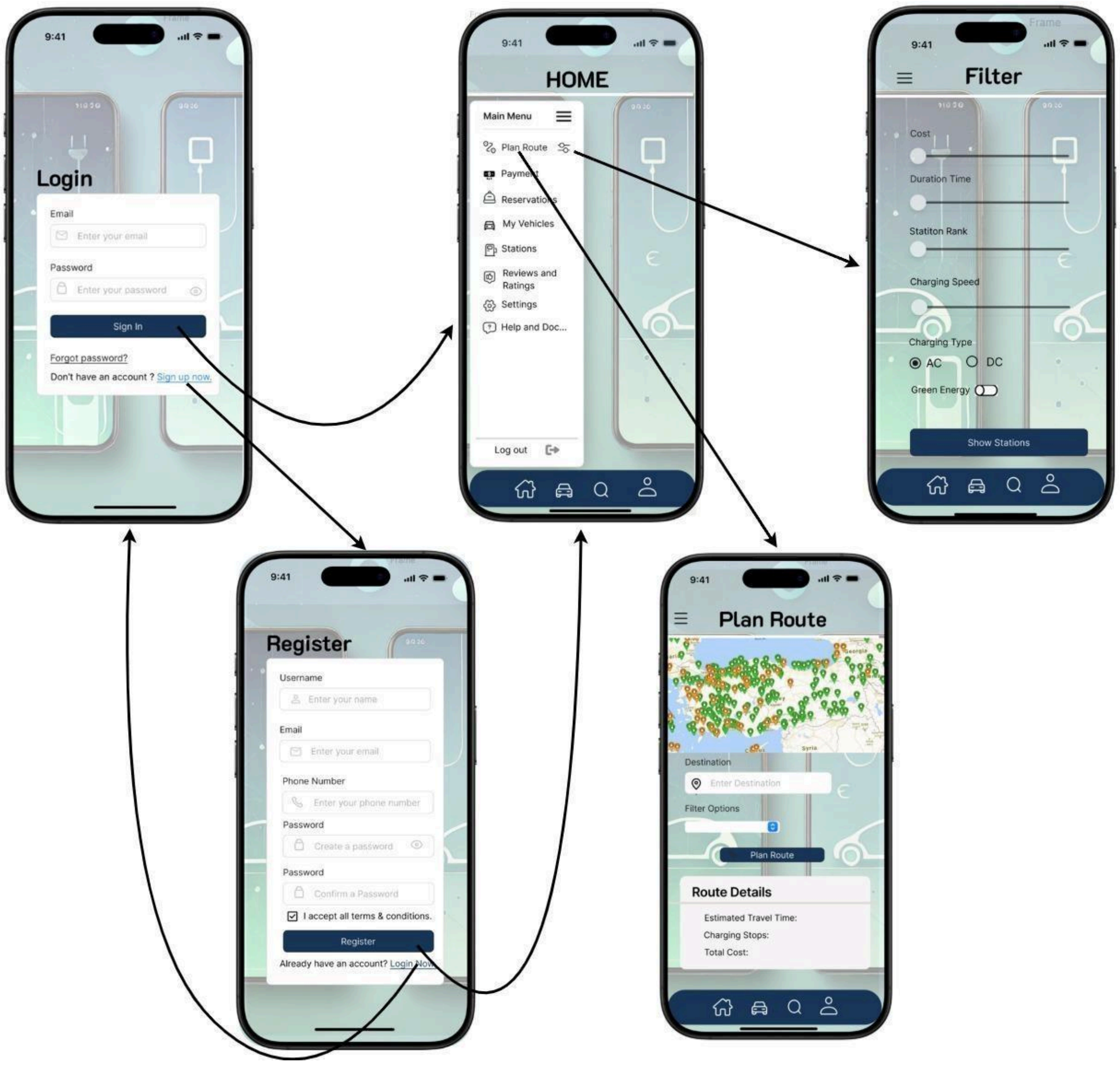


Figure 20: UI Navigation Schema

5.5 Requirements Matrix

Requirement Description	Design Component	Verification Method
User registration and login	Frontend (React Native)	Unit Testing, UI Testing
Route planning with station availability	Backend (Java/Spring Boot), ML Model	Integration Testing
Real-time station status retrieval	Backend, API Integration	System Testing, API Testing
Secure payment processing	Backend, Payment Gateway	Security Testing, Penetration Testing, Functional Test
Response time under 2 seconds	Backend, Load Balancer	Performance Testing
High availability (99.7% uptime)	Redundant Servers, Failover	Stress Testing, Monitoring
Data encryption during transmission	TLS/SSL, OAuth 2.0	Penetration Testing
User-friendly interface	Frontend (React Native)	Usability Testing
Compatibility with Android and iOS devices	Frontend	Compatibility Testing

6. Conclusion

The ChargeMind represents a major step in enhancing the e-charging ecosystem with a better user experience and AI-based approach. In our literature review, we have identified key trends, challenges, and existing solutions that are done by other applications within the domain. Additionally, the work plan outlines a clear roadmap, guaranteeing that the project's milestones can be reached within the allotted time. The System Requirements Specification (SRS) document establishes a foundation for the functional and non-functional requirements, ensuring that all critical user needs and technical limitations are taken into account. The System Design Document (SDD) converts the requirements into architectural design and allows for a reliable implementation.

Our main goal is to improve customer satisfaction and support sustainable transportation infrastructure for e-charge vehicles, and these preparation activities together offer a thorough framework for creating a dependable, effective, and creative e-charging solution.

7. References

[1] "PlugShare," Wikimedia Foundation, Inc, [Online]. Available: https://en.wikipedia.org/wiki/PlugShare. [Accessed: 08 November 2024].

[2] "PlugShare – EV Charging Station Payment," PlugShare, Inc, [Online]. Available: https://pay.plugshare.com/. [Accessed: 08 November 2024].

[3] "Lixhium," [Online]. Available: https://lixhium.com/. [Accessed: 08 November 2024].

[4] "Lixhium Mobility Innovation Summit Presentation," YouTube, [Online]. Available: https://www.youtube.com/watch?v=EY7ddAO-8bY&t=3s. [Accessed: 08 November 2024].

[5] "Lixhium Pro App," Google Play, [Online]. Available: https://play.google.com/store/apps/details?id=com.lixhium.lixhiumproapp&hl=tr&pli=1. [Accessed: 08 November 2024].

[6] "Open Charge Map," [Online]. Available: https://openchargemap.org/site. [Accessed: 08 November 2024].

[7] "Voltla," [Online]. Available: https://voltla.com.tr/. [Accessed: 08 November 2024].

[8] "ChargePrice," [Online]. Available: https://www.chargeprice.net/. [Accessed: 08 November 2024].

[9] R. Shahbazian, L. D. Pugliese, F. Guerriero, and G. Macrina, "Integrating machine learning into vehicle routing problem: Methods and applications" [Accessed: 08 November 2024].

[10] European Environment Agency. (n.d.). *Greenhouse gas emissions from transport*. European Environment Agency. Available: https://www.eea.europa.eu/en/analysis/indicators/greenhouse-gas-emissions-from-transport

[11] Hu, Y., Xie, T., Chi, N., & Yang, Y. (2023). Design of energy big data and carbon emission monitoring system based on the perceptron model in the context of carbon neutral and carbon peaking. *State Grid Sichuan Electric Power Company Tianfu New District Power Supply Company*, Chengdu, Sichuan, China. Received November 3rd, 2022; Accepted March 8th, 2022; Available online August 28th, 2023.

[12] "Electric vehicle", *Wikipedia.* Available: https://en.wikipedia.org/wiki/Electric_vehicle (accessed Dec. 4, 2024).

[13] *ScienceDirect*. Available: https://www.sciencedirect.com/topics/computer-science/non-volatile-memory (accessed Dec. 1, 2024).

[14] "RESTful API", *TechTarget*. Available: https://www.techtarget.com/searchapparchitecture/definition/RESTful-API (accessed Dec. 2, 2024).

[15] *Turhost*. Available: https://www.turhost.com/ (accessed Dec. 3, 2024).

[16] *Wikipedia*. Available: https://www.wikipedia.org/ (accessed Dec. 2, 2024).

[17] "What is SSL/TLS?", *Amazon Web Services (AWS)*. Available: https://aws.amazon.com/tr/what-is/ssl-certificate/] (accessed Dec. 1, 2024).

[18] "Internet protocol suite", *Wikipedia*. Available: https://en.wikipedia.org/wiki/Internet_protocol_suite. (accessed Dec. 1, 2024).

[19] "What is MFA?", *Amazon Web Services (AWS).* Available: https://aws.amazon.com/tr/whatis/mfa/ (accessed Dec. 2, 2024).

[20] "What is SSO?", *Amazon Web Services (AWS).* Available: https://aws.amazon.com/tr/whatis/sso/. (accessed Dec. 2, 2024).

[21] *ReactNative*. Available: https://reactnative.dev/ (accessed Dec. 2, 2024).

[22] "Advanced Encryption Standard (AES)", *TechTarget.* Available: https://www.techtarget.com/searchsecurity/definition/Advanced-Encryption-Standard#:~:text=The%20Advanced%20Encryption%20Standard%20(AES)%20is%20a%20symmetric%20block%20cipher,world%20to%20encrypt%20sensitive%20data. (accessed Dec. 2, 2024).

[23] "HMAC", *Wikipedia*. Available: https://en.wikipedia.org/wiki/HMAC (accessed Dec. 1, 2024).

[24] "Institute Of Electrical And Electronics Engineers", *IEEE Recommended Practice for Software Requirements Specifications*. New York: IEEE, 1998. (accessed Dec. 3, 2024).

[25] "Jira, Atlassian", *Atlassian.* Available: https://www.atlassian.com/software/jira (accessed Dec. 2, 2024).

[26] "JPype documentation — JPype 1.5.2.dev0 documentation", *ReadTheDocs.* Available: https://jpype.readthedocs.io/en/latest/. (accessed Dec. 1, 2024).

[27] "Welcome to Py4J — Py4J", *Py4J*. Available: https://www.py4j.org/ (accessed Dec. 1, 2024).

[28] "*PyTorch*", *PyTorch*. Available: https://pytorch.org/ (accessed Dec. 4, 2024).

[29] "TensorFlow", *TensorFlow*. Available: https://www.tensorflow.org/ (accessed Dec. 4, 2024).

[30] "Load balancing (computing)", *Wikipedia*. Available: https://en.wikipedia.org/wiki/Load_balancing_(computing) (accessed Dec. 4, 2024).

[31] "What are Microservices?", *Amazon Web Services (AWS).* Available: https://aws.amazon.com/microservices/ (accessed Dec. 3, 2024).

[32] "What is hashing?," *Built In.* Available: https://builtin.com/articles/what-ishashing#:~:text=What%20Is%20Hashing%3F-,Hashing%20is%20the%20practice%20of%20transforming%20a%20given%20key%20or,are %20very%20difficult%20to%20decode. (Accessed Dec. 23, 2024).

[33] "What is Java?," *Amazon Web Services (AWS)*. Available: https://aws.amazon.com/whatis/java/?nc1=h_ls. (Accessed Dec. 24, 2024).

[34] "Introduction to NoSQL," *GeeksforGeeks*. Available: https://www.geeksforgeeks.org/introduction-to-nosql/. (Accessed Dec. 22, 2024).

[35] "React Native," *Netguru*. Available: https://www.netguru.com/glossary/react-native. (Accessed Dec. 23, 2024).

[36] "What is Java Spring Boot?," *Microsoft Azure*. Available: https://azure.microsoft.com/enus/resources/cloud-computing-dictionary/what-is-java-spring-boot#:~:text=Java%20Spring%20Boot%20is%20an,computing%20platforms%20for%20app %20development. (Accessed Dec. 24, 2024).

[37] “What is SSL?,” *Cloudflare*. Available: https://www.cloudflare.com/learning/ssl/what-isssl/. (Accessed Dec. 23, 2024).

[38] “What is TensorFlow?,” *NVIDIA*. Available: https://www.nvidia.com/enus/glossary/tensorflow/. (Accessed Dec. 25, 2024).

[39] “Transport Layer Security (TLS),” *Cloudflare*.Available: https://www.cloudflare.com/learning/ssl/transport-layer-security-tls/.(Accessed Dec. 27, 2024).