ÇANKAYA UNIVERSITY

Software Requirements Specification

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

Lorin Melek Vural - 202011035, Selin Deniz - 202011051, Hanife Müge Karakaya - 202011044, Meleksu Özdoğan - 202011054, Kaan Baydemir - 202011070

06/12/2024

Table of Contents

1. Introduction

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.

1.2 Scope of Project

Electric Vehicle [1] 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.

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.[2]
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.[3]
JSON	A stand-alone data transfer format called JSON was created to express basic data structures.[4]
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. [5]
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).[6]
TCP/IP	The protocol suite of the internet, is a framework used for organizing the set of communication protocols.[7]
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. [8]
SSO	SSO is an authentication solution that allows users to sign in to multiple applications and websites with one-time authentication.[9]
React Native	A JavaScript Framework used to create cross-platform applications.[10]
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.[11]
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.[12]
RBAC	According to their responsibilities, users' authorizations and system access are managed using this access control model.

1.4 Definitions, Acronyms, and Abbreviations

- **SRS:** Software Requirements Specification
- **GPS:** Global Positioning System
- **IEEE**: Institute of Engineers and Everyone Else
- **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

1.5 References

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

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

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

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

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

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

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

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

[9] “What is SSO?”, *Amazon Web Services (AWS)*. Available: https://aws.amazon.com/tr/what-is/sso/. (accessed Dec. 2, 2024).

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

[11] “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).

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

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

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

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

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

[17] “*PyTorch*”, *PyTorch*. Available: https://pytorch.org/ (accessed Dec. 4, 2024).

[18] “TensorFlow”, *TensorFlow*. Available: https://www.tensorflow.org/ (accessed Dec. 4, 2024).

[19] “Load balancing (computing)”, *Wikipedia*. Available:

https://en.wikipedia.org/wiki/Load_balancing_(computing) (accessed Dec. 4, 2024).

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

1.6 Document Conventions

The document adheres to standard naming conventions, terminology, and formatting guidelines specified by IEEE Software Requirement Specification document [13] standards for clarity and consistency.

1.7 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.

2. Overall Description

2.1 Product Perspective

ChargeMind: AI-Based E-Charge Management and Planning Application is a smart solution for optimizing and planning electric vehicle [5] (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.

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 [14]. 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.

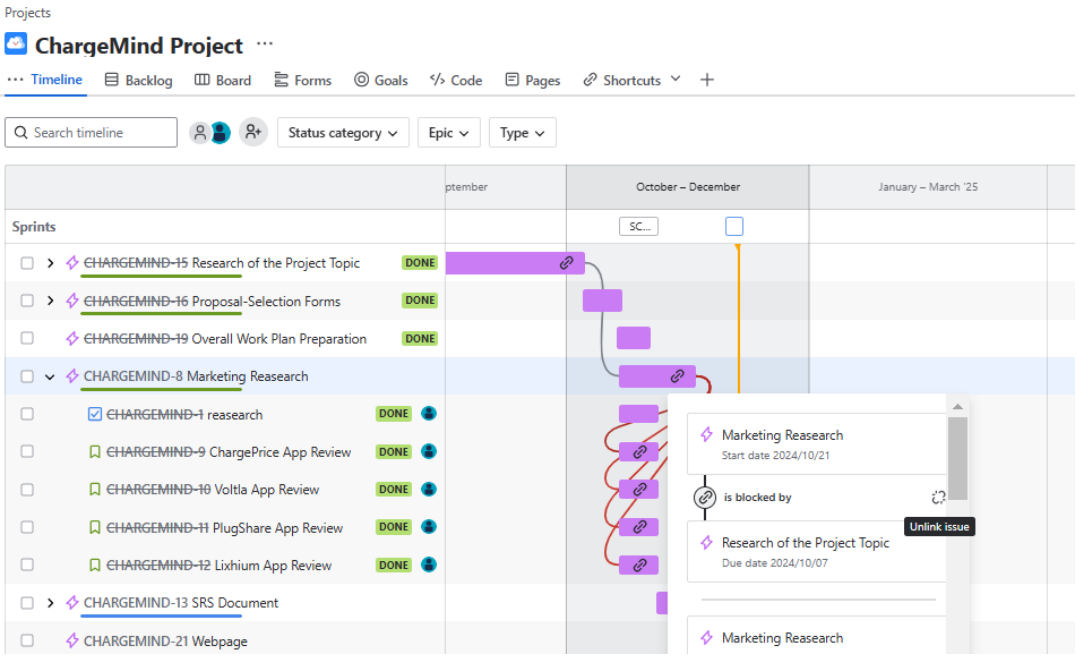


Figure 1 A Part of the Project on the Timeline Table

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.

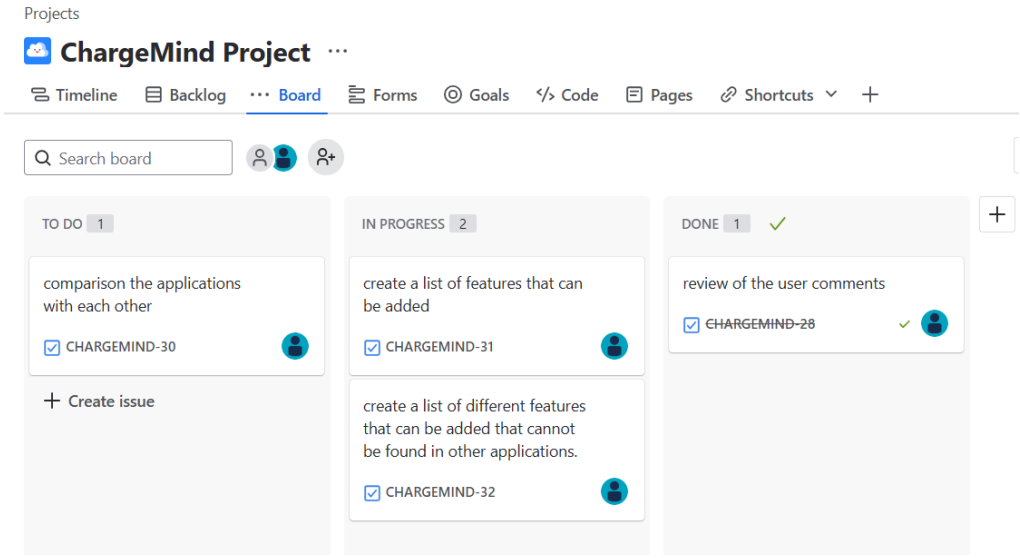


Figure 2 Marketing Research Sprint Backlogs on the Kanban board

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.

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.

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.

3.1 External Interfaces

3.1.1 User Interface

3.1.1.1 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.

3.1.1.2 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.

3.1.1.3 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.

3.1.1.4 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.

3.1.1.5 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.

3.1.1.6 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.

3.1.1.7 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.

3.1.1.8 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.

3.1.1.9 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.

3.1.1.10 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.

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.

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 [15] or Py4J [16], 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 [17] or TensorFlow [18]. 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.

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, real-time 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.

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 Authenticator)
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 two-factor 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.

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.

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.

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.

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.

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.

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 [19] techniques will be used to meet increasing user demands. Thus, the traffic of increasing user demands is managed.

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.

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.

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.

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.

3.4.2 Fault Tolerance and Error Handling

ChargeMind uses a micro/macroservices [20] 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.

3.4.3 Continuous Testing

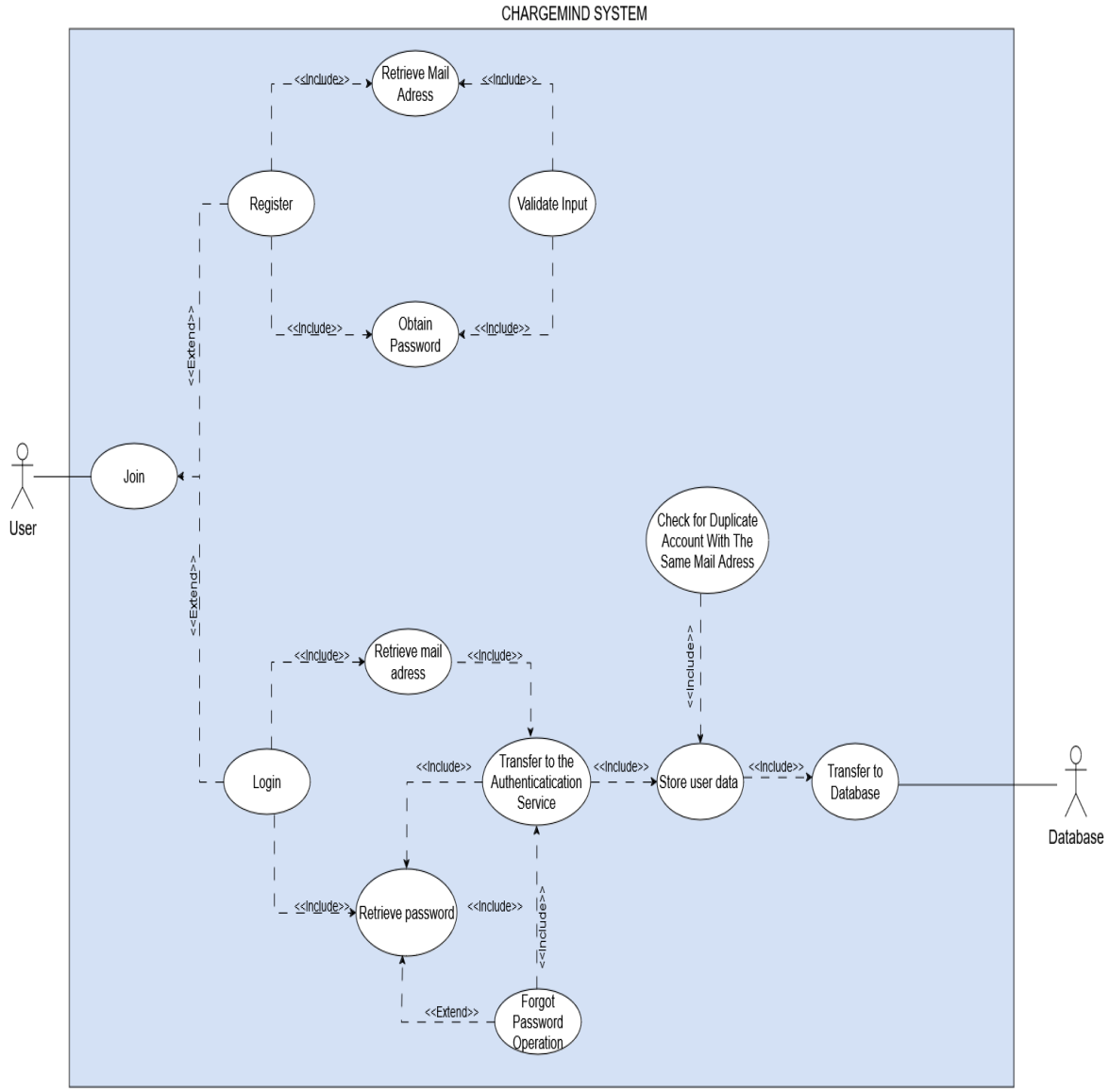
ChargeMind develops a comprehensive security testing strategy, which includes unit testing, integration testing, and end-to-end security testing, to verify the system's resilience against potential threats. Critical components such as user authentication, data encryption, and access control methods are 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. End-to-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. Use Case Diagrams

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.

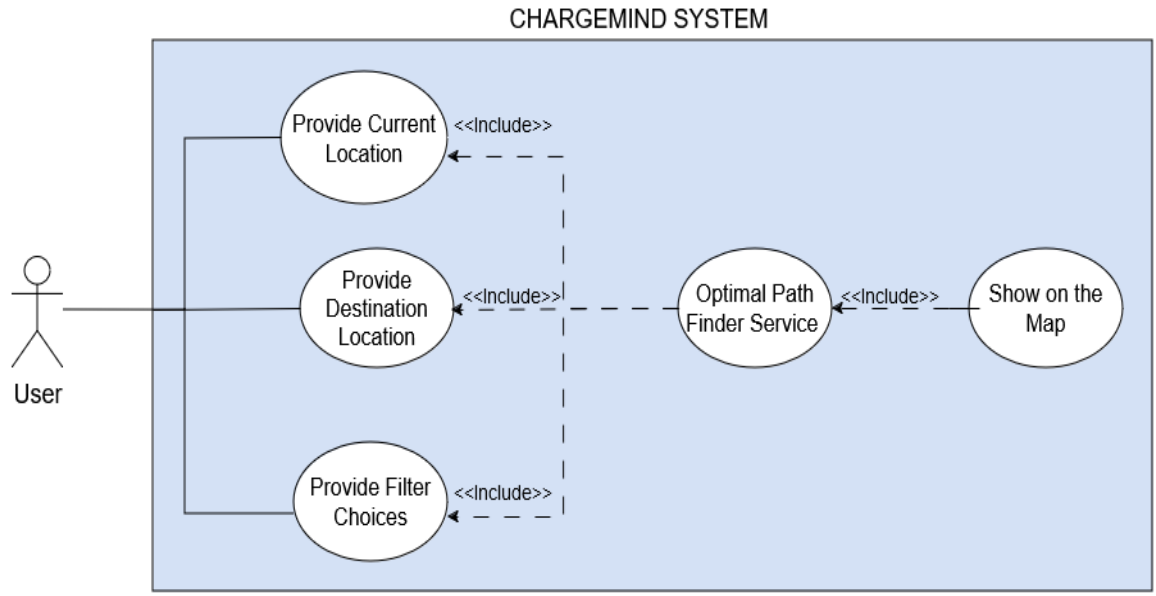
	2.If the password requirements are not met, the user is requested for a new password. 3.If the user tries to register with an already registered e-mail address, they are informed that the e-mail address already has an account.



4.2 UC-2

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

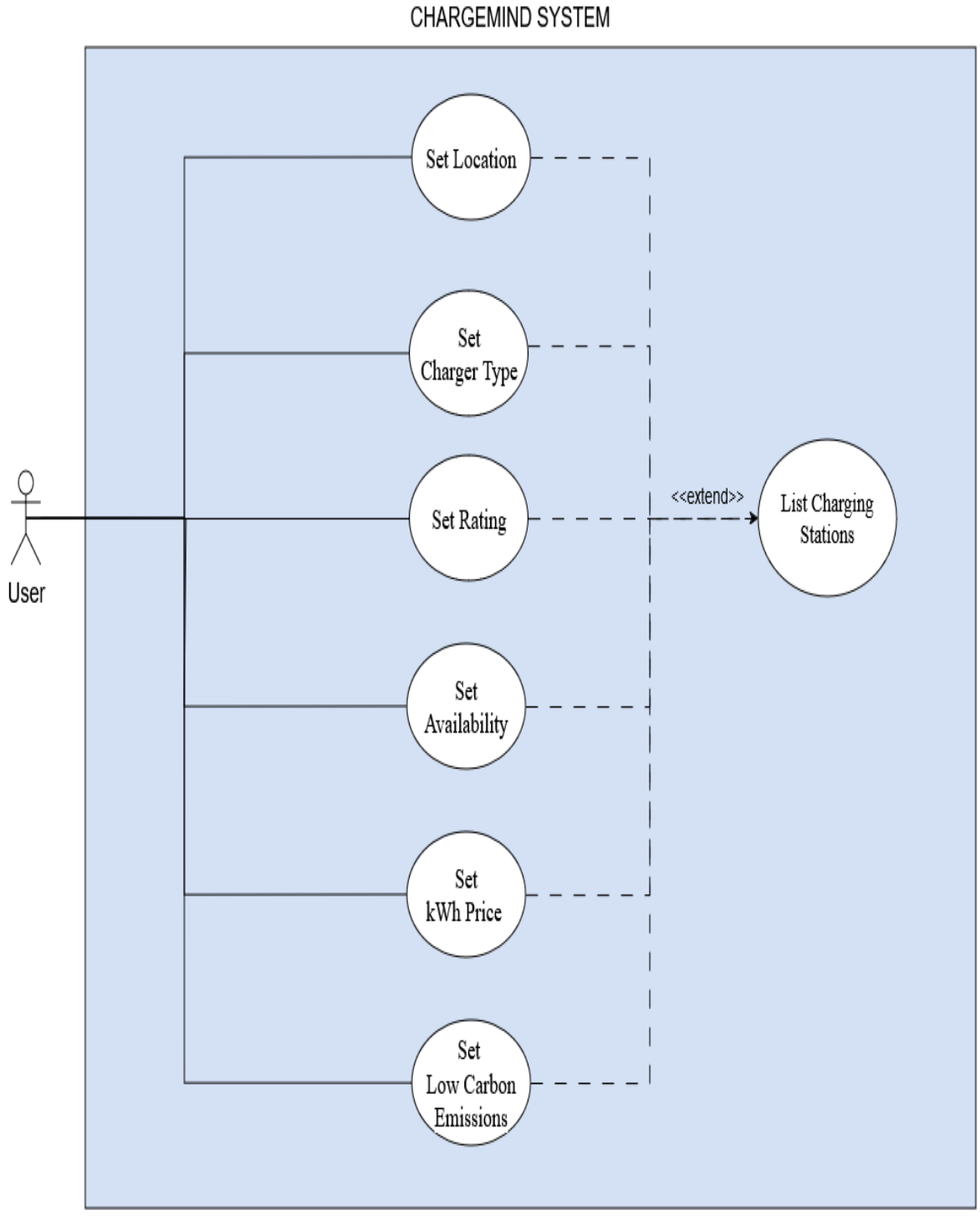
Description	The system finds the optimal path for the user based on their current location, destination, and filter choices.
Precondition	The user must provide their current location, destination location, and filter choices (if any).
Scenario	1.The user opens the system. 2.The user enters their current 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.



4.3 UC-3

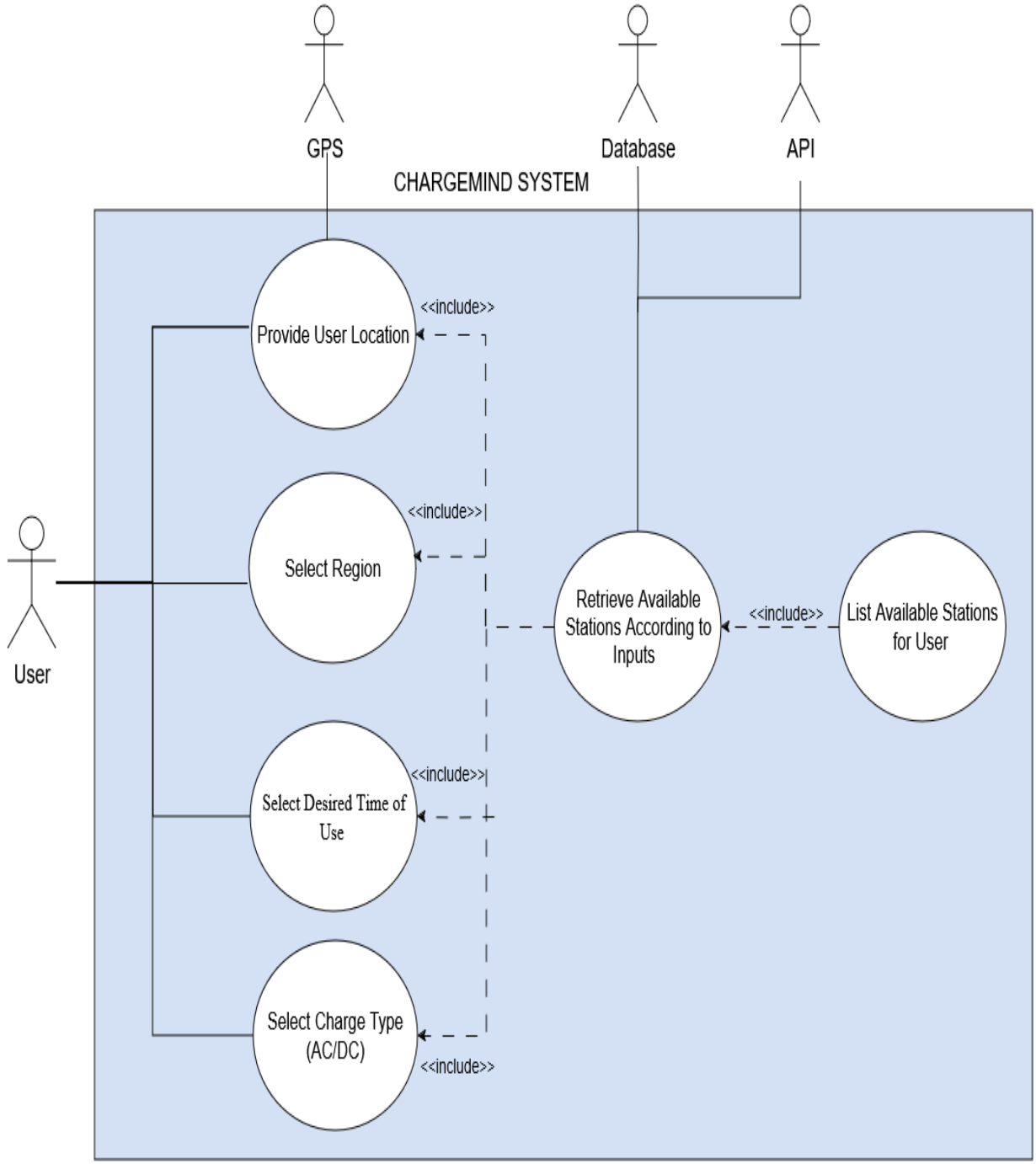
Name	Filter
Use Cases	UC-3
Actor	User

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

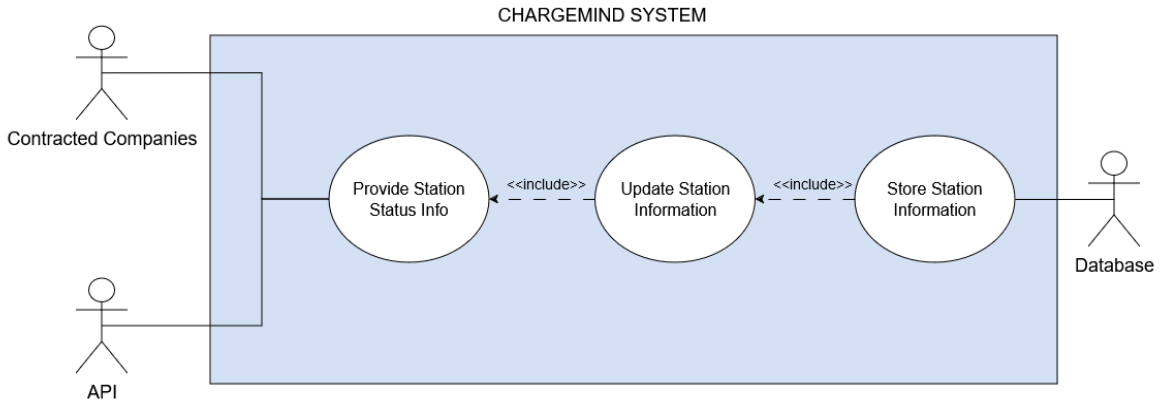
4.4 UC-4

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>>
Retrieve Available Stations According to Inputs
<<include>>
List Available Stations for User
Select Desired Time of Use
<<include>>
Select Charge Type (AC/DC)
<<include>>
User

4.5 UC-5

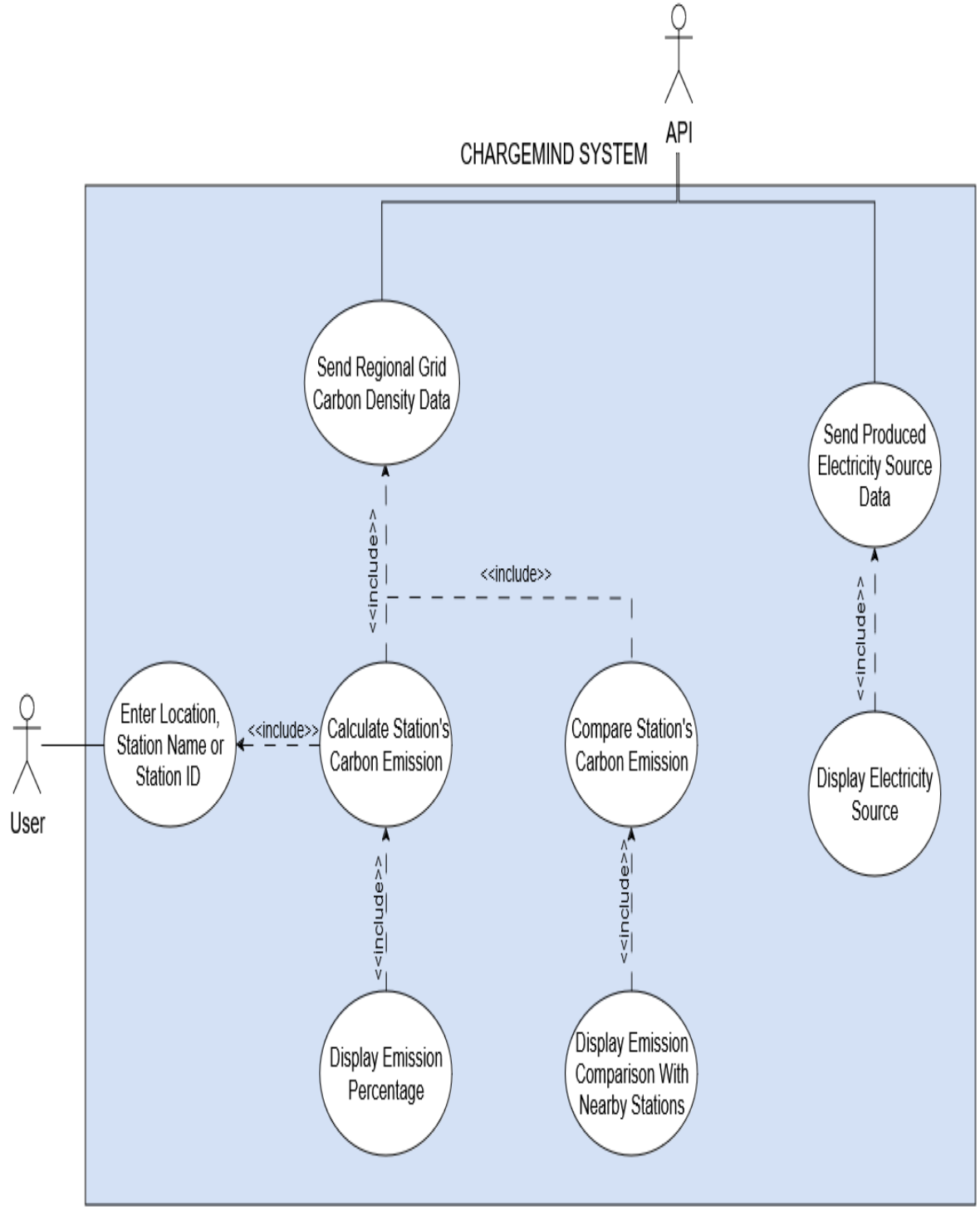
Name	Stations Information Updates
Use Cases	UC-5
Actor	Contracted Companies, API, Database
Description	The system updates the status and information of charging stations based on data provided by contracted companies and APIs. The updated information is stored in the database.
Precondition	Contracted 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.



4.6 UC-6

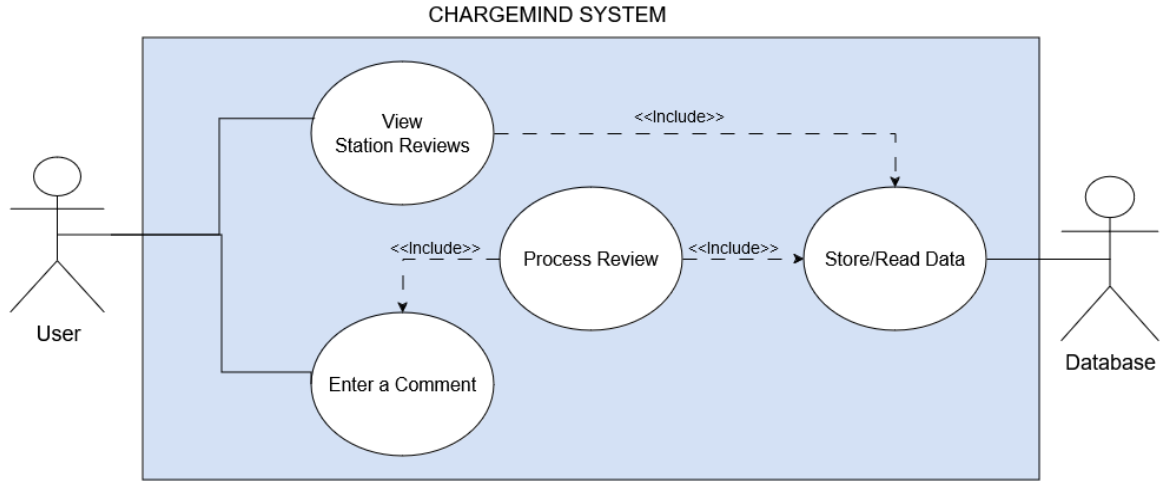
Name	Carbon Emission Calculation
Use Cases	UC-6
Actor	User, API
Description	The system calculates the carbon emissions of a charging station based on the regional grid carbon density and the electricity source data. It also allows the user to compare the emissions with nearby stations and displays the 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>>
Enter Location, Station Name or Station ID
<<include>>
Calculate Station's Carbon Emission
Compare Station's Carbon Emission
Display Electricity Source
User
<<include>>
<<include>>
Display Emission Percentage
Display Emission Comparison With Nearby Stations

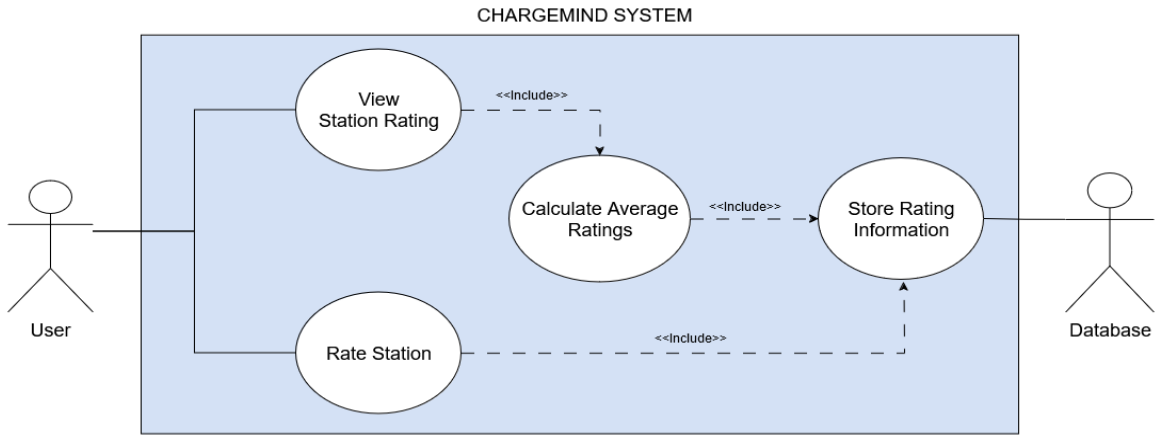
4.7 UC-7

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).



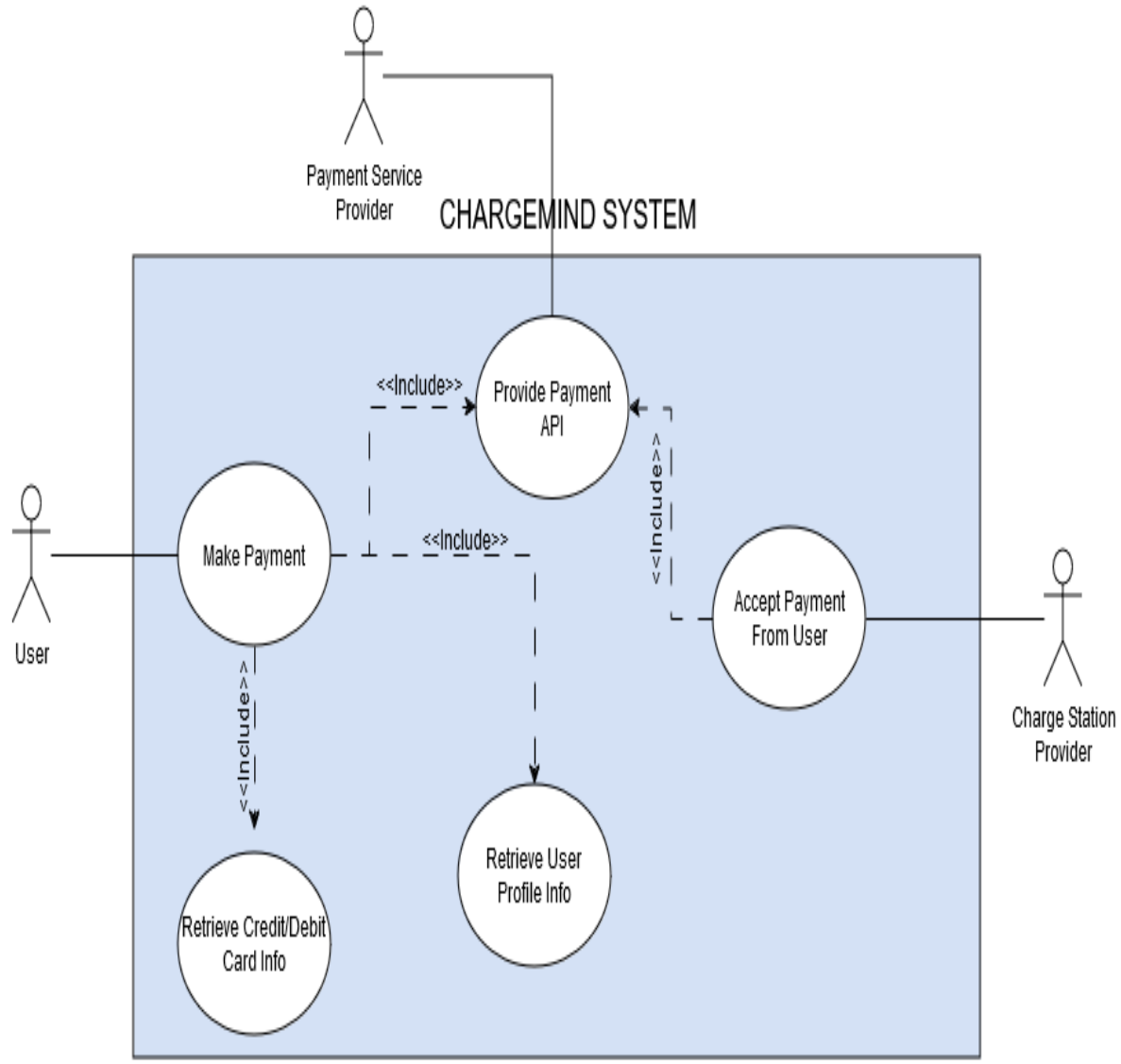
4.8 UC-8

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.



4.9 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>>
Make Payment
<<Include>>
<<Include>>
Accept Payment From User
User
Charge Station Provider
<<Include>>
Retrieve Credit/Debit Card Info
Retrieve User Profile Info